

# **AN EFFICIENT PERFORMANCE ANALYSIS OF UT SUTRA ARCHITECTURE USING 32nm CMOS TECHNOLOGY**

Submitted in partial fulfillment of requirements for the award of

Bachelor of Engineering in

Electronics and Communication Engineering

**VANAMU LALIK (39130497)**

**THOTA LAVANYA (39130475)**



**DEPERTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

**SCHOOL OF ELECTRICAL AND ELECTRONICS**

## **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY  
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI – 600119**

**APRIL – 2023**



# **SATHYABAMA**

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE**

**www.sathyabama.ac.in**

## **DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING**

### **BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of THOTA LAVANYA (39130475) and VANAMU LALIK (39130497) who carried out the project entitled "**AN EFFICIENT PERFORMANCE ANALYSIS OF UT SUTRA ARCHITECTURE USING 32nm CMOS TECHNOLOGY**" under our supervision from Nov 2022 to April 2023.

**Internal Guide**

Ms. Mary Sajin Sanju I, M.Tech., Ph.D.

Department of ECE

**Head of the Department**

Dr.T.RAVI M.E., Ph.D.

---

**Submitted for Viva voce Examination held on** 19-04-2023

**Internal Examiner**

**External Examiner**

## DECLARATION

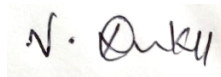
We, VANAMU LALIK(39130497) and THOTA LAVANYA (39130475) hereby declare that the Project Report entitled **“AN EFFICIENT PERFORMANCE ANALYSIS OF UT SUTRA ARCHITECTURE USING 32nm CMOS TECHNOLOGY”** done by us under the guidance of **Ms MARY SAJIN SANJU I, M.Tech., Ph.D,** Assistant Professor is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Electronics and Communication Engineering.

**DATE:** 19-04-2023

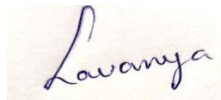
**PLACE:** CHENNAI

**SIGNATURE OF THE CANDIDATES**

1.

Handwritten signature of N. Dinku in black ink.

2.

Handwritten signature of Lavanya in blue ink.

## ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to Board of Management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr.N.M.NANDHITHA, M.E., Ph.D.**, Dean, School of Electronics and Communication Engineering and **Dr.T.RAVI M.E., Ph.D.** Head of the Department, Department of **Electronics and Communication Engineering** for providing me necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Ms.Mary Sajin Sanju I,M.Tech., Ph.D.** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of **Electronics and Communication Engineering** who were helpful in many ways for the completion of the project.

## ABSTRACT

Describes a design architecture performs mathematical operations using Vedic sutra for upward compatibility in pipeline manner. In spite of increasing the area, performance and reducing power, Vedic architecture have observed to be inherently compatible with higher efficiency for pipeline architecture. However Vedic architecture leads to additional flexibility starting from 2-bit base modules to 8-bit modules ( $L=1,2,3$ ) and pipeline can be compactable to any base modules for any given length. Many researchers proposed arithmetic algorithms at simulation level using vedic sutra. These algorithms have been evaluated with better performance, area and speed. The literature has been widely found to be towards individual arithmetical operators like multiplier, square and cube and so on. A consolidated computing architecture, especially Nbit ALU is yet be realized Generalized N-bit ALU, can always be realized using pipeline modular architecture. The proposition is on realizing N-bit using '2L ' as base modules using 'K' modules in pipelining. The authors have extensively verified modular architecture for 4-Bit modules for 16 bit and 32 bit pipelined operations. Individually multiplication using Urdhva Tiryakbhyam, division using Dhvajanka sutra, square using Dwandwayoga sutra. MAC unit which involves multiplication algorithms used in FFT and IFFT using sutras of Vedic mathematics and it is possible to achieve reduce version interms of speed and delay, compared to different generations of ALU The authors are now exploring N-bit ALU architecture FPGA implementation using Vedic sutras with flexible modular pipeline architecture and mainly targeted for Digital Signal processing applications

## **TABLE OF CONTENTS**

| <b>CHAPTER No.</b> | <b>TITLE</b>                  | <b>PAGE No.</b> |
|--------------------|-------------------------------|-----------------|
|                    | <b>ABSTRACT</b>               | <b>V</b>        |
|                    | <b>LIST OF TABLES</b>         | <b>IX</b>       |
|                    | <b>LIST OF FIGURES</b>        | <b>IX</b>       |
| <b>1</b>           | <b>INTRODUCTION</b>           | <b>1</b>        |
|                    | 1.1 INTRODUCTION              | 1               |
|                    | 1.2 ARRAY MULTIPLIER          | 2               |
|                    | 1.3 VEDIC MULTIPLIER          | 2               |
|                    | 1.4 FGPA                      | 3               |
|                    | 1.5 MOS TECHNOLOGY            | 3               |
|                    | 1.6 CMOS TECHNOLOGY           | 4               |
|                    | 1.6.1 NMOS                    | 4               |
|                    | 1.6.2 PMOS                    | 5               |
|                    | 1.6.3 CMOS                    | 5               |
|                    | 1.6.4 CMOS LOGIC GATES        | 6               |
| <b>2</b>           | <b>LITERATURE SURVEY</b>      | <b>8</b>        |
|                    | 2.1 LITERATURE SURVEY         | 8               |
|                    | 2.2 LITERATURE SUMMARY        | 13              |
|                    | 2.3 PROBLEM DEFINITION        | 13              |
|                    | 2.4 OBJECTIVES OF THE PROJECT | 13              |
| <b>3</b>           | <b>EXISTING SYSTEM</b>        | <b>15</b>       |

|          |                                            |           |
|----------|--------------------------------------------|-----------|
|          | 3.1 8X8 BIT MULTIPLIER                     | 15        |
|          | 3.2 PIPELINE ARCHITECTURE                  | 15        |
|          | 3.3 ADVANTAGES OF PIPELINE ARCHITECTURE    | 16        |
|          | 3.4 DISADVANTAGES OF PIPELINE ARCHITECTURE | 16        |
| <b>4</b> | <b>PROPOSED SYSTEM</b>                     | <b>17</b> |
|          | <b>REFERENCES</b>                          | <b>18</b> |

### **LIST OF TABLES**

| <b>TABLE NO.</b> | <b>TITLE</b>                                                           | <b>PAGE No.</b> |
|------------------|------------------------------------------------------------------------|-----------------|
| 5.1              | PERFORMANCE EVOLUTION OF POWER, DELAY AND POWER DELAY PRODUCT          | V               |
| 5.2              | PERFORMANCE EVOLUTION OF POWER, DELAY AND POWER DELAY PRODUCT IN CNFET |                 |

### **LIST OF FIGURES**

| <b>CHAPTER No.</b> | <b>TITLE</b>                                                     | <b>PAGE No.</b> |
|--------------------|------------------------------------------------------------------|-----------------|
| <b>1.1</b>         | MOS Transistor Symbols                                           | <b>V</b>        |
| <b>1.2</b>         | NMOS transistor                                                  |                 |
| <b>1.3</b>         | PMOS Transistor                                                  |                 |
| <b>1.4</b>         | CMOS logic gate                                                  |                 |
| <b>1.5</b>         | CMOS Inverter                                                    |                 |
| <b>1.6</b>         | Description of the multiplication process by Urdhava Triyakbhyam |                 |
| <b>1.7</b>         | CMOS NAND Gate                                                   |                 |
| <b>1.8</b>         | HALF ADDER                                                       |                 |

|             |                                              |
|-------------|----------------------------------------------|
| <b>1.9</b>  | 2-BIT HALF ADDER TRUTH TABLE                 |
| <b>1.10</b> | LOGIC CIRCUIT FOR HALF ADDER                 |
| <b>1.11</b> | FULL ADDER                                   |
| <b>1.12</b> | TRUTH TABLE OF FULL ADDER                    |
| <b>1.13</b> | LOGIC DIARAM OF FULL ADDER                   |
| <b>3.1</b>  | 8 X 8 MULTIPLIER                             |
| <b>3.2</b>  | PIPELINE ARCHITECTURE                        |
| <b>4.1</b>  | HSPICE SOFTWARE                              |
| <b>4.2</b>  | SELECTION OF INPUTS AND OUTPUTS              |
| <b>4.3</b>  | SIMULATED SCREEN                             |
| <b>4.4</b>  | COSMOSCOPE SOFTWARE                          |
| <b>5.1</b>  | SIMULATION RESULT OF AND GATE                |
| <b>5.2</b>  | SIMULATION RESULT OF NAND GATE               |
| <b>5.3</b>  | SIMULATION RESULT OF OR GATE                 |
| <b>5.4</b>  | SIMULATION RESULT OF NOR GATE                |
| <b>5.5</b>  | SIMULATION RESULT OF XOR GATE                |
| <b>5.6</b>  | SIMULATION RESULT OF HALF ADDER              |
| <b>5.7</b>  | SIMULATION RESULT OF FULL ADDER              |
| <b>5.8</b>  | SIMULATION RESULT OF 2-bit RCA               |
| <b>5.9</b>  | SIMULATION RESULT OF 4-bit RCA               |
| <b>5.10</b> | SIMULATION RESULT OF 8-bit RCA               |
| <b>5.11</b> | SIMULATION RESULT OF 2x2 VEDIC<br>MULTIPLIER |
| <b>5.12</b> | SIMULATION RESULT OF 4x4 VEDIC<br>MULTIPLIER |
| <b>5.13</b> | SIMULATION RESULT OF 8x8 VEDIC<br>MULTIPLIER |



# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 INTRODUCTION**

Very large-scale integration was made possible with the wide adoption of the MOS transistor, originally invented by Mohamed M. Atalla and Dawon Kahng at Bell Labs in 1959. Atalla first proposed the concept of the MOS integrated circuit chip in 1960, followed by Kahng in 1961, both noting that the MOS transistor's ease of fabrication made it useful for integrated circuits. General Microelectronics introduced the first commercial MOS integrated circuit in 1964. In the early 1970s, MOS integrated circuit technology allowed the integration of more than 10,000 transistors in a single chip. This paved the way for VLSI in the 1970s and 1980s, with tens of thousands of MOS transistors on a single chip (later hundreds of thousands, then millions, and now billions).

The first semiconductor chips held two transistors each. Subsequent advances added more transistors, and as a consequence, more individual functions or systems were integrated over time. The first integrated circuits held only a few devices, perhaps as many as ten diodes, transistors, resistors and capacitors, making it possible to fabricate one or more logic gates on a single device. Now known retrospectively as small-scale integration (SSI), improvements in technique led to devices with hundreds of logic gates, known as medium-scale integration (MSI). Further improvements led to large-scale integration (LSI), i.e. systems with at least a thousand logic gates. Current technology has moved far past this mark and today's microprocessors have many millions of gates and billions of individual transistors.

At one time, there was an effort to name and calibrate various levels of large-scale integration above VLSI. Terms like ultra-large-scale integration (ULSI) were used. But the huge number of gates and transistors available on common devices has rendered such fine distinctions moot. Terms suggesting greater than VLSI levels of integration are no longer in wide spread use.

In 2008, billion-transistor processors became commercially available. This became more commonplace as semiconductor fabrication advanced from the then- current generation of 65 nm processes. Current designs, unlike the earliest devices, use extensive design automation and automated logic synthesis to lay out the transistors, enabling higher levels of complexity in the resulting logic functionality. Certain high- performance logic blocks like the SRAM (static random-access memory) cell, are still designed by hand to ensure the highest efficiency.

## **1.2 ARRAY MULTIPLIER**

An array multiplier is a digital combinational circuit used for multiplying two binary numbers by employing an array of full adders and half adders. This array is used for the nearly simultaneous addition of the various product terms involved. To form the various product terms, an array of AND gates is used before the Adder array. Checking the bits of the multiplier one at a time and forming partial products is a sequential operation that requires a sequence of add and shift micro-operations. The multiplication of two binary numbers can be done with one micro-operation by means of a combinational circuit that forms the product bits all at once. This is a fast way of multiplying two numbers since all it takes is the time for the signals to propagate through the gates that form the multiplication array. However, an array multiplier requires a large number of gates, and for this reason it was not economical until the development of integrated circuits.

## **1.3 VEDIC MULTIPLIER**

Vedic Multiplier is a key tool in rapidly growing technology especially in the immense domain of Image processing, Digital Signal Processing, real-time signal. Multipliers are important block in digital systems and play a critical role in digital designs. Along with accuracy demand for minimizing time area, power, and delay of the processor by enhancing speed is the focus point. Vedic mathematics rules and Algorithms generate partial products concurrently and save time. This paper is a review of the application and modification of Vedic multiplier in different fields and a

comparison of Vedic multiplier with other multipliers for enhancing performance parameters.

The word “Urdhva-Tiryakbhyam” resources vertical and crosswise multiplication. This multiplication formula is pertinent to all cases of algorithm for N bit numbers. Conventionally this sutra is used for the multiplication of two numbers in decimal number system. The same concept can be applicable to binary number system which is being discussed in this paper. Advantage of using this type of multiplier is that as the number of bits increases, delay and area increases very slowly as compared to other multipliers.

