LOW POWER AND HIGH SPEED MULTIPLIER DESIGN USING SWITCHED OUTPUT DIFFERENTIAL STRUCTURE

Thesis submitted in the partial fulfillment of requirement for the award of degree of

Master of Technology

in

VLSI Design

Submitted by:

Shakun

Roll No: 601061022

Under the guidance of:

Mr. Arun Kumar Chatterjee

Assistant Professor

T.U, Patiala

ELECTRONICS AND COMMUNICATION ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

(Established under the section 3 of UGC Act, 1956)

PATIALA – 147004 (PUNJAB)

DECLARATION

I, **Shakun**, hereby certify that the work which is being presented in this thesis entitled **"Low Power and High Speed Multplier Design using Switched Output Differential Structure"** by me in partial fulfillment of the requirements for the award of degree of Master of Technology in VLSI Design from Thapar University (Deemed University), Patiala, is an authentic record of my own work carried out under the supervision of **Mr. Arun Kumar Chatterjee.**

The matter presented in this thesis has not been submitted in any other University / Institute for the award of any other degree.

Date: 14-06-2012

(Shakun)
Roll No. 601061022

It is certified that the above statement made by the student is correct to the best of my knowledge and belief.

Date: 14/06/2012

Arun Kumar Chatterjee.

(Mr Arun Kumar Chatterjee)
Assistant Professor
ECED

Countersigned by:

(Dr. Rajesh Khanna)
Professor and Head ECED
Thapar University, Patiala
Date:

(Dr. S.K. Mohapatra)
Dean of Academic Affairs
Thapar University, Patiala
Date:

ACKNOWLEDGEMENT

First of all, I would like to express my gratitude to **Mr. Arun Kumar Chatterjee** Assistant Professor**,** Electronics and Communication Engineering Department, Thapar University, Patiala for his patient guidance and support throughout this research work. I am truly very fortunate to have the opportunity to work with him.

I convey my sincere thanks to **Head of the Department, Dr. Rajesh Khanna** as well as **PG Coordinator, Dr. Kulbir Singh, Assistant Professor,** Electronics and Communication Engineering Department, entire faculty and staff of Electronics and Communication Engineering Department for their encouragement and cooperation. I would also like to thanks to SMDP I & II, Govt. of India for providing the tool to carry out this thesis work.

My greatest thanks are to all who wished me success especially my parents. Above all I render my gratitude to the Almighty who bestowed self-confidence, ability and strength in me to complete this work for not letting me down at the time of crisis and showing me the silver lining in the dark clouds. I do not find enough words with which I can express my feelings of thanks to my dear friends for their help, inspiration and moral support which went a long way in successful competition of the present study.

Shakun

ABSTRACT

Today the use of a CMOS differential style has reached a wide range of development such that complex Boolean functions are performed with less transistor count and less propagation delay. In the CMOS logic structure, the redundant logic has to implement in the PMOS block that same logic have to implement in the NMOS. PMOS has to be large (3 to 4 times) to gain same speed or resistance of NMOS that increases the area required to implement any logic using CMOS structure. There are high speed techniques that are used to reduce the area of the LOAD block, in which the minimum number of PMOS transistor has been used that decreases the area of the any logic design and increases the speed. Some high speed structures such as DCVS-domino, ECDL and SODS structure have been discussed here, in terms of speed and area. The disadvantage of DCVS-Domino and SODS structure is minimum logical depth that increases the delay. The SODS structure has been modified. In the modified SODS, differential structure has been used to implement the output based on the difference generated by the NMOS tree. The difference generated by the NMOS tree is used as an input for cross coupled inverter pair to generate the rail to rail swing at the output.

The modified SODS structure has been used in designing of a multiplier, which gives better result than that of the basic SODS adder. The 2-bit and 4-bit multipliers have been designed and simulated on Cadence Spectre. The simulation results have shown that delay of modified structure has been reduced in comparison of basic SODS structure. The delay for 2-bit multiplier has been reduced from 390 ps to 360 ps and also for 4-bit multiplier reduced from 1.25 ns to 950 ps. In this thesis work, Braun algorithm is used for the design of multiplier architecture. The low power technique, MTCMOS, has been used to reduce power dissipation of the multiplier circuit. The low power technique reduced the power dissipation from 272 nW/MHz to 210 nW/MHz for the sum circuit. The design has been simulated at UMC 180nm Technology with power supply of 1.8V in Cadence Analog Design Environment. Layout has been drawn and LVS has been verified. RCX extraction has also been generated with the help of tool.

TABLE OF CONTENTS

List of Abbreviations

Abbreviation	Meaning
MOSFET	Metal oxide semiconductor field effect transistor
CMOS	Complementary MOSFET
NMOS	n-type MOSFET
PMOS	p-type MOSFET
SODS	Switched output differential structure
DCVS	Differential Cascade voltage switched logic
ECDL	Enable/Disable CMOS differential logic
LVS	Layout Verses schematic match
RCX	Resistance and capacitance extraction
DSP	Digital signal processing
VLSI	Very large scale integration
DRC	Design rule check
MTCMOS	Multiple threshold voltage CMOS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

Multiplier is such an important element which contributes substantially to the total power consumption of the system. On VLSI level, the area also becomes quite important as more area means more system cost. Speed is another key parameter while designing a multiplier for a specific application. These three parameters i.e. power, area and speed are always traded off. Speaking of DSP processors, area and speed are the most important factors. But sometimes, increasing speed also increases the power consumption, so there is an upper bound of speed for a given power criteria. Considering the battery operated portable multimedia devices, low power and fast designs of multipliers are more important than area. So I am using here high speed circuit for the designing of multiplier. The use of CMOS differential-type logic has recently reached a wide range of development such that Complex Boolean functions are performed with less transistor-count and less propagation delay (equivalent to one gate), making this logic especially suitable for applications in high-speed VLSI circuits and the power consumption increases because there is always a trade between power and delay if delay decreases than the power dissipation increases thus the low power is used to reduce the power dissipation in the circuit. Logic functions are performed only with NMOS transistors. For this reason, inputs do not feed gates of PMOS transistors, and therefore, input load decreases. The general scheme of a differential-type circuit is basically composed of two blocks called NMOS-tree and load. The NMOS-tree performs the logic operation, and the load block acts as the pull-up for CMOS logic. However, one of the most important drawbacks prior to definite acceptance of differential logic circuits is the excessive hardware required to implement the load block. Although this excess relatively decreases when the complexity of the Boolean function increases, a very important task is to reduce hardware requirements to make these circuits competitive with their conventional equivalents.

Multiplication is one of the basic functions used in digital signal processing (DSP). Most high performance digital signal processing systems rely on hardware multiplication to achieve high data throughput. The important point is how the

multiplication is implemented. The design issue of the multiplier from circuit level to architecture level will be discussed and explored in this research work.

1.1 Motivation

In the past, many novel ideas for multipliers have been proposed to achieve high performance. They fall into two general categories: “tree" multiplier and “array" multiplier [7]. Another recent algorithm used to design multiplier is Braun Algorithm [2]. Tree multipliers add as many partial products in parallel as possible and therefore are very high performance Architectures. Unfortunately, tree multipliers are also very irregular, hard to layout and hence Large Area. Array multipliers, on the other hand, are very regular, small in size, but suffer in latency and propagation delay [1], [7]. The goal of a practical design is not only high performance but also a reasonable price. Reasonable pricing should be considered from the aspect of both speed and power dissipation, which means a high speed and a low power design. The problem with contemporary FPGA-based multiplier designs is that they are too slow and too large so the power Dissipation is high.

1.2Methods

Several tools were used in the design approach of the multiplier. Cadence was used for the logic simulation and the layout of the multiplier. Cadence integrates the back-end and front-end phase of the design process. This tool provides a digital simulator and a schematic editor for the front end of the design process. For the back end, it contains a layout editor and a design rule checker (DRC). Layout-versus-schematic (LVS) check integrates the front and the back end.

This report describes the design of an 4-bit multiplier, under which the basic cell i.e 1-bit full adder has been designed using a high speed circuit technique “Switched Output differential logic” [9], [10], [11] and low Power is achieved by using the MTCMOS technique[1].

1.3 Organization

This report is organized into five subsequent chapters. Chapter 2 discuss various circuit Design Techniques in detail. Chapter3 discuss various Multiplier Architecture

in detail. Chapter 4 discuss Low Power Design in detail. Chapter 5 discuss the Conclusion and future scope.

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 a top 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 [7]. Although most people think of multiplication only in base 10, this technique applies equally to any base, including binary. Figure 1.1 shows the data flow for the basic multiplication technique just described. Each black dot represents a single digit [7].

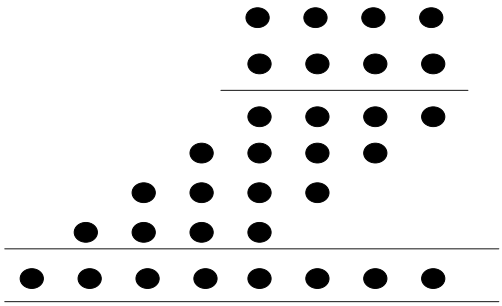


Figure 1.1: Basic Multiplication

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 which is shown in figure 1.2. 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.

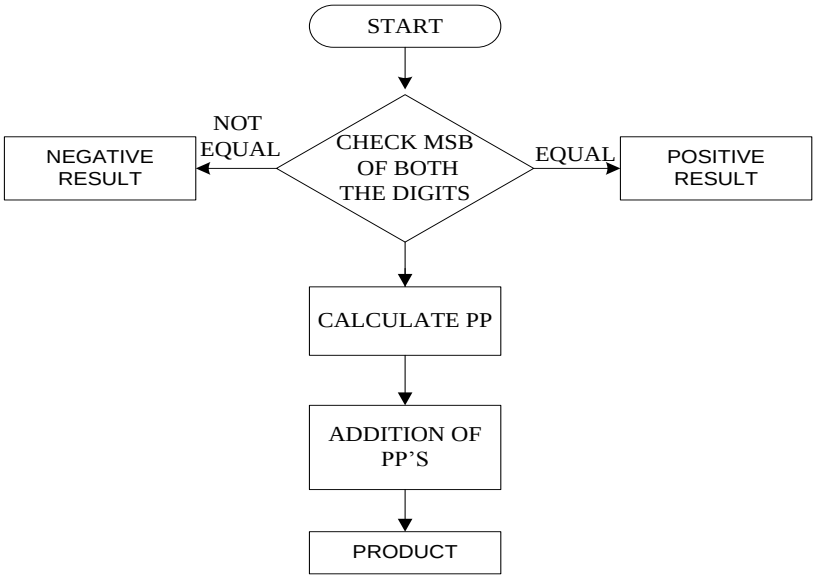


Figure 1.2: Signed Multiplication Algorithm

1.4.1 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 $x = x_{n-1}.....x_1\ x_0$ and the multiplier $y = y_{n-1}.....y_1y_0$ in order to form the product up to n shifted copies of the multiplicand are to be added for unsigned multiplication. The entire process consists of three steps, partial product generation, partial product reduction and final addition.

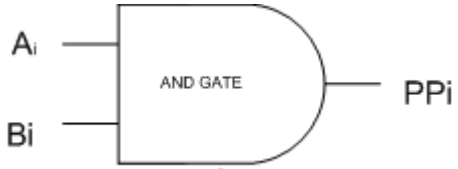


Figure 1.3(a): AND gate showing i^{th} partial term

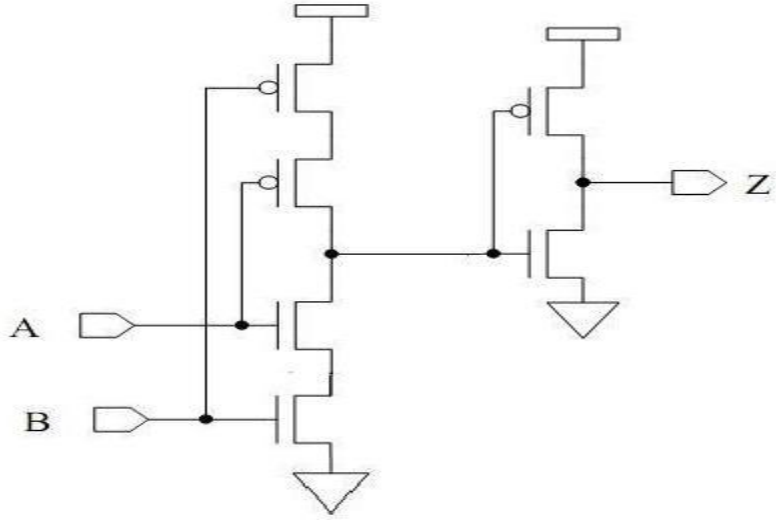


Figure 1.3(b): AND gate circuit level design

A simple AND gate can be used to generate the i-th partial product, hence we would require 16 AND gates to generate all the partial products in parallel at same instance to reduce the latency.

1.4.2 Partial Product Generation

In the digital multiplication, as in initial step, one needs to generate n shifted copies of the multiplicand, which may be added in the coming stage. The value of the multiplier bit determines whether the shifted copy is to be added or not: if the i-th bit of the multiplier is '1', then the shifted copy of the multiplicand is added. If the bit is '0', it is not added. A logical AND gate can implement this operation, by performing the function AND ($x_i y_i$) [7]. The resulting values are called partial products. This process is called partial product generation as all the bits are formed in the parallel in the PPA, thus the static delay of each bit is equal. The width of the PPA is proportional to the size of the multiplier. The bits in a particular column will be added later on and some columns have more bits than others.

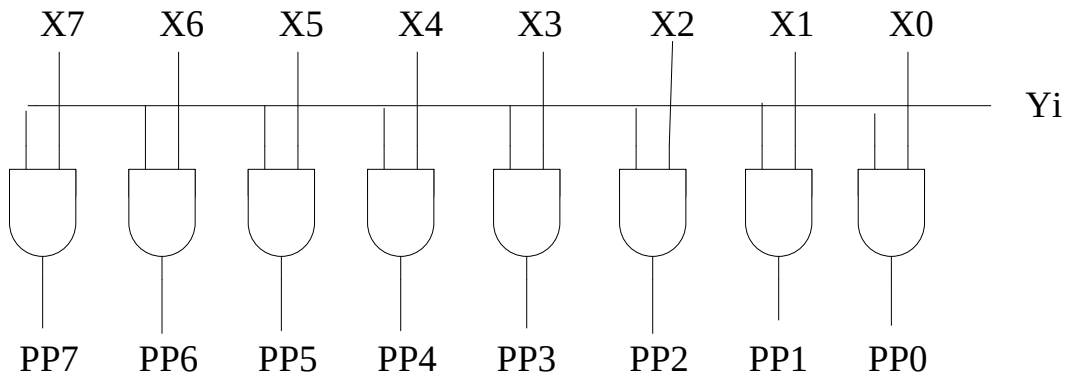


Figure 1.4: Partial Product Generation Logic [7]

1.4.3 Partial Product Reduction

Efficient implementation of a digital multiplier depends on the method of the addition of partial product array bits. Since each shifted version of the multiplicand will give a delay proportional to the width of the multiplicand, the multiplier blocks will require a large amount of time to perform the operation if conventional adders were used to implement this addition. Hence, the partial products are reduced using a technique called carry-save addition, which allows successive additions in one global step. [7].

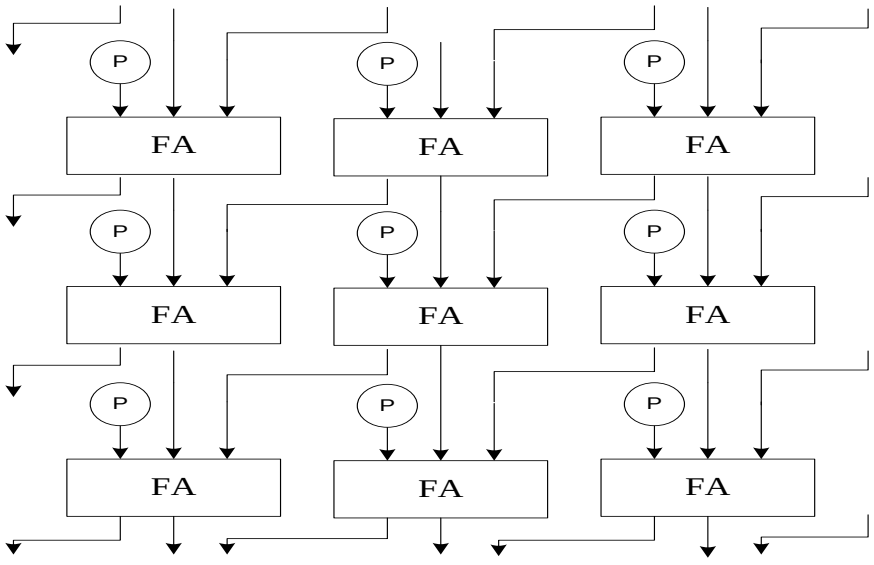


Figure 1.5: Carry save Array [6]

Considering the addition of two bits from two vectors, x and y , conventional full adders can be used which takes in three bits and output a sum and a carry bit, so the block adds two bits at a given position with the carry in from the previous bit position.

Considering the case of adding two bit vectors, two bits are added at the lower bit positions added carry is propagated to the next bit position. At the higher positions, two inputs and the carry bit are to be combined and a carry out is generated. The rippling technique of adding two n-bit numbers requires O (n) sequential bit additions, hence a delay of O (n). For the addition of three bit vectors of size n. X, Y, Z, total number of bit additions of n shifted copies of an n-bit multiplicand is O(n) where the total delay is $O(n^2)$. The carry save algorithm is shown in figure 1.5.

Even though the result comes from the combination of all operations, a certain amount of independence exists between each operation, considering the addition on a particular column. All the bits in a column must be added together along with the carry in bits coming from the previous column. Carry save addition influences that addition in separate columns can be performed independently. For example, in order to add three vectors of bits, full adders can be used to perform the addition of three bits in each column. Except the lowest and the highest bit positions, the result is a carry and a sum bit in each bit position. So, the three bit vectors have been reduced to two bit vectors.

Using the carry save addition technique, a set of vectors which are to be added together, can be reduced to two bit vectors. Carry save addition is one of the ways to make a multiplication faster than the conventional methods, considering the number of necessary additions.

1.4.4 Compressor

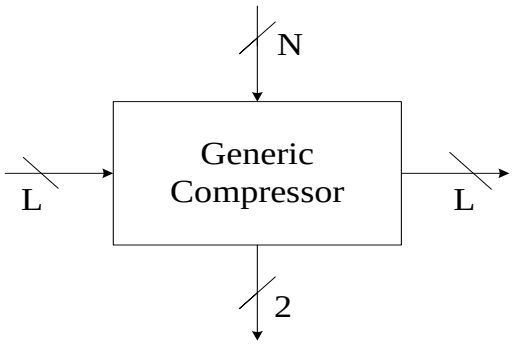


Figure 1.6: A Generic Compressor

Compressors are mostly used in multipliers to reduce the operands while adding terms of partial products. A Compressor Ci is a combinational device that compresses N input lines in the position i to 2 output lines i.e. sum and carry. In addition, there are L inputs lines coming to the compressor to different levels j. Figure 3.9 shows a simple compressor.

1.4.4.1 [3:2] Compressor

A [3:2] compressor is basically a full adder. It has 3 inputs A, B and C to be summed up and provides 2 outputs (sum and carry). Gate level diagram of [3:2] compressor is shown in figure 1.7.

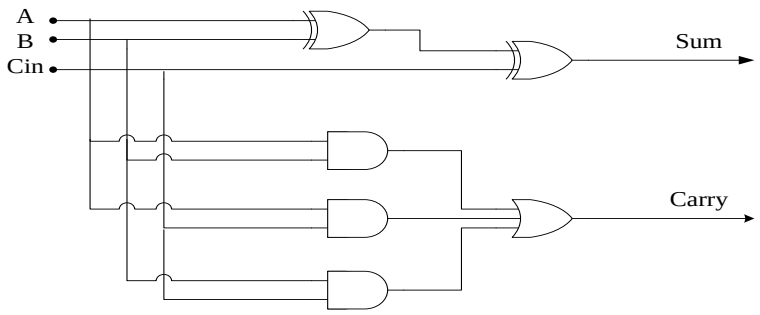


Figure 1.7: Gate level design of [3:2] compressor [7]

CHAPTER 2

CIRCUIT DESIGN

Circuit design plays an important role in the design of the multiplier. First to guarantee the multiplier to work at the desired clock rate, the designer has to know the delay of the critical path and the required time of inserting a pipeline stage. Second to reduce the area of the multiplier, several architectures of adders are investigated. Circuit analysis helps the designer to verify the functions and performances of adders. The architecture of the adder has been determined first. The size of the multiplier should be as small as possible if all the requirements can be met. The design of a 4-bit multiplier, under which the basic cell 1-bit full adder has been used. Some Logic structures have been discussed in this Chapter.

2.1 CMOS Logic Structures

There are a number of alternative CMOS logic structures. CMOS complementary logic structure, pass transistor logic etc are used in the design of the multiplier. In this report CMOS (Complementary Metal Oxide Semiconductor logic) and other high speed circuit design styles is discussed.

