International Journal of Advance Research In Science And Engineering http://www.ijarse.com
IJARSE, Vol. No.4, Special Issue (02), February 2015 ISSN-2319-8354(E)

A COMPARITIVE ANALYSIS OF MULTIPLIERS USING GDI TECHNIQUE

Duraisingh Ruth Anita Shirley[1], Shanmuga Priya.K[2], Nagalingam Rajeswaran[3]

[1]PG students / VLSI Design, SNS College of Technology, Coimbatore, (India)
[2,3] Dept.of ECE, SNS College of Technology, Coimbatore,(India)

ABSTRACT

A bountiful of adders has been designed over the years in order to simplify the multiplication with various improvements. A comparison of Complementary Pass Transistor Logic and Shannon Adder and Gate-Diffusion Input has been performed to determine the latter adder as the optimised one in terms of power consumption and area. This adder is then implemented in three types of multiplies: Array multiplier, Wallace-tree multiplier and Modified Baugh-Wooley multiplier. The modified Baugh-Wooley multiplier is more advantages than the other two multipliers due to reduction in latency caused while passing the partial products from one adder to the other. A comparison table of the power consumed by the three multipliers is draw and it is found that the Baugh-Wooley multiplier utilizes less amount of power than the rest. This paper presents the power consumption comparisons and delay of various designs of multipliers.

I INTRODUCTION

A binary multiplier is an electronic circuit used in digital electronics, such as a computer, to multiply two binary numbers. It is built using binary adders. A variety of computer arithmetic techniques can be used to implement a digital multiplier. Most techniques involve computing a set of partial products, and then summing the partial products together. This process is similar to the method taught to primary schoolchildren for conducting long multiplication on base-10 integers, but has been modified here for application to a base-2 (binary) numeral system. 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 (Ayman.A et al (2001)). 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 rather complex circuits and must typically operate at a high system clock rate, reducing the delay of a multiplier is an essential part of satisfying the overall design

II. LITERATURE REVIEW OF DIFFERENT MULTIPLIER CIRCUITS

2.1. Complementary Pass Transistor Logiic (CPL) Adder

The Complementary pass-transistor logic (CPL) full adder having 32 transistors and using the CPL gates has been designed. The complexity of full CMOS pas gate logic can be reduced dramatically by adopting another circuit called CPL. The main idea behind CPL is to use a purely NMOS pas transistor network for the logic operations. Al the inputs are applied in complementary form. i.e.: every input signal and its inverse must be provided. The circuit also produces complimentary output, to be used by subsequent CPL.