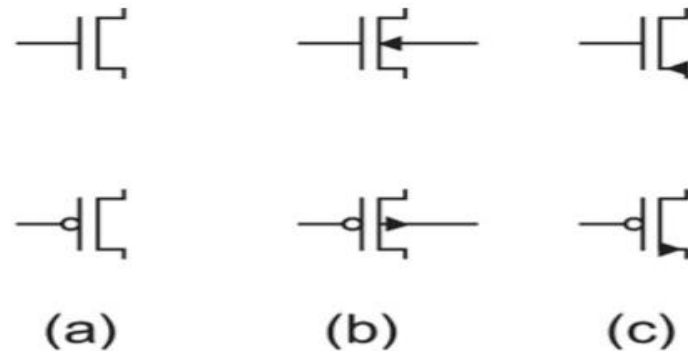
#### **1.4 FPGA**

Field Programmable Gate Array (FPGA) Fully fabricated FPGA chips containing thousands or even more, of logic gates with programmable interconnects, are available to users for their custom hardware programming to realize desired functionality. This design style provides a means for fast prototyping and also for cost-effective chip design, especially for low-volume applications. A typical field programmable gate array (FPGA) chip consists of I/O buffers, an array of configurable logic blocks (CLBs), and programmable interconnect structures. The programming of the interconnects is accomplished by programming of RAM cells whose output terminals are connected to the gates of MOS pass transistors. Thus, the signal routing between the CLBs and the I/O blocks is accomplished by setting the configurable switch matrices accordingly.

#### **1.5 MOS TECHNOLOGY**

MOS technology is considered as one of the very important and promising technologies in the VLSI design process. The circuit designs are realized based on pMOS, nMOS, CMOS devices. The pMOS devices are based on the p-channel MOS transistors. Specifically, the pMOS channel is part of a n-type substrate lying between two heavily doped p+ wells beneath the source and drain electrodes. Generally speaking, a pMOS transistor is only constructed in consort with an NMOS transistor. The nMOS technology and design processes provide an excellent background for other technologies. In particular, some familiarity with nMOS allows a relatively easy transition to CMOS technology and design.

MOS Transistor Symbol:



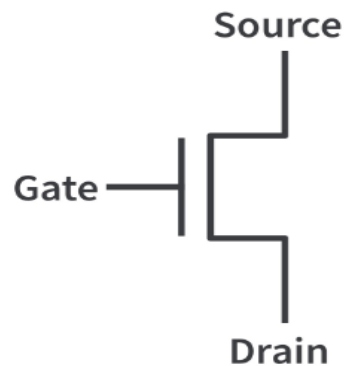
**Fig 1.1 MOS Transistor Symbols**

## 1.6 CMOS TECHNOLOGY

CMOS technology is a predominant technology for manufacturing integrated circuits. CMOS stands for “Complementary Metal Oxide Semiconductor”. Microprocessors, batteries, and digital sensors among other electronic components make use of this technology due to several key advantages. This technology uses both NMOS and PMOS to realize various logic functions. Both N and P MOSFET channels are designed to have matching characteristics. Before CMOS, PMOS and NMOS logic were widely used for implementing logic gates. PMOS was then replaced by the NMOS Technology, which used to be the standard IC fabrication technology. Initially, CMOS was slower and more expensive than NMOS. The main advantages of NMOS technology are simple physical process, functional density, processing speed and manufacturing efficiency. The main disadvantages of NMOS technology are its electrical asymmetry and static power dissipation. These drawbacks are minimized by using CMOS Technology. The main advantage of CMOS is the minimal power dissipation as this only occurs during circuit switching. This results in much better performance as it allows integrating more CMOS gates on an IC.

### 1.6.1 NMOS

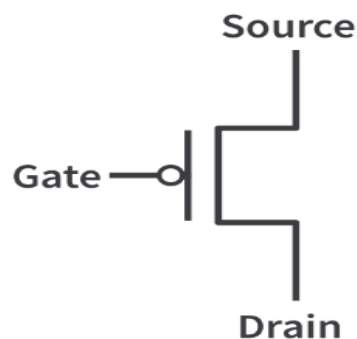
N-channel MOSFET consists of an N-type source and drain diffused on a P-type substrate. The majority carriers are electrons. When the applied voltage to the gate is high enough, the NMOS will conduct; otherwise, it will not. Since the majority carriers (electrons) travel faster than holes, NMOS are considered to be faster than PMOS.



**Fig 1.2: NMOS transistor**

### **1.6.2 PMOS**

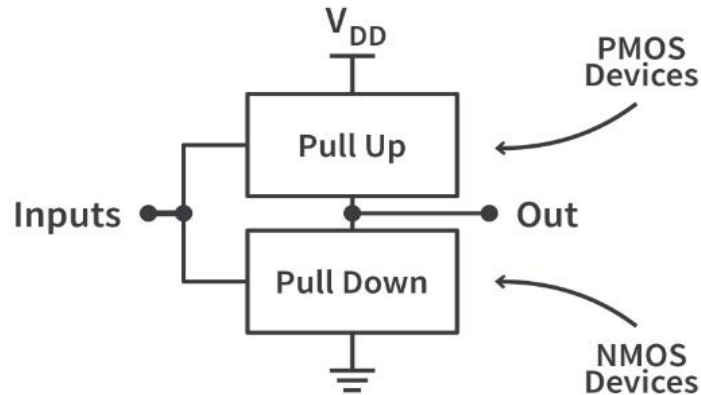
P-channel MOSFET also has a Source and Drain diffused on a substrate. The Source is P-type while the substrate is N-type. The majority carriers are holes. PMOS will conduct when a low voltage is applied. When a high voltage is applied to the gate, the PMOS will not conduct.



**Fig 1.3: PMOS Transistor**

### **1.6.3 CMOS**

Since CMOS technology uses both N-type and P-type transistors to design logic functions, a signal which turns ON a transistor type is used to turn OFF the other transistor type. This eliminates the need for pull-up resistors in favour of simple switches. In CMOS logic gates N-type MOSFETs are arranged in a pull-down network between the output and the low voltage supply rail ( $V_{SS}$  or ground) while P-type MOSFETs are in a pull-up network between the output and the higher-voltage rail (often  $V_{DD}$ ). Thus, the N-type MOSFET will be ON when the P-type MOSFET is OFF, and vice-versa. For any input pattern, one of the networks is ON and the other is OFF.



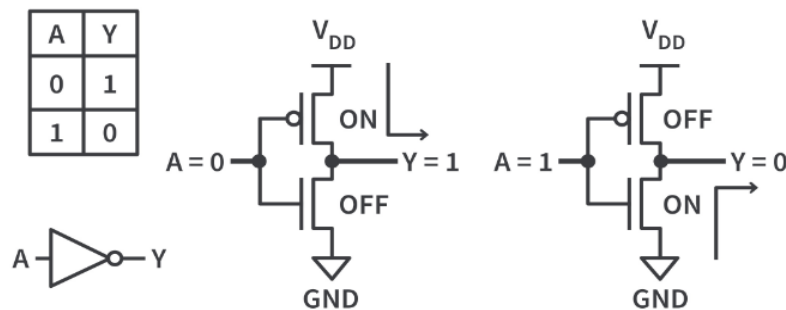
**Fig 1.4 CMOS logic gate**

CMOS advantages are high speed, low power dissipation, high noise margins in both states, and a wide range of source and input voltages (fixed source voltage).

#### **1.6.4 CMOS Logic Gates**

##### **1) CMOS Inverter**

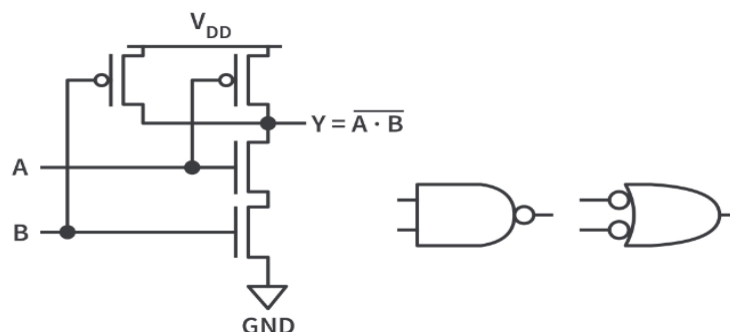
The CMOS inverter is the simplest CMOS logic gate. The circuit consists of PMOS and NMOS FET. Input A serves as the gate voltage for both transistors while Y is the output.



**Fig 1.5: CMOS Inverter**

The NMOS transistor has an input from  $V_{SS}$  or ground and the PMOS transistor has an input from  $V_{DD}$ . When the input (A) is low ( $<V_{DD}$ ,  $\sim 0$  V, Logic 0), the NMOS is OFF while the PMOS is ON.  $V_{DD}$  will appear at the output through the P-channel MOSFET path. Hence, there is output (Logic 1) with the circuit pulled up to  $V_{DD}$ . When the input is high ( $\sim V_{DD}$ , Logic 1), the PMOS is OFF while the NMOS is ON. The output is pulled down and is therefore low (Logic 0).

A 2-input NAND gate has two N-channel MOSFETs connected in series between Y (output) and GND and two P-channel MOSFETs connected in parallel between  $V_{DD}$  and Y.

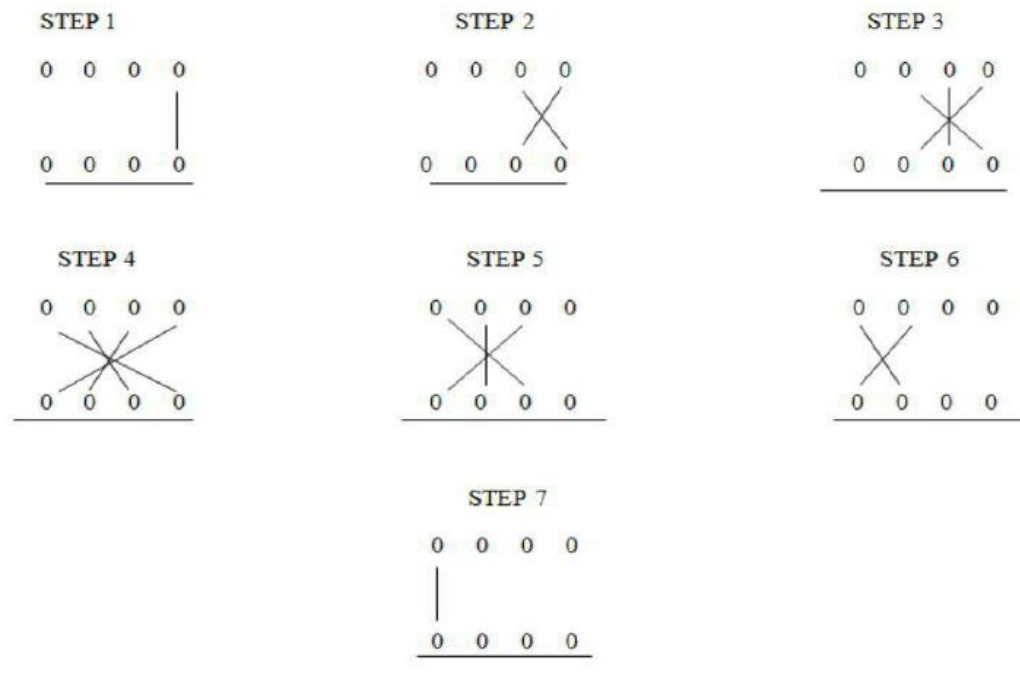


**Fig 1.6: CMOS NAND Gate**

If either A or B is low (Logic 0), at least one of the NMOS transistors will be OFF. This breaks the path from Y to GND since the NMOS transistors are connected in series. But in this case at least one of the PMOS transistors is ON, completing a path from Y to  $V_{DD}$ . This makes the output Y high (Logic 1). For Y to be low, both A and B should be high to ensure that both NMOS transistors are ON so that the path from Y to GND is complete. For all the other combinations of the inputs, Y will be high. The truth table of NAND logic gate is given below.

### **1.6.5 URDHVA TRIYAKBHYAM (U-T Sutra)**

The multiplier principle is based on the Urdhva-Tiryakbhyam(UT) algorithm of ancient Indian-Vedic mathematics. In either case, Urdhva-Tiryakbhyam Sutra is a broad variety of multiplication formulas. This basically means 'vertical and cross-sectional.' It is founded on the idea that all partial products can be generated simultaneously by adding other partial products. The parallelism of the development and summation of the partial product is described in Figure, using Urdhava-Tiryakbhyam. The algorithm can be generalized to a bit size of  $n \times n$ . Because the device properties are calculated in addition to their quantity, the multiplier is independent of the clock frequency of the processor. The multiplier then carries the output variable independently and is thus independent of the clock frequency.



**Fig 1.7: Description Of The Multiplication Process By Urdhava-Triyakbhyam**

## 1.8 ADDERS

An adder or summer is a digital circuit that performs addition of numbers. In many computers and other kinds of processors, adders are used not only in the arithmetic logic units, but also in other parts of the processor, where they are used to calculate addresses, table indices, increment and decrement operators, and similar operations.

Although adders can be constructed for many numerical representations, such as binary-coded decimal or excess-3, the most common adders operate on binary numbers. In cases where two's complement or ones' complement is being used to represent negative numbers, it is trivial to modify an adder into an adder subtractor. Other signed number representations require more logic around the basic adder.

Binary carry-propagate adders have been extensively published, heavily attacking problems related to carry chain problem. Binary adders evolve from linear adders, which have a delay approximately proportional to the width of the adder, e.g. ripple-carry adder (RCA), to logarithmic - delay adders, such as the carry-look ahead adder (CLA). There are some additional performance enhancing schemes, including the carry-increment adder and the Ling adder that can further enhance the carry chain,

however, in Very Large Scale Integration (VLSI) digital systems, the most efficient way of offering binary addition involves utilizing parallel-prefix trees, this occurs because they have the regular structures that exhibit logarithmic delay. Parallel-prefix adders compute addition in two steps: one to obtain the carry at each bit, with the next to compute the sum bit based on the carry bit. Unfortunately, prefix trees are algorithmically slower than fast logarithmic adders, such as the carry propagate adders, however, their regular structures promote excellent results when compared to traditional CLA adders.

This happens within VLSI architectures because a carry-look ahead adder, such as the one implemented in one of Motorola's processors, tends to implement the carry chain in the vertical direction instead of a horizontal one, which has a tendency to increase both wire density and fan-in/out dependence. Therefore, although logarithmic adder structures are one of the fastest adders algorithmically, the speed efficiency of the carry-look ahead adder has been hampered by diminishing returns given the fan-in and fan-out dependencies as well as the heavy wire load distribution in the vertical path. In fact, a traditional carry-look ahead adder implemented in VLSI can actually be slower than traditional linear-based adders, such as the Manchester carry adder.