2.1.1 CMOS Complementary Logic

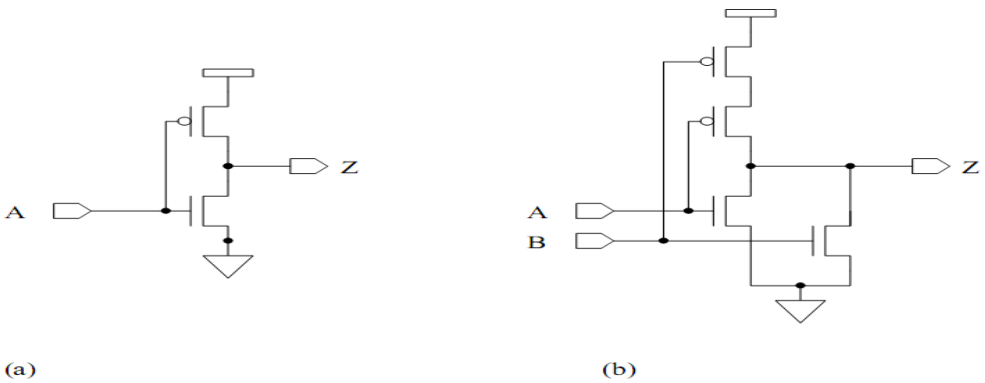


Figure 2.1: (a) CMOS Inverter, (b) CMOS NOR Gate

Figure 2.1 (a) shows a CMOS inverter. There is a pull-up PMOS transistor and a pull-down NMOS transistor. The steady state of the output will be independent of the ratio of the pull-up and pull-down transistor sizes. Because of this, in CMOS complementary logic does not have to worry about signal degradation problems. Because the power supply-to-ground path closes only during the transition, it almost consumes no static power. The CMOS complementary gate has two function determining blocks an n-block and a p-block. There are normally 2n transistors in an n-input gate.

2.2 Differential structure

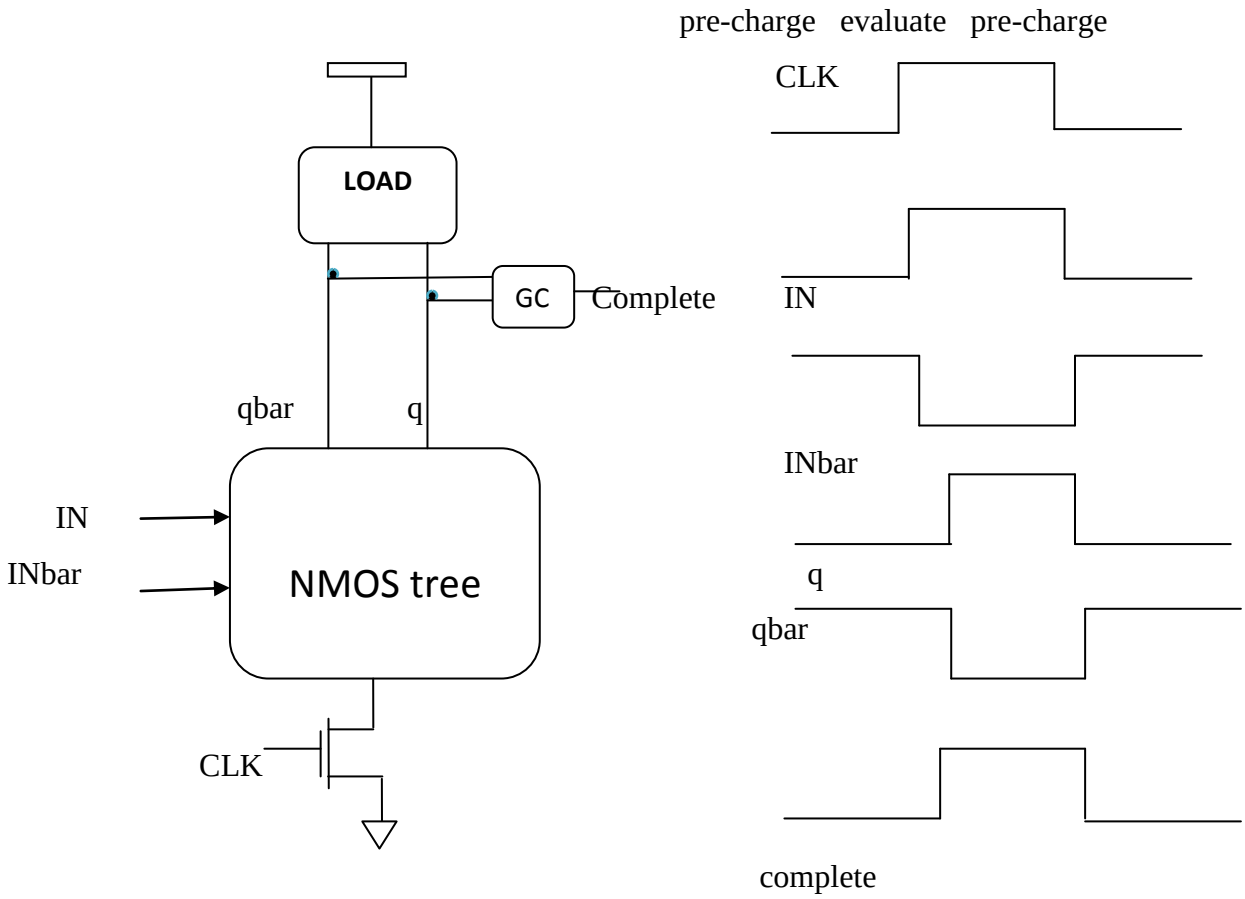


Figure2.2: General schematic of differential structure and Waveforms [10]

The general schematic of a differential-type circuit including a clock signal is shown in Fig. 2.2. The lower NMOS transistor appears only in some structures. The NMOS-tree, comprised of a double NMOS transistor tree, generates the logic function with double output q and qbar. There are some systematic methods for the optimized implementation of Boolean functions such as full adder circuit, although the use of

differential-pair transistors is quite straightforward to design simple functions**.** To perform such logic function, inputs in double-rail code are needed, i.e., INPUT (IN) and OUTPUT (q and qbar)**,** operating according to a coding scheme as shown in figure 2.2 [10].

Input data signal	Clock	Valid data output (Evaluation phase)
In(1/0)	0(during pre-charge)	1 1
In(1/0)	1(during evaluation)	1 0 OR 0 1

Table 2.1 Data input values and pre-charge and evaluation phase values

The load block acts as pull-up for CMOS logic, generating the high logic level "1." Furthermore, it may add regeneration capability to the circuit. During the pre-charge phase, the pull-up disables the logic function generated by the NMOS-tree, setting the output to the pre-charge value. This phase can be made to coincide with the low phase of clock. In the evaluation phase, the pull-up turns off and depending on the input pattern, one of the output nodes, either q OR qbar, is discharged through the corresponding branch of the NMOS-tree, generating the logic function. The other output node remains unchanged at the logic "1" level.

2.2.1 DCVS-DOMINO logic (Differential Cascade Voltage Switched Domino Logic):

DCVS-DOMINO configuration is shown in Fig. 2.3 below. Basically, it is the DCVS structure including two "weak" feedback PMOS transistors to regenerate the high '1' logic level and thus ensure that there is no deterioration of that level. DCVS-DOMINO logic enables performing more rapid circuits than that of conventional CMOS logic but at the cost of area and power consumption penalty.

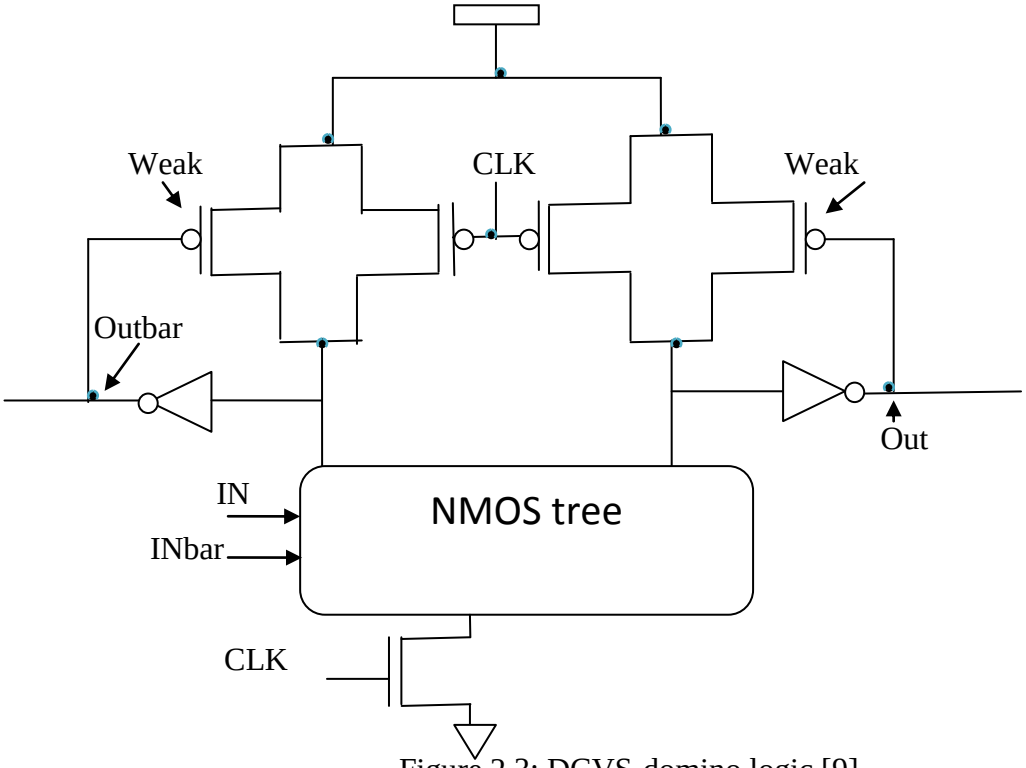


Figure 2.3: DCVS-domino logic [9]

2.2.2 ECDL (Enable/Disable CMOS Differential Logic) logic

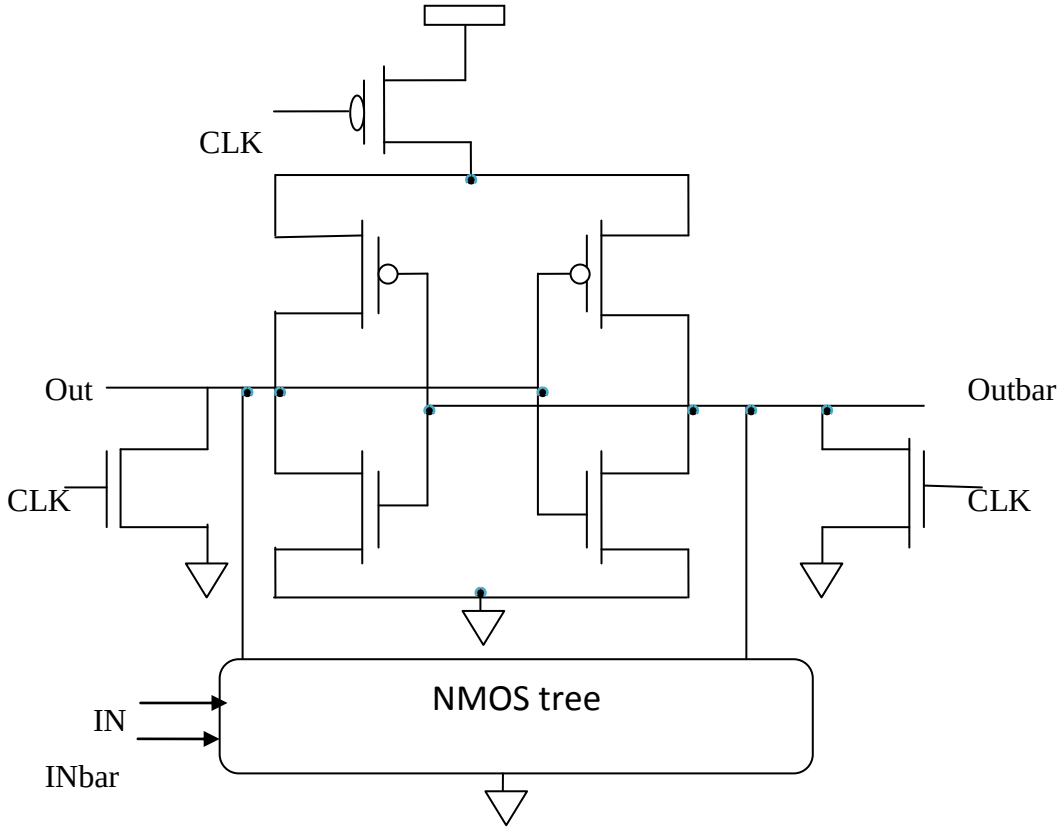


Figure 2.4: ECDL Structure [9]

ECDL Logic structure uses less number of transistor in LOAD block in comparison of DCVS-Domino structure and Static buffer is not required in case of ECDL circuit Design For the adder design number of PMOS is 14 and NMOS is 14, but in PMOS hole mobility is lower than the electron mobility thus the speed of electrons is higher than the holes. So for increasing the speed, we have to increase the width of the PMOS 3 to 4 times than that of the NMOS transistor that increases the area of the circuit. Thus the load area decreases significantly in ECDL than that of the conventional CMOS full adder design as well as the number of PMOS transistors used is decreased. The total number of PMOS transistors used is three in the carry-out and sum as compare to conventional CMOS transistor in which total number of PMOS transistors is 14. The ECDL family as shown below in Figure 2.4. Outputs are connected to Gnd during pre-charge phase by two NMOS transistors. It does not require output buffers and uses less area than DCVS-DOMINO circuits [9], [10], [11].

2.2.3 DIFFERENTIAL STRUCTURE – SODS:

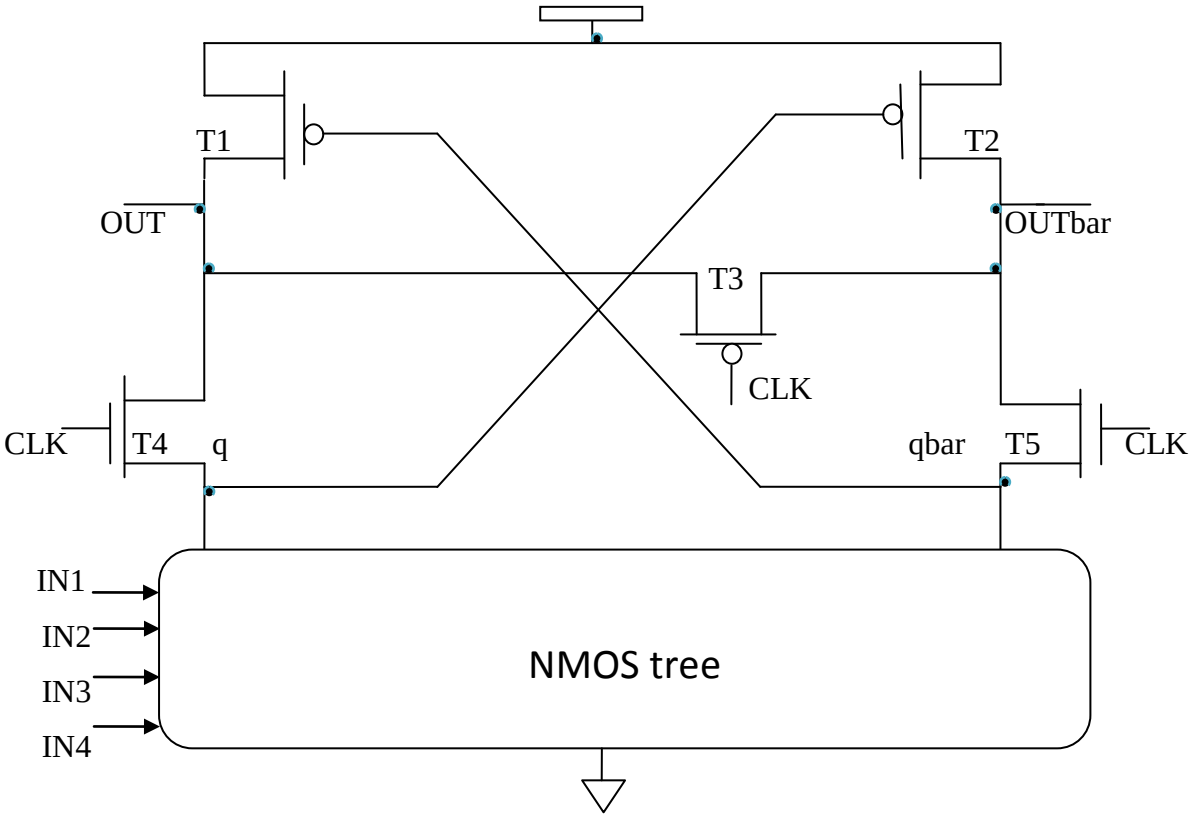


Figure 2.5: SODS Structure [10]

SODS is a high speed circuit design style. The speed of the any circuit design using SODS (Switched Output Differential Structure) is higher than the conventional CMOS circuit. The SODS circuit has minimum delay as compared to ECDL logic.

	DCVS-domino	ECDL	SODS
Transistor count	6 PMOS+3NMOS	3PMOS+4NMOS	3PMOS+2NMOS

Table 2.2: Comparison of load block based on number of transistors

The operation of the SODS circuit is as follows: [9], [10], [11], During pre-charge phase (CLK="0") T4 and T5 are OFF, while T3 is ON connecting output out and out to each other. Since the only forbidden input data is IN INbar=00, for every other input value one of the nodes q and qbar are connected to GND through the NMOS-TREE, setting that node to 0. This 0 turns TI or T2 ON, forcing the correspondent output (OUT OR OUTbar) to 1, and to turn T3 ON, both output are 11. Since T4 and T5 are OFF, there is no path between Vdd and Gnd that connect Vdd and Gnd and hence no power consumption. In the evaluation phase (CLK ="1") nodes out and out are connected to q and qbar respectively since transistors T4 and T5 turn ON, thus passing the generated function to the output. Since T3 turns OFF, data remain stable due to the action of the latch formed by TI and T2.

2.2.3.1 Characteristics of SODS:

1. Fully differential logic
2. Suitable for high speed circuits
3. Especially suited to detect “Operation Finished”

2.2.3.2 Performance improvements in SODS:

The main notice-able improvement in SODS is in terms of area, power and speed. The following improvements are as follows in compare to DCVS-DOMINO and ECDL-------

1. SODS provide high performance.
2. SODS provide low power consumption.
3. SODS reduce propagation delay.

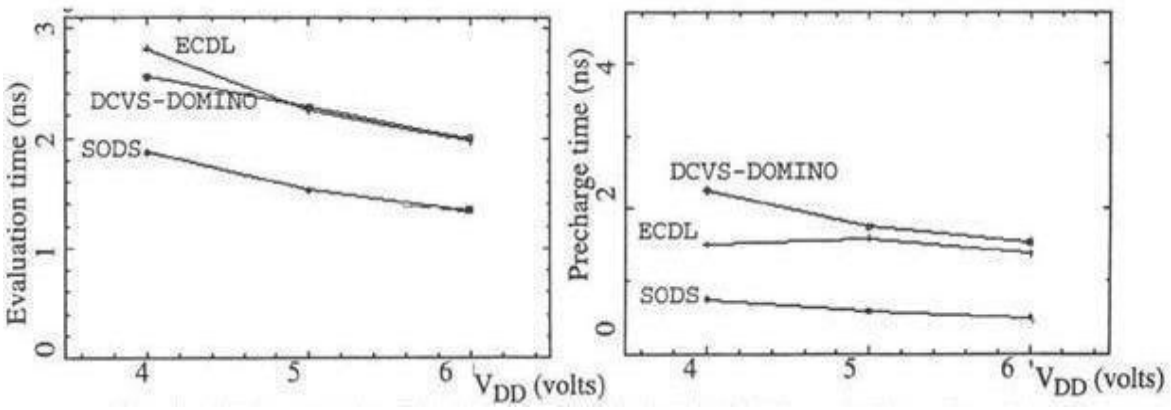


Figure 2.6 Comparison between DCVS-DOMINO, ECDL and SODS logic [9]

The above two graphs shows the SODS improvement in terms of Evaluation time and pre- charge time in first and second graph respectively.