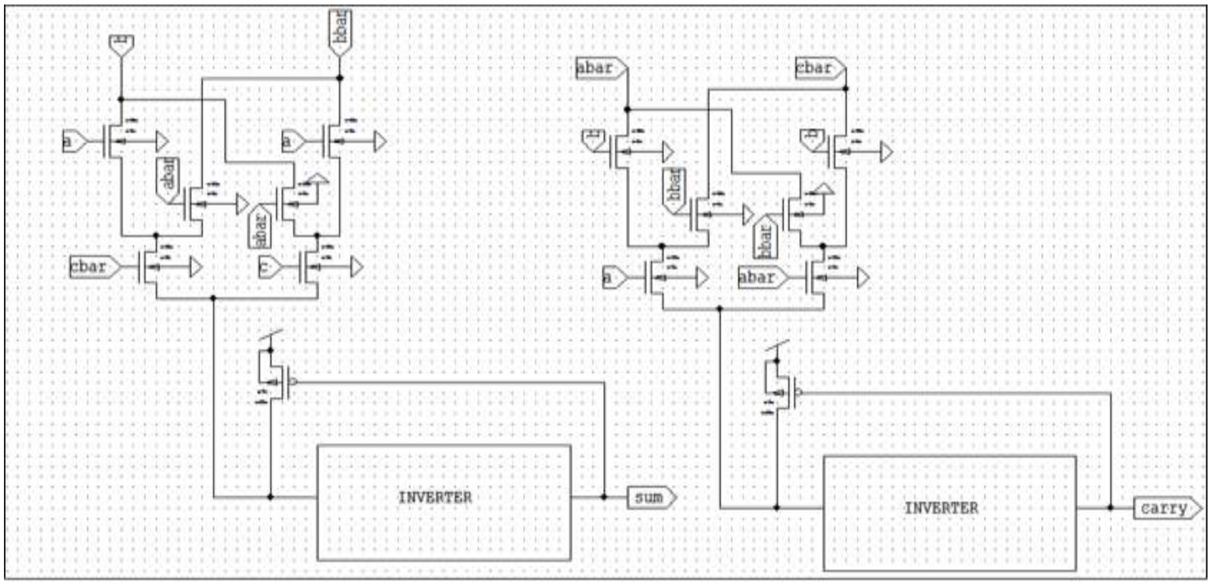


Fig. 1 CPL Adder

The CPL has certain drawbacks due top source follower action, body effect, limited fan-out capability, high leakage power when not cross coupled. Duality principle enables logic function such as AND-OR, NAND-NOR, XOR-XNOR. By just interchanging the inputs AND, OR basic logic gates, MUX circuits can be constructed.

2.2 Shannon Full Adder

The MCIT technique is developed by using Karnaugh map from the Boolean expressions for the sum and carry signals from the standard truth table for the full adder circuit. The Boolean expressions are given as:

$$C= AB + BC + CA \qquad \text{----(1)}$$

$$S= ABC + A'B'C + AB'C' + A'BC' \qquad \text{----(2)}$$

By using expressions (1) and (2), the pass transistor functions can be implemented. When the expression result =0, the pass transistor function is given by the complement of the input variable. If the expression result = 1, the pass transistor function is given by the input variable. To implement the pass transistor function for ‘n’ input variables, we use n-1 control input data and only one input data. The source input acts as an input of the signal.

The formulae for the i^{th} stage are given as:

$$C_{i+1}=A_iB_i+(A_i \text{ xor } B_i)\, C_i$$

$$S_i=A_i \text{ xor } B_i \text{ xor } C_i$$

According to Shannon’s theorem any logic expression is divided into two terms. One with a particular variable set to 1and multiplying it by a variable and then set the variable to 0 and multiplying it by the inverse. The fullest reduction can be obtained by continuously repeating the Shannon theorem. This method is useful especially to multiplier and pass transistor circuit design. The Shannon’s theorem in a generalised way can be stated as a function of many variables, f (b0, b1, b2, y, bi, y, bn) can be written as the sum of two terms, say one with a particular variable ai , set to 0, and one with it set to 1.

$$f(b0, b1, b2, \ldots\ldots, bi,\ldots\ldots y, bn) = bi' f (b0, b1, b2, \ldots\ldots, 0,\ldots\ldots y, bn) +$$
$$bi\ f (b0, b1, b2, \ldots\ldots, 1,\ldots\ldots y, bn)$$