### 1.8.1 HALF ADDER

So, coming to the scenario of half adder, it adds two binary digits where the input bits are termed as augend and addend and the result will be two outputs one is the sum and the other is carry. To perform the sum operation, XOR is applied to both the inputs, and AND gate is applied to both inputs to produce carry.



***Fig 1.8 Half Adder***

Whereas in the full adder circuit, it adds 3 one-bit numbers, where two of the three bits can be referred to as operands and the other is termed as bit carried in. The produced output is 2-bit output and these can be referred to as output carry and sum. By using a half adder, you can design simple addition with the help of logic gates. Let's see an example of adding two single bits.

The 2-bit half adder truth table is as below:

| INPUTS |   | OUTPUTS |       |
|--------|---|---------|-------|
| A      | B | SUM     | CARRY |
| 0      | 0 | 0       | 0     |
| 0      | 1 | 1       | 0     |
| 1      | 0 | 1       | 0     |
| 1      | 1 | 0       | 1     |

***Fig 1.9 2-Bit Half Adder Truth Table***

$$0+0=0$$

$$0+1 = 1$$

$$1+0 = 1$$

$$1+1 = 10$$

These are the least possible single-bit combinations. But the result for 1+1 is 10, the sum result must be re-written as a 2-bit output. Thus, the equations can be written as

$$0+0=00$$

$$0+1 = 01$$

$$1+0 = 01$$

$$1+1 = 10$$

The output '1' of '10' is carry-out. 'SUM' is the normal output and 'CARRY' is the carry-out. Now it has been cleared that a 1-bit adder can be easily implemented with the help of the XOR Gate for the output 'SUM' and an AND Gate for the 'Carry'. For instance, when we need to add, two 8-bit bytes together, then it can be implemented

by using a full-adder logic circuit. The half-adder is useful when you want to add one binary digit quantities.

A way to develop two-binary digit adders would be to make a truth table and reduce it. When you want to make a three binary digit adder, the half adder addition operation is performed twice. In a similar way, when you decide to make a four-digit adder, the operation is performed one more time. With this theory, it was clear that the implementation is simple, but development is a time taking process.

The simplest expression uses the exclusive OR function:

$$\text{Sum} = A \text{ XOR } B$$

$$\text{Carry} = A \text{ AND } B$$



**Fig 1.10 Logic Circuit For Half Adder**

And an equivalent expression in terms of the basic AND, OR, and NOT is:

$$\text{SUM} = A.B + A.B'$$

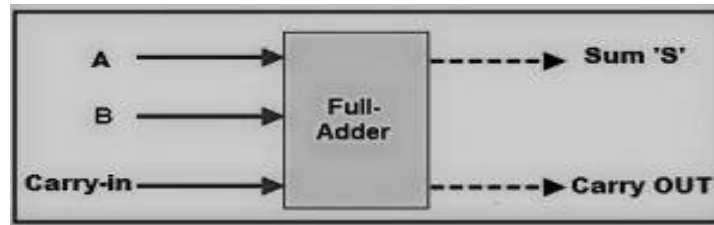
The implementation of half adder can be done through high-speed

CMOS digital logic integrated circuits like 74HCxx series which includes the SN74HC08 (7408) & SN74HC86 (7486).

The main reason to call these binary adders like Half Adders is, that there is no range to include the carry bit using an earlier bit. So, this is a main limitation of HAs once used like binary adder particularly in real-time situations which involve adding several bits. So this limitation can be overcome by using the full adders.

### 1.8.2 FULL ADDER

This adder is difficult to implement when compared to half-adder.



**Fig 1.11 FULL ADDER**

The full-adder has three inputs and two outputs, whereas half adder has only two inputs and two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. When a full-adder logic is designed, you string eight of them together to create a byte-wide adder and cascade the carry bit from one adder to the next.

| INPUTS |   |      | OUTPUT |   |
|--------|---|------|--------|---|
| A      | B | C-IN | C-OUT  | S |
| 0      | 0 | 0    | 0      | 0 |
| 0      | 0 | 1    | 0      | 1 |
| 0      | 1 | 0    | 0      | 1 |
| 0      | 1 | 1    | 1      | 0 |
| 1      | 0 | 0    | 0      | 1 |
| 1      | 0 | 1    | 1      | 0 |
| 1      | 1 | 0    | 1      | 0 |
| 1      | 1 | 1    | 1      | 1 |

**Fig 1.12 TRUTH TABLE OF FULL ADDER**

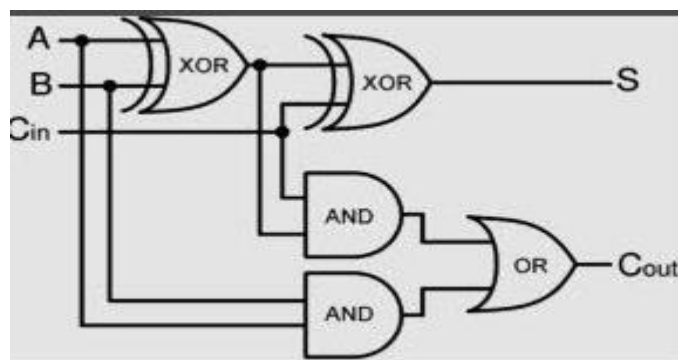
The difference between a half-adder and a full-adder is that the output carry is designated as C-OUT and the normal output is represented as S which is 'SUM'.

With the above full adder truth-table, the implementation of a full adder circuit can be understood easily. The SUM 'S' is produced in two steps: By XORing the provided inputs 'A' and 'B'. The result of A XOR B is then XORed with the C-IN.

This generates SUM and C-OUT is true only when either two of three inputs are HIGH, then the C-OUT will be HIGH. So, we can implement a full adder circuit with the help of two half adder circuits. Initially, the half adder will be used to add A and B to produce a

partial Sum and a second-half adder logic can be used to add C-IN to the Sum produced by the first half adder to get the final S output.

If any of the half adder logic produces a carry, there will be an output carry. So, C-OUT will be an OR function of the half-adder Carry outputs. Take a look at the implementation of the full adder circuit shown below.



**Fig 1.13 LOGIC DIAGRAM OF FULL ADDER**

The implementation of larger logic diagrams is possible with the above full adder logic a simpler symbol is mostly used to represent the operation. Given below is a simpler schematic representation of a one-bit full adder.

With this type of symbol, we can add two bits together, taking a carry from the next lower order of magnitude, and sending a carry to the next higher order of magnitude. In a computer, for a multi-bit operation, each bit must be represented by a full adder and must be added simultaneously. Thus, to add two 8-bit numbers, you will need 8 full adders which can be formed by cascading two of the 4-bit blocks.

## **CHAPTER 2**

### **LITERATURE SURVEY**

This literature survey discusses about the designing and implementation architecture that performs mathematical operations using Vedic sutra for upward compatibility in pipeline manner

#### **2.1 LITERATURE SURVEY**

Vedic mathematics or ancient mathematics is a unique technique of calculations based on 16 sutras. It provides an innovative way of computation of almost all the mathematical operations. In this era of digitization, engineers are working on increase speed of the digital circuits while reducing the size and power consumed. Arithmetic operations are the basic units of all the digital circuitry and hence optimizing these unit increases efficiency of the entire digital design. Unlike conventional mathematics, Vedic math provides different techniques to compute basic arithmetic operations. Vedic math reduces the computational steps required to achieve the result. Designers have implemented many computer architectures based on Vedic math. In this we review these architectures as well as several extended work in the area. In addition, we also review several state-of-art applications that take full advantage of such simple ancient Vedic Mathematical technique.

In spite of increasing the area, performance and reducing power, Vedic architecture have observed to be inherently compatible with higher efficiency for pipeline architecture. Many researchers proposed arithmetic algorithms at simulation level using vedic sutra. These algorithms have been evaluated with better performance, area and speed. The literature has been widely found to be towards individual arithmetical operators like multiplier, square and cube and so on. Individually multiplication using Urdhva Tiryakbhyam, division using Dhvajanka sutra, square using Dwandwayoga sutra. MAC unit which involves multiplication algorithms used in FFT and IFFT using sutras of Vedic mathematics and it is possible to achieve reduce version in terms of speed and delay, compared to different generations of ALU.

The pipeline architecture ( $N = 2L * K$ ) has been successfully implemented by the authors in their earlier work on 32 bit MAC unit used for DSP applications. Hardware implementation of the Arithmetic unit has been verified on Spartan 3E Board/Basys3(artix\_7) family. For the Xilinx Spartan and Artix family, it is found that the gate delay significantly reduces by using Vedic math sutras. The present work proposes a flexible, modular and pipeline facilitation for a generic arithmetic unit to be used for any application.

Malti Bansal, Jasmeet Singh, **“Comparative Analysis of 4-bit CMOS Vedic Multiplier and GDI Vedic Multiplier using 18nm FinFET Technology”**, Proceedings of the International Conference on Smart Electronics and Communication (ICOSEC 2020). Today's modern market needs application computational blocks to execute complex operations with an ease to satisfy low power requirements, wherein relentless efforts are being made by the researchers to optimize these computational blocks in terms of speed, power, and area requirements. The core key component in these computational blocks is Multipliers. Since, these multiplier circuits directly effects the performance of overall computation block or overall system, where there is a need to optimize the multiplier circuit. Since, it is already known that the multiplication requires several steps and is a lengthy process, the algorithm of Vedic multiplication is used to speed-up the multiplication process. Moreover, by implementing the Vedic multiplier circuit with the GDI technique, there will be a further reduction in number of transistors and propagation delay. So, here in this paper the Vedic multiplier circuit is implemented by using GDI technique and also 18nm FinFET is deployed for analyzing simulation results. Here, the primary objective is to optimize the proposed multiplier circuitry, where an experimental analysis is conducted by comprising the parameters such as propagation Delay, Power, Area and Power-Delay-Product (PDP). All experimental analysis is done using Cadence Virtuoso.

CHANDRASHEKARA M N , ROHITH S **“Design of 8 Bit Vedic Multiplier Using Urdhva Tiryagbhyam Sutra With Modified Carry Save Adder ”**, 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT-2019), MAY 17th & 18th 2019 .Describes the

design of 8-bit Vedic multiplier and its performance comparison with existing multiplier such as i) Booth multiplier ii) Array multiplier iii) Wallace tree multiplier. Vedic calculations are the olden scheme of mathematics, which has a procedure of mathematical calculations to compute the multiplication of two 8-bit number. In this work Urdhva Tiryagbhyam (vertical and crosswise) Vedic sutra is used for multiplier design which provides better performance and consumes lesser time for computation. The Urdhva Tiryagbhyam is the finest sutra and universal one among additional sutras and which represents the different multiplication process compared to normal multiplication. In this work, Modified Carry Save Adder (MCSA) is used to compute the sum of partially generated products. It reduces the computational delay towards the addition of unfinished products. The proposed design uses the Verilog HDL to develop the algorithm. The XILINX 14.7 software tool is used to simulate and synthesize the code. The proposed design is also verified on Spartan-6 Field Programmable Gate Array (FPGA). Finally, the proposed 8-bit multiplier design is compared with 8-bit Booth multiplier, Array multiplier and Wallace tree multiplier in terms of Area, Memory and Delay. The result shows proposed 8-bit Vedic multiplier is efficient and consumes 14.219ns time for the multiplication process which is better compared to 8-bit, Booth multiplier, Array multiplier and Wallace tree multiplier.

A Abdul Hayum, S Chinnapparaj, G Sujatha, G Thamarai Selvi, Mohd Naved, P Mohanraj , **“Review of Vedic Multiplier Using Various Full Adders”**, Proceedings of the Fifth International Conference on Computing Methodologies and Communication (ICCMC 2021). The complexity of the chip is increasing as the advances in VLSI technology leads to the accumulation of more and more devices on a single chip. Since the chip has high density, the power dissipated in the chip or the amount of the heat produced is also increased resulting in the low power CMOS VLSI designs. In digital computer systems and digital signal processors, multiplication is one of the basic reckoning processes. This paper reviews the 2 by 2 vedic multiplier design using various high speed and low power adders.

Pallavi Singh, C. Dinendra, Hemanth. R. J, N. Mitra Vivek, **“Design Of High-Speed Vedic Multiplier Using Urdhva Tiryakbhyam Sutra”**, 2022 First International

Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT). Multiplication is the major process in all the arithmetic applications. Processors make use of adders and multipliers for all the arithmetic applications. The proposed work presents the efficient usage of Vedic multiplier than the conventional multiplier. The objective is to achieve low power consumptions and reduced propagation delay. This is achieved by implementing the Vedic multiplier using Adiabatic Logic and Vedic sutra. Vedic mathematics suggests many sutras, where Urdhava Tiryakbhyam is dedicated for multiplication of  $N \times N$  numbers. In the proposed work partial sum and products are obtained in single iterative step. ac power supply is used by Adiabatic Logic instead of dc power supply. In the proposed work, the design of Vedic Multiplier is done using 180nm CMOS processing technology through Cadence Virtuoso tool.