Delay Calculation:

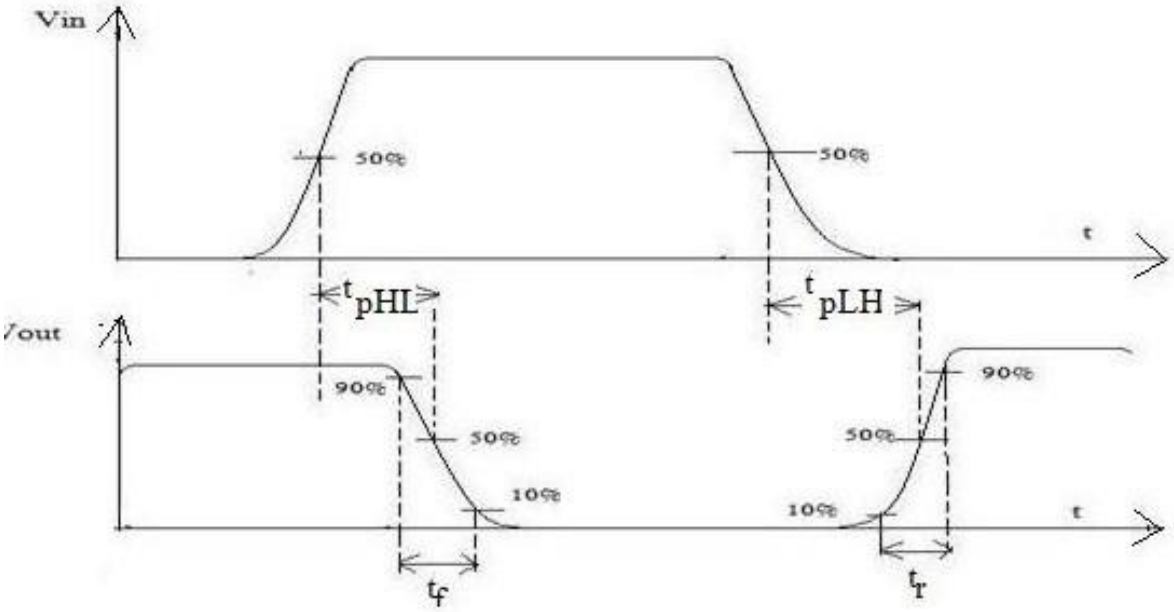


Figure 2.7: Waveform of Invertor Input / Output representing delay t_{pLH} , t_{pLH} , t_r & t_f[7]

From a System Designer perspective, the performance of a digital circuit expresses the computational load that the circuit can manage. The propagation delay tp of a gate defines how quickly it responds to a change at its input(s). It expresses the delay experienced by a signal when passing through a gate. It is measured between the 50%

transition points of the input and output waveforms, as shown in Figure 2.7 for an inverting gate. Because a gate displays different response times for rising or falling input waveforms, two definitions of the propagation delay are necessary. The t_{pLH} defines the response time of the gate for a low to high (or positive) output transition, while t_{pHL} refers to a high to low (or negative) transition. The propagation delay tp is defined as the average of the two.

$$t_p = \frac{t_{pHL} + t_{pLH}}{2}$$

The 50% definition is inspired the assumption that the switching threshold VM is typically located in the middle of the logic swing.

Power and Energy Consumption

The power consumption of a design determines how much energy is consumed per operation, and how much heat the circuit dissipates. These factors influence a great number of critical design decisions, such as the power-supply capacity, the battery lifetime, supply-line sizing, packaging and cooling requirements. Therefore, power dissipation is an important property of a design that affects feasibility, cost, and reliability. In the world of high-performance computing, power consumption limits, dictated by the chip package and the heat removal system, determine the number of circuits that can be integrated onto a single chip, and how fast they are allowed to switch. With the increasing popularity of mobile and distributed computation, energy limitations put a firm restriction on the number of computations that can be performed given a minimum time between battery recharges. Depending upon the design problem at hand, different dissipation measures have to be considered. For instance, the peak power Ppeak is important when studying supply-line sizing. When addressing cooling or battery requirements, one is predominantly interested in the average power dissipation Pavg. Both measures are defined below:

$$P_{peak} = i_{peak} x V_{supply} = \max[p(t)] \quad [7]$$

$$P_{avg} = (1/T) \int_0^t p(t)dt = (V_{supply}/T) \int_0^t i_{supply}(t)dt \quad [7]$$

Where p(t) is the instantaneous power, i_{supply} is the current being drawn from the supply Voltage V_{supply} over the interval t Î[0,T], and i is the maximum value of i_{supply} over that interval.

2.3 Adder

The adder is the basic element in computer arithmetic. It is also the critical element of the multiplier. The truth table of a one-bit full adder is shown on Table 2.2. We can derive the Boolean function in sum of products from this truth table as the following:

SUM = ABC + Abar Bbar C + ABbar Cbar + Abar B Cbar……..Equation (3.1)

CARRY = AB + BC + AC…….……...Equation(3.2)

A and B are the adder inputs, C is the carry input, SUM is the sum output, and CARRY is the carry output. The generate signal, G, occurs when a carry output (CARRY) is internally generated within the adder. When the propagate signal, P, is true, the carry-in signal (C) is passed to the carry output (CARRY) when C is true. Because the carry-in of the current bit is determined by lower bits of two operands, the delay of the adder depends on the generation of the carry.

2.3.1 CMOS Full Adder

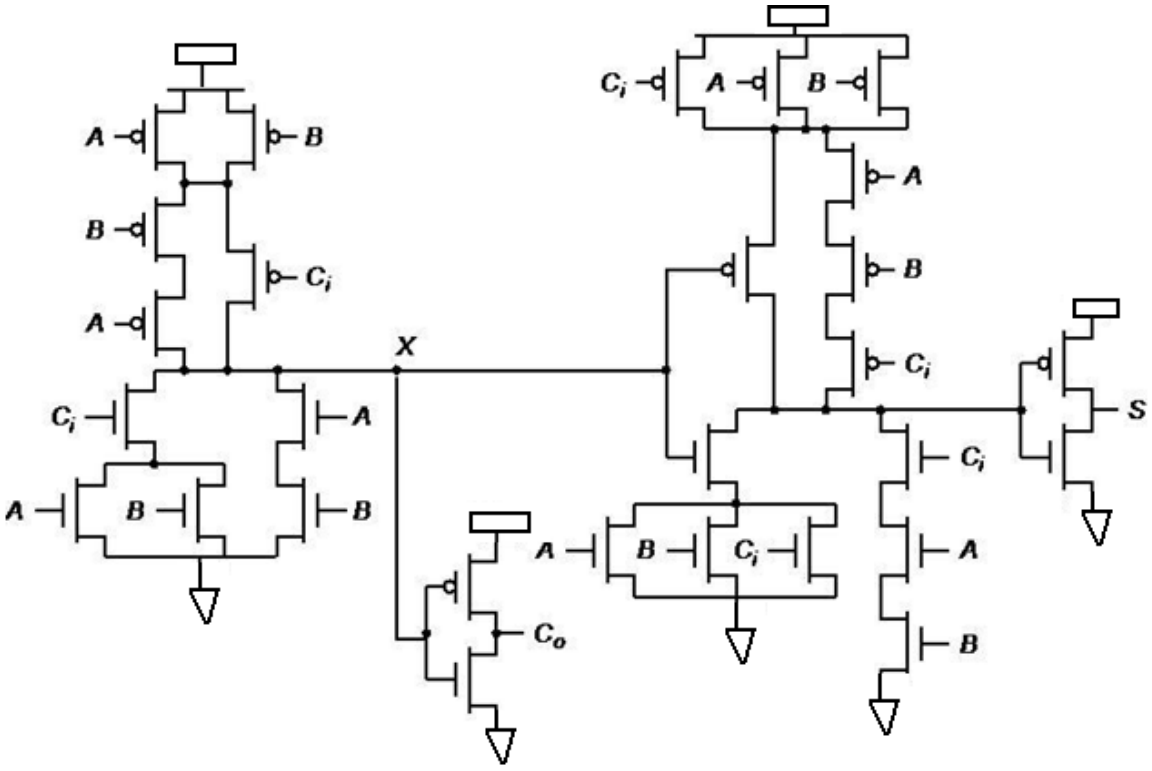


Figure 2.8: CMOS full adder[7]

The total number of transistors used in Conventional CMOS full adder Design is 14 NMOS and 14 PMOS Transistors. The adder is the basic element in computer

arithmetic. It is also the critical element of the multiplier. This implementation is shown in Figure 2.8. Equation 3.2 may be factored as follows:

$C_o = AB + C_i (A + B)$……Equation (3.3)

$C_o = AC_i + BC_i + AB$……Equation (3.4)

From Equation 3.1 and Equation 3.4, SUM can be expressed as follows:

$S = ABC_i + (A+B+C_i)\, C_o bar$……Equation (3.5)

C	A	B	$A \times B(G)$	A+B(P)	$A \oplus B$	SUM	$CARRY$
0	0	0	0	0	0	0	0
0	0	1	0	1	1	1	0
0	1	0	0	1	1	1	0
0	1	1	1	1	0	0	1
1	0	0	0	0	1	1	1
1	0	1	0	1	1	0	1
1	1	0	0	1	1	0	1
1	1	1	1	1	0	1	1

Table 2.3: Truth Table of full adder

2.3.1.1 Simulation Results:

Sr NO.	A_i Input logic	B_i Input logic	C_{i-1} (Input carry logic)	S_i (Sum(delay in output(ps)))	C_i (Carryout(delay in output(ps)))
1.	1	1	Vpulse	219.00	No change
2.	0	1	Vpulse	247.00	183.50
3.	1	0	Vpulse	275.00	200.60
4.	Vpulse	1	0	290.50	212.00
5.	1	Vpulse	1	210.50	No change
6.	1	Vpulse	0	295.00	213.24
7.	Vpulse	1	1	No Change	207.50

Table2.4 Delay in generation of sum and carry for different input combinations

Where Vpulse logic 0 for 0 ns to 1 ns and 1 for 1 ns to 2 ns and 0 ns for 2 ns to 3 ns. Logic 0 is 0 volt and Logic 1 is 1.8volt.

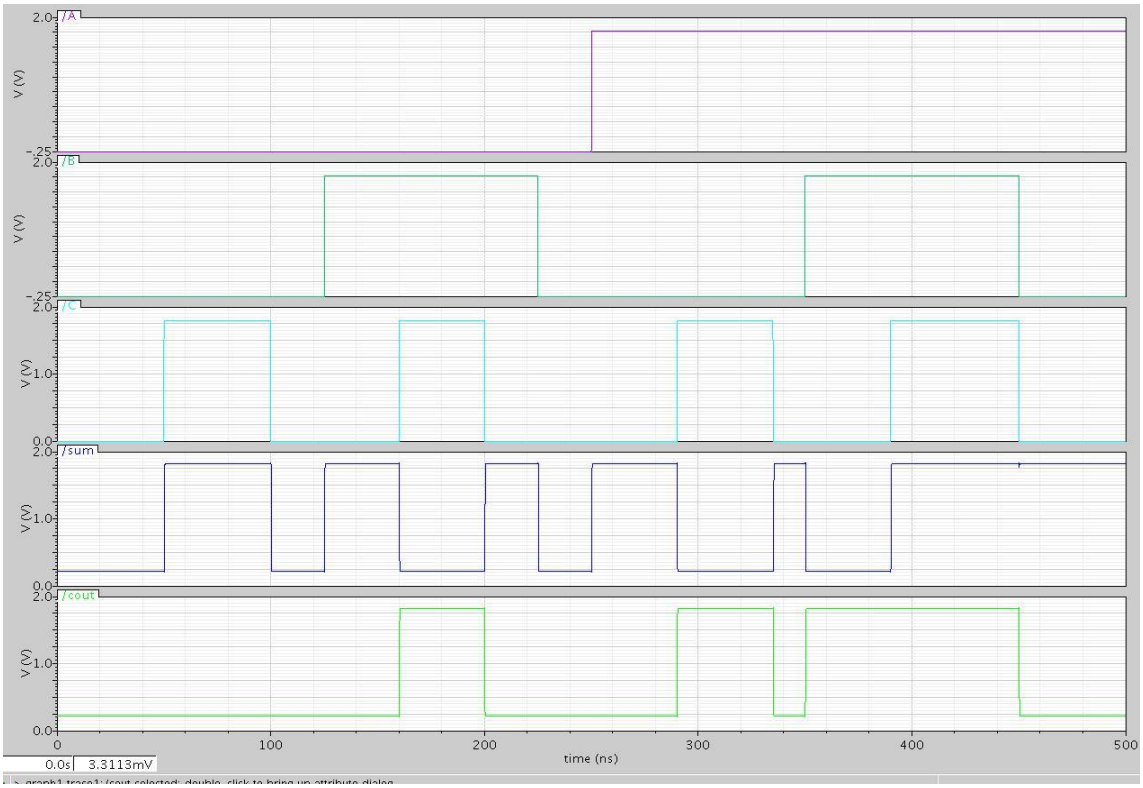


Figure 2.9(a) Waveforms of CMOS full adder output for different input combinations

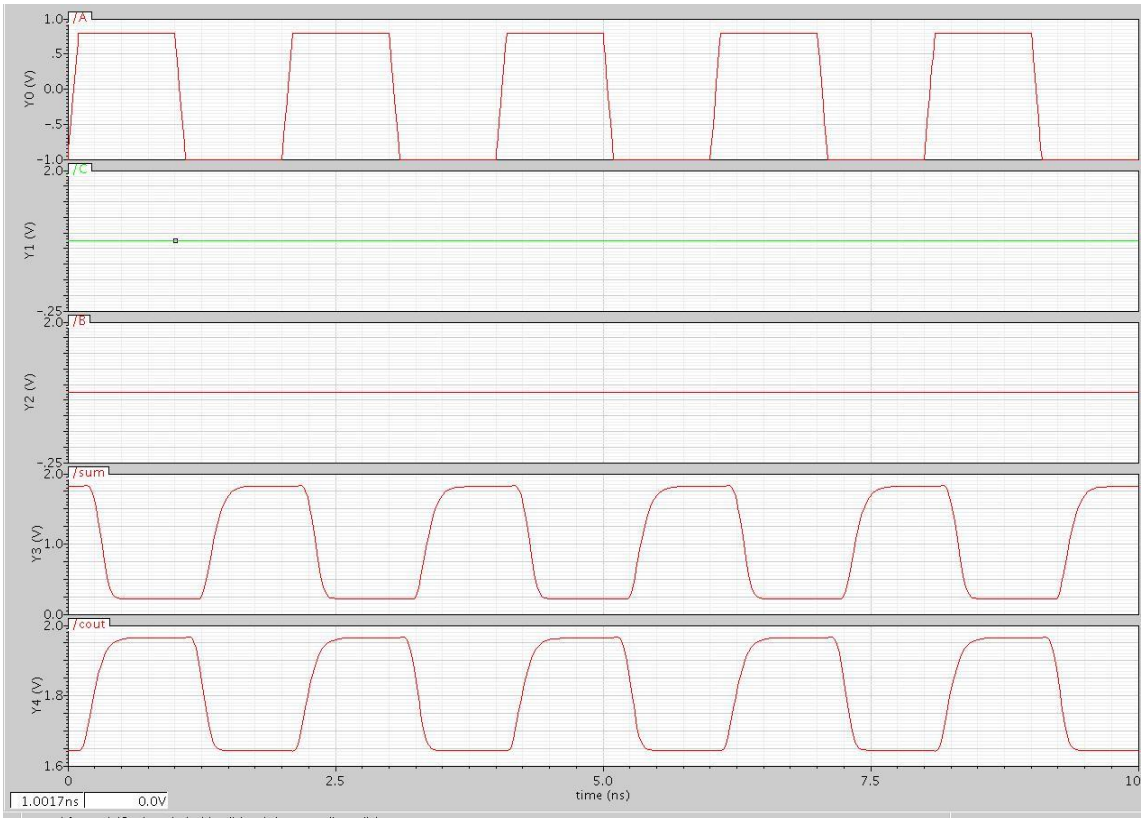


Figure 2.9(b) Waveforms of CMOS full adder output for one of the input combination

2.3.2 DIFFERENTIAL STRUCTURE - Basic SODS Adder

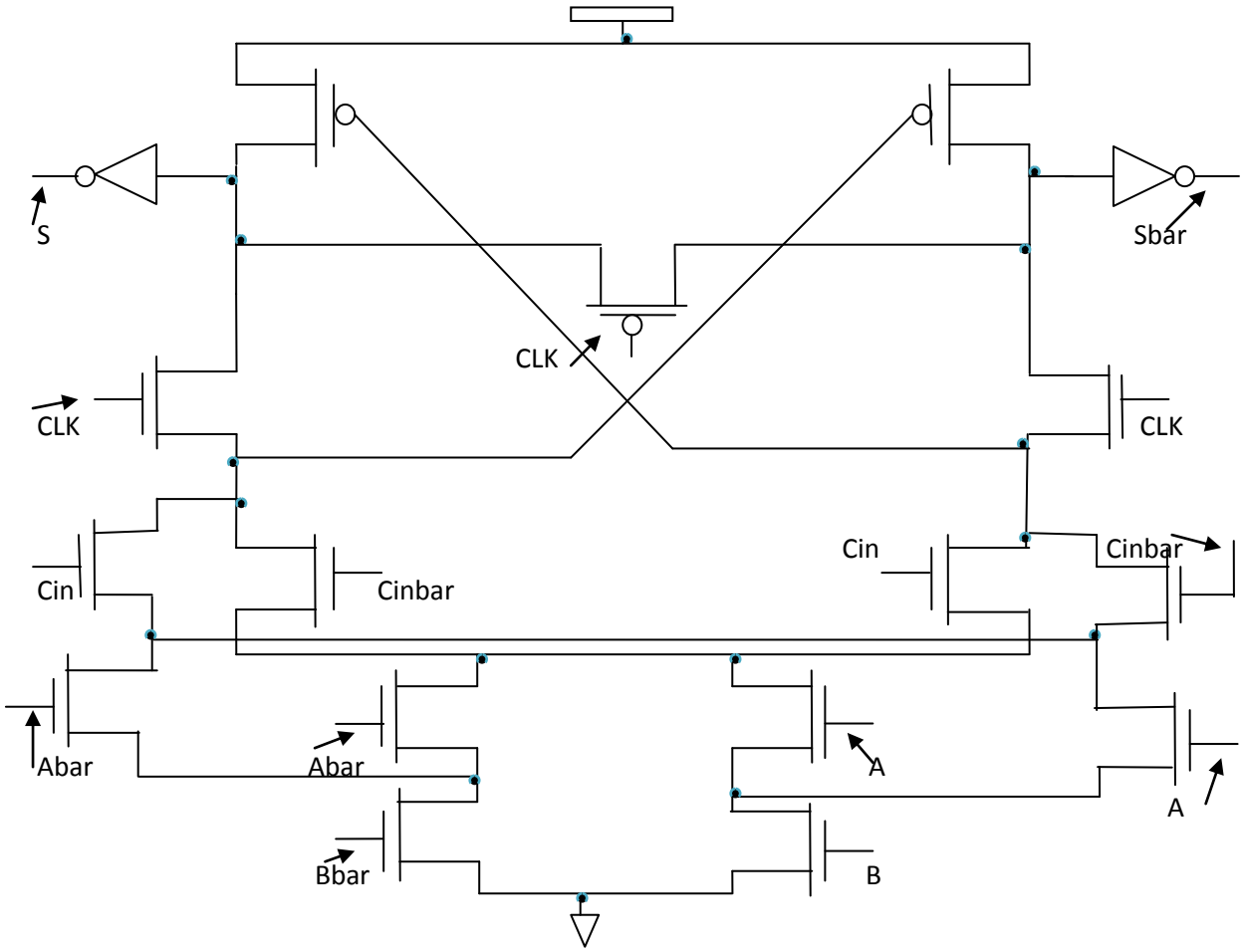


Figure 2.10: Sum using basic SODS full adder circuit design

Full Adder design using SODS is a high speed circuit. The speed of the full adder using Switched Output Differential logic is higher than the conventional CMOS full adder design.

The total number of PMOS transistors used in much less then that used in Conventional CMOS transistor. The SODS circuit has reduced delay as compared ECDL logic as we discussed above in differential logic styles. Adder Circuit is implemented using this technique in figure 2.9. As we can compare the full adder design, the number of stages is reduces in full adder design using High Speed Circuit SODS than the Conventional CMOS full adder Design and so the speed of the circuit is increased. The SODS circuit suffers from the problem of minimum logical depth that increases the delay for Logic "0' because in SODS circuit design the Logic "0" is

obtained through NMOS tree ,it is the only one path for connecting ground in the circuit so modification can be done to reduce the delay in the circuit.