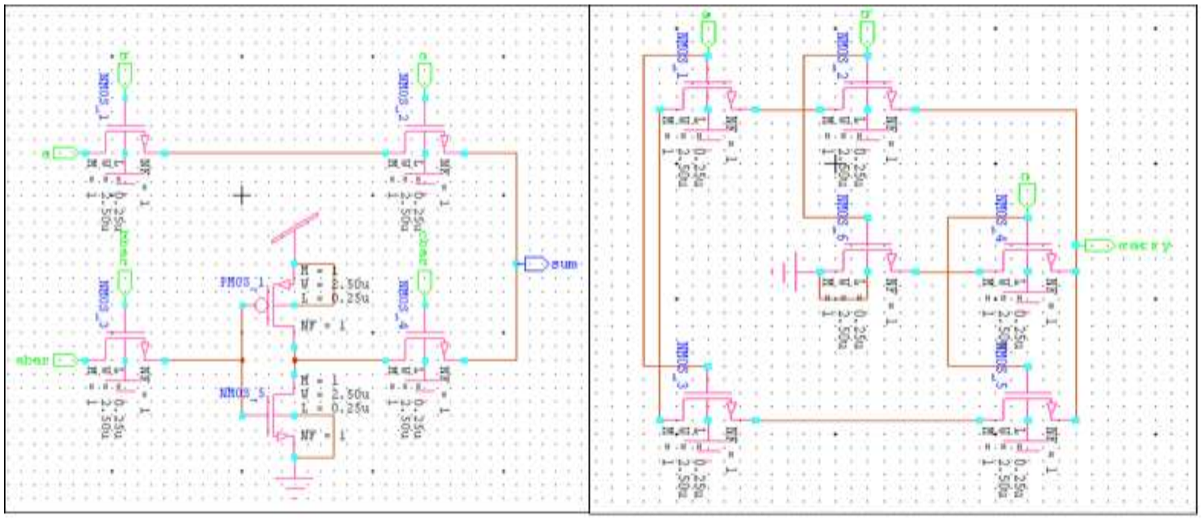


Fig. 2 Shannon Adder

Shannon’s theorem is applied to the logical function using n-1 variables as control inputs and three data lines set to a logical ‘1’. These source inputs are then connected to the VDD lines (logical ‘0’), which are connected to the ground. The remaining nth variable is connected from the data input to the source input. The data signals flow horizontally and the control signals flows vertically. Remove pairs of transistors when they cancel each other. The Shannon expression output depends upon the pass logic ‘1’ or logic ‘0’. If it has logic’0’ then the connection input is given by 0 and by ‘1’ for the connection input ‘1’.

2.3 Gate Diffusion Input

Gate diffusion input is a novel technique for low power digital circuit design in an embedded system. This technique allows reduction in power consumption, delay and area of the circuit. This technique can be used to reduce the number of transistors compared to conventional CMOS design. Recently, a novel design called Gate-Diffusion Input (GDI) is proposed by Morgenshtein et. al.. It is a genius design which is very flexible for digital circuits. Besides, it is also power efficient without huge amount of transistor count. Although GDI has the above advantages, it still has some difficulties that are needed to be solved. The major problem of a GDI cell is that it requires twin-well CMOS or silicon on insulator (SOI) process to realize. Thus, it will be more expensive to realize a GDI chip. However, if only standard p-well CMOS process can be used, the GDI scheme will face the problem of lacking driving capability which makes it difficult to realize a feasible chip. The basic GDI cell is shown in figure. It should be noted that the source of the PMOS in a GDI cell is not connected to VDD while the source of the NMOS in a GDI cell is not connected to GND. This feature gives the GDI cell two extra input pins to use which makes the GDI design more flexible than a usual CMOS design. However, this feature is also the major cause of its disadvantage: special CMOS process required. To be more specific, the GDI scheme requires twin-well CMOS or silicon on insulator (SOI) process to implement which is of course more expensive than the standard p-well CMOS process.

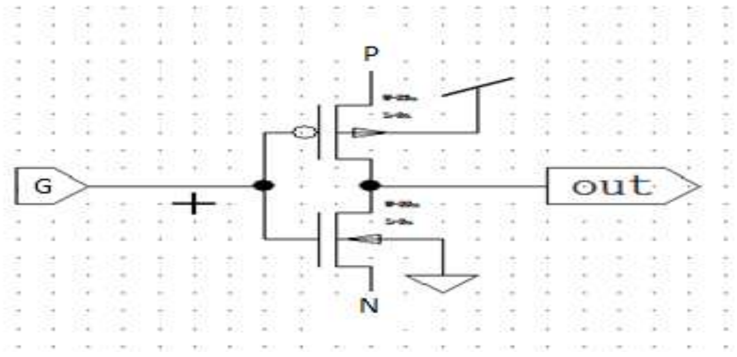


Fig. 3 Basic GDI Cell

GDI cell consists of three input terminals G, P and N. The various functions that can be implemented with basic GDI cell, which consists of only two transistors is as shown in below.