K. Rajesh, G. Umamaheswara Reddy, “ **FPGA Implementation of Multiplier-Accumulator Unit using Vedic multiplier and Reversible gates**”, International Conference on Inventive Systems and Control (ICISC 2019). The design of Multiplier-Accumulator (MAC) unit can be implemented by using the Vedic multiplier along with the reversible logic gates. The designing of Vedic multiplier is designed by using the new sutra called “Urdhava Triyagbhayam”. The performance of the MAC operation depends on the multiplier unit and the adder units. Here the designing of a multiplier and an adder can be designed by using the reversible gates to get the high speed of operation and also a Vedic multiplier is used for the higher performance, lesser area and to reduce the partial products. Nowadays reversible computing will take a more preferable for low power dissipation, higher speed of operation. Here, we proposed an 8, 16, 32, 64-bit Vedic multiplier is designed by using the carry save adder(CSA), the kogge stone adder(KSA) and the DKG adder out of these the proposed DKG gate adder is having the high speed of operation. The comparative analysis is carried out among the ripple carry adder (RCA), carry save adder, kogge stone adder. Finally, it has been proved that the proposed DKG gate with Vedic multiplier adder is having the high speed of operation. The overall Simulation and synthesis process is carried out with Xilinx ISE 14.7 and is dumped on the FPGA Zynq board.

Yaswanth Sai, **“High Speed Sixty Four Bit Vedic Multiplier”**, 2021 5th Conference on Information and Communication Technology (CICT). Multiplication operation is the core of many techniques like convolution, correlation, fast fourier transform etc. 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. Optimising multiplier (in terms of delay or area or power) will have a huge impact on system performance. One of the methods adopted to reduce delay is the use of Vedic Multiplier based on “Vedic Mathematics Sutras”. In this project “Urdhava - Tiryakbhayam” Sutra is used to perform multiplication and is simulated and implemented in Xilinx ISE Design.

Avinash Jain, Somya Bansal, Shaheen Khan, Shamim Akhter, Saurabh Chaturvedi, **“Implementation of an Efficient N×N Multiplier Based on Vedic Mathematics and BoothWallace Tree Multiplier”** 2019 IEEE. The HDL implementation of a novel multiplier algorithm based on the combination of Vedic mathematics and Booth-Wallace tree multiplier. An 8×8 multiplier is implemented in VHDL. The HDL code is simulated and synthesized using ModelSim and Xilinx ISE 14.1, respectively. The performance parameters of 8-bit multipliers implemented using various algorithms are compared in this paper. The comparison results exhibit that the proposed algorithm is faster than other multiplier algorithms.

Palak Yash, Mansi Thakare, and Babita Jajodia, **“Optimized Hardware Implementation of Vedic Binary Multiplier using Nikhilam Sutra on FPGA”**, 2022 IEEE 13th Latin America Symposium on Circuits and System (LASCAS). An efficient and optimized method of binary multiplication based on Nikhilam Sutra (algorithm) of Ancient Vedic Mathematics feasible for practical and real-time hardware implementations. This work demonstrates hardware implementation results of varying input bit-lengths (4-bit, 8-bit, 16-bit, 32-bit, and 64-bit) on Field Programmable Gate Array (FPGA) platform with an advantage of the reduced combinational delay and low device utilization (no. of slice LUTs) over existing alternative multiplication units and state-of-the-art Urdhva Tiryakbhayam and Nikhilam Sutra-based multiplication architectures. The proposed binary multiplier architecture is also demonstrated on the

Virtex-7 (xc7vx485t-3ffg1157) FPGA device. Hardware implementation results on Xilinx Virtex-7 FPGA device for  $8 \times 8$  proposed multiplication unit show that the design consumes 24 slice LUTs and maximum frequency up to 257.1 MHz.

S. Dhanasekar, P. Malin Bruntha, L. Jubair Ahmed, G. Valarmathi, V. Govindaraj, C. Priya, **“An Area Efficient FFT Processor using Modified Compressor adder based Vedic Multiplier”**, 2022 6th International Conference on Devices, Circuits and Systems (ICDCS). Fast Fourier Transform (FFT) technique has been used for processing the images, signals. It is used in spectrum analysis, sound filtering, image filtering, and data compression to name but a few. In the mentioned applications, FFT is used to translate the time domain signals into a spectrum representation. An optimized area for a 64-point FFT processor using a Vedic multiplier with a modified compressor adder has been proposed. Multi-radix approaches such as radix-23, radix-22, and radix-24 decrease the computational cost of the FFT processor. In the FFT processor, 8-bit Urdhva Tiryakbhyam vedic multipliers have been introduced, which reduces the total length of multiplication operations. A modified 4:2 Compressor adder will be used to speed up the vedic multipliers, minimizing carry delay. Urdhva Tiryakbhyam vedic multipliers uses modified compressor adders in the multi-radix FFT processor to reduce the silicon chip hardware area. The proposed multi-radix FFT processor uses a Field Programmable Gate Array (FPGA) chip named Virtex-6 XC6VLX75T device with 3548 slice registers and 19247 slice Look up Tables (LUTs). It operates with a clock frequency of 148.57 MHz as its maximum. The FFT algorithm developed decreases the slice register to 87.8% and slice LUTs to 48%. It operates by the clock frequency up to 148.57 MHz.

Aditi Awasthy, **“VLSI Implementation of Multiplier and Adder Circuits with Vedic Algorithm Computation”**, 2022 First International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT). This work aims to build a Vedic Multiplier using the Indian Vedic Mathematics technique as the best alternative for multiplying algorithm. The performance of a high-speed CPU is heavily dependent on a component known as a multiplier. In this project, we will use the Vedic mathematics algorithm with detector and compressor circuits to overcome these major

challenges of delay and complexity. We will focus on minimizing the processing delay of the digital circuit thereby increasing the speed. Also, reducing the switching activities, that will reduce the power consumption. The algorithm that we will use is 'Urdhva-Tiryagbhyam Sutra'. Simulation will be done using Xilinx ISE platform with Verilog language. Finally, the goal of this study is to design an effective Vedic Multiplier employing the Urdhva-Tiryabhyam algorithm, followed by a comparison of the proposed and conventional multipliers based on area, propagation delay, and power, with improved performance factors.

## **2.2 LITERATURE SUMMARY**

Above implementation and analysis it can lead to frame independent computing when the application word length is smaller than the architecture word length, N bit processor by 3 Authorized licensed use limited to: Sathyabama Institute of Science and Technology. Downloaded on July 16,2022 at 09:07:46 UTC from IEEE Xplore. Restrictions apply. product flexible and it posses potentially becoming parallel processor also. If 64 bit processor is will have control signals to enable and disable the lower components (32 bit or 16 bit) as per user requirements and can use for both pipeline and parallel architecture modes. As an alternate for downward compactable applications user will have choice of successive blocks. Proposed architecture works well for multiplication in pipeline, since addition and subtraction happened to be redundant in pipeline it continues to be implemented in array form. Division can also be pipelined with multiplication, provided fractional dividend is replaced by integer  $72/4$  is represented as  $3 \cdot (24/4)$  where dividend  $24/4$  is equivalent to  $6 \cdot 3$  hence for division any initial preprocessing is required for implementation in pipelining and array division is well implemented by Dhvajanka sutra.

## **2.3 PROBLEM DEFINITION**

To evaluate an efficient Multiplier for Digital applications.

## **2.4 OBJECTIVES OF THE PROJECTS**

- To increase the efficiency of the Multiplier for Low Power applications.

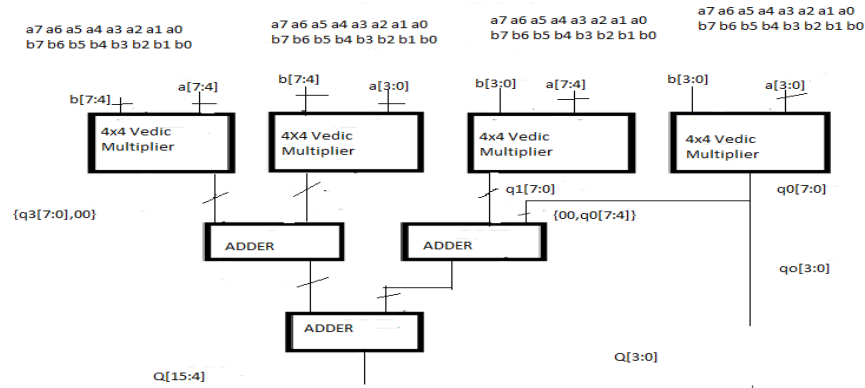
- In order to improve Efficiency Low Power Techniques will be applied in all Circuits.
- To improve, the characteristics of MOSFET Technology in terms of Power, Delay and Power Delay Product.
- To identify the Feasability of Multiplier for low power applications

## CHAPTER 3

### EXISTING METHODOLOGY

In this we discuss about Vedic multiplier to measure the average power and delay.

#### 3.1 8 X 8 BIT MULTIPLIER



**Fig 3.1: 8 X 8 MULTIPLIER**

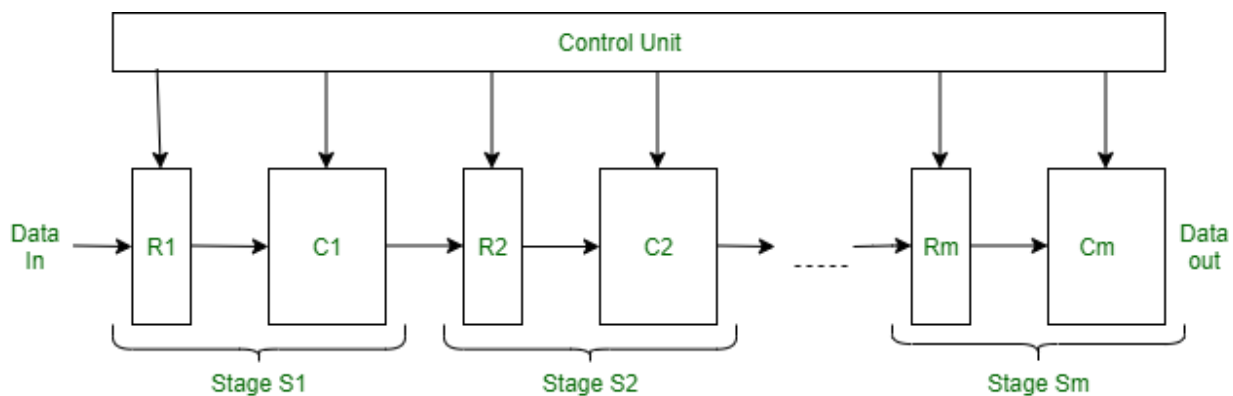
The Arithmetic module designed in this work, makes use of 4 components that are, Adder, Subtractor, and Multiplier. As a subtraction, and multiplication on  $n$ -bit data. The arithmetic unit uses conventional adder and subtractor, while the multiplier unit are made using Vedic Mathematics Algorithm. The control signals which guide the Arithmetic unit to perform a particular operation, i.e. Addition, subtraction, multiplication operation are  $s_0$  and  $s_1$ , which are provided by the control circuit. The status of control lines  $s_0$  and  $s_1$  and the corresponding arithmetic operation being performed.

#### 3.2 PIPELINE ARCHITECTURE

Pipeline Processor consists of a sequence of  $m$  data-processing circuits, called stages or segments, which collectively perform a single operation on a stream of data operands passing through them. Some processing takes place in each stage, but a final result is obtained only after an operand set has passed through the entire pipeline. As shown in figure, a stage  $S(i)$  contains a multiword input register or latch  $R(i)$ , and a data path circuit  $C(i)$ , that is usually combinational. The  $R(i)$ 's hold partially processed

results as they move through the pipeline; they also serve as buffers that prevent neighbouring stages from interfering with one another.

A common clock signal causes the  $R(i)$ 's to change state synchronously. Each  $R(i)$  receives a new set of input data  $D(i-1)$  from the preceding stage  $S(i-1)$  except for  $R(1)$  whose data is supplied from an external source.  $D(i-1)$  represents the results computed by  $C(i-1)$  during the preceding clock period. Once  $D(i-1)$  has been loaded into  $R(i)$ ,  $C(i)$  proceeds to  $D(i-1)$  to compute a new data set  $D(i)$ . Thus in each clock period, every stage transfers its previous results to the next stage and computes a new set of results.



**Fig 3.2: Pipeline Architecture**

### 3.3 ADVANTAGES OF PIPELINE ARCHITECTURE

- More efficient use of processor.
- Quicker time of execution of large number of instructions

### 3.4 DISADVANTAGES OF PIPELINE ARCHITECTURE

- Pipelining involves adding hardware to the chip.
- Inability to continuously run the pipeline at full speed because of pipeline hazards which disrupt the smooth execution of pipeline.