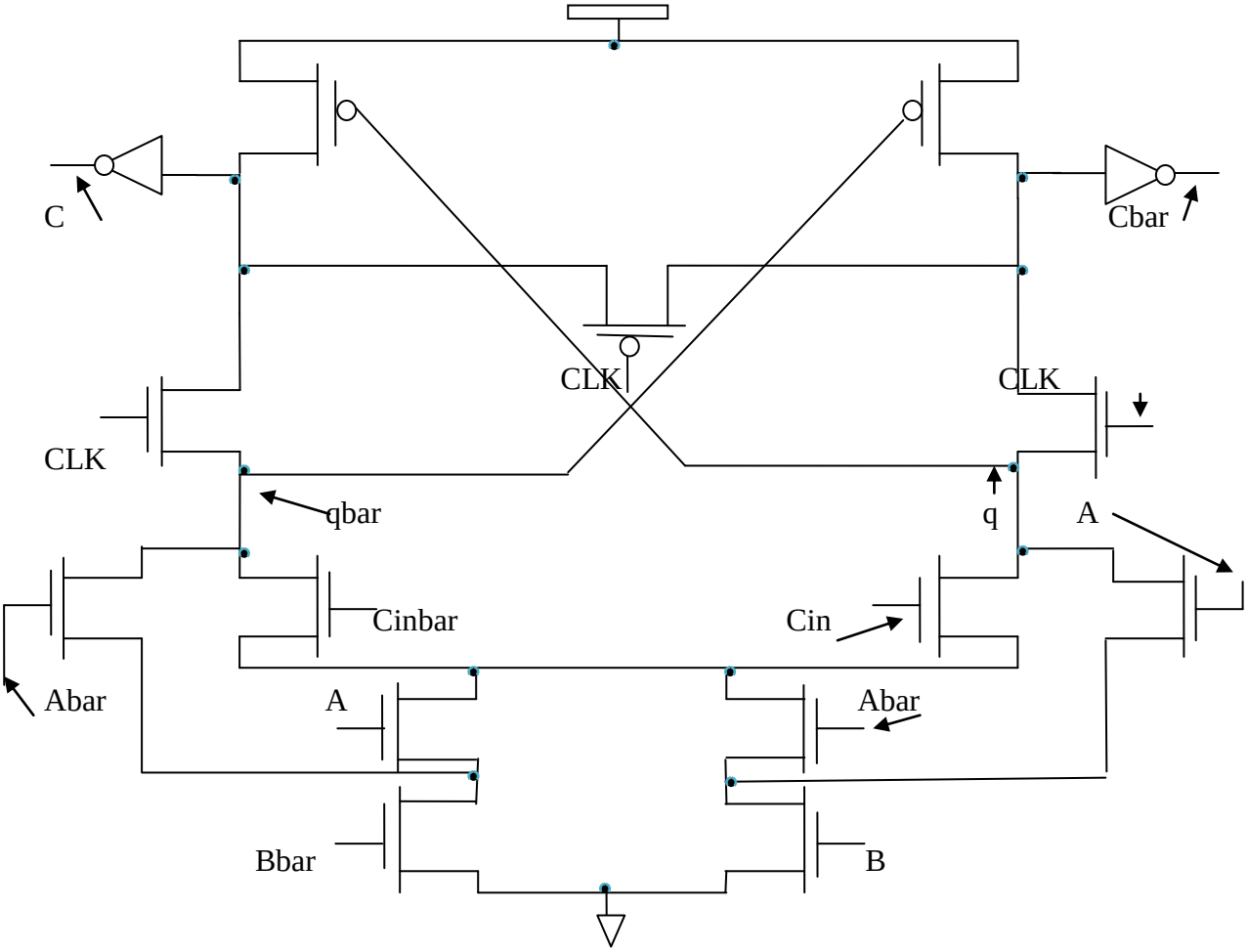


Figure 2.11: Carryout generation using basic SODS full adder circuit design

S = (Abar Bbar + AB) Cinbar + (Abar B + A Bbar) Cin

C = A B + (Abar B + A Bbar) Cin

Sbar = (Abar Bbar + A B) Cinbar+ (Abar B + A Bbar) Cin

Cbar = Abar Bbar + (Abar B + A Bbar) Cinbar [8]

Where A , B , Cin are the inputs and Abar , Bbar are the inverse of the A , B , Cin is the input carry and Cinbar is the inverse of Cin.

2.3.3 MODIFIED STRUCTURE OF SODS:

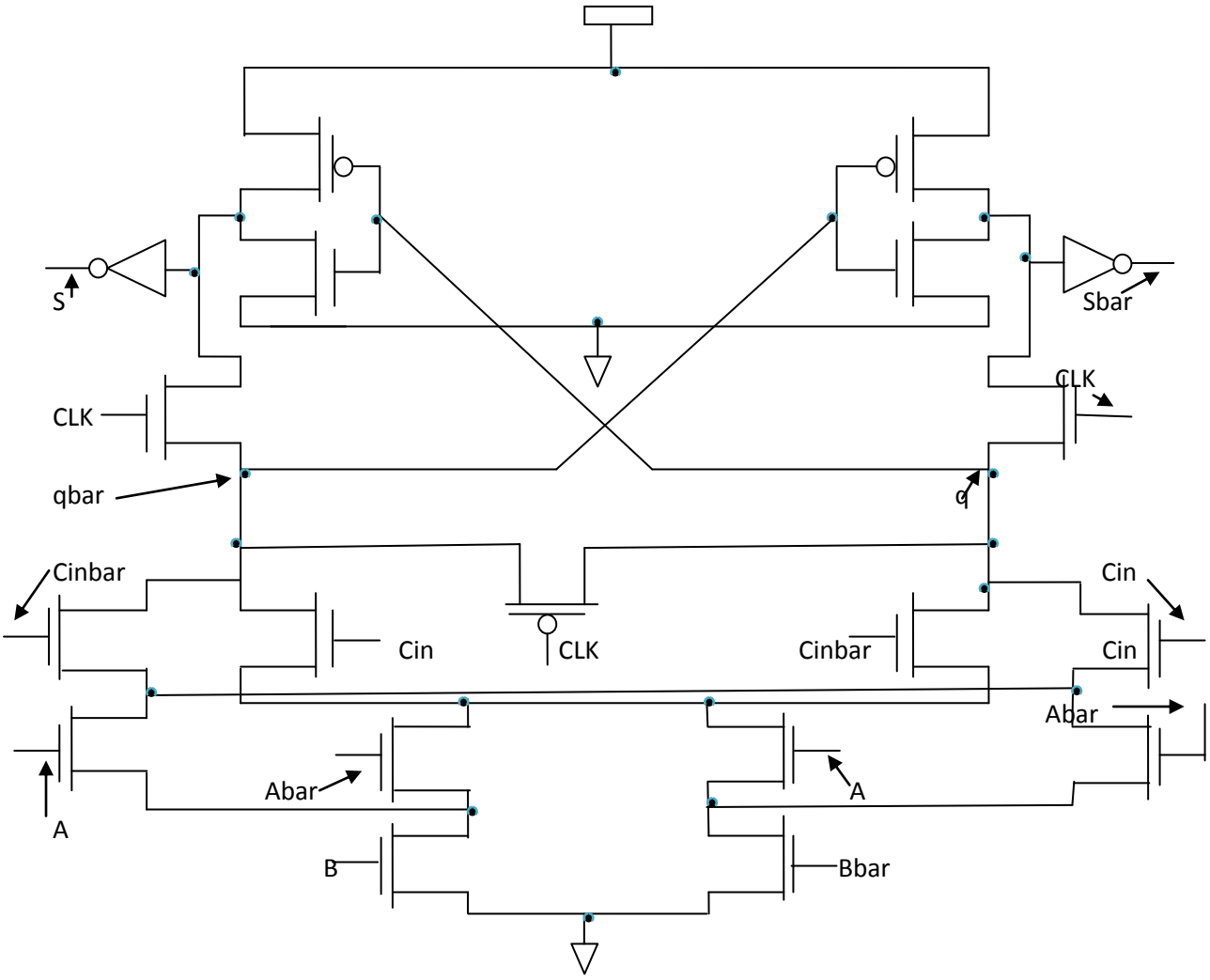


Figure 2.12: Sum generation using modified SODS full adder circuit design

In the previous discussed structure basic SODS, the circuit achieve the logic '1' level through the load Block and logic '0' level is achieved through the NMOS tree so the basic structure of SODS is suffered from the problem of minimum logical depth and the fall delay is higher for the circuit because the logic '0' is achieved only through the load block as the depth increased, more path have to follow in the basic SODS thus the circuit is investigated to do modification in the load block. In the modified structure of SODS, the NMOS tree is used only to generate the difference between the high and low logic level and the rest full logic is developed using cross coupled invertor pair in the load block. The PMOS used that connects both the nodes q and qbar at same potential in pre-charge state that cause the output and its complement at the same level.

This PMOS transistor is ON only when the clock signal is not present in the circuit. The location of the PMOS transistor is also changed in the modified structure because changing the placement of the PMOS reduces the capacitor caused by the PMOS at the output of the circuit. The circuit is designed for FO4 load implies that the circuit is capable in driving the load equal to four unit invertors at the output.

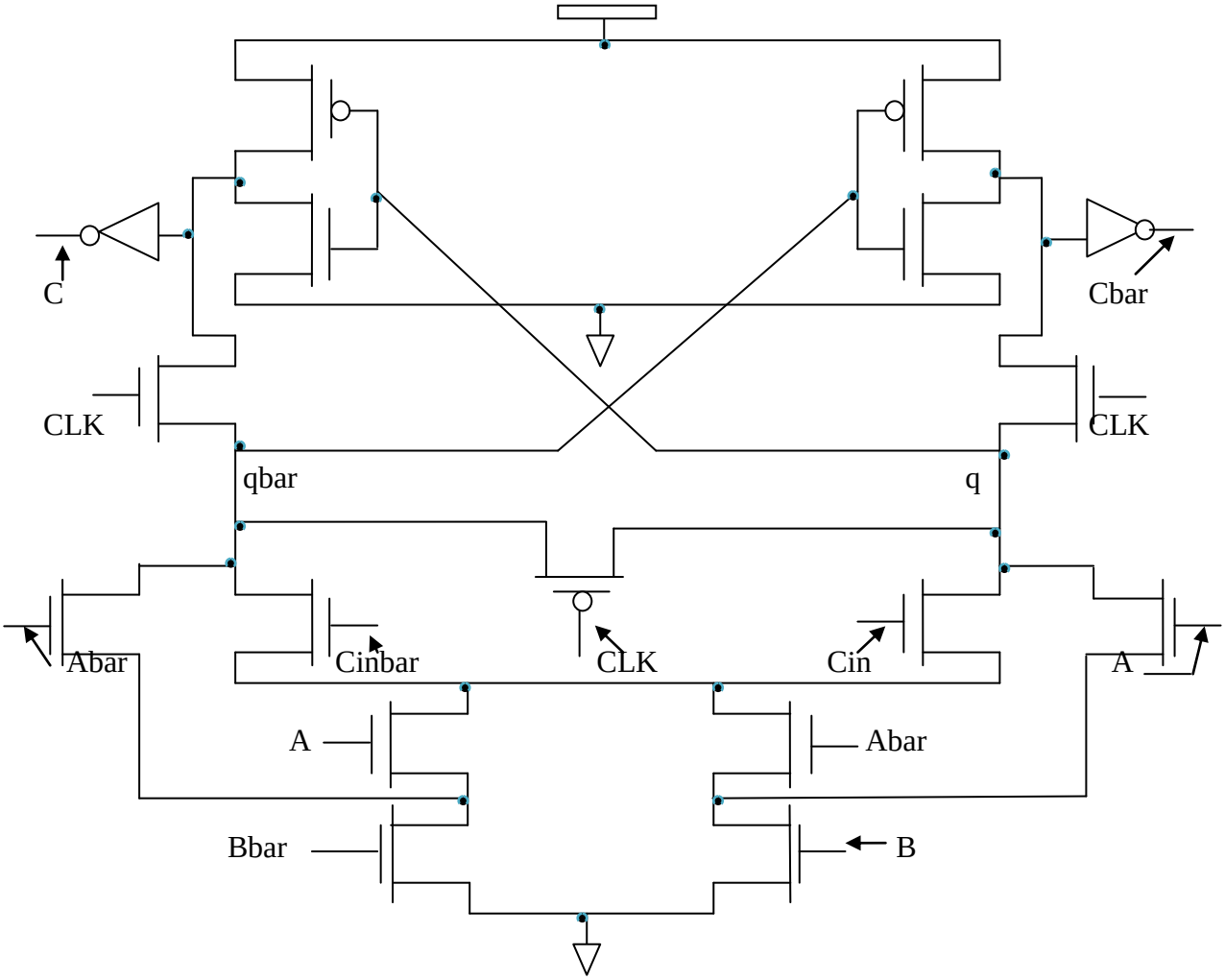


Figure 2.13: Carryout generation using modified SODS full adder circuit design

2.3.3.1 Simulation Results:

SODS FULL ADDER

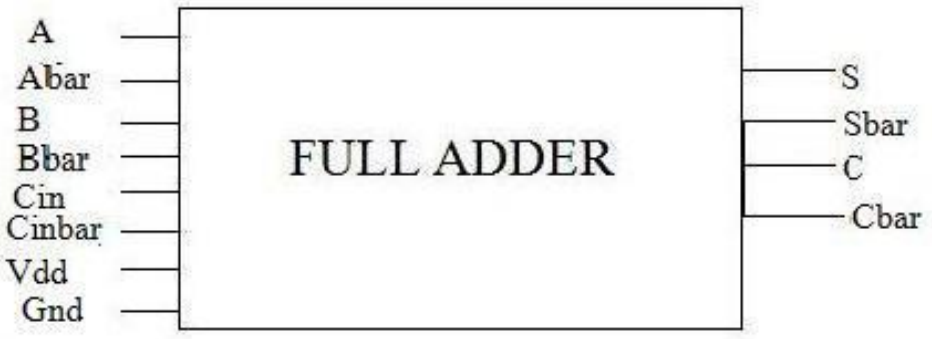


Figure 2.14: Symbol of full adder

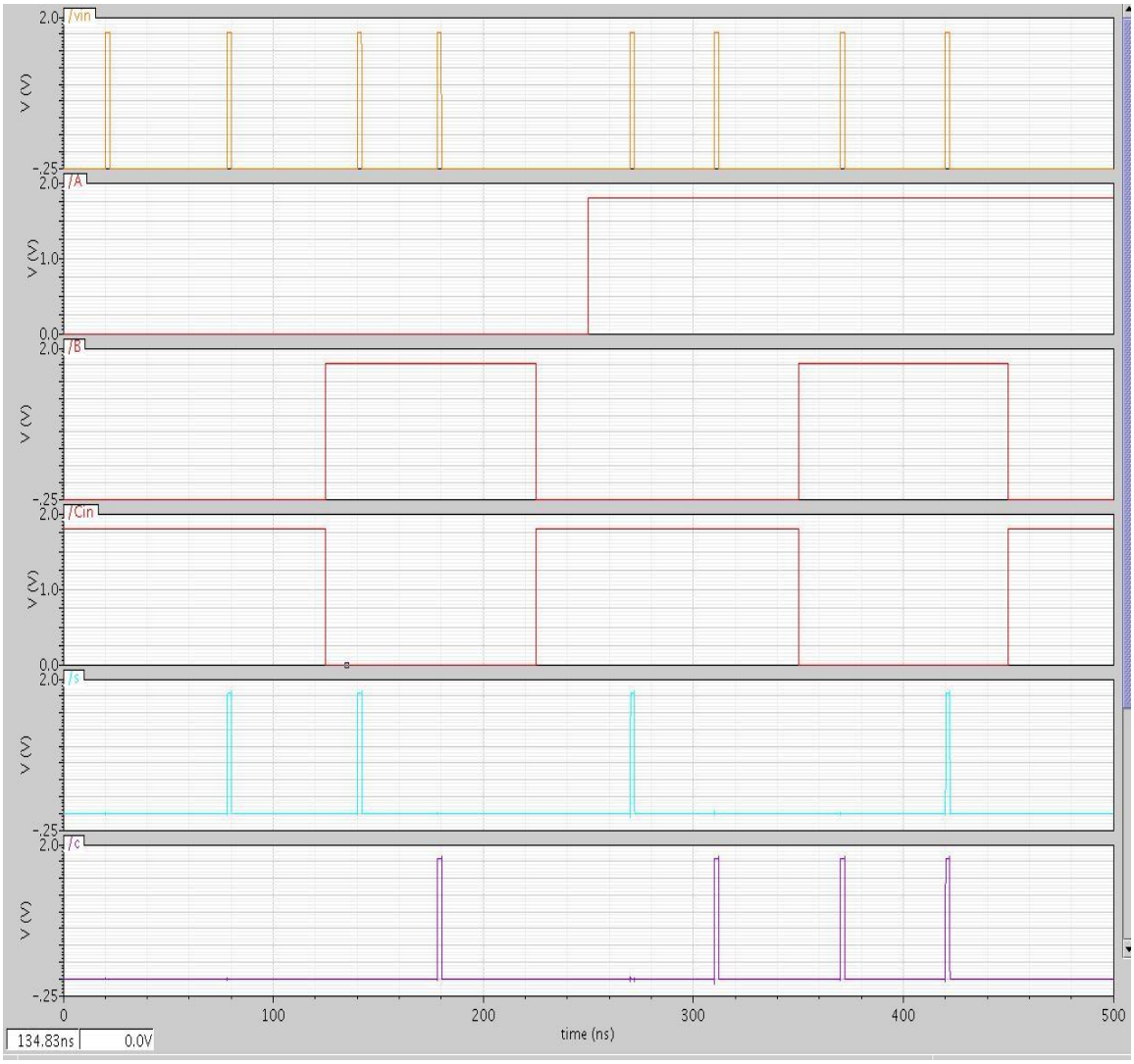


Figure 2.15: Waveforms for all input combinations of modified SODS full adder for carryout and sum

Inputs (ABCin) => outputs(S Cout)	SUM Delay (ps)		SUM Power (nW/MHz)		Cout Delay (ps)		Cout Power (nW/MHz)	
	Basic SODS Adder	Modified SODS Adder	Basic SODS Adder	Modified SODS Adder	Basic SODS Adder	Modified SODS Adder	Basic SODS Adder	Modified SODS Adder
000=>00	136.50	135.00	594.00	642.00	145.00	115.00	62.52	627.28
001=>10	191.25	162.50	130.00	263.40	147.70	129.50	637.40	654.00
010=>10	189.00	155.74	130.00	249.00	153.00	115.00	630.52	630.00
011=>01	137.50	131.50	614.50	651.18	195.84	147.00	116.00	211.70
100=>10	189.35	155.50	130.00	272.00	157.50	121.50	566.00	640.00
101=>01	135.00	131.00	614.50	650.00	191.00	116.50	106.40	180.40
110=>01	136.00	135.00	596.00	642.00	188.30	157.50	125.16	255.76
111=>11	191.50	162.50	130.00	262.00	186.00	139.50	250.32	180.60

Table 2.5: Resulting showing comparison for delay and power of basic and modified SODS full adder.

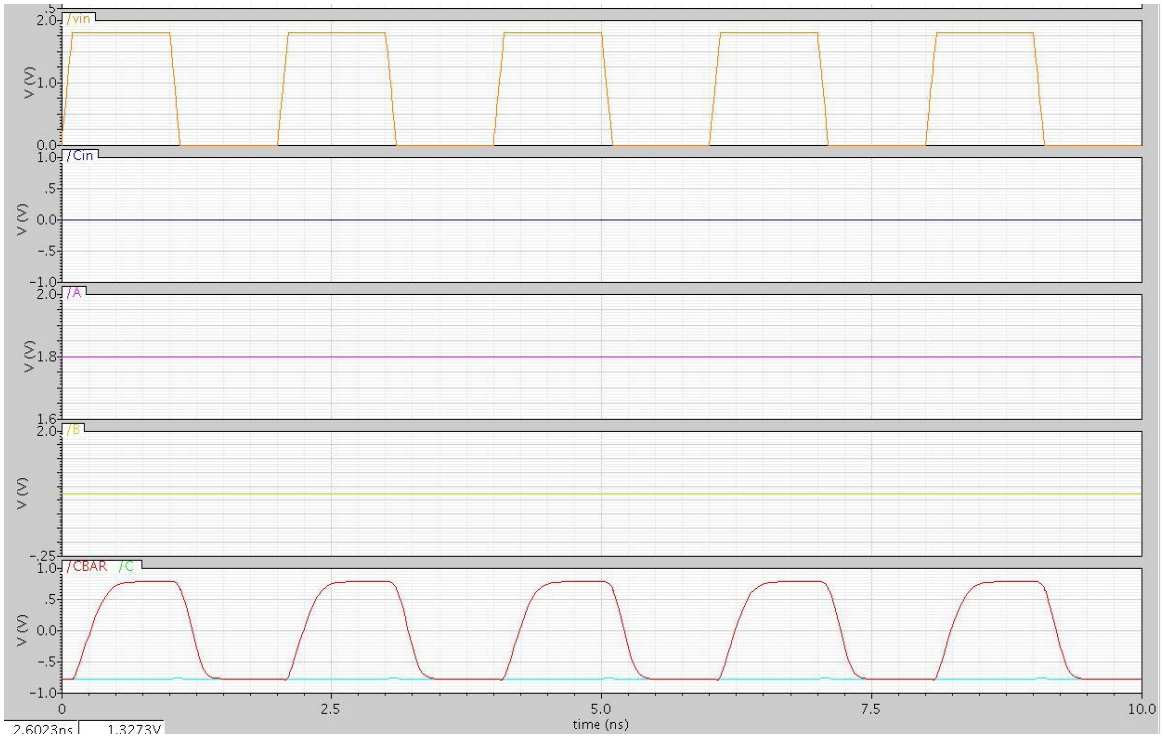


Figure 2.16: Waveforms for one of the input combination for basic SODS carry circuit