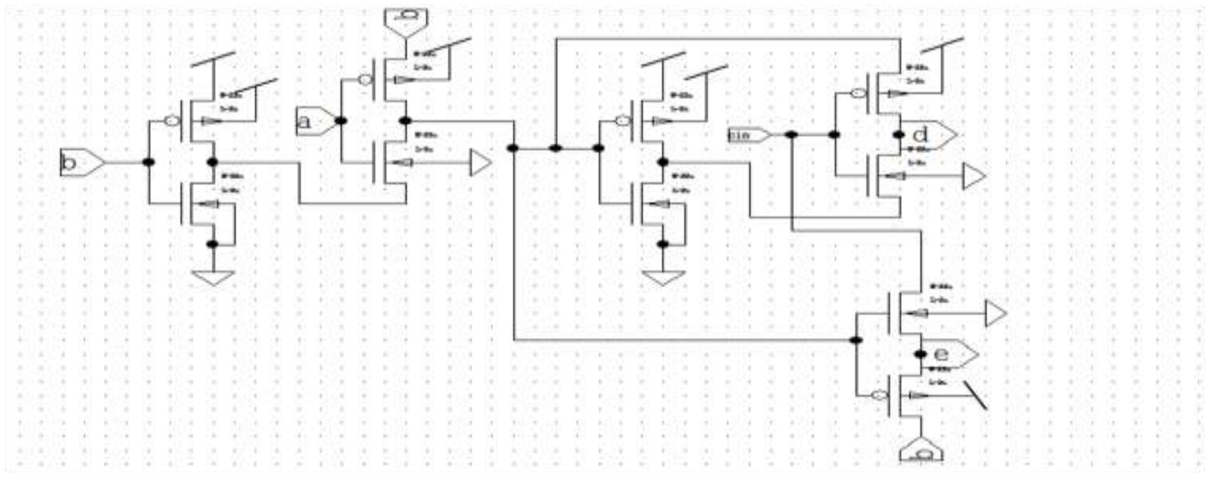


Fig. 4 GDI Full Adder

A full adder is implemented with the help of GDI technique. Here two multiplexers and an XOR gate operate together to form the full adder. It is observed that GDI adder is the optimum adder based on power consumed. Hence the multipliers are implemented with the GDI adder.

2.4 Array Multiplier

The array multiplier originates from the multiplication parallelogram. As shown in Fig.5, each stage of the parallel adders should receive some partial product inputs. The carry-out is propagated into the next row. The bold line is the critical path of the multiplier. In a non-pipelined array multiplier, all of the partial products are generated at the same time. It is observed that the critical path consists of two parts: vertical and horizontal. Both have the same delay in terms of full adder delays and gate delays. For an n-bit by n-bit array multiplier, the vertical and the horizontal delays are both the same as the delay of an n-bit full adder.

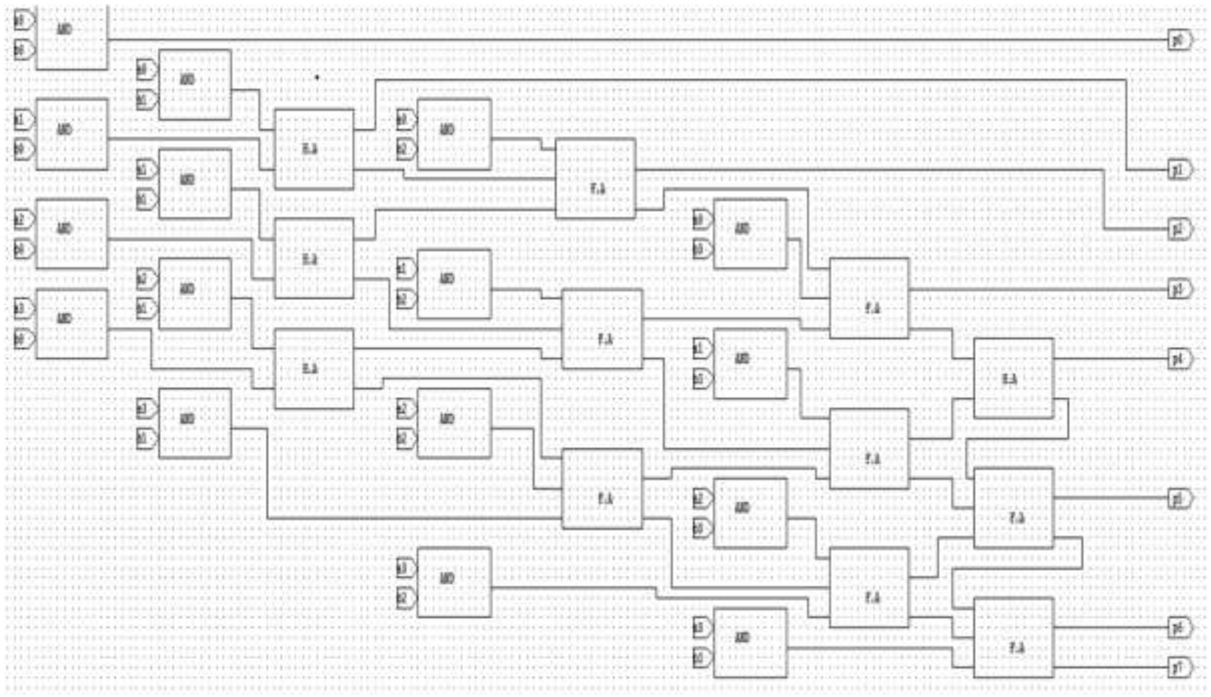


Fig. 5 Array Multiplier

One advantage of the array multiplier comes from its regular structure. Since it is regular, it is easy to layout and has a small size. The design time of an array multiplier is much less than that of a tree multiplier. A second advantage of the array multiplier is its ease of design for a pipelined architecture. The main disadvantage of the array multiplier is the worst-case delay of the multiplier proportional to the width of the multiplier. The speed will be slow for a very wide multiplier.