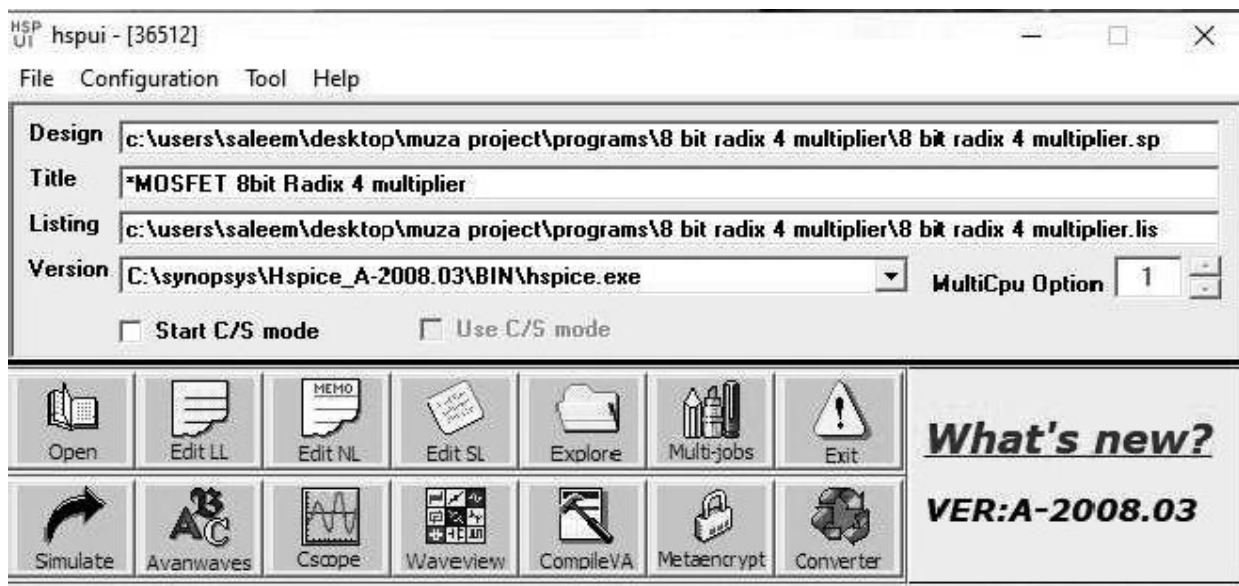
## CHAPTER 4

### PROPOSED METHODOLOGY

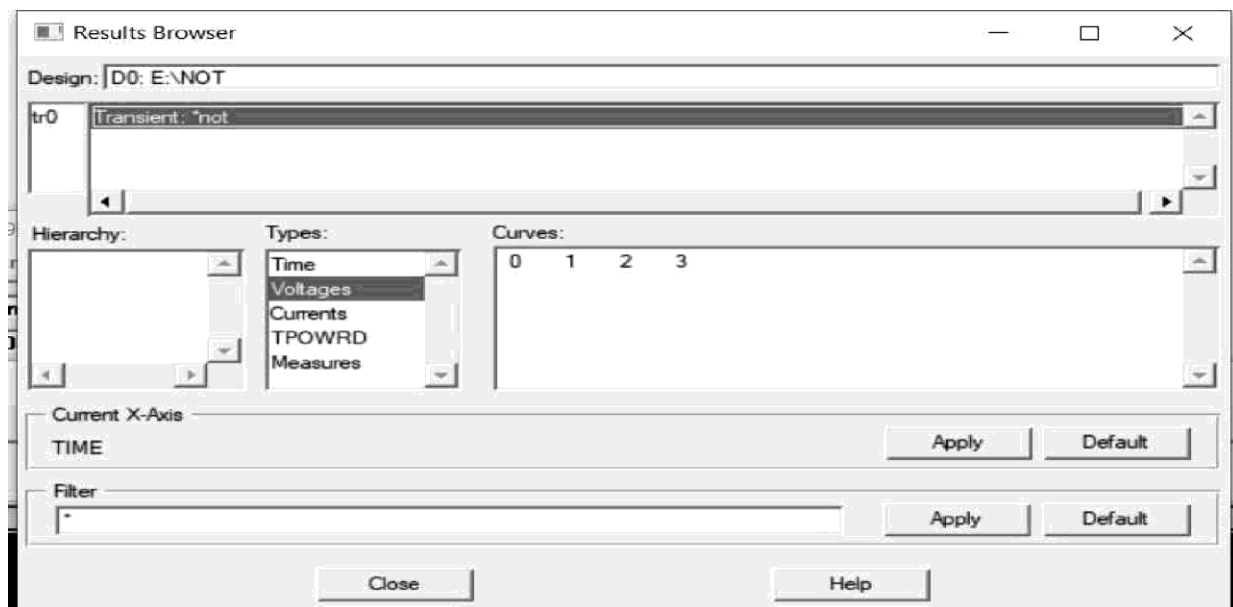
In order to improve efficiency existing multiplier will be analysed using 130nm, 32nm CMOS Technology. Further, multiplier will be modified and the proposed multiplier will be analysed and compared using 130nm, 32nm CMOS Technology in terms of power, delay and power delay product which can be used for digital applications.

#### 4.1 HSPICE SOFTWARE

Hewlett Simulation Program with Integrated Circuit is the full name of HSPICE. Unlike board-level designs made up of individual components, integrated circuits cannot be bread boarded before being manufactured. Furthermore, due to the high cost of photolithographic masks and other manufacturing requirements, it is critical to design the circuit to be as close to perfect as possible before it is designed. Until committing to producing an integrated circuit, the industry-standard method of verifying circuit operation at the transistor level is to simulate the circuit with SPICE.



**Fig 4.1 Hspice Software**



**Fig 4.2 Selection Of Input And Outputs**

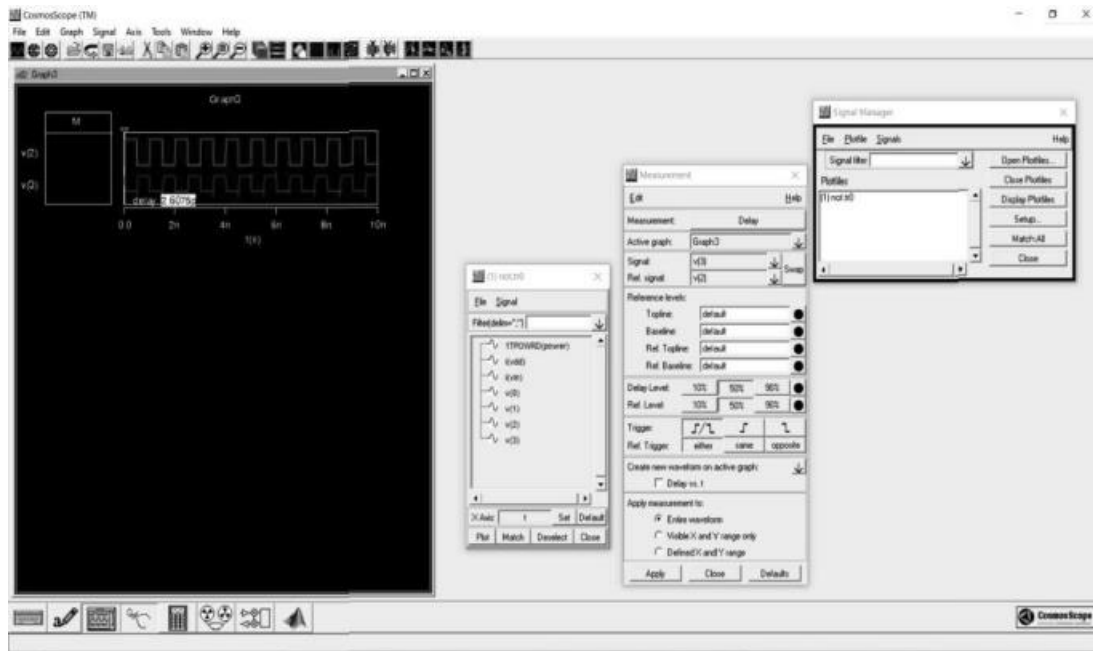
#### **4.1.1: SIMULATED SCREEN**



**Fig 4.3 Simulated Screen**

## 4.2 COSMOSCOPE

Cosmoscope is a software which is used to calculate the delay value.



**Fig 4.4 Cosmoscope Software**

### 4.2.1: POWER

Electric power is the rate at which electrical energy is transmitted by an electric Circuit per unit of time. It is the pace at which work is completed. Electrical power is the rate at which electric all energy is transmitted by an electric circuit per unit time in terms of an electric circuit.

The unit of the Electric Power = Watt

$$\text{Power} = V \cdot I (W)$$

Where:

W = power in watts

V = potential in volts

I = current in amps  
Q = charge in coulombs  
t = time in seconds

#### **4.2.2: DELAY**

The time it takes for a signal to reach its destination is referred to as delay. The time it takes for the consequence of a shift in input to be noticeable at the output is known as the delay of a logic gate.

#### **4.2.3: POWER DELAY PRODUCT**

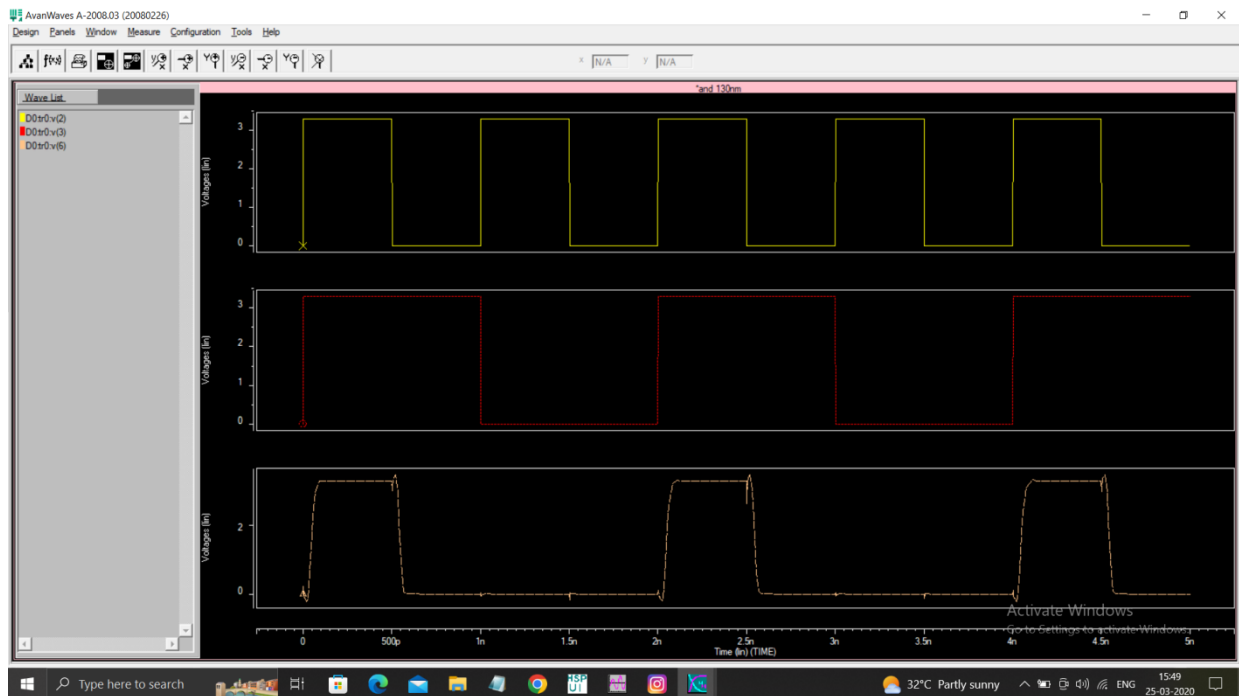
Power Delay Product is the product of the power and delay.

$$\text{PDP} = \text{Power} * \text{Delay}$$

## CHAPTER 5

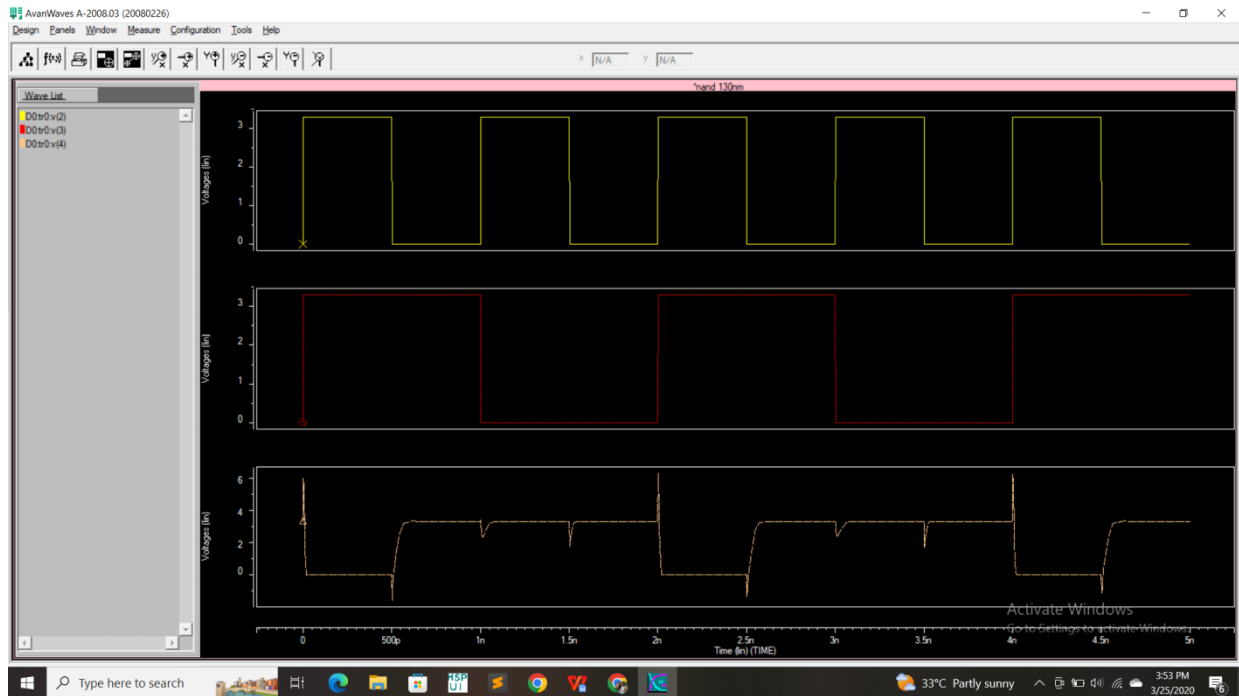
### RESULTS AND DISCUSSIONS

The existing and proposed architecture are simulated by using AvanWaves simulator. Output of the existing 8x8 Vedic Multiplier and proposed radix 4 booth multipliers are analysed from the results and it is concluded that the proposed architecture reduces the power consumption compared with the existing system. The Synthesis is done using Hspice.