2.3.3.2 Layout:

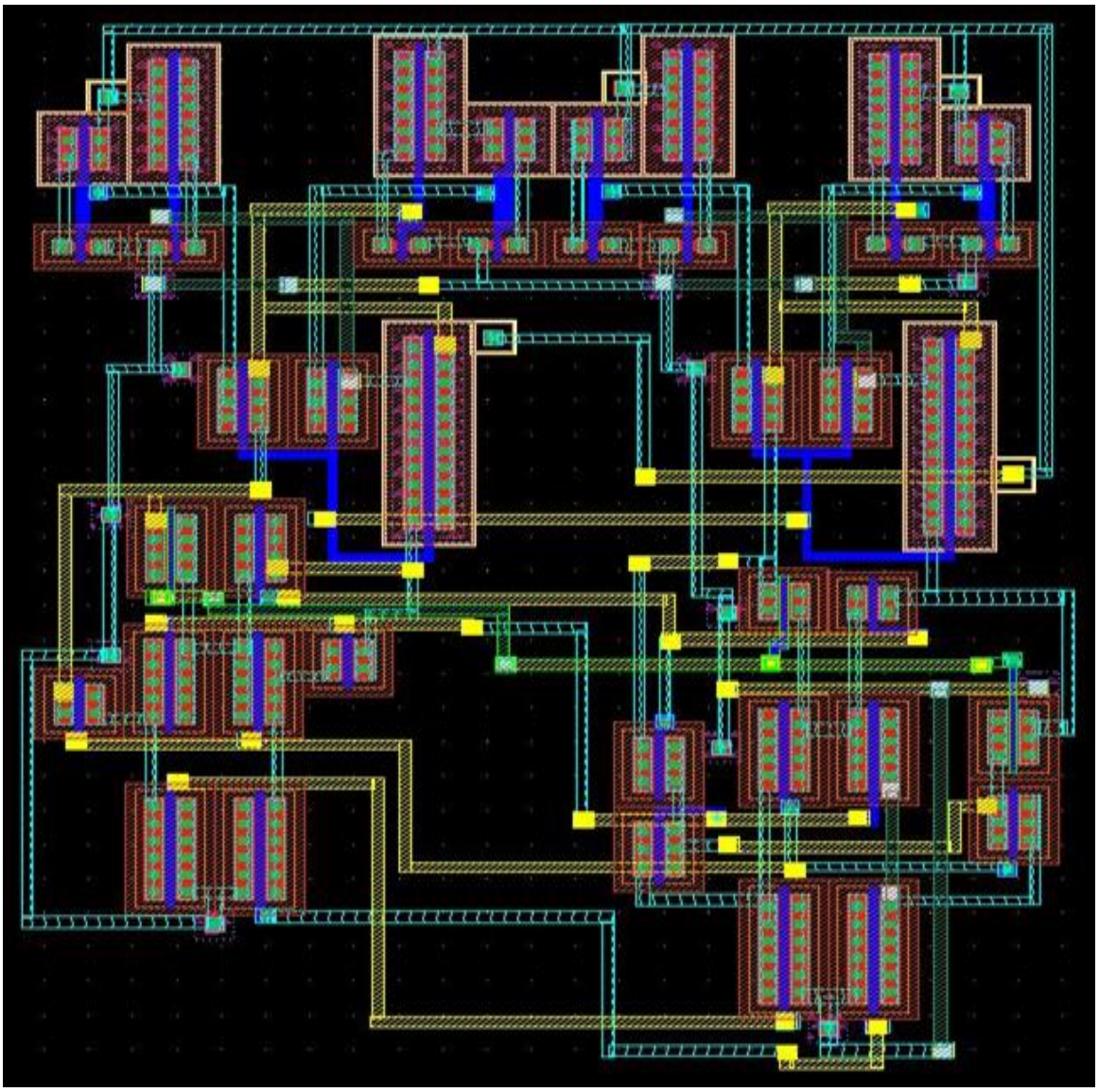


Figure 2.17: Layout of the modified SODS

2.3.3.3 LVS report:

Figure 2.18: LVS result

2.3.3.4 RCX :

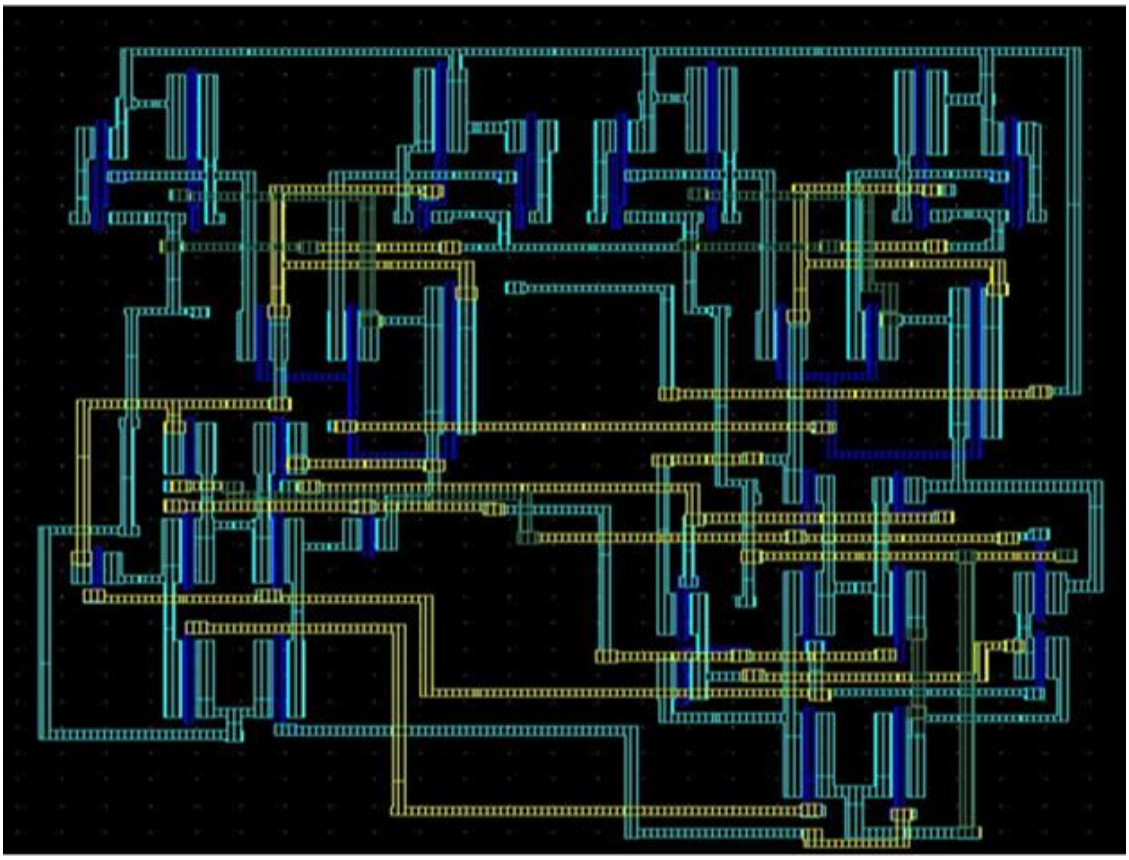


Figure 2.19: RCX view

```
                                Show Environment
*****************************************************************************************************************
Library         : layout
Cell Name       : shakun
View Name       : av_extracted
File Name       : /home/shakun/Cadence_WorkDir/layout/shakun/av_extracted/layout.cdb
View Type       : maskLayout
Edit Mode       : Edit
Display Levels  : 0 - 0
Entry Layer     : DIFF drawing
*****************************************************************************************************************

                                Layer Object Statistics
*****************************************************************************************************************
Layer      Purpose    Arc Bend Donut Dot Ellipse Label Line Path Polygon Rect Trl Taper TextDisplay Total
*****************************************************************************************************************
POl        net          0    0     0   0       0     0    0    0      35   40   0     0     0         75
ME1        net          0    0     0   0       0     0    0    0     142   33   0     0     0        175
ME2        net          0    0     0   0       0     0    0    0      52    7   0     0     0         59
ME3        net          0    0     0   0       0     0    0    0      14    1   0     0     0         15
*****************************************************************************************************************
Total                   0    0     0   0       0     0    0    0     243   81   0     0     0        324

                                Instance Statistics
*****************************************************************************************************************
Master Name  View Name  Library Name  Count
*****************************************************************************************************************
N_18_MM      symbol     UMC_18_CMOS      30
P_18_MM      symbol     UMC_18_CMOS      10
presistor    symbol     UMC_18_CMOS     306
pcapacitor   symbol     UMC_18_CMOS     229

Total number of instances placed : 575
Total number of mosaics placed   : 0

                                Contact Statistics
*****************************************************************************************************************
Master Name  View Name  Library Name  Count
*****************************************************************************************************************

Total number of contacts placed  : 0

*****************************************************************************************************************
```

Figure 2.20: File showing results of RCX

CHAPTER 3

MULTIPLIER ARCHITECTURES

There are enormous architectures available for multipliers. Mainly multiplier architectures can be classified into two main blocks, these are:

3.1 Serial Multipliers

3.2 Parallel Multipliers

3.1 Serial Multipliers

The advantage of this design over the parallel circuit is the reducing of required hardware and input, output routing when the high clock rate is not important factor in application. This Multiplier is used when area is of main concern in the circuit design.

3.2 Parallel Multipliers

Parallel multipliers are fast as compared to serial multipliers. These can be further classified into enormous architectures, but in this report I am discussing only following architectures:

3.2.1 Array Multiplier Architecture

3.2.2 Baugh Wooley Multiplier Architecture

3.2.3 Braun Multiplier Architecture

3.2.1 Array Multiplier

3.2.1.1 Simple Array Multiplier

In this type of array, the output of each row of counters (3:2 compressors) is the input to the next row of counters. In the simple array, each row of [3:2] compressors adds a partial product to the partial sum, generating a new partial sum and a sequence of carries. The delay of the array depends on the depth of the array. Therefore, the summing time for the simple array is N-2 [3:2] Compressor delays, where N is the number of partial products. The drawback of this type of array is the hardware is

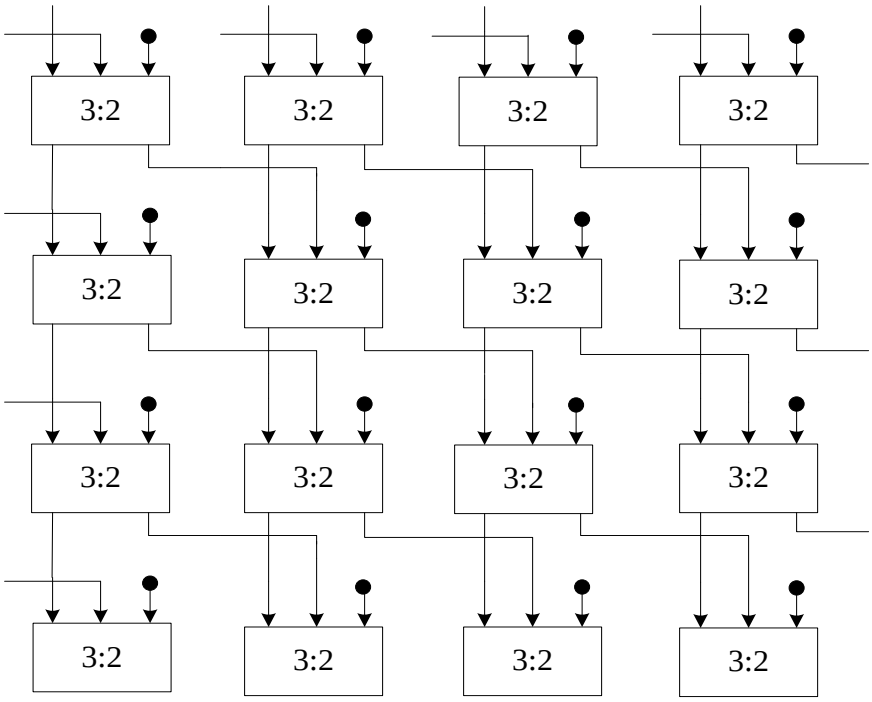


Figure 3.1: Simple array mechanism

underutilized. The counters are used only once in the calculation of the result, for the remaining time, they are idle. Array Multiplier is an efficient layout of a combination multiplier. It accepts all bits simultaneously [9].

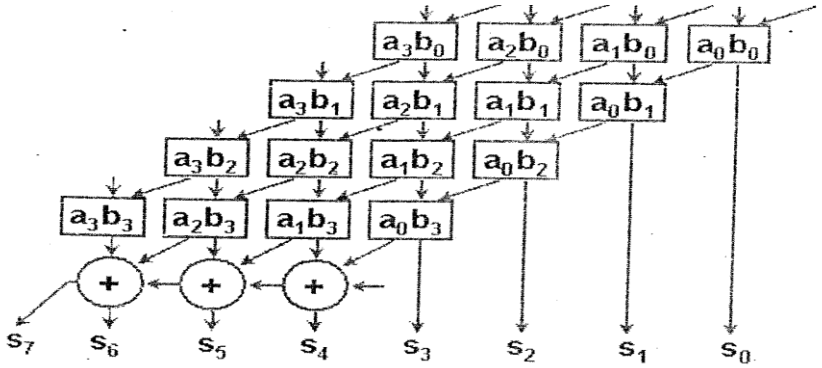


Figure 3.2: Architecture of array multiplier [1]

This drawback can be diminished by pipelining the array so that several multiplications can occur simultaneously. Pipelining would increase the throughput of the multiplier, but would also increase the latency and area of the multiplier. A fully pipelined array is normally avoided, since the array would be faster than the clock of processor. Figure 3.2 depicts the layout of a simple array topology. The dots represent the partial

products. The longest product calculation delay in it depends on the speed of the adders. An n-bit multiplier requires n(n-1) full adders and n^2 AND gates. It is possible to decompose Array multipliers in two parts. The first part is dedicated to the generation of partial products, and the second one collects and adds them.

Multiplier circuit is based on add and shift algorithm. Each partial product is generated by the multiplication of the multiplicand with one multiplier bit. The partial product are shifted according to their bit orders and then added. Addition is mainly done by carry save algorithm in which every carry and sum signal is passed to the adders of the next stage. Final product is obtained in a final adder by any fast adder. In array multiplication we need to add, as many partial products as there are multiplier bits. Now as both multiplicand and multiplier may be positive or negative 2's complement number system is used to represent them and care for sign bit extension must be taken. When 2's complement partial products are added in carry save arithmetic all numbers to be added in one adder stage have to be of equal bit length. Therefore, the sign bits of the partial product(s) in the first row and the sum and carry signals of each adder row are extended up to the most significant sign bit of the number with the largest absolute value to be added in this stage. The sign bit extension results in a higher capacitive load of the sign bit signals compared to the load of other signals and accordingly slows down the speed of the circuit. Here we'll discuss signed multiplication.

3.2.1.2 Reduction of the partial products

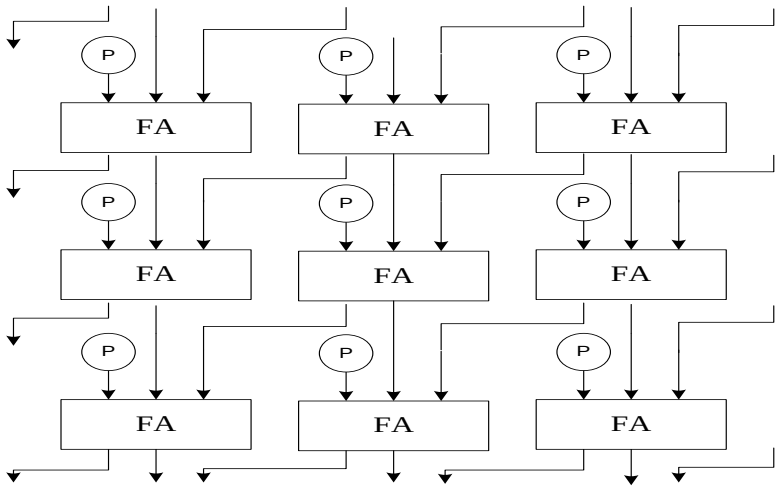


Figure 3.3: Reduction of partial products using array method

Conventional linear array multipliers consist of rows of carry-save adders (CSA). A portion of an array multiplier with the associated routing can be seen in figure 3.3.

3.2.1.3 Advantages of 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. A fully pipelined array of the multiplier with a stage delay equal to the delay of a 1-bit full adder plus a register has been successfully designed for high-speed DSP applications. Also it can be easily pipelined by inserting latches after CSA (carry save adders) or after every few rows.

3.2.1.4 Limitations of Array Multiplier

The biggest problem with full linear array multipliers is that they are very large. As operand sizes increase, linear arrays grow in size at a rate equal to the square of the operand size. This is because the number of rows in the array is equal to the length of the multiplier, with the width of each row equal to the width of multiplicand. The large size of full arrays typically prohibits their use, except for small operand sizes, or on special purpose math chips where a major portion of the silicon area can be assigned to the multiplier array.

Another problem with array multipliers is that the hardware is underutilized. As the sum is propagated down through the array, each row of CSA's computes a result only once, when the active computation front passes that row. Thus, the hardware is doing useful work only a very small percentage of the time. This low hardware utilization in conventional linear array multipliers makes performance gains possible through increased efficiency.

2-bit Multiplier Design using High Speed Circuit SODS:

In the circuit below, ha stands for half adder and ha in the block is designed using SODS high speed circuit to increase the speed of operation. The partial Product a_0b_1, a_1b_0, a_1b_1

and a_0b_0 are generated using AND gate and how the Multiplier operation is obtained is as shown in figure 3.4.

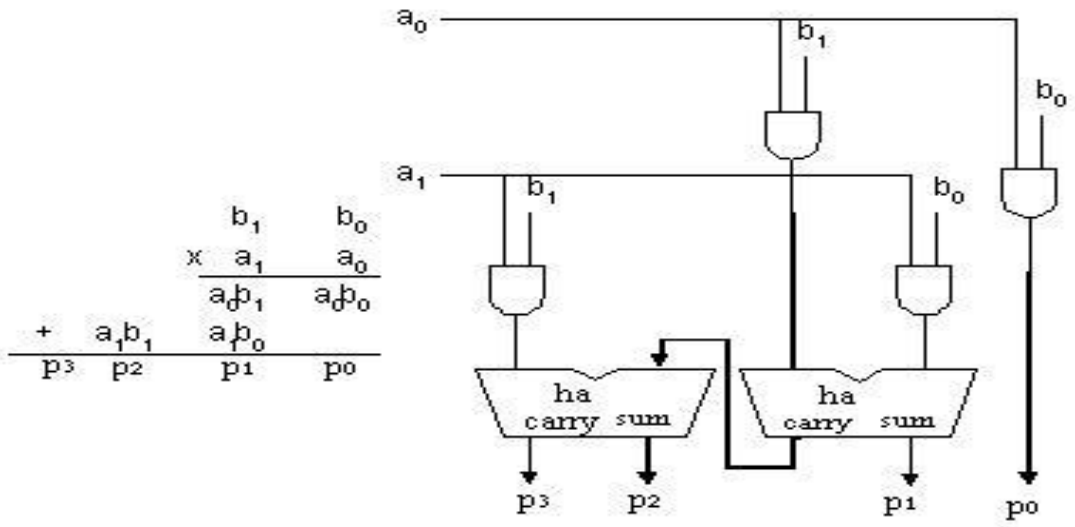


Figure 3.4: 2-bit multiplier design

Multiplier input bits (A0B0*A1B1)	**Output bits**	**Delay of 2-bit Multiplier using SODS adder circuit as a unit cell (ps)**		**Delay of 2-bit Multiplier using SODS adder circuit as a unit cell (uW/MHz)**	
		Basic	**Modified**	**Basic**	**Modified**
11*11	1001	390.00	360.00	1.71	1.94
11*10	0110	400.00	350.00	1.365	1.847

Table 3.1 Comparison of delay and power of 2-bit multiplier for basic and modified SODS

3.2.2 Baugh-Wooley Multiplier:

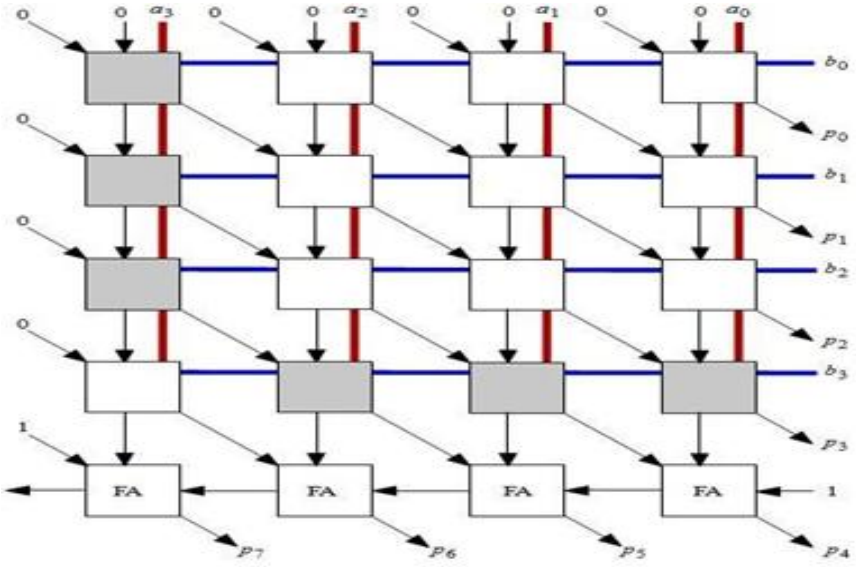


Figure 3.5: Baugh-Wooley multiplier [4]

The Baugh-Wooley technique was developed to design direct multipliers for 2's complement Numbers[4].