2.5 Wallace Tree Multiplier

Several popular and well-known schemes, with the objective of improving the speed of the parallel multiplier, have been developed in past. Wallace introduced a very important iterative realization of parallel multiplier. This advantage becomes more pronounced for multipliers of bigger than 16 bits. In Wallace tree architecture, all the bits of all of the partial products in each column are added together by a set of counters in parallel without propagating any carries. Another set of counters then reduces this new matrix and so on, until a two-row matrix is generated. The most common counter used is the 3:2 counters which is a Full Adder. The final results are added using usually carry propagate adder. The advantage of Wallace tree is speed because the addition of partial products is now O (logN). A block diagram of 4 bit Wallace Tree multiplier is shown in below. As seen from the block diagram partial products are added in Wallace tree block. The result of these additions is the final product bits and sum and carry bits which are added in the final fast adder (CRA).

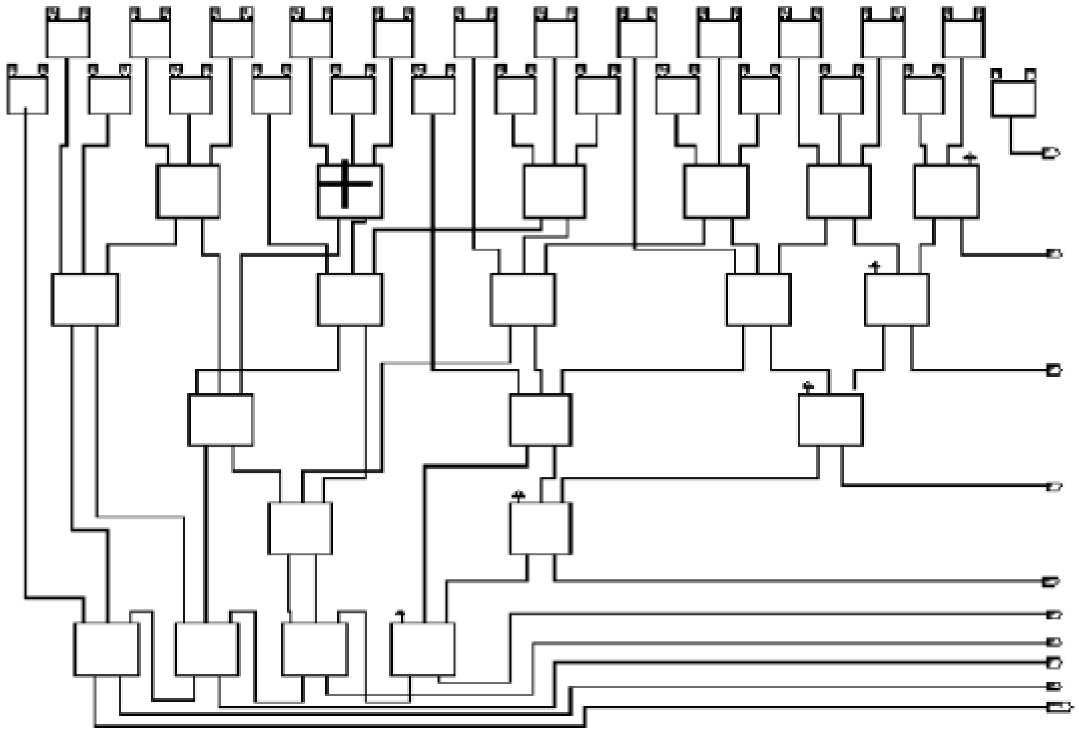


Fig. 6 Wallace Tree Multiplier

A typical Wallace tree multiplier is designed and implemented in Fig. 6. The Wallace tree is designed with the help of full adders and AND logic gates.

2.6 Modified Baugh-Wooley Multiplier

One important complication in the development of the efficient multiplier implementations is the multiplication of two's complement signed numbers. The Modified Baugh-Wooley Two's Complement Signed Multiplier is the best known algorithm for signed multiplication because it maximizes the regularity of the multiplier logic and allows all the partial products to have positive sign bits.