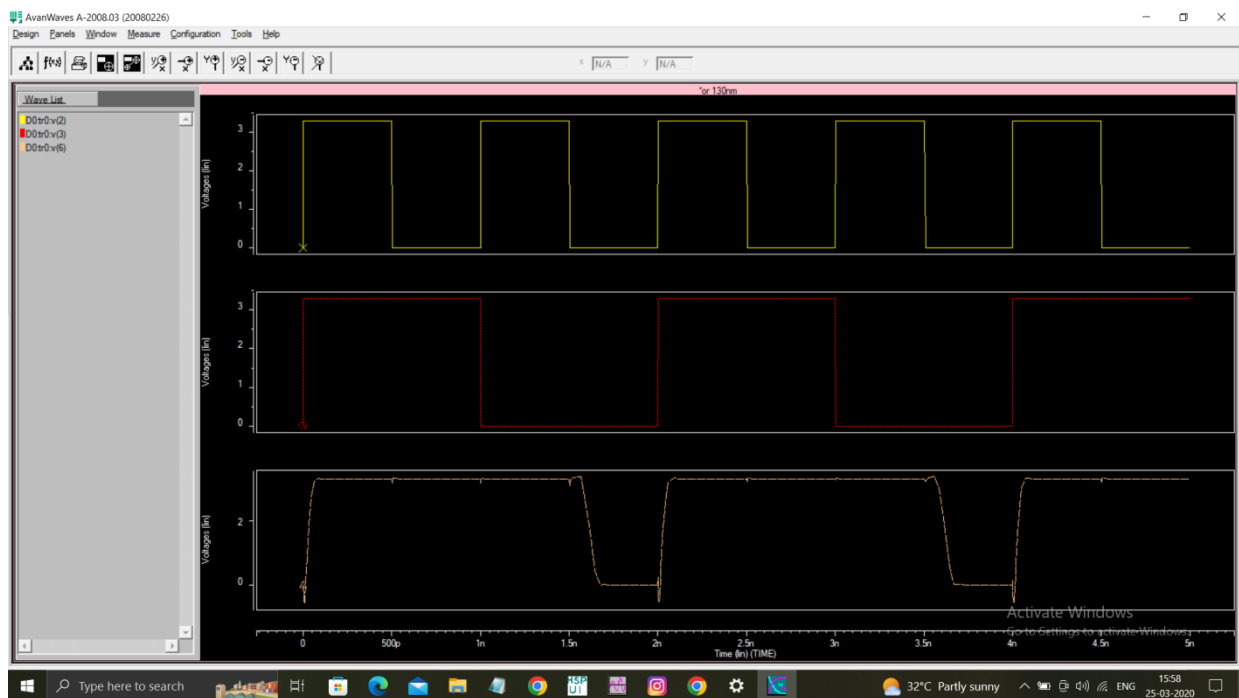
***Fig 5.1 Simulation Result Of And Gate***

Figure 5.1 shows the simulation output of a AND Gate.



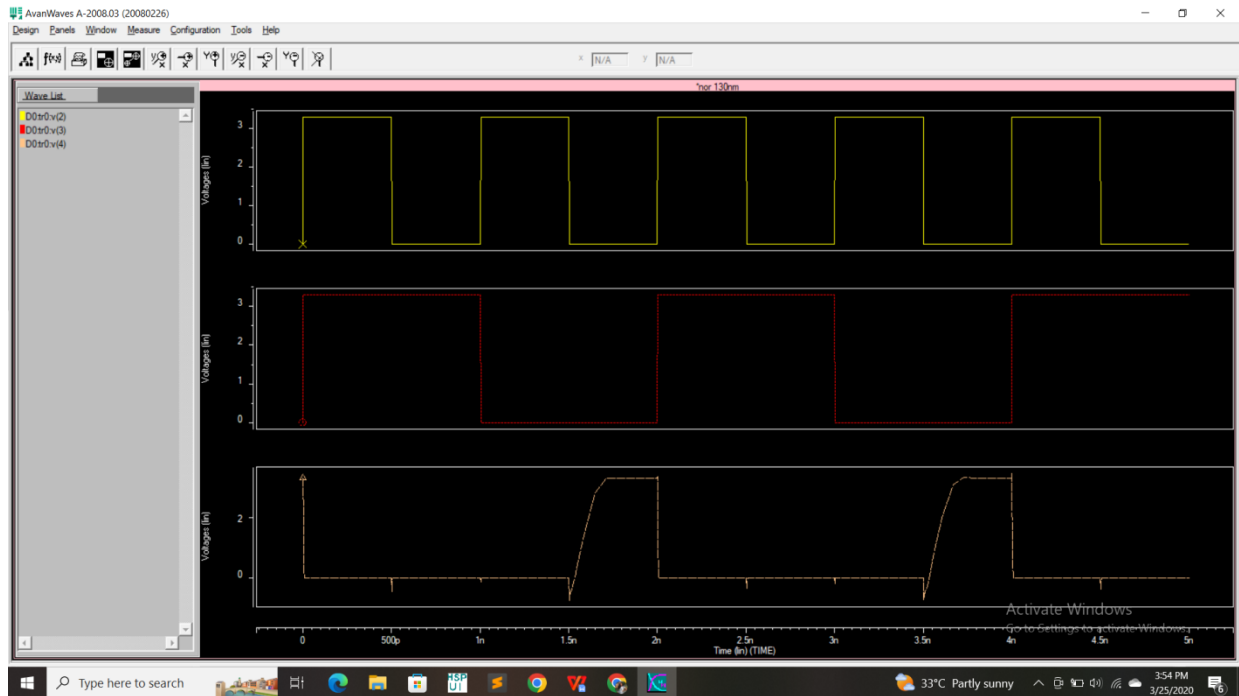
**Fig 5.2 Simulation Result Of Nand Gate**

Figure 5.2 shows the simulation output of a NAND Gate.



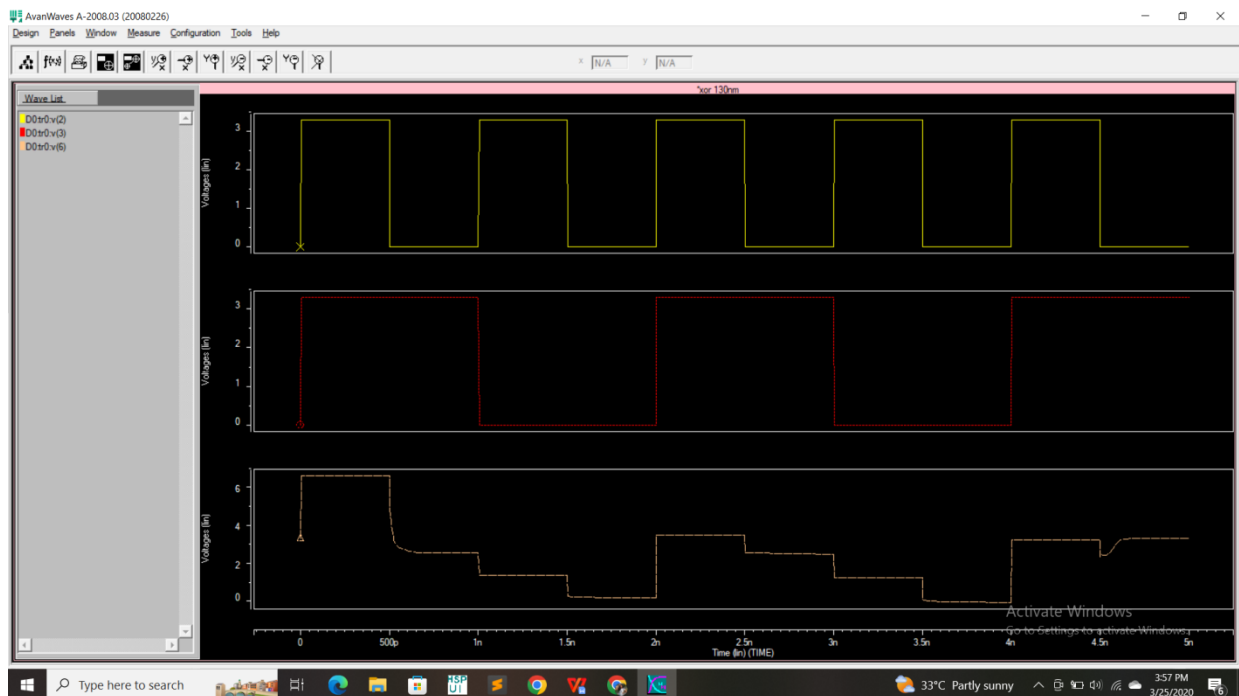
**Fig 5.3 Simulation Result Of Or Gate**

Figure 5.3 shows the simulation output of an OR Gate.



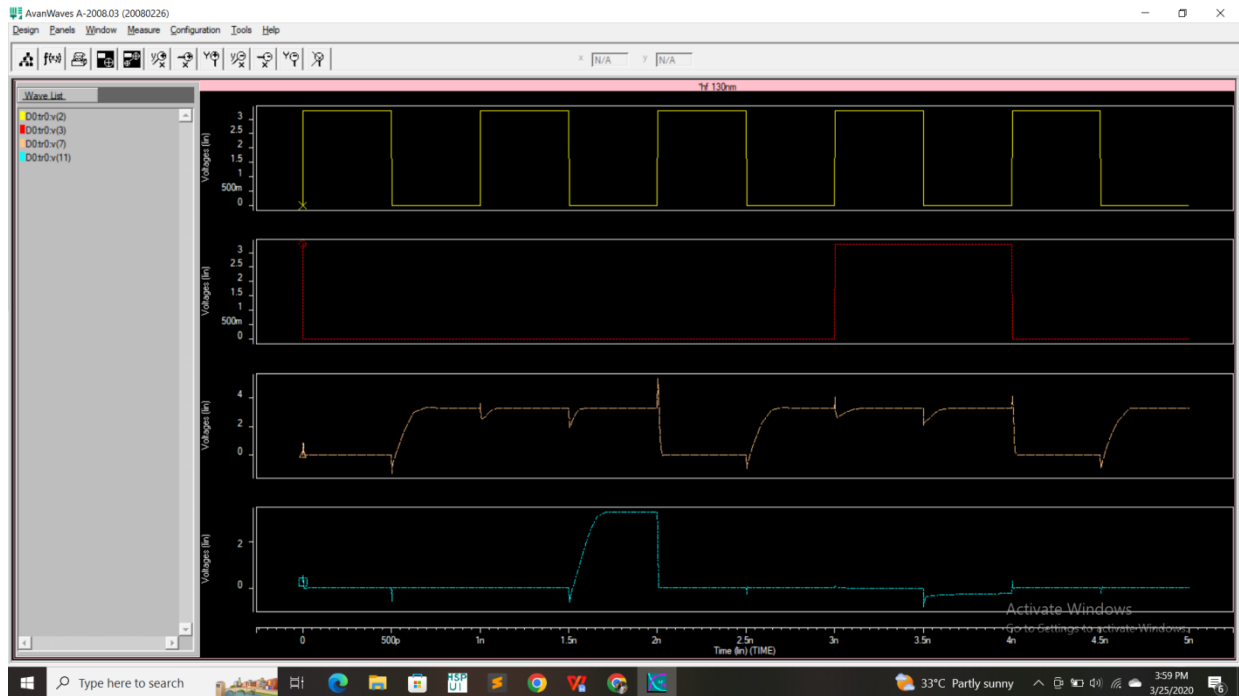
**Fig 5.4 Simulation Result Of Nor Gate**

Figure 5.4 shows the simulation output of a NOR Gate.



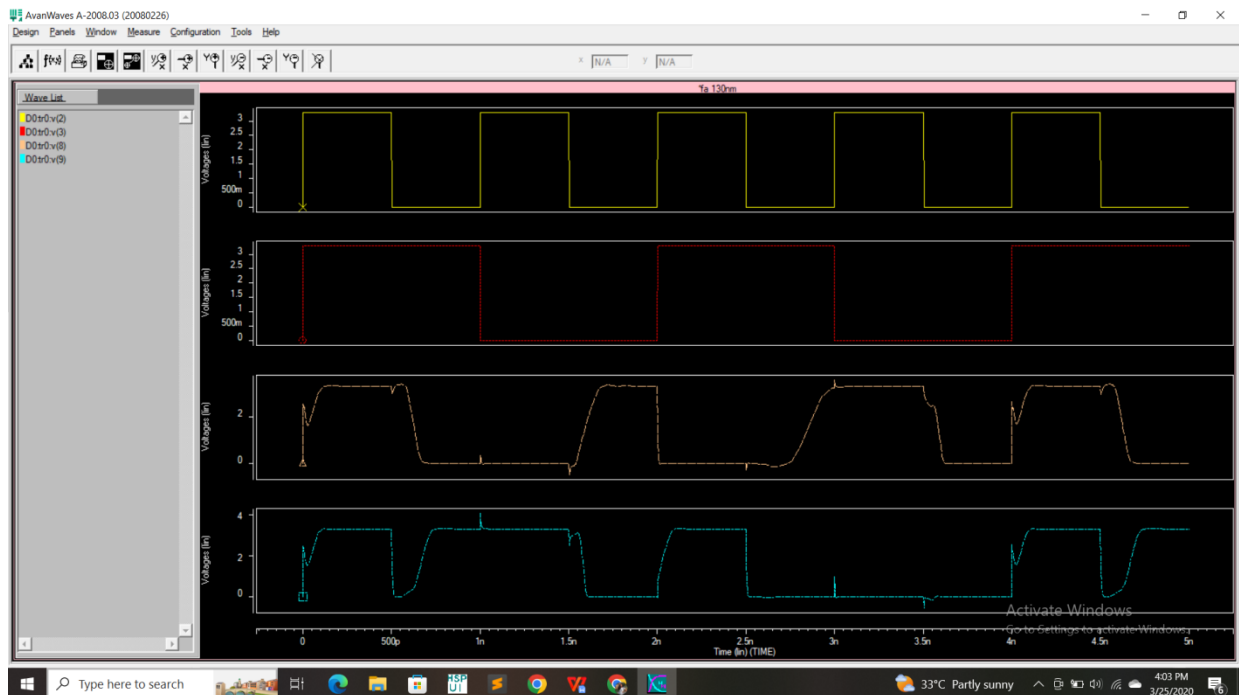
**Fig 5.5 Simulation Result Of Xor Gate**

Figure 5.5 shows the simulation output of a XOR Gate.