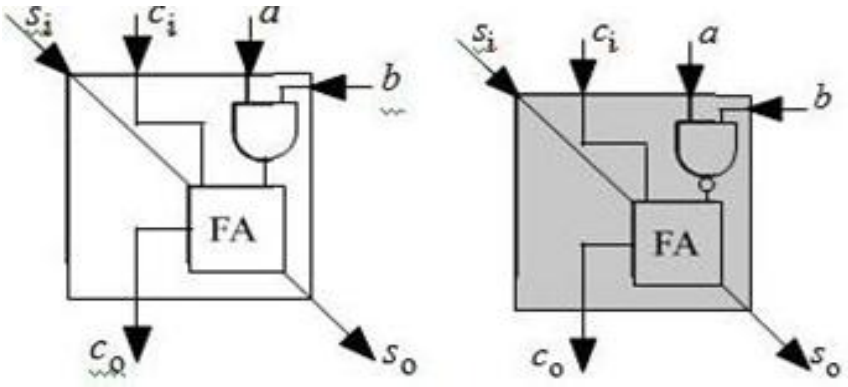


Figure 3.6: Baugh-Wooley Multiplier basic cell construction (a) Baugh-Wooley Multiplier White box , (b)Baugh-Wooley Multiplier Gray box[4]

When multiplying 2'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 CSA tree. It has been restricted to perform signed bits. [4].

3.2.3 Braun Multiplier

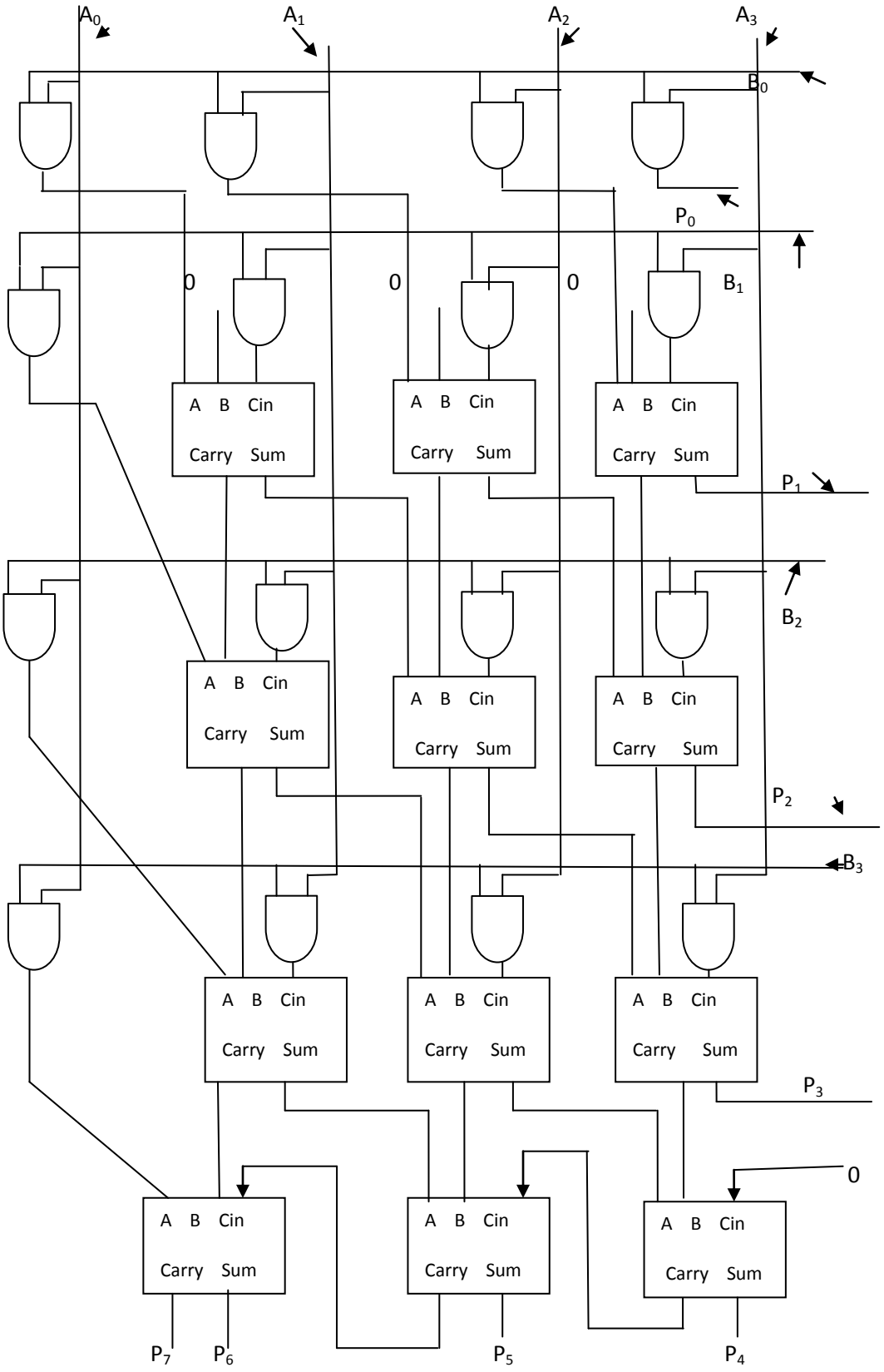


Fig3.7: Braun multiplier [2]

Where, A: 4-bit multiplicand

B: 4-bit multiplier

P: 8-bit product of X and Y

Pn = Ai*Bj is a product bit

An n*n bit Braun multiplier is constructed with n (n-1) adders and n2 AND gates as shown in the figure 3.7, [2]. An array implementation of parallel multiplier is Braun multiplier. It is a simple parallel multiplier generally called as carry save array multiplier. The structure consists of array of AND gates and adders arranged in the iterative manner and no need of logic registers. This can be called as non–addictive multiplier Architecture.

The shifting would carry out with the help of Carry Save Adder (CSA) and the Ripple carry adder should be used for the final stage of the output. Braun multiplier performs well for the unsigned operands that are less than 16 bits in terms of speed, power and area. But it is simple structure when compared to the other multipliers. The main drawback of this multiplier is that the potential susceptibility of glitching problem due to the Ripple Carry Adder in the last stage. The delay depends on the delay of the Full Adder and also a final adder in the last. It has the advantages of low capacitance and fast speed in comparison to array multiplier because in Braun multiplier critical path is propagated diagonally rather than horizontally in array multiplier .

The circuit design used inside the full adder is high speed modified SODS. One other advantage of this design style is it is a clocked circuit implies that the output or operation is performed only when the clock is high and other advantage is that the complete signal can be generated.

Using NAND gate for the next stage of the multiplier operation because it is a differential structure and generated the output and its complement simultaneously and the complete signal generation is described in section 2.2. In the 1st row of the multiplier Figure 3.7, the clock is used and for the next stages the complete signal is generated using the previous stages output. By using the complete signal generation technique, the stage is put into evaluation mode only when the clock by the complete signal or after the output obtained at previous stage.Thus the output of the Multiplier is obtained in asychronously and the glitches can be removed because the next stage is evaluated only after previous stage evaluation has been done. Modified SODS gives better result in Braun multiplier than basic SODS adder used as a unit cell in Braun multiplier design.

Sr. No.	**Input bits 4bit*4bit (decimal)**	**Output bits of 4-Bit Braun multiplier**	**4-bit Braun multiplier delay using SODS adder**	
			Basic	**Modified**
1.	**14*12**	168(10101000)	1.36 ns	586 ps
2.	**15*12**	180(10110100)	1.34 ns	990 ps
3.	**13*12**	156(10011100)	1.32 ns	1.13 ns
4.	**12*12**	144(10010000)	1.23 ns	1.13 ns
5.	**8*8**	64(01000000)	1.32 ns	1.18 ns
6.	**9*8**	72(01001000)	1.35 ns	1.11 ns
7.	**11*9**	99(01100011)	1.28 ns	1.10 ns
8.	**13*13**	169(10101001)	1.39 ns	1.26 ns
9.	**11*11**	121(01111001)	1.43 ns	944 ps
10.	**15*11**	165(10100101)	854 ps	311 ps
11.	**15*9**	135(10000111)	1.15 ns	320 ps
12.	**9*13**	117(01110101)	1.43 ns	990 ps
13.	**11*13**	143(10001111)	1.39 ns	1.19 ns
14.	**14*13**	182(10110110)	1.43 ns	723 ps
15.	**15*15**	225(11100001)	1.25 ns	950 ps
16.	**7*15**	105(01101001)	1.41 ns	1.05 ns
17.	**7*14**	98(01100010)	1.33 ns	1.05 ns

Table 3.2: Results showing comparison of Delay for basic and modified 4-bit Braun multiplier

Sr. No.	**Input Bits 4Bit*4Bit (decimal)**	**Output Bits of 4-Bit Braun multiplier**	**4-bit Braun multiplier power using SODS adder (uW/MHz)**	
			Basic	**Modified**
1.	**14*12**	168(10101000)	9.80	13.22
2.	**15*12**	180(10110100)	8.08	11.42
3.	**13*12**	156(10011100)	8.46	12.68
4.	**12*12**	144(10010000)	11.62	15.52
5.	**8*8**	64(01000000)	14.00	18.76
6.	**9*8**	72(01001000)	12.10	16.96
7.	**11*9**	99(01100011)	9.58	13.22
8.	**13*13**	169(10101001)	7.84	10.94
9.	**11*11**	121(01111001)	7.48	11.48
10.	**15*11**	165(10100101)	6.30	7.60
11.	**15*9**	135(10000111)	8.42	11.00
12.	**9*13**	117(01110101)	8.98	13.10
13.	**11*13**	143(10001111)	10.28	14.00
14.	**14*13**	182(10110110)	7.92	11.18
15.	**15*15**	225(11100001)	9.48	12.34
16.	**7*15**	105(01101001)	8.76	10.80
17.	**7*14**	98(01100010)	10.00	12.70

Table 3.3: Results showing comparison of power for basic and modified multiplier

3.2.3.1 Simulation Results:

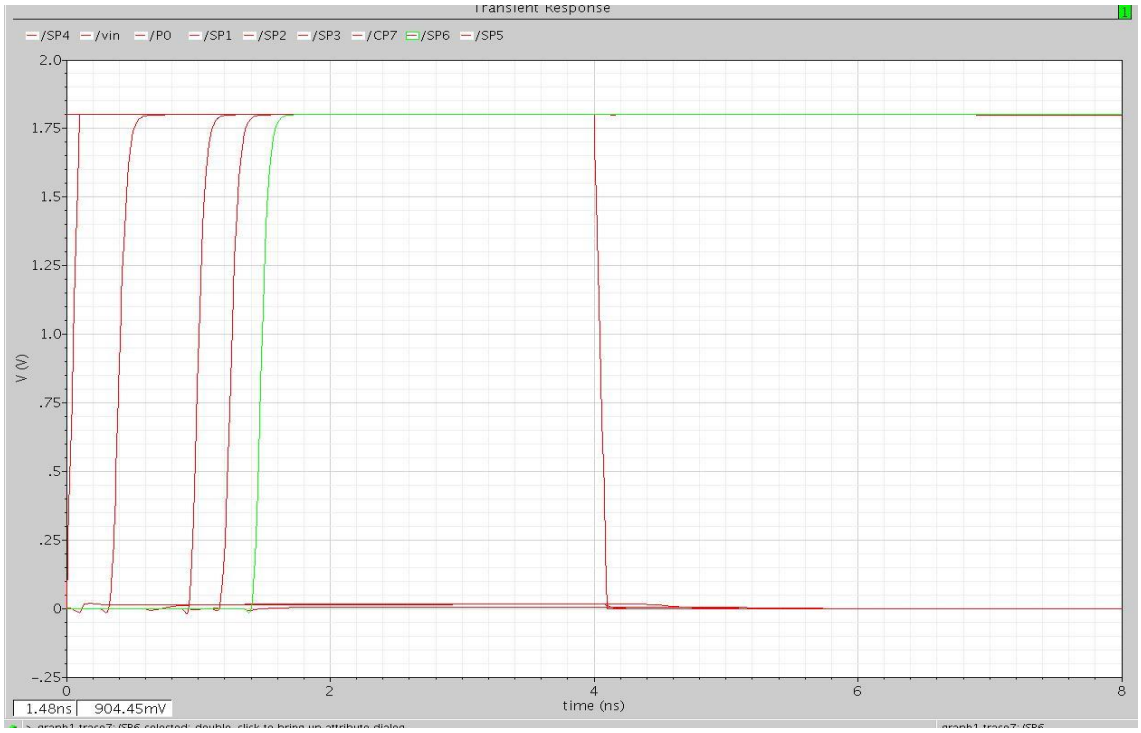


Figure3.8: 9*13 basic SODS 4-bit Braun multiplier output

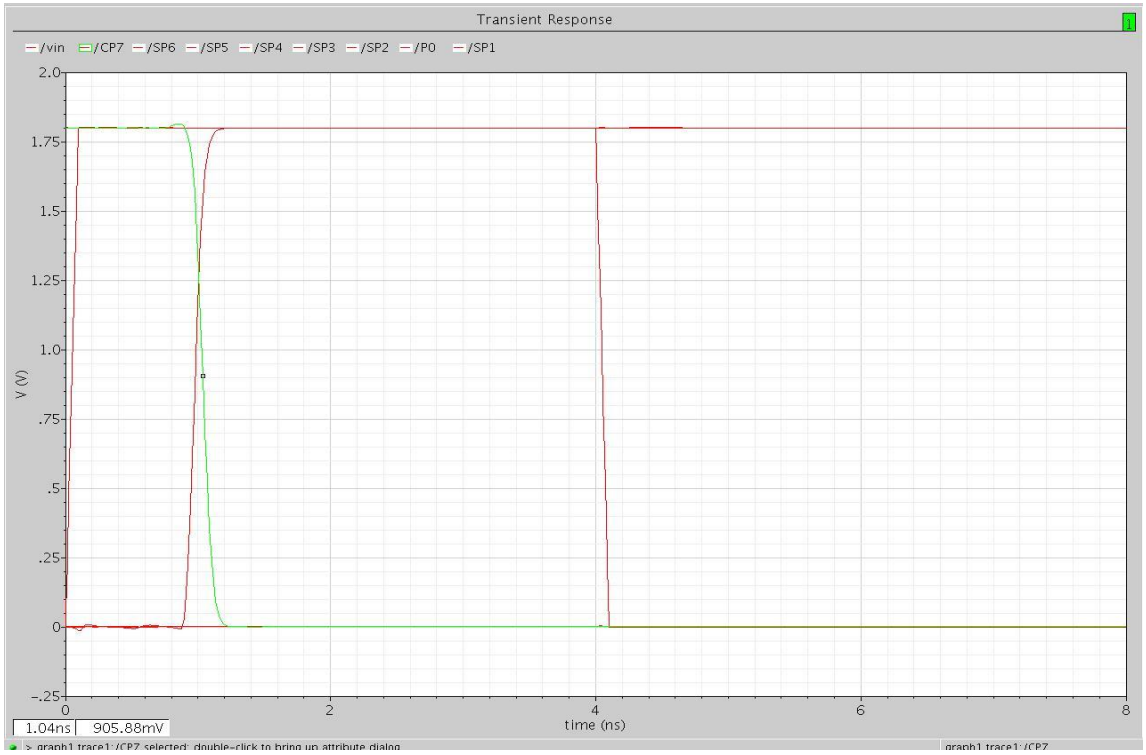


Figure 3.9: 9*13 modified SODS 4-bit Braun multiplier output

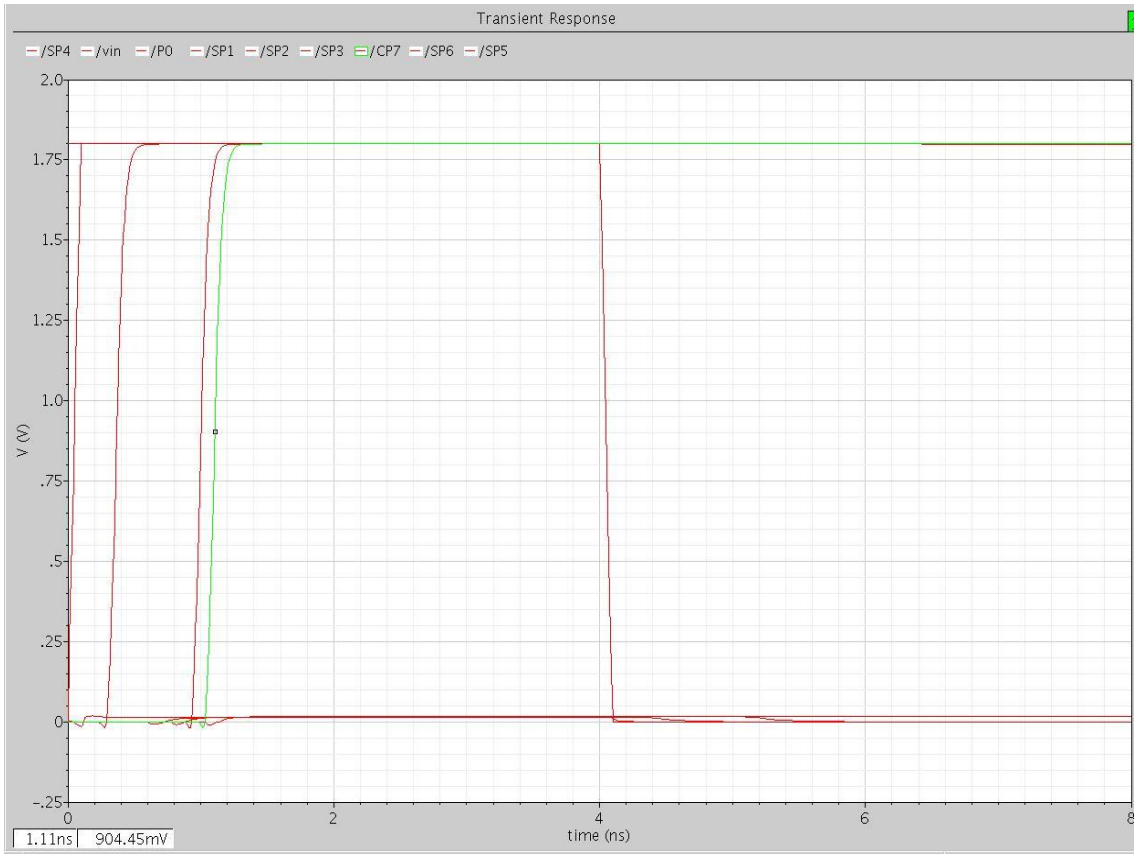


Figure 3.10: 15*11 basic SODS 4 bit Braun multiplier output

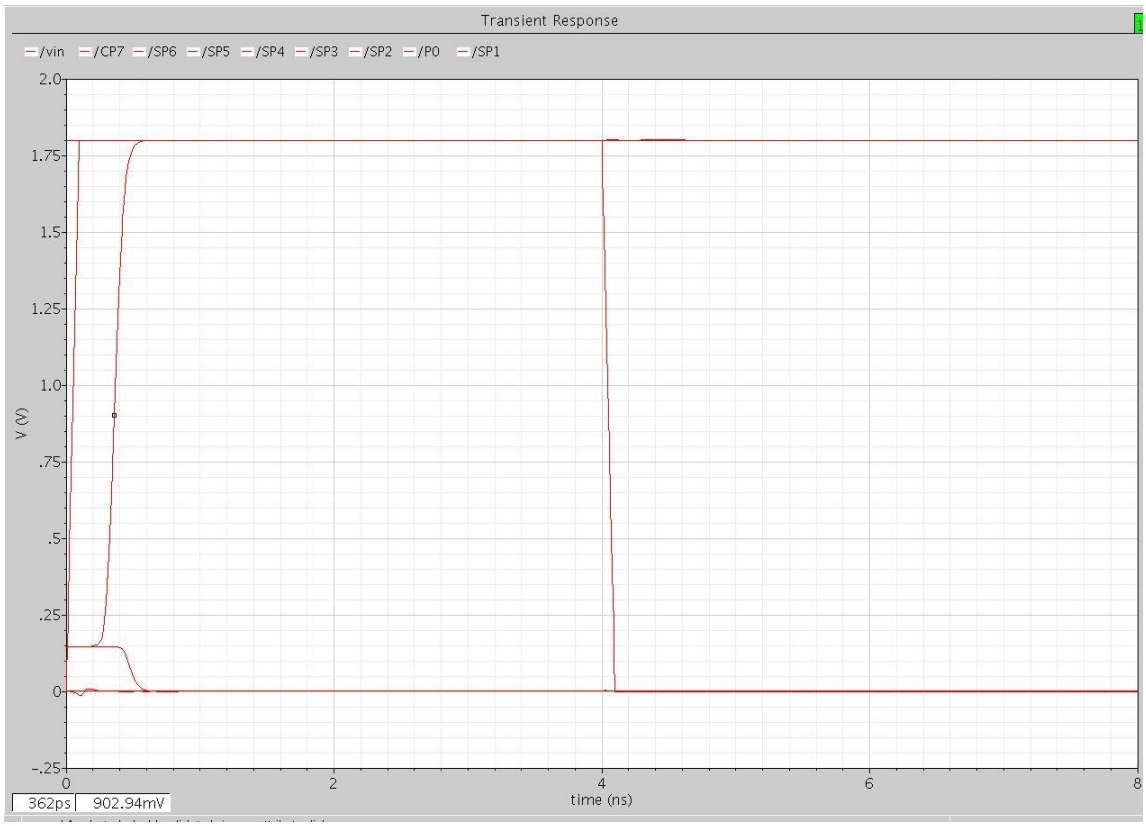


Figure 3.11: 15*11 modified SODS 4 bit Braun multiplier output

Chapter 4

LOW POWER DESIGN

4.1 Low Power Design Techniques:

In order to reduce the power consumption in all the selected multipliers all contributors to low power must be targeted. There are several power reduction techniques which are used to reduce the power dissipation of the circuits [10], [11], [12]:

1) Power gating: is best technique to reduce power reduction.