Baugh-Wooley technique was developed to design direct multipliers for two's complement numbers. When multiplying two's complement numbers directly, each of the partial products to be added is a signed number. Thus, each partial product has to be sign-extended to the width of the final product in order to form the correct sum by the Carry Save Adder tree. According to the Baugh Wooley approach, an efficient method of adding extra entries to the bit matrix is suggested to avoid having to deal with the negatively weighted bits in the partial product matrix.

Knowing that the sign bit in two's complement numbers has a negative weight, hence the term can be written in terms of

$$-a_3b_0 = a_3(1-b_0) - a_3 = a_3\bar{b}_0 - a_3$$

The unsigned multiplication of two four-bit digits results in the formation of 8 terms P1 to P8, where each term represents the total addition of the partial products formed. Here P8 also indicates the carry formed during the multiplication process.

The 2's complement of the 4x4 multiplication can be represented as:

$$\begin{array}{cccccccc}
 & & & & a_3 & a_2 & a_1 & a_0 \\
 & & & & b_3 & b_2 & b_1 & b_0 \\
\cline{5-8}
 & & & & -a_3b_0 & a_2b_0 & a_1b_0 & a_0b_0 \\
 & & & -a_3b_1 & a_2b_1 & a_1b_1 & a_0b_1 & \\
 & & -a_3b_2 & a_2b_2 & a_1b_2 & a_0b_2 & & \\
 & a_3b_3 & -a_2b_3 & -a_1b_3 & -a_0b_3 & & & \\
\hline
P8 & P7 & P6 & P5 & P4 & P3 & P2 & P1 \\
\hline
\end{array}$$

Hence, the term $-a_3b_0$ is replaced with $a_3\bar{b}_0$ and $-a_3$. If a_3 is used instead of $-a_3$, the column sum increases by $2a_3$. Thus, $-a_3$ must be inserted in the next higher column in order to compensate the effect of $2a_3$. The same is done for $a_3\bar{b}_1$, $a_3\bar{b}_2$ and $a_3\bar{b}_3$. In each column, a_3 and $-a_3$ cancel each other out. The P7 column gets a a-a_3 entry, which is replaceable by $\bar{a}$-1. This can be repeated for all entries, yielding to the insertion of b_3 in the P4 column, and $\bar{b}_4$-1 in the column P8. There are two -1's in the eighth column now, which is equivalent to a -1 entry in P8 and that can be replaced with a 1 and borrow into the non-existing tenth column.

Baugh-Wooley method increases the height of the longest column by two, which may lead to a greater delay through the Carry Save Adder tree. This can reduce the extra delay caused by the additional Carry Save Adder level. Thus, the maximum number of entries in one column becomes six, which can be implemented with three level Carry Save Adder tree.

All negatively weighted a_3b_3 terms can be transferred to the bottom row, which leads to two negative numbers in the last two rows, where a subtraction operation from the sum of all the positive elements is necessary. Instead of subtracting a_3b_3 two's complement of a can be added b_4 times.

Modified form of the Baugh-Wooley method, is more preferable since it does not increase the height of the columns in the matrix. However, this type of multiplier is suitable for applications where operands with less than 32 bits are processed, like digital filters where small operands like 6, 8, 12 and 16 bits are used. Baugh-Wooley scheme becomes slow and area consuming when operands are greater than or equal to 32 bits.