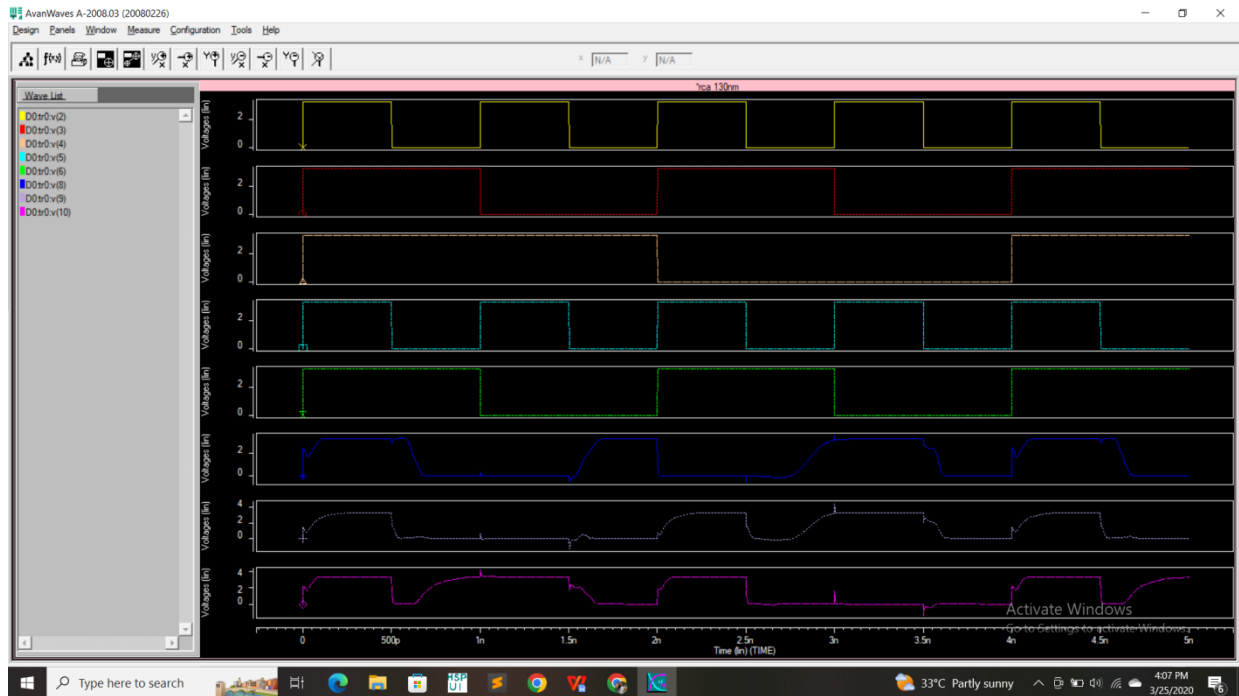
**Fig 5.6 Simulation Result of Half Adder**

Figure 5.6 shows the simulation output of a Half adder.



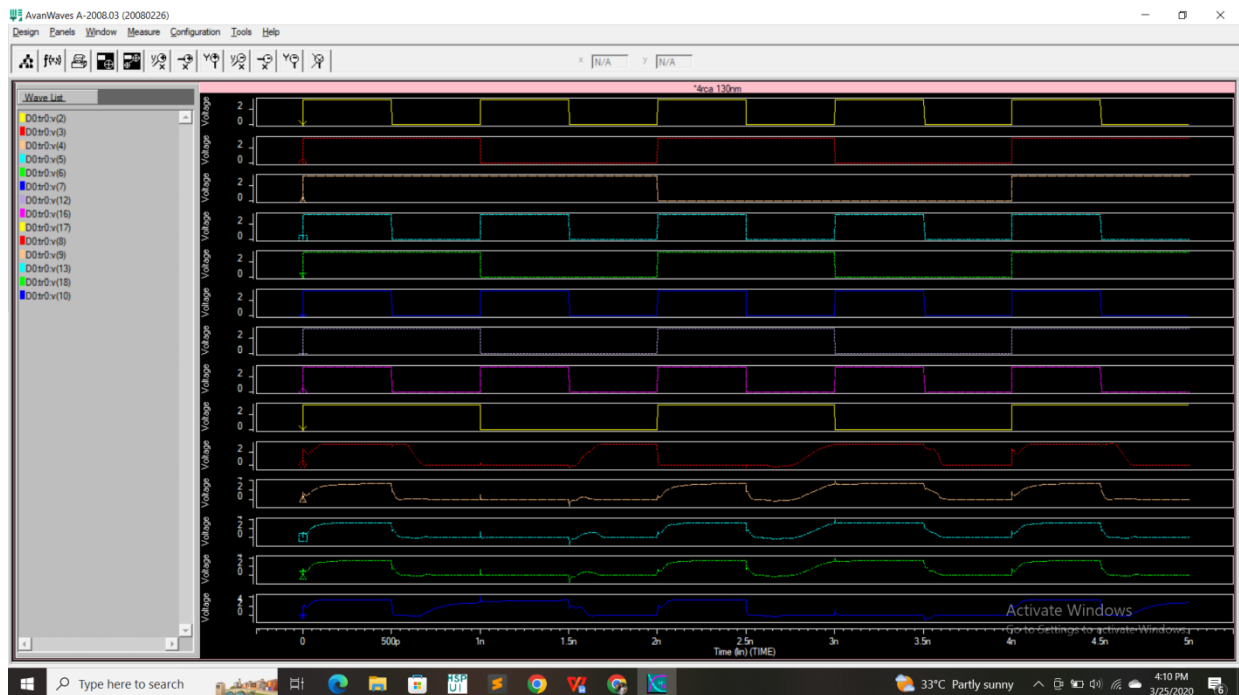
**Fig 5.7 Simulation Result of Full Adder**

Figure 5.7 shows the simulation output of a full adder.



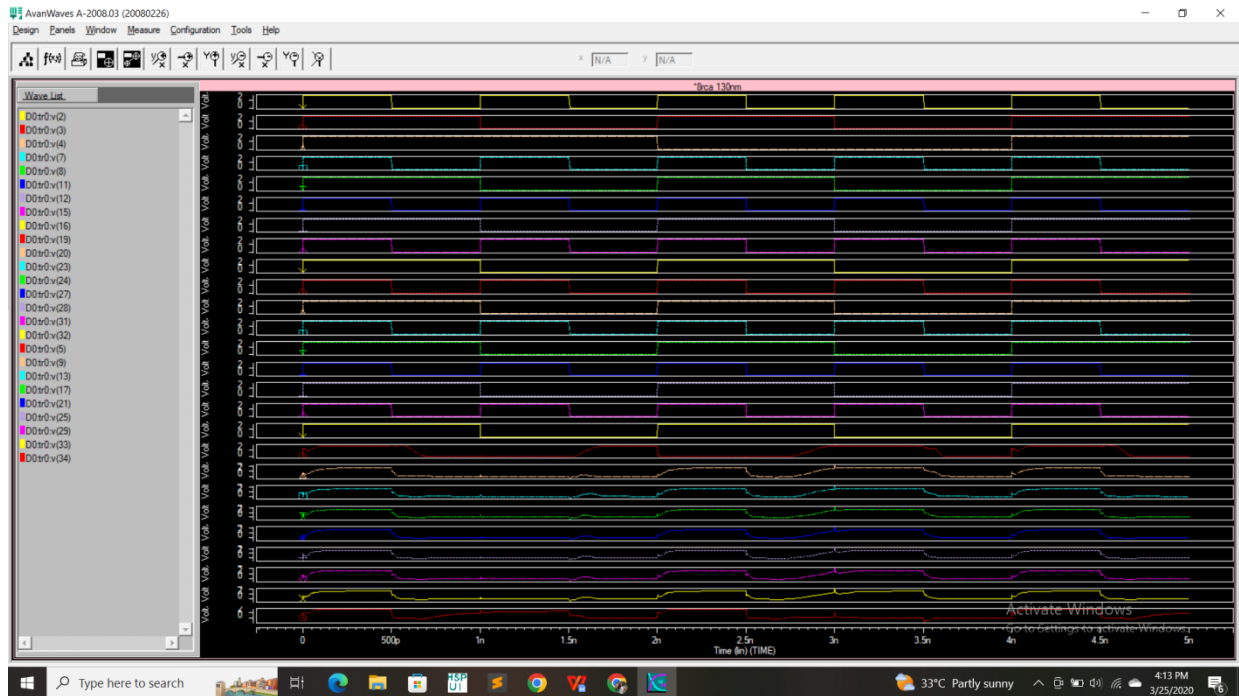
**Fig 5.8 Simulation Result Of 2-Bit Rca**

Figure 5.8 shows the simulation output of a 2-bit RCA.



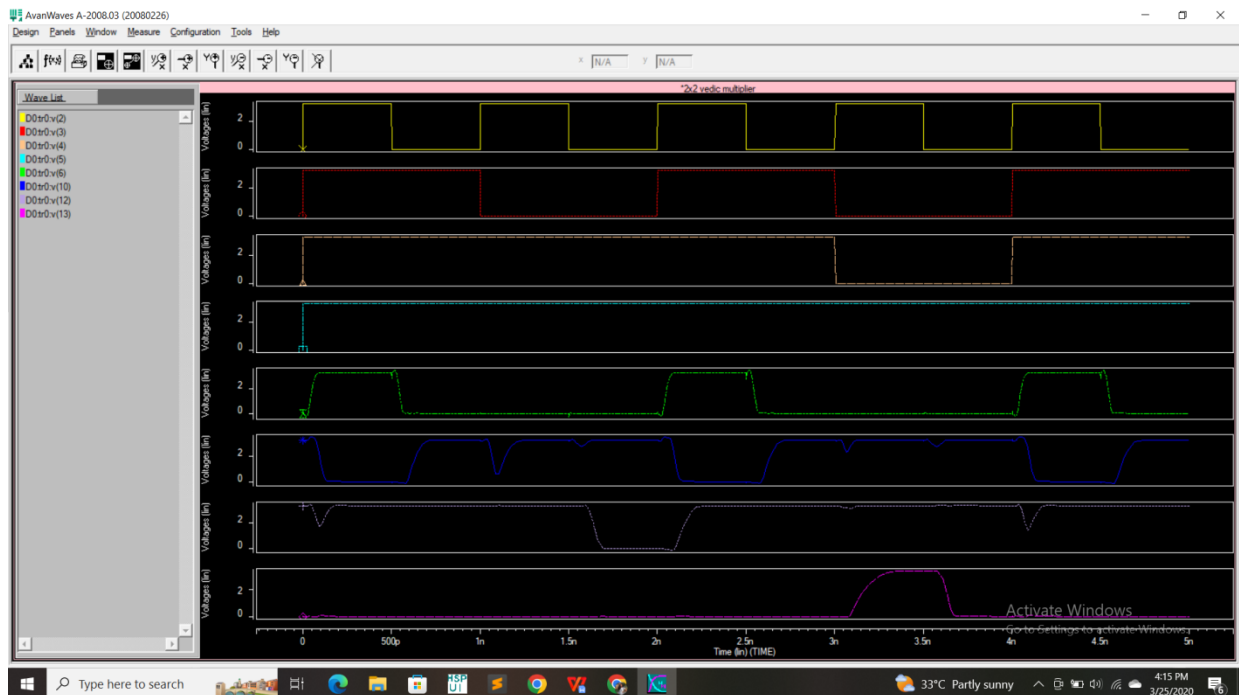
**Fig 5.9 Simulation Result Of 4-Bit Rca**

Figure 5.9 shows the simulation output of a 4-bit RCA.



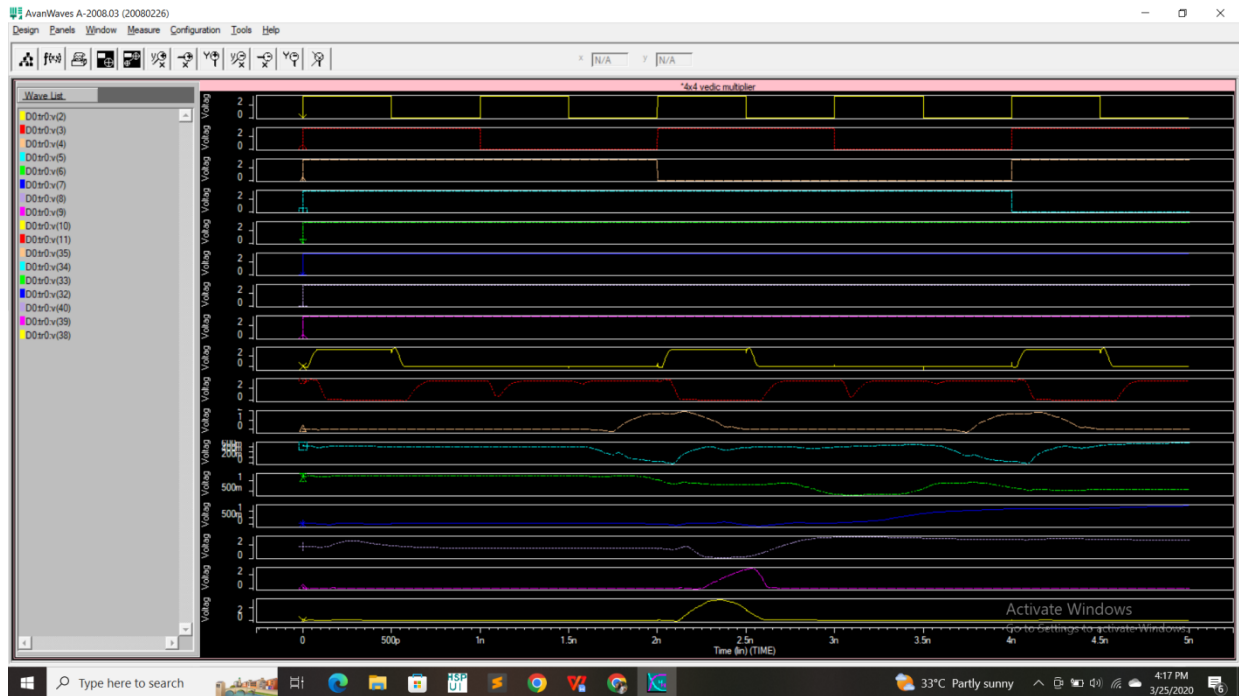
**Fig 5.10 Simulation Result Of 8-Bit RCA**

Figure 5.10 shows the simulation output of a 8-bit RCA.