2) Gate level optimization: Energy delay product can be improved by avoiding wasting of energy,

i) By avoiding node transitions that are of necessary,

ii) The idle blocks do not consume any power,

iii) By controlling the consumption power of the blocks. Most of the times, these blocks are the clock system and buses. This leads to a significant reduction in the static power Multi Vdd [9]:- Since the dynamic power is proportional to Vdd^2, by lowering Vdd on selected blocks can reduce power significantly. In this design lowering the voltage increases the delay Multi V_t technique [9]: Many libraries today offer two or three version of their cells. Low V_t, Standard V_t, and High V_t. The implementation tools can take advantage of these libraries to optimize timing and power simultaneously. It is now common to use "Dual V_t "flow during synthesis. The goal of this approach is to minimize the total number of fast, leaky low V_t transistors by deploying them when it is required to meet timing. MTCMOS is used to reduce the power consumption in the ideal state.

4.1.1 MTCMOS Technique:

MTCMOS logic is effective standby leakage control technique, but difficult to implement since sleep transistor sizing is highly dependent on discharge pattern within the circuit block. They showed dual V_t domino logic avoids the sizing difficulties and inherent performance associated with MTCMOS. High V_t cells are used where leakage has to be prevented whereas low V_t cells are employed where

speed is of concern. Both cells are effectively used in MTCMOS technique [3], [10], [11]. In active mode of operation the high V_t transistors are turned off and the logic gates consisting of low V_t transistors can operate with low switching power dissipation and smaller propagation delay. In standby mode the high V_t transistors are turned off thereby cutting off the internal low V_t circuitry.

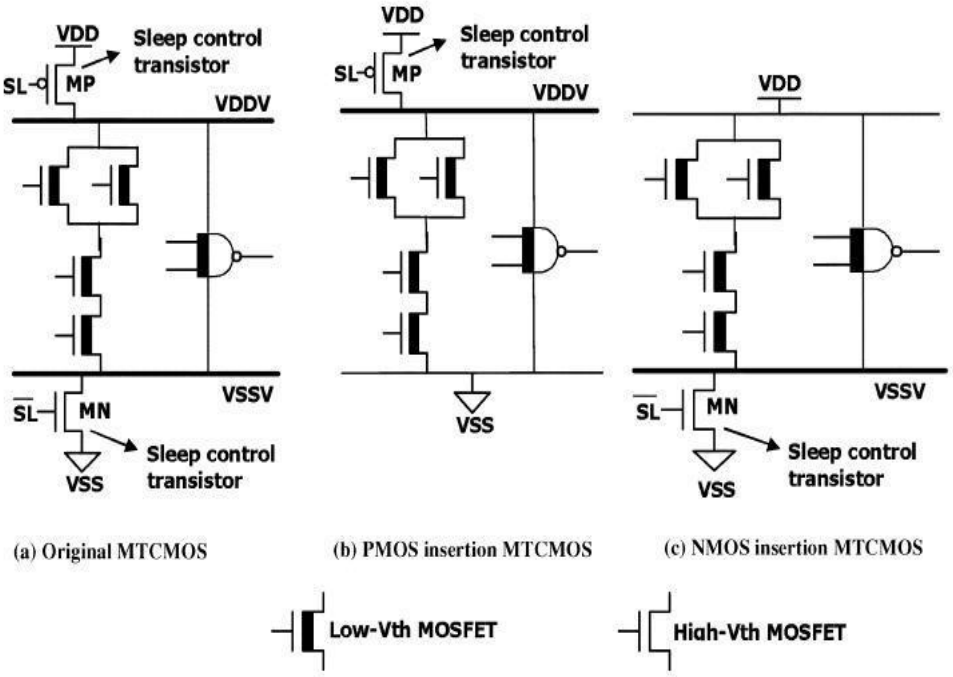


Figure4.1: Schematic of MTCMOS technique [11]

Multi-threshold-voltage CMOS*:* Multi-threshold voltage CMOS (MTCMOS) reduces the leakage by inserting high-threshold devices in series to low circuitry .Fig. 4.1(a) shows the schematic of an MTCMOS circuit. A sleep control scheme is introduced for efficient power management. In the active mode, SL is set low and sleep control high transistors (MP and MN) are turned on. Since their on-resistances are small, the virtual supply voltages (VDDV and VSSV) almost function as real power lines. In the standby mode, SL is set high, MN and MP are turned off, and the leakage current is low. In fact, only one type of high transistor is enough for leakage control.

Fig. 4.1(b) and (c) shows the PMOS insertion and NMOS insertion schemes, respectively. The NMOS insertion scheme is preferable, since the NMOS on-resistance is smaller at the same width; therefore, it can be sized smaller than

corresponding PMOS. MTCMOS can be easily implemented based on existing circuits. A 1-V digital signal processor chip for mobile phone applications has been developed recently. However, MTCMOS can only reduce the standby leakage power, and the large inserted MOSFETs can increase the area and delay. Moreover, if data retention is required in the standby mode, an extra high memory circuit is needed to maintain the data. Instead of using high sleep control transistors as MTCMOS, super cut-off CMOS (SCCMOS) technique uses low transistors with an inserted gate bias generator. For the PMOS (NMOS) insertion, the gate is applied to 0V (VDD) in the active mode, and the virtual VDD (VSS) line is connected to supply VDD (VSS). In the standby mode, the gate is applied to VDD (VSS to fully cut off the leakage current).

4.2 SODS circuit working by using 'MTCMOS Technique':

For reducing the power consumption, MTCMOS technique is used in the modified SODS. For this high V_t transistor is connected before Vdd and Gnd terminal in the circuit as shown in figure 4.2 and after using low power technique, the power reduced but overall delay of the circuit increased. MTCMOS technique used in this circuit high V_t transistors are added before the Vdd and Gnd in the circuit, rest the circuitry remains the same. Low V_t transistors are not used for the internal block, first the NMOS and PMOS at the two end point are charged to Vdd and Gnd, than the circuit puts into evaluation mode and sum output is evaluated and the power and delay is as shown in table 4.1. For one of the combination, the circuit is verified; it will provide low power surely for other input combinations.

Inputs (ABCin)	**Output (S/Sbar)**	**Power (nW/Hz)**		**Delay (ps)**	
			After using low power technique		**After using Low Power technique**
011	0/1	270	210	155	250

Table 4.1: Result for reduction of power using low power technique and increment of delay

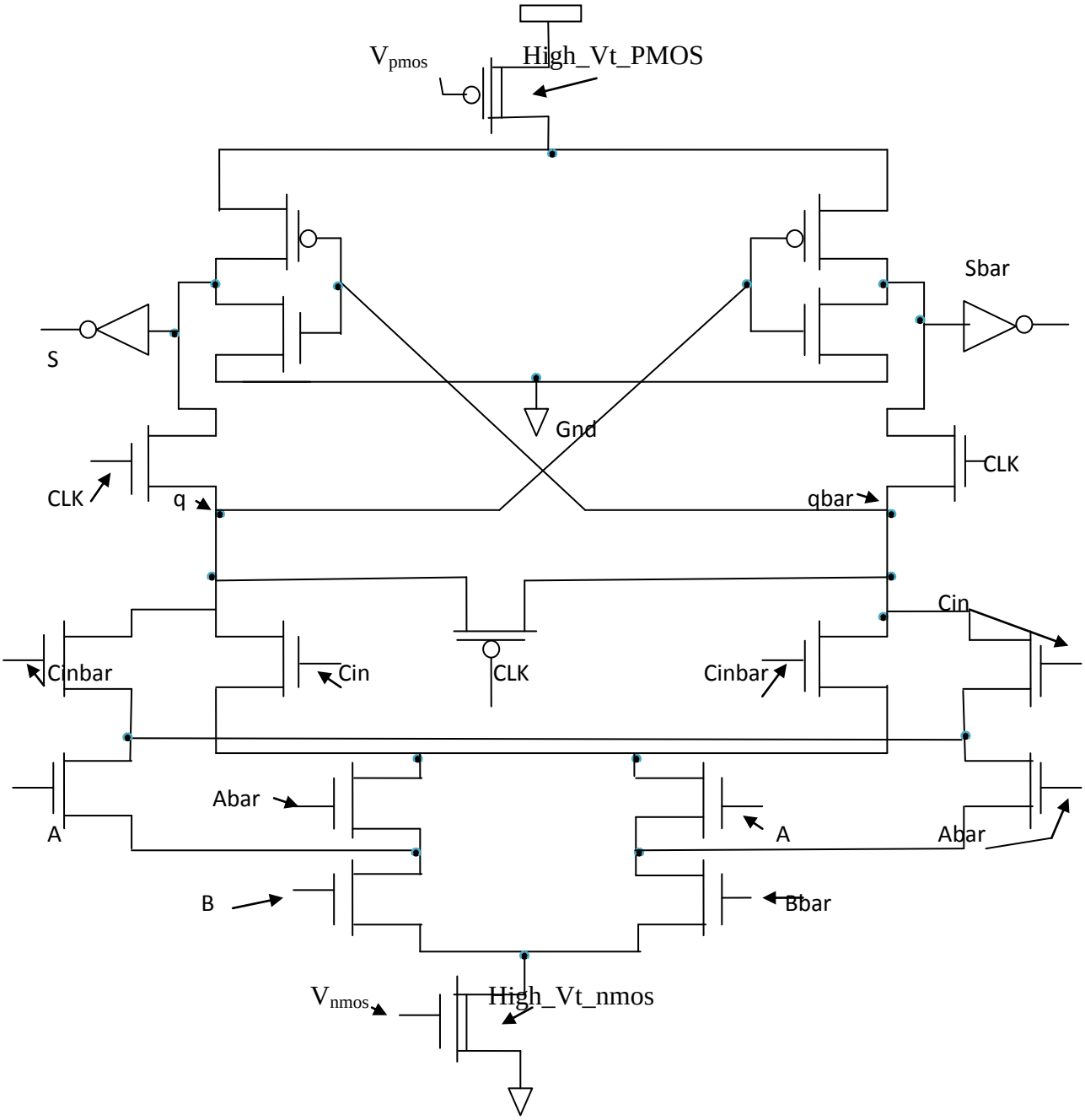


Figure 4.2: Low Power Circuit design of modified SODS sum circuit

4.2.1Simulation Results:

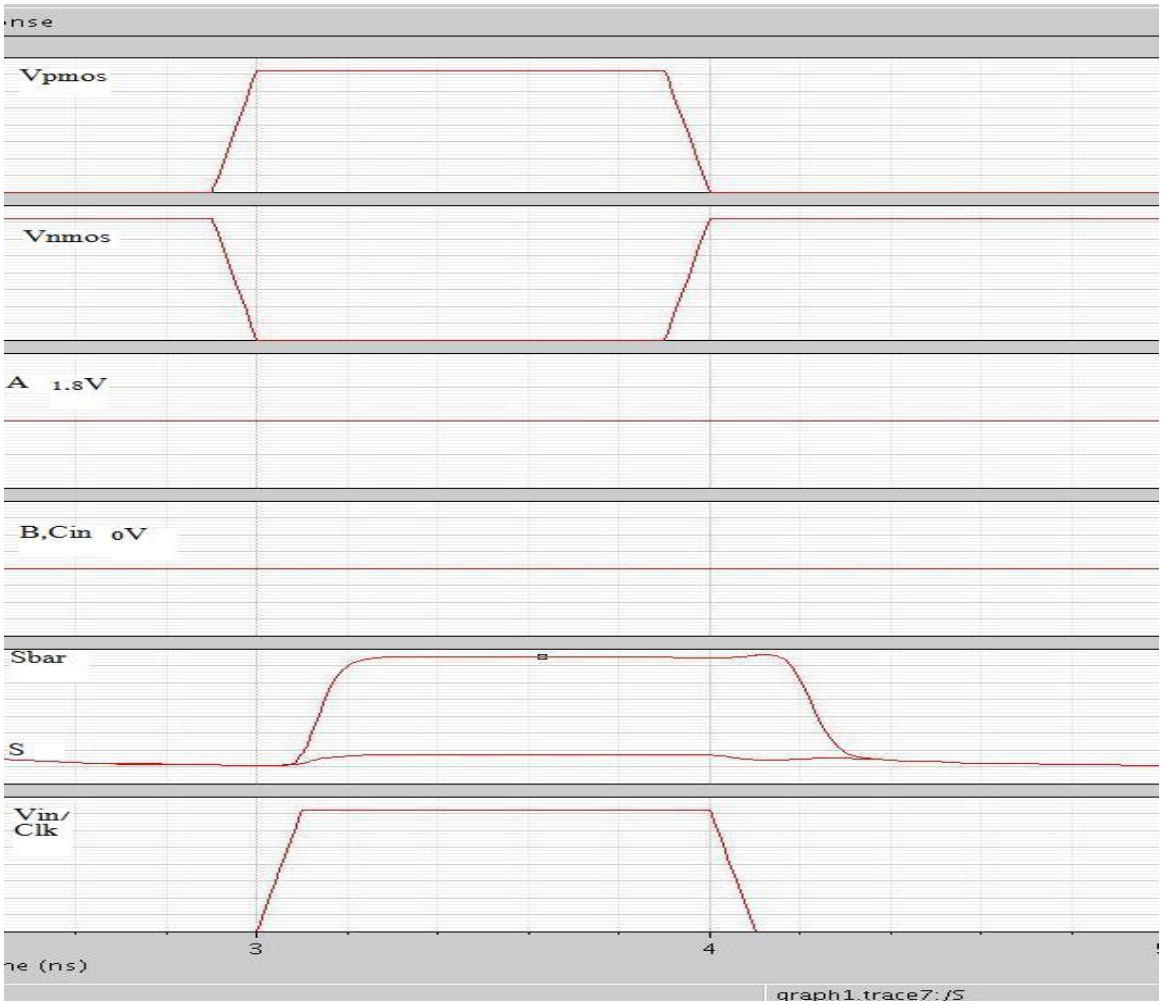


Figure 4.3: Results showing for low power circuit technique

Chapter 5
CONCLUSION

Conclusion:

The modified structure of SODS has been investigated that gives a speed higher than the basic SODS circuit. This structure is designed for full adder circuit as shown in figure 2.8 and figure 2.9 and results showing comparison between the two structures basic and modified SODS (full adder) circuit is given in Table 2.4.In the basic SODS structure NMOS tree is used as a pull-down network. The disadvantage of basic SODS is its minimum logical depth that causes to increase the size of the transistor used in NMOS tree but the self-loading limits the maximum size of the transistor in the NMOS tree. So a modified circuit is proposed to overcome this problem. In the modified circuit cross coupled inverter pair is used to generate the full logic '0' and logic '1' and NMOS tree used to play a role to generate the difference between the two logic levels '0' and '1' and not to generate the complete logic level. Thus it has been seen that in the basic structure the NMOS tree is responsible for the propagation delay. The delay has been further reduced due to the use of cross coupled inverter pair in the modified structure of SODS.

The modified SODS full adder has been used in designing of 4-bit multiplier and better result has been shown as compare to the basic structure or higher speed in comparison to basic SODS circuit as given in Table 3.2. The Braun algorithm has been used for multiplier design instead of the array multiplier because in Braun algorithm, load on the critical path is less than load on the critical path of 4-bit array multiplier algorithm. Further low power technique 'MTCMOS' is used to reduce the power dissipation of the circuit, the low power technique has reduced the power dissipation of the circuit but at the cost of delay as given in Table 4.1. It has been observed that the Braun multiplier using SODS circuits improves the performance.

Future Scope:

In the future scope, this high speed circuit can be used for signed multiplication in Baugh-Wooley multiplier, this will give better result for delay for modified structure as compare to the basic structure of SODS full adder and also for Floating point numbers multiplication by converting first in binary for multiplication and then after multiplication operation converted back to binary.

REFERENCES

[1] T. Esther Rani , Dr. Rameshwar Rao, "Area and Power Optimized Multiplier with Minimum Leakage", Electronics Computer Technology, Vol. 3,pp. 284-287, 2011.

[2] R. Anitha , V. Bagyeereswaram, "Comparative study of Braun's Multiplier Using FPGA Devices", IJEST, Vol. 3 No. 6, pp. 4785-4793, 3 June 2011.

[3] P. V. Rao , Cyril Prasanns Raj P , S. Ravi, "VLSI Design and Analysis of Multiplier for Low Power", Fifth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, pp. 1354-1357, 2009.

[4] Magnus Sjalander, Per Larsson-Edefors, "High Speed and Low Power Multipliers using the Baugh-Wooley Algorithm and HPM Reduction Tree", proceedings of the 15th International Conference on Electronics, Circuits and Systems, pp. 33-36, 2008.

[5] John P. Uyemura, "Chip Design for Submicron VLSI", Thomson, India Edition-2007.

[6[22] Anitha R,Bagayaveereswaran, "Braun's Multiplier Implementation using FPGA with Bypassing Techniques", VLSICS Vol. 2, No. 3, pp. 225-231, Sept. 2011.

[7] J. Rabaey and B. Nikolic, A. Chandrakasan, "Digital Integrated Circuits: A Design Perspective", Prentice Hall, 2003.

[8] "Evaluation of Two Summed Adders Implemented in ECDL CMOS Differential Logic", IEEE Journal of solid state circuits, Vol. 26, No. 8, pp. 1152-1160, August 1991.

[9] J. Acosta, M. Valencia, A. Barriga, M. J. Bellido, J. L. Huertas, "SODS: A New CMOS Differential-Type Structure", IEEE Journal of Solid-State Circuits, vol. 30, no. 7, pp. 835-838, July 1995.

[10] Kaushik Roy, Kiat-Seng Yeo, "Low Voltage, Low-power VLSI Subsystems", McGraw-Hill, Edition no. 1, pp.124-141, 2009.

[11] Kaushik Roy, Saibal Mukhopadhayay Hamid Mahmoodi-Meimand, "Leakage Current Mechanism and Leakage Reduction Technique in Deep Submicrometer CMOS-Circuits", Proceedings of the IEEE, vol. 91, no. 2, pp. 305-327, February 2003.

[12] Roy and Prasad, "Low Power CMOS VLSI Circuit Design", John Wiley & sons, 2 Feb. 2009.

[13] A. J. Acosta, R. Jimenez, A. Barriga, M. J. Bellido, M. Valencia, J. L. Huertas, "Design and characterization of a CMOS VLSI self-timed Multplier architecture based on a bit-level pipelined-array structure", IEEE Proc. -Circuits Devices Syst.., Vol. 145, No. 4, pp. 247-253, August 1998.

[14] "SELF: A Self-timed Systems Design Technique", Electronics Letter, Vol. 23, No. 6, pp. 269-270, 12th March 1987.

[15] Joh P. Uyemera, "CMOS LOGIC CIRCUIT DESIGN", Springer international edition-2005.

[16] Neil H. E. Weste, David harris, Ayan Banerjee, "CMOS VLSI DESIGN", Third edition Pearson Education India Edition-2006.

[17] Dauglas A. Pucknell, Karman Eshragian, "BASIC VLSI DESIGN", Prentice Hall of India Pvt. Ltd. Third Edition-2005.

[18] M. Alioto and G. Palumbo, "Analysis and comparison on full adder block in submicron technology", IEEE Trans. Very Large Scale Integer. (VLSI) Syst., vol. 10, no. 6, pp. 806-823, Dec. 2002.

[19] J. H. Chang, J. Gu, M. Zhang, "A review of 0.18-m full adder performances for tree structured arithmetic circuits", IEEE trans. Very Large Scale Integer. (VLSI) Syst., vol. 13, no. 6, pp. 686-695, Jun. 2005.

[20] A. M. Shams, T. K. Darwish, M. A. Bayoumi, "Performance analysis of Low-power 1-bit CMOS full adder cells", IEEE Very Large Scale Integer. (VLSI) Syst., vol. 10, no. 1, pp. 20-29, Feb. 2002.

