MULTI-STATE LOGIC FOR COMPUTATION

A thesis submitted to the
University of Petroleum and Energy Studies

For the Award of
Doctor of Philosophy
in
Computer Science & Engineering

BY
Amit Verma

Novomber 2019

SUPERVISOR
Dr. Manish Prateek

UPES
UNIVERSITY WITH A PURPOSE

School Of Computer Science
University Of Petroleum & Energy Studies
Dehradun-248007: Uttarakhand

MULTI-STATE LOGIC FOR COMPUTATION

A thesis submitted to the
University of Petroleum and Energy Studies

For the Award of
Doctor of Philosophy
in
Computer Science & Engineering

BY
Amit Verma
(SAP ID 500048922)

Novomber 2019

SUPERVISOR
Dr. Manish Prateek
Professor & Dean
School Of Computer Science
University Of Petroleum & Energy Studies

School Of Computer Science
University Of Petroleum & Energy Studies
Dehradun-248007: Uttarakhand

November 2019

DECLARATION

I declare that the thesis entitled **MULTI-STATE LOGIC FOR COMPUTATION** has been prepared by me under the guidance of **DR. MANISH PRATEEK**, Professor & Dean of School Of Computer Science, University Of Petroleum & Energy Studies. No part of this thesis has formed the basis for the award of any degree or fellowship previously.

Amit Verma
School Of Computer Science
University Of Petroleum & Energy Studies
P.O. Bidholi Via-Prem Nagar,
Dehradun-248007
DATE :

UPES
UNIVERSITY WITH A PURPOSE

CERTIFICATE

I certify that **Amit Verma** has prepared his thesis entitled **MULTI-STATE LOGIC FOR COMPUTATION**, for the award of PhD degree of the University of Petroleum & Energy Studies, under my guidance. He has carried out the work at the Department of Informatics, University of Petroleum & Energy Studies.

Dr. Manish Prateek
Professor & Dean of School Of Computer Science
University Of Petroleum & Energy Studies P.O. Bidholi Via-Prem Nagar,
Dehradun-248007
DATE :

CORPORATE OFFICE: 210, 2nd Floor,
Okhla Industrial Estate, Phase III,
New Delhi - 110 020, India.
T: +91 11 41730151/53, 46022691/5
F: +91 11 41730154

ENERGY ACRES: Bidholi Via
Prem Nagar, Dehradun - 248 007
(Uttarakhand), India.
T: +91 135 2770137, 2776053/54/91, 2776201
F: +91 135 2776090/95

KNOWLEDGE ACRES: Kandoli Via
Prem Nagar, Dehradun - 248 007
(Uttarakhand), India
T: +91 8171979021/2/3, 7060111775

NGINEERING | COMPUTER SCIENCE | DESIGN | BUSINESS | LAW | HEALTH SCIENC

ABSTRACT

In this thesis, work is divided in mainly two parts contributing in the field of multiplication operation and multivalued logic.

Multiplication is one of the important arithmetic operation both in manual mathematics and digital machine, methods of multiplication travels from the time of Vedas till today. Many methods of performing multiplication are mention in Vedic literature but most of the researcher mainly used *urdhva tiryakbhya* and *nikhilam* sutra for designing binary multiplexer which are more complex and require large number of electronic components with interconnection overhead results in delay. Moreover, previously developed Vedic multiplexers for binary machine are totally based on Vedic sutra, no modification has been done or no novel approach is derived from the Vedic literature. In this work, a novel algorithm is proposed for the multiplication of binary numbers motivated from various untouched methods mention in the Vedic literature. The algorithm mainly consist of one main algorithm that is am-Multiplication and two sub-algorithms MIN & MAX, a binary multiplexer circuit is also design on the basis of the proposed algorithm for performing binary multiplication.

Multivalued logic simply means increasing the logical states of machine on the basis of the level of voltages, which can be consider as an alternate for the traditional binary system to improve the computation speed & storage efficiency. Furthermore for reducing the number of electronic components, interconnection overhead, delay and chip size. Here, a novel algorithm is proposed for performing ternary addition where ternary system comprises of three logical states(bits) {0, 1, & 2} instead of two logical states {0 & 1} as in case of traditional binary system. Stack of size 1 is used in the algorithm for holding generated carry bit, where in case of ternary addition carry is only generated when both the bits are non-zero and either of the bit is 2. Truth table for various ternary operators are proposed, based on which the circuits are design using op-amp 741 IC, 7432 IC & 7208 IC. Single ternary inverter(t-NOT) circuit is proposed which can be consider as an extension of binary NOT gate replacing the previously developed three separate circuits for ternary inverter. The circuit for ternary decoder, ternary NAND & NOR gate also proposed in the work. The t-NOT gate is considered as building block for designing circuits based on ternary logic. And on the basis of these ternary operators an adder circuit is designed which comprises of separate circuit for computing sum and generated carry. The proposed ternary adder circuit is based on the truth table representing the sum and carry for all possible combinations of two ternary bits either of which can be from the set of three logical bits {0, 1, & 2}.

keywords : Vedic literature, multiplication, multivalued Logic, ternary inverter.