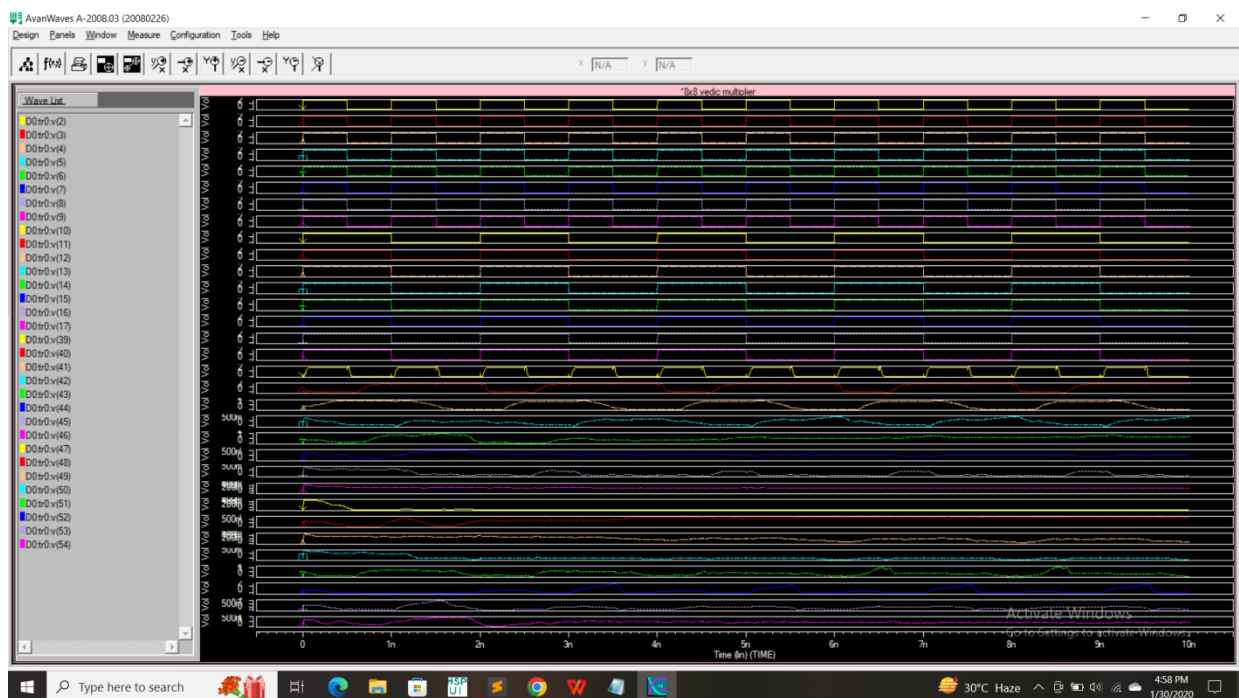
**Fig 5.11 Simulation Result of 2x2 Vedic Multiplier**

Figure 5.11 shows the simulation output of a 2x2 Vedic multiplier.



**Fig 5.12 Simulation Result of 4x4 Vedic Multiplier**

Figure 5.12 shows the simulation output of a 4x4 Vedic multiplier.



**Fig 5.13 Simulation Result of 8x8 Vedic Multiplier**

Figure 5.13 shows the simulation output of a 8x8 Vedic multiplier.

## 5.1 PERFORMANCE ANALYSIS

**Table 5.1. PERFORMANCE EVOLUTION OF POWER, DELAY AND POWER DELAY PRODUCT**

| Device | 130nm                  |                          |                         | 32nm                   |                          |                         |
|--------|------------------------|--------------------------|-------------------------|------------------------|--------------------------|-------------------------|
|        | Power(W)               | Delay(s)                 | PDP(J)                  | Power(W)               | Delay(s)                 | PDP(J)                  |
| NOT    | $7.878 \times 10^{-9}$ | $498.37 \times 10^{-12}$ | $3.926 \times 10^{-18}$ | $4.849 \times 10^{-9}$ | $438.54 \times 10^{-12}$ | $2.126 \times 10^{-18}$ |
| AND    | $1.790 \times 10^{-9}$ | $486.36 \times 10^{-12}$ | $0.870 \times 10^{-18}$ | $1.661 \times 10^{-9}$ | $406.36 \times 10^{-12}$ | $0.674 \times 10^{-18}$ |
| NOR    | $7.474 \times 10^{-9}$ | $497.63 \times 10^{-12}$ | $3.719 \times 10^{-18}$ | $6.315 \times 10^{-9}$ | $492.11 \times 10^{-12}$ | $3.107 \times 10^{-18}$ |
| NAND   | $4.096 \times 10^{-9}$ | $489.56 \times 10^{-12}$ | $2.005 \times 10^{-18}$ | $2.215 \times 10^{-9}$ | $436.69 \times 10^{-12}$ | $0.973 \times 10^{-18}$ |
| HA     | $5.410 \times 10^{-9}$ | $416.96 \times 10^{-12}$ | $2.255 \times 10^{-18}$ | $3.777 \times 10^{-9}$ | $400.06 \times 10^{-12}$ | $1.511 \times 10^{-18}$ |
| FA     | $6.523 \times 10^{-9}$ | $469.14 \times 10^{-12}$ | $3.060 \times 10^{-18}$ | $4.324 \times 10^{-9}$ | $459.52 \times 10^{-12}$ | $1.986 \times 10^{-18}$ |
| 2 RCA  | $11.32 \times 10^{-9}$ | $478.35 \times 10^{-12}$ | $5.414 \times 10^{-18}$ | $7.603 \times 10^{-9}$ | $437.58 \times 10^{-12}$ | $3.326 \times 10^{-18}$ |
| 4 RCA  | $21.66 \times 10^{-9}$ | $498.87 \times 10^{-12}$ | $1.080 \times 10^{-17}$ | $11.45 \times 10^{-9}$ | $490.88 \times 10^{-12}$ | $5.620 \times 10^{-18}$ |
| 8 RCA  | $51.34 \times 10^{-9}$ | $466.86 \times 10^{-12}$ | $2.396 \times 10^{-17}$ | $21.82 \times 10^{-9}$ | $427.46 \times 10^{-12}$ | $9.327 \times 10^{-18}$ |
| 2x2 VM | $101.2 \times 10^{-9}$ | $344.7 \times 10^{-12}$  | $3.488 \times 10^{-17}$ | $81.15 \times 10^{-9}$ | $110.59 \times 10^{-12}$ | $8.974 \times 10^{-18}$ |
| 4x4 VM | $195.2 \times 10^{-9}$ | $196.83 \times 10^{-12}$ | $3.842 \times 10^{-17}$ | $102 \times 10^{-9}$   | $107.44 \times 10^{-12}$ | $1.095 \times 10^{-17}$ |
| 8x8 VM | $265.7 \times 10^{-9}$ | $497.34 \times 10^{-12}$ | $1.321 \times 10^{-16}$ | $155.5 \times 10^{-9}$ | $408.33 \times 10^{-12}$ | $6.349 \times 10^{-17}$ |

**TABLE 5.2. PERFORMANCE EVOLUTION OF POWER, DELAY AND POWER DELAY PRODUCT IN CNFET**

| Device | CNFET                   |                          |                         |
|--------|-------------------------|--------------------------|-------------------------|
|        | Power(W)                | Delay(s)                 | PDP(J)                  |
| 2 RCA  | $4.214 \times 10^{-9}$  | $500.74 \times 10^{-12}$ | $2.110 \times 10^{-18}$ |
| 4 RCA  | $7.003 \times 10^{-9}$  | $500.22 \times 10^{-12}$ | $3.503 \times 10^{-18}$ |
| 8 RCA  | $11.884 \times 10^{-9}$ | $500.3 \times 10^{-12}$  | $5.945 \times 10^{-18}$ |
| HA     | $2.214 \times 10^{-9}$  | $499.82 \times 10^{-12}$ | $1.106 \times 10^{-18}$ |
| FA     | $2.546 \times 10^{-9}$  | $500.41 \times 10^{-12}$ | $1.274 \times 10^{-18}$ |
| 2X2 VM | $51.85 \times 10^{-9}$  | $82.55 \times 10^{-12}$  | $4.280 \times 10^{-18}$ |
| 4X4 VM | $87.25 \times 10^{-9}$  | $107.44 \times 10^{-12}$ | $9.374 \times 10^{-18}$ |
| 8X8 VM | $108.22 \times 10^{-9}$ | $352.85 \times 10^{-12}$ | $3.818 \times 10^{-17}$ |

## REFERENCES

- [1] A Abdul Hayum, S Chinnapparaj, G Sujatha, G Thamarai Selvi, Mohd Naved, P Mohan raj, "Review of Vedic Multiplier Using Various Full Adders", Proceedings of the Fifth International Conference on Computing Methodologies and Communication (ICCMC 2021).
- [2] Aditi Awasthy, "VLSI Implementation of Multiplier and Adder Circuits with Vedic Algorithm Computation ",2022 First International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT).
- [3] Anbumani V, Soviya S, Sneha S, "Speed and Power Efficient Vedic Multiplier using Adders with MUX", 2021 Innovations in Power and Advanced Computing Technologies (i-PACT).
- [4] Avinash Jain, Somya Bansal, Shaheen Khan, Shamim Akhter, Saurabh Chaturvedi, "Implementation of an Efficient  $N \times N$  Multiplier Based on Vedic Mathematics and BoothWallace Tree Multiplier" , 2019 IEEE.
- [5] CHANDRASHEKARA M N , ROHITH S, "Design of 8 Bit Vedic Multiplier Using Urdhva Tiryagbhyam Sutra With Modified Carry Save Adder" , 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT-2019).
- [6] Harpreet Singh Dhillon and Abhijit Mitra, "A Reduced- Bit Multiplication Algorithm for Digital Arithmetic's", International Journal of Computational and Mathematical Sciences, Spring 2008.
- [7] Himanshu Thapliyal, Saurabh Kotiyal and M. B Srinivas, "Design and Analysis of A Novel Parallel Square and Cube Architecture Based On Ancient Indian Vedic Mathematics", Centre for VLSI and Embedded System Technologies, International Institute of Information Technology, Hyderabad, 500019, India, 2005 IEEE .
- [8] K.Arunkumar, Dr.P.Mangayarkarasi, Dr.Beulah Jackson, Dr.A.Anitha Juliette, "Design of High Speed, Low Power 16x16 Vedic Multiplier With Adiabatic Logic" , 2022 8th International Conference on Smart Structures and Systems (ICSSS).

[9] K. Rajesh, G. Umamaheswara Reddy , “FPGA Implementation of Multiplier-Accumulator Unit using Vedic multiplier and Reversible gates” , International

[10] Malti Bansal, Jasmeet Singh, “Comparative Analysis of 4-bit CMOS Vedic Multiplier and GDI Vedic Multiplier using 18nm FinFET Technology” , Proceedings of the International Conference on Smart Electronics and Communication (ICOSEC 2020).

[11] M.Bala Murugesh, S.Nagaraj, J.Jayasree, G.Vijay Kumar Reddy, “Modified High Speed 32-bit Vedic Multiplier Design and Implementation”, Proceedings of the International Conference on Electronics and Sustainable Communication Systems (ICESC 2020).

[12] Palak Yash, Mansi Thakare, Babita Jajodia, “Optimized Hardware Implementation of Vedic Binary Multiplier using Nikhila Sutra on FPGA”, 2022 IEEE 13th Latin America Symposium on Circuits and System (LASCAS).

[13] Pallavi Singh, C. Dinendra, Hemanth, R. J, N. Mitra Vivek, “Design Of High-Speed Vedic Multiplier Using Urdhva Tiryakbhyam Sutra” , 2022 First International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT).

[14] Ruchi anchaliya, Chiranjeevi G N, Subhash Kulkarni,” Efficient Computing Techniques using Vedic Mathematics Sutras”, international journal of innovative research in electrical, Electronics, Instrumentation and Control engineering, volume 3, Issues 5, May 2015.

[15] S. Dhanasekar, P. Malin Bruntha, L. Jubair Ahmed , G. Valarmathi, V. Govindaraj, C. Priya, “An Area Efficient FFT Processor using Modified Compressor adder based Vedic Multiplier” , 2022 6th International Conference on Devices, Circuits and Systems (ICDCS).

[16] Sumita Vaidya and Deepak Dandekar, “Delay- Power Performance comparison of Multipliers in VLSI Circuit Design”, International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, July 2010.

[17] Umesh Akare, T.V. More and R.S. Lonkar, "Performance Evaluation and Synthesis of Vedic Multiplier", National Conference on Innovative Paradigms in Engineering & Technology (NCIPET- 2012), proceedings published by International.

[18] Vaijyanath Kunchigi, linganagouda kulkarni and subhash kulkarni 32-bit MAC unit design using Vedic multiplier – published at: "International Journal of Scientific and Research Publications (IJSRP), Volume 3, Issue2, February 2013 Edition".

[19] Yaswanth Sai , "High Speed Sixty Four Bit Vedic Multiplier" , 2021 5th Conference on Information and Communication Technology (CICT).