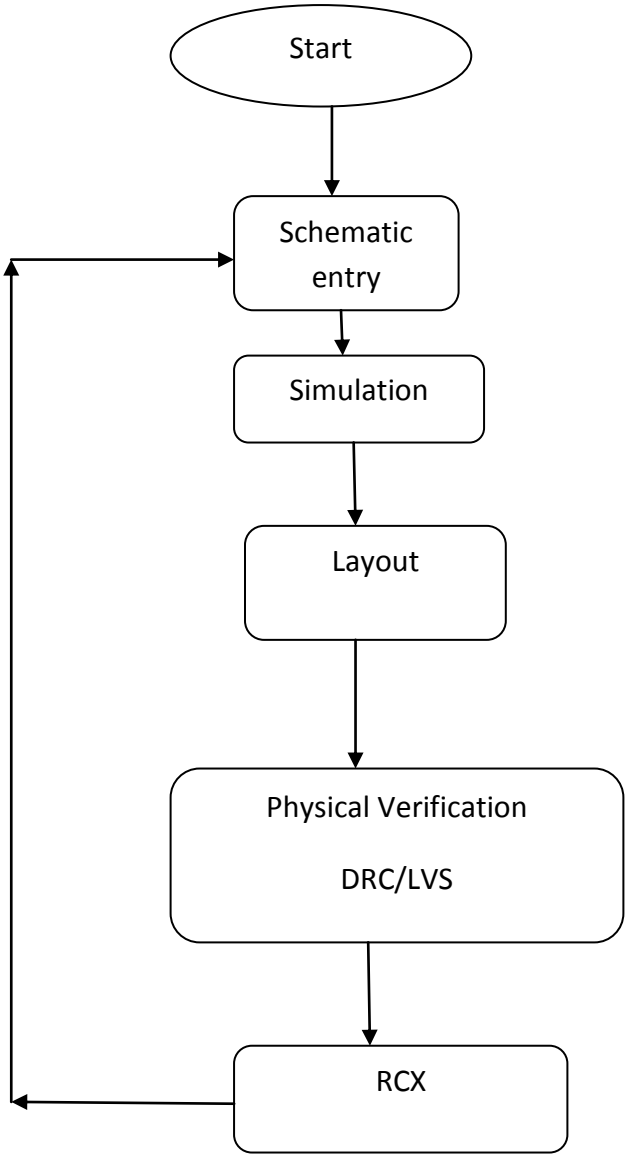
[21] Ronak Bajaj, Saransh Chhabra, Sreehari Veeramchaneni, M B Srinias, "A Novel, Low Power Arrray Multplier Architecture", pp. 119-123, IEEE 2009.

[22] Angus Wu, Tat Chee Ave., "High Perfomance Adder Cell for Low Power Pipelined Multiplier", Circuit and Systems, pp. 57-60, 1996..

[23] D. Sinha,T. Sharma, K.g. Sharma, B. P. Singh, " Design and analsis of low power 1-bit full adder cell", ICECT, Vol. 2, pp. 303-305, 8-10 April 2011.

APPENDIX

Cadence Flow:



Cadence 1.8V NMOS Model file

```
**********************************************************************************
simulator lang=spectre insensitive=yes
//          0.18um 1.8v/3.3v MIXEDMODE Triple-well technology, 1.8V NMOS Model
//            *
//
**********************************************************************************
                                model n_bpw_18_mm                   + version=3.2000e+00
                                bsim3v3 type=n                      binunit=1.0000e+00
                                                                    mobmod=1.0000e+00
+ capmod=2.0000e+00             + tox=4.2000e-09 +                  + xj=1.6000e-07
nqsmod=0.0000e+00               dtox_n_bpw_18_mm                    nch=3.7446e+17
                                toxm=4.2000e-09                     rsh=8.0000e+00
+ ngate=1.0000e+23              + k1=4.5780e-01                     + k3b=2.3790e-01
vth0=3.0750e-01 +               k2=-2.6380e-02          k3=-        w0=-8.8130e-08
dvth0_n_bpw_18_mm               1.0880e+01                          nlx=4.2790e-07
+ dvt0=4.0420e-01               + dvt0w=3.8300e-01                  + lint=1.5870e-08
dvt1=3.2370e-01                 dvt1w=6.0000e+05                    wint=1.0220e-08
dvt2=-8.6020e-01                dvt2w=-2.5000e-02                   dwg=-3.3960e-09
+ dwb=1.3460e-09                + ua=-9.2010e-10                    + vsat=7.1580e+04
u0=3.1410e+02 +                 ub=1.9070e-18                       a0=1.9300e+00
du0_n_bpw_18_mm                 uc=4.3550e-11                       ags=5.0720e-01
+ b0=1.4860e-06                 + a1=0.0000e+00                     + nfactor=1.0380e+00
b1=9.0640e-06                   a2=1.0000e+00                       cit=-1.5110e-03
keta=1.7520e-02                 voff=-1.0880e-01                    cdsc=2.1750e-03
+ cdscd=-5.0000e-04             + etab=-1.4590e-03                  + pdiblc1=3.0610e-03
cdscb=8.2410e-04                dsub=1.5920e-03                     pdiblc2=1.0000e-06
eta0=1.0040e-03                 pclm=1.0910e+00                     pdiblcb=0.0000e+00
+ drout=1.5920e-03              + pvag=-2.9580e-01                  + prwb=0.0000e+00
pscbe1=4.8660e+08               rdsw=4.9050e+00                     wr=1.0000e+00
pscbe2=2.8000e-07               prwg=0.0000e+00                     alpha0=0.0000e+00
+ alpha1=0.0000e+00             + cgso=2.3500e-10 +                 + cgdo=2.3500e-10 +
beta0=3.0000e+01                dcgso_n_bpw_18_mm                   dcgdo_n_bpw_18_mm
xpart=1.0000e+00                                                    cgbo=0.0000e+00
+ cgsl=0.0000e+00               + cf=1.5330e-10                     + dlc=2.9000e-08
cgdl=0.0000e+00                 clc=1.0000e-07                      dwc=0.0000e+00
ckappa=6.0000e-01               cle=6.0000e-01                      vfbcv=-1.0000e+00
+ noff=1.0000e+00               + moin=1.5000e+01                   + lmin=1.8000e-07
voffcv=0.0000e+00
acde=1.0000e+00
+ lmax=5.0000e-05               + xl= - 1.0500e-08 +                + xw=0.0000e-00 +
wmin=2.4000e-07                 dxl_n_bpw_18_mm                     dxw_n_bpw_18_mm
wmax=1.0000e-04                                                     js=1.0000e-06
+ jsw=7.0000e-11                + mj=4.4300e-01                     + cjsw=1.3400e-10 +
cj=1.0300e-03 +                 pb=8.1300e-01                       dcjsw_n_bpw_18_mm
dcj_n_bpw_18_mm                                                     mjsw=3.3000e-01
```

+ tnom=2.5000e+01 + kt1l=-4.1750e-09 + ub1=-2.6730e-18
ute=-1.2860e+00 kt1=-2.2550e-01 kt2=-2.5270e-02 uc1=-3.8320e-11
+ prt=-1.0180e+01 ua1=2.1530e-09 at=1.4490e+04
xti=3.0000e+00 + wln=1.0000e+00 + wwl=0.0000e+00
wl=0.0000e+00 ww=7.2620e-16 ll=-1.0620e-15
+ lw=2.9960e-15 wwn=1.0000e+00 lln=1.0000e+00
lwn=1.0000e+00 + llc=-2.1400e-15 + wlc=0.0000e+00
lwl=0.0000e+00 lwc=0.0000e+00 wwc=0.0000e+00
+ lvth0= - 1.0000e-03 + lwlc=0.0000e+00 wwlc=0.0000e+00
dlvth0_n_bpw_18_mm + wvth0=6.027e-02 + + lnlx=-2.8540e-08
dwvth0_n_bpw_18_mm wnlx=0.0000e+00
pvth0=0 + pnlx=0.0000e+00
dpvth0_n_bpw_18_mm
+ wua=-1.8800e-11 + pub=3.8000e-20 + weta0=0.0000e+00
wu0=5.4000e-01 + pw0=1.3000e-09 wetab=0.0000e+00
dwu0_n_bpw_18_mm wrdsw=0.0000e+00 leta0=1.5740e-03
+ letab=0.0000e+00 + wpclm=0.0000e+00 + pvoff=-3.7880e-04
peta0=0.0000e+00 wvoff=-4.0780e-04 wa0=-4.7310e-02
petab=0.0000e+00 lvoff=-4.2080e-03 la0=-4.6670e-01
+ pa0=-2.6490e-02 + pags=0.0000e+00 + pketa=0.0000e+00
wags=4.2420e-03 wketa=0.0000e+00 wute=6.3730e-02
lags=3.0280e-01 lketa=-1.9420e-02 lute=0.0000e+00
+ pute=0.0000e+00 + pvsat=0.0000e+00 + + wat=7.0670e+03
wvsat=5.0660e+03 dpvsat_n_bpw_18_mm wprt=0.0000e+00
lvsat=0.0000e+00 lpdiblc2=-4.7520e-03
+ ldif=8.0000e-08 + pbsw=8.8000e-01 + ctp=9.1400e-04
hdif=2.6000e-07 cjswg=5.0000e-10 + ptp=9.2400e-04
n=1.0000e+00 dcjgate_n_bpw_18_mm cta=9.1900e-04
+ pta=1.5800e-03
elm=5.0000e+00
tlevc=1.0000e+00

Cadence 1.8V PMOS Model file

```
**********************************************************************************
//        0.18um 1.8v/3.3v MIXEDMODE Twin-well technology, 1.8V PMOS Model
//          *
//
**********************************************************************************

model p_18_mm bsim3v3 type=p
+ tox=4.2000e-09 + dtox_p_18_mm toxm=4.2000e-09
+ k2=6.9730e-03 k3=-2.8330e+00 k3b=1.3260e+00
+ dvt1w=2.4790e+04 dvt2w=6.9150e-01 lint=-1.0410e-08
+ ub=2.6460e-19 uc=-9.5870e-02 vsat=5.3400e+04
+ a2=1.0000e+00 voff=-1.0730e-01 nfactor=1.5350e-00
+ dsub=1.9191e+00 pclm=2.6530e+00 pdiblc1=0.0000e+00
+ rdsw=7.9210e+02 prwg=0.0000e+00 prwb=0.0000e+00
+ cgso=2.0540e-10 + dcgso_p_18_mm xpart=1.0000e+00
+ cle=6.0000e-01 dwc=0.0000e+00 vfbcv=-1.0000e+00
+ wmin=2.4000e-07 wmax=1.0000e-04

+ jsw=4.1200e-11 cj=1.1400e-03 + dcj_p_18_mm
+ tnom=2.5000e+01 ute=-4.4840e-01 kt1=-
+ mobmod=3.0000e+00 version=3.2000e+00 capmod=2.0000e+00
+ xj=1.0000e-07 nch=6.1310e+17 ngate=1.0000e+23
+ w0=-1.9430e-07 nlx=2.5300e-07 dvt0=4.8850e-01
+ wint=-1.5250e-07 dwg=-1.1510e-07 dwb=-1.0390e-07
+ a0=1.3500e+00 ags=3.8180e-01 b0=-3.0880e-07
+ cit=-1.0670e-03 cdsc=7.5780e-04 cdscd=-2.8830e-05
+ pdiblc2=5.0000e-06 pdiblcb=0.0000e+00 drout=1.4570e+00
+ alpha0=0.0000e+00 alpha1=0.0000e+00 beta0=3.0000e+01
+ cf=1.5330e-10 dlc=5.6000e-08 cgsl=0.0000e+00
+ noff=1.0000e+00 voffcv=0.0000e+00 acde=1.0000e+00
+ xl= - 2.0000e-09 + dxl_p_18_mm

+ mj=3.9500e-01 pb=7.6200e-01

+ kt1l=-8.2040e-09 kt2=-9.4870e-03
+ binunit=1.0000e+00 nqsmod=0.0000e+00

+ vth0= - 4.5550e-01 + dvth0_p_18_mm k1=5.7040e-01
+ dvt1=7.5780e-02 dvt2=1.2870e-01 dvt0w=-1.2610e-01
+ u0=1.1450e+02 + du0_p_18_mm ua=1.5400e-09
+ b1=0.0000e+00 keta=1.0440e-02 a1=0.0000e+00
+ cdscb=1.0000e-04 eta0=1.0710e+00 etab=-9.2910e-01
+ pscbe1=4.8660e+08 pscbe2=2.8000e-07 pvag=1.1620e+00
+ cgdo=2.0540e-10 + dcgdo_p_18_mm cgbo=0.0000e+00
+ cgdl=0.0000e+00 ckappa=6.0000e-01 clc=1.0000e-07
+ moin=1.5000e+01 lmin=1.8000e-07 lmax=5.0000e-05
+ xw=0.0000e+00 + dxw_p_18_mm js=3.0000e-06
+ cjsw=1.7400e-10 + dcjsw_p_18_mm mjsw=3.2400e-01
+ ub1=-6.0260e-18 uc1=-9.8500e-02
```

2.1940e-01
+ prt=0.0000e+00
xti=3.0000e+00
ww=1.2360e-14
+ lln=1.0000e+00
lwn=1.0000e+00
lwl=0.0000e+00
+ lvth0=4.4000e-03 +
dlvth0_p_18_mm

+ wrdsw=1.0070e+01
weta0=0.0000e+00
wetab=0.0000e+00
+ pub=0.0000e+00
wuc=0.0000e+00
luc=0.0000e+00
+ pa0=8.6610e-02
wags=-4.1770e-02
lags=4.4540e-02
+ pute=0.0000e+00
wvsat=-1.4200e+04
lvsat=0.0000e+00
+ wprt=2.1660e+02
n=1.0000e+00
pbsw=6.6500e-01
+ rd=0.0000e+00
rsc=0.0000e+00
rdc=0.0000e+00
+ noic=2.61260020285845E-
11 ef=1.1388

ua1=4.5710e-09
+ lw=-2.8730e-16
ll=6.6350e-15
wl=0.0000e+00
+ llc=-7.4500e-15
lwc=0.0000e+00
lwlc=0.0000e+00
+ wvth0= - 1.4800e-02 +
dwvth0_p_18_mm

+ wpclm=0.0000e+00
wua=2.6300e-09
lua=-8.1530e-11
+ puc=0.0000e+00
wvoff=-9.8160e-03
lvoff=-9.8710e-04
+ pags=-4.0760e-02
wketa=0.0000e+00
lketa=-1.2000e-02
+ pvsat= - 4.3400e+02 +
dpvsat_p_18_mm
lpdiblc2=3.0120e-03
+ cta=1.0000e-03
ctp=7.5300e-04
pta=1.5500e-03
+ hdif=2.6000e-07
rs=0.0000e+00

+ em=41000000

at=1.2030e+04
+ wln=1.0000e+00
wwn=1.0000e+00
wwl=0.0000e+00
+ wlc=0.0000e+00
wwc=0.0000e+00
wwlc=0.0000e+00
+ pvth0=3.2000e-03 +
dpvth0_p_18_mm
lnlx=-1.5840e-08
+ pua=5.8550e-11
wub=0.0000e+00
lub=0.0000e+00
+ pvoff=-9.8330e-05
wa0=-4.8070e-02 la0=-
2.8100e-01
+ pketa=0.0000e+00
wute=-2.6820e-01
lute=0.0000e+00
+ cjswg=4.200e-10 +
dcjgate_p_18_mm
wat=-6.4050e+03
+ ptp=1.2400e-03
ldif=8.0000e-08
rsh=8.0000e+00
+ noimod=2
noia=3.57456993317604E+18
noib=2500

Cadence 3.3V NMOS Model file

```
**********************************************************************
* 0.18um 1.8v/3.3v MIXEDMODE Twin-well technology, 3.3V NMOS Model *
**********************************************************************
.model N_33_MM NMOS
*****Model Selectors/Controllers**********************************
+ LEVEL = 4.9000E+01 VERSION = 3.1000E+00
+ MOBMOD = 1.0000E+00 BINUNIT = 2.0000E+00
+ CAPMOD = 2.0000E+00 NQSMOD = 0.0000E+00
*****Process Parameters*******************************************
+ TOX = '7.0000E-09+DTOX_N_33_MM' XJ = 2.5000E-07
+ NCH = 1.7000E+17 RSH = 8.0000E+00
*****Basic Model Parameters***************************************
+ LINT = '0.0000E+00+DLINT_N_33_MM'
+ WINT = '0.0000E+00+DWINT_N_33_MM'
+ DWG = 3.5527E-15 DWB = 6.2000E-09
+ VTH0 = '5.9241E-01+DVTH0_N_33_MM' K1 = 7.1375E-01
+ K2 = -2.5827E-02 K3 = 6.2000E+01
+ DVT0 = 4.7502E+00 DVT1 = 6.3402E-01
+ DVT2 = -1.1392E-01 DVT0W = 1.5000E-01
+ DVT1W = 5.9500E+05 DVT2W = 0.0000E+00
+ NLX = 1.2508E-07 W0 = 5.6800E-06
+ K3B = 3.2000E+01 VSAT = 9.8280E+04
+ UA = -5.8547E-10 UB = 1.6519E-18
+ UC = 5.5177E-11 U0 = 3.4130E-02
+ A0 = 1.2753E+00 KETA = 5.0000E-03
+ A1 = 1.0000E-02 A2 = 9.9000E-01
+ AGS = 2.6785E-01 B0 = 1.0000E-08
+ B1 = 9.0000E-07 VOFF = -1.3609E-01
+ NFACTOR = 1.0000E+00 CIT = 0.0000E+00
+ CDSC = 2.4000E-04 CDSCB = 0.0000E+00
+ CDSCD = 2.4000E-04 ETA0 = 1.0808E-01
+ ETAB = -1.5000E-03 DSUB = 5.2100E-01
+ PCLM = 6.3180E-01 PDIBLC1 = 9.4999E-04
+ PDIBLC2 = 1.0000E-03 PDIBLCB = 0.0000E+00
```

+ DROUT = 5.6000E-01 PSCBE1 = 4.2400E+08
+ PSCBE2 = 1.0000E-05 PVAG = 0.0000E+00
+ DELTA = 1.0000E-02
*****Parameters for Asymmetric and Bias-Dependent Rds Model******
+ RDSW = 5.9591E+02 PRWB = 5.7155E-02
+ PRWG = 2.3069E-02 WR = 1.0000E+00
*****Impact Ionization Current Model Parameters******************
+ ALPHA0 = 0.0000E+00 ALPHA1 = 0.0000E+00
+ BETA0 = 3.0000E+01
*****Gate-Induced Drain Leakage Model Parameters*****************
*****Gate Dielectric Tunneling Current Model Parameters**********
*****Charge and Capacitance Model Parameters**********************
+ CGDO = '2.7000E-10+DCGDO_N_33_MM'
+ CGSO = '2.7000E-10+DCGSO_N_33_MM'
+ CGBO = 1.0000E-10 XPART = 1.0000E+00
+ CF = 1.3920E-10

Cadence 3.3V PMOS Model file

```
**********************************************************************

* 0.18um 1.8v/3.3v MIXEDMODE Twin-well technology, 3.3V PMOS Model *

**********************************************************************

.model P_33_MM PMOS
*****Model Selectors/Controllers**********************************
+ LEVEL = 4.9000E+01 VERSION = 3.1000E+00
+ MOBMOD = 1.0000E+00 CAPMOD = 2.0000E+00
+ NQSMOD = 0.0000E+00
*****Process Parameters*******************************************
+ TOX = '7.0000E-09+DTOX_P_33_MM' XJ = 1.8000E-07
+ NCH = 4.3096E+17 RSH = 8.0000E+00
+ RSH = 8.0000E+00
*****Basic Model Parameters***************************************
+ LINT = '2.2396E-08+DLINT_P_33_MM'
+ WINT = '3.3613E-08+DWINT_P_33_MM'
+ DWG = -1.4371E-08 DWB = 5.9839E-09
+ VTH0 = '-7.2044E-01+DVTH0_P_33_MM' K1 = 7.6674E-01
+ K2 = 5.4287E-02 K3 = 2.0893E+00
+ DVT0 = 7.3947E+00 DVT1 = 9.6349E-01
+ DVT2 = 7.8053E-03 DVT0W = 1.7970E+00
+ DVT1W = 2.8763E+06 DVT2W = 1.0262E-01
+ NLX = 0.0000E+00 W0 = 2.1247E-07
+ K3B = 1.0028E+00 VSAT = 1.0008E+05
+ UA = -3.9783E-10 UB = 1.6947E-18
+ UC = -3.6495E-11 U0 = 8.0759E-03
+ A0 = 1.0736E+00 KETA = -1.0347E-02
+ A1 = 1.0000E-02 A2 = 9.2123E-01
+ AGS = 1.1384E-01 B0 = 4.3581E-07
+ B1 = 1.7379E-06 VOFF = -1.2391E-01
+ NFACTOR = 9.9641E-01 CIT = 5.3337E-04
+ CDSC = 3.1657E-03 CDSCB = 2.4630E-03
+ CDSCD = 0.0000E+00 ETA0 = 5.4211E-02
+ ETAB = -1.8932E-02 DSUB = 4.5943E-01
+ PCLM = 6.1644E-01 PDIBLC1 = 1.1236E-02
+ PDIBLC2 = 3.5434E-04 PDIBLCB = 1.4779E-01
+ DROUT = 1.4014E-01 PSCBE1 = 4.6428E+08
+ PSCBE2 = 2.8700E-05 PVAG = -1.3070E-01
+ DELTA = 1.0000E-02
*****Parameters for Asymmetric and Bias-Dependent Rds Model******
+ RDSW = 6.1702E+02 PRWB = -7.7253E-01
+ PRWG = 3.3490E-02 WR = 9.9132E-01
```

*****Impact Ionization Current Model Parameters**********
+ ALPHA0 = 0.0000E+00 BETA0 = 3.0000E+01
*****Gate-Induced Drain Leakage Model Parameters**********
*****Gate Dielectric Tunneling Current Model Parameters**********
*****Charge and Capacitance Model Parameters**********
+ CGDO = '2.6000E-10+DCGDO_P_33_MM'
+ CGSO = '2.6000E-10+DCGSO_P_33_MM'
+ CGBO = 1.0000E-10 XPART = 1.0000E+00
+ CF = 1.3920E-10