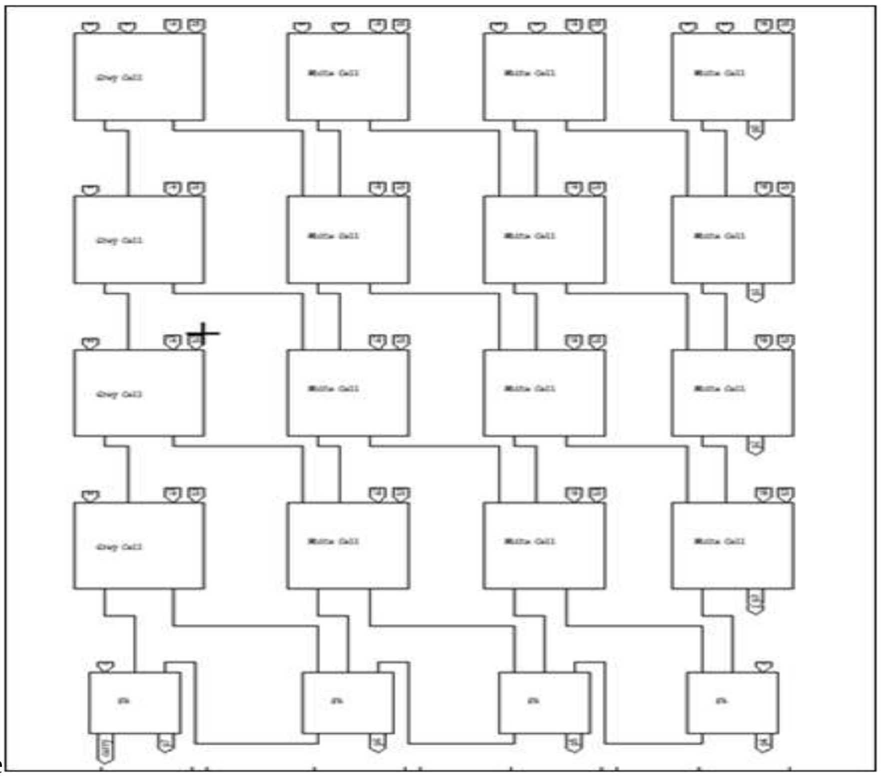
$$\begin{array}{cccccccc}
 & & & & a_3 & a_2 & a_1 & a_0 \\
 & & & & b_3 & b_2 & b_1 & b_0 \\
\hline
 & & & & \overline{a3b0} & a_2b_0 & a_1b_0 & a_0b_0 \\
 & & & \overline{a3b1} & a_2b_1 & a_1b_1 & a_0b_1 & \\
 & & \overline{a3b2} & a_2b_2 & a_1b_2 & a_0b_2 & & \\
 & a_3b_3 & \overline{a2b3} & \overline{a1b3} & \overline{a0b3} & & & \\
\hline
P8 & P7 & P6 & P5 & P4 & P3 & P2 & P1 \\
\hline
\end{array}$$

The Baugh-Wooley multiplier is implemented with the help of two blocks known as

- White cell
- Grey Cell

The white cell consists of a AND gate to which two inputs are fed from a and b respectively. The output of the AND gate is fed to a full adder circuit along with the sum bit and carry bit from the previous stages. A grey cell is similar to the white cell with the exception that a NAND gate is used in place of the AND gate.

The Baugh-Wooley multiplier that is designed with white and grey cell is shown in Fig. 7.



e

Fig 7. Modified Baugh Wooley Multiplier

III. SIMULATION AND ANALYSIS

3.1 Simulation Environment

All the circuits have been simulated using 90 nm technologies on Tanner EDA tool. To make the impartial testing environment all the circuits has been simulated on the same input patterns. Tanner EDA provides of a complete line of software solutions for the design, layout and verification of Analog and Mixed-Signal (A/MS) ICs.T-Spice Pro is Tanner EDA's design entry and simulation system includes S-Edit for schematic capture T-Spice for circuit simulation, and W-Edit for waveform probing.

3.2 Performance Analysis

TABLE 1 depicts a comparison of adders of which GDI Technique shows minimum Power consumption. Based on the optimum adder, the multipliers designed are compared in terms of power and delay. It is found that Modified Baugh-Wooley multiplier is the optimum multiplier in terms of power and delay.

TABLE 1: Power Consumed Comparison of different Adder Circuit

PARAMETER	CPL	Shannon Adder	GDI Technique
Transistor Count	18	12	10
Power Consumed	0.1179	0.05624	0.002714

TABLE 2: Power Delay Comparison of different Multiplier Circuit

PARAMETER	Array Multiplier	Wallace Tree Multiplier	Modified Baugh-Wooley
Transistor Count	18	12	10
Power Consumed	0.5373	0.5118	0.4983
Delay	1.34	1.005	0.87

The simulation result for the GDI adder is as given in fig. 8.

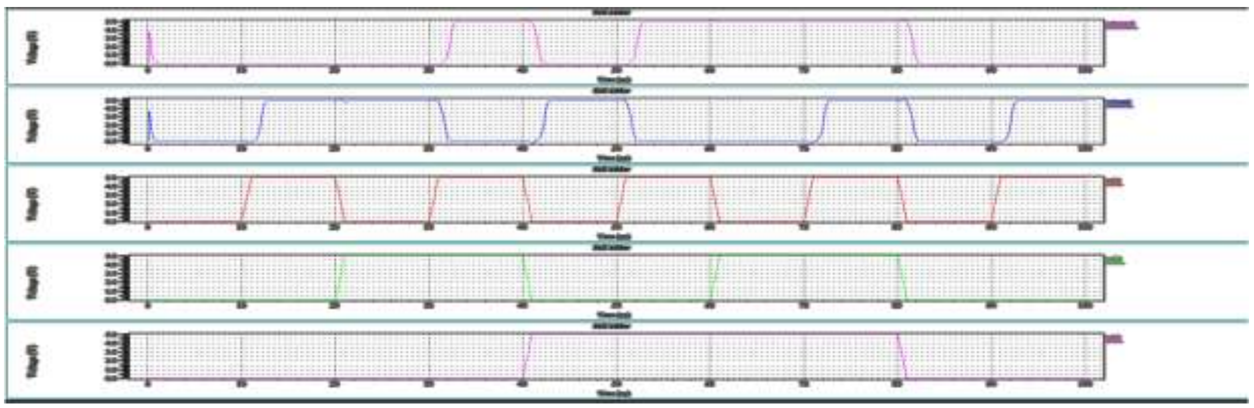


Fig. 8 Simulation result for GDI adder circuit

IV. CONCLUSION

This project resulted in a 4-bit multiplier able to handle signed multiplication. It was constructed and tested in the Tanner EDA software environment. The multiplier is implemented using the Baugh-Wooley algorithm, which is designed to use very few logical operations in each step of the multiplication operation and therefore does not suffer large cumulative gate delays. In addition, the algorithm uses common logical units to simplify design and deals with signed inputs more efficiently than other algorithms. This produces a relatively efficient calculation time, while maintaining low-cost production costs. Although the algorithm is not able to work with unsigned inputs, the goal of the project was a signed multiplier.

REFERENCES

[1] Yano, K., Sasaki, Y., Rikino, K. and Seki, K. (1996) "Top-down pass transistor logic design", IEEE Journal of Solid-State Circuits 31(6), 792–803.

[2] Pramodini Mohanty and Rashmi Ranjan, "An Efficient Baugh Wooley Architecture for both Signed and Unsigned Multiplication", International Journal of Computer Science & Engineering Technology (IJCSET), Vol. 3 No. 4 April 2014.

[3] Niichi Itoh, Yuka Naemura, Hiroshi Makino, Yasunobu Nakase, Tsutomu Yoshihara, and Yasutaka Horiba, "A 600-MHz 54x54-bit Multiplier with Rectangular-Styled Wallace Tree" IEEE Journal of Solid State Circuits, vol. 36, no.2, Feb 2001.

[4] Antonio G. M. Strollo and Davide De Caro, "Booth Folding Encoding for High Performance Squarer Circuits" IEEE transactions on circuits and systems—ii: analog and digital signal processing, vol. 50, no. 5, May 2003.

[5] K.-S. Chong, B.-H. Gwee and J.-S. Chang, "Low energy 16-bit Booth leapfrog array multiplier using dynamic adders" IET Circuits Devices Syst., vol. 1, no. 4, Feb 2007.

[6] Shiann-Rong Kuang, *Member*, *IEEE*, Jiun-Ping Wang, and Cang-Yuan Guo, "Modified Booth Multipliers With a Regular Partial Product Array" IEEE transactions on circuits and systems—ii: express briefs, vol. 56, no. 5, May 2009.

[7] Jin-Hao Tu and Lan-Da Van, Member, IEEE "Power-Efficient Pipelined Reconfigurable Fixed-Width Baugh-Wooley Multipliers", IEEE transactions on computers, vol. 58, no. 10, Oct 2009.

[8] Theo A. Drane, Thomas M. Rose, and George A. Constantinides, Senior Member, IEEE, "On the Systematic Creation of Faithfully Rounded Truncated Multipliers and Arrays" IEEE transactions on computers, vol. 63, no. 10, Oct 2014.

[9] E. Walters and M. Schulte, "Fast, Bit-Accurate Simulation of Truncated-Matrix Multipliers and Squarers," Proc. 2010 Conf. Record of the 44th Asilomar Conf. Signals, Systems and Computers (ASILOMAR), pp. 1139-1143, Nov. 2010.