ACKNOWLEDGEMENT

I bow my head humbly to pay heart felt regards to Almighty God for giving me the strengths and blessing in completing this thesis.

There are quite a few people that have helped me in one way or another to the completion of this work. It is with great pleasure, I would like to thank all of you from very deep inside.

Foremost, I would like to express my sincere gratitude to my thesis advisor Prof. Manish Prateek, for picking me up as a student at the critical stage of my career and the continuous support of my Ph.D study and research, for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

It is absolutely difficult to succeed in the process of finding and developing an idea without the help of a specialist in the domain. I found in my advisor not only the source of wonderful ideas to develop, but also the support that a Ph.D student needs. I could not have imagined having a better advisor and mentor for my Ph.D study.

Besides my advisor, I would like to thank Chancellor Dr. S. J. Chopra, Vice chancellor Dr. Deependra Kumar Jha, Dean Dr. Kamal Bansal at the University of Petroleum and Energy Studies for their encouragement, suggestions and valuable support for my research work.

I would like to express my special thanks to Dr. J K Pandey, R&D, Director and Dr. Rakhi Ruhal, Program Manager-Ph.D, University of Petroleum and Energy Studies, for his assistance during my research work. I am grateful to the University of Petroleum and Energy Studies, for giving me an opportunity to pursue my research and for providing all facilities in the Department of Studies in School of Computer

Science and Engineering.

I would like to thank all the Heads of School of Computer Science Departments, doctoral students for their feedback, cooperation, and of course friendship. In addition I would like to express my gratitude to all colleagues in the university. In particular, I am grateful to Dr. Pradeep Kumar for his patiance, support and being with me in all my hard times.

Last but not the least, I would like to thank my family especially to my mother Smt. Raj Rani Verma for supporting me spiritually throughout writing this thesis and my life in general.

TABLE OF CONTENTS

LIST OF TABLES

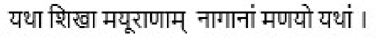
LIST OF FIGURES

CHAPTER 1
INTRODUCTION

This section is divided into two major parts one is about the multiplication operation and another is ternary logic. In the section of multiplication operation, early sources of multiplication, various Vedic methods of multiplication are discussed with the introduction about the proposed work which includes the development of novel multiplication algorithm inspired by various Vedic multiplication methods. The proposed multiplication algorithm is extended for the multiplication of binary numerals. And the corresponding multiplexer circuit is also developed based on the proposed algorithm. In the section of ternary logic, basic information about multi-state logic is discussed with the proposed circuitry and functionality of various operators, decoder, and adder based on ternary logic. This is also discussed how the proposed work can be a better approach as compared to the already existing approaches.

1.1 Multiplication Operation

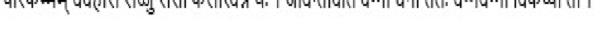
Multiplication is one of the basic and very important arithmetic operations from the Vedic period. We have tried to find out the starting point of the evolution of mathematics for that various Vedic literature is considered which are having traces of mathematics. According to various commentaries and the translation is done for Vedas, Vedanga Jyotisa [1, 2, 3] as one of the early evidence that mentions about ganita basically, considered as the mathematics which when joining together with jyotisa that is astronomy. Vedanga Jyotisa is placed on the top position in all the sciences same the way crests is situated at the head of peacocks and gems on the heads of snakes is mention in the sloke 1.1. It is considered as one of the oldest text whose actual author is not known, has given the concept of the lunisolar calendar by providing the accurate length of days, month and years according to the motion of the sun, the moon and the planets. Also, provide the rules for relatively calculating the length of day and night. Has provided the actual length of shortest and the longest day, 12 and 18 muhurtas, according to [4] muhurtas is one-third of the whole day.



यथा शिखा मयूराणाम् नागानां मणयो यथां ।

Figure 1.1: Sloke mention in Vedanga Jyotisa

Buddhist literature also mentions ganita considered as among early evidence, it states ganita as in three classes namely mudra meaning finger arithmetic, ganana meaning mental arithmetic, samkhyana meaning higher arithmetic in general [5]
The scope of hindu ganita is extremely large [2] including the mathematical geometry, and beautifully elaborated in the sthanangasutra mention in the sloke 1.2 in which each word represent the versatility of the hindu mathematics. The sutra comprises of the following words, Parikarma “fundamental operations”, vyavahara “determination”, Rajju “rope” that means geometry, Rasi “rule of three”, Kalasavarna “fractional operations”, Yavat tavat “simple equations”, Varga “quadratic equations”, Ghana “cubic equations”, Varga-varga “biquadratic equations, and Vikalpa “permutations and combinations”. The word Rajju or sulba is also used in Atharvaveda and Yajurveda is considered as the measuring instrument and used for constructing various mathematical geometry[6].



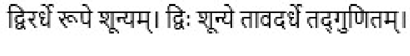
परिकम्मम् ववहारो राज्जु रासी कलास्वन्ने यः । जावन्तावति वग्गो घनो ततः वग्गवग्गो विकप्पो तां ।

Figure 1.2: Sloke for Sthanangasutra

In later studies ganita is considered as samkhyana as mention in Buddhist literature and rest is excluded from the scope of mathematics. And the words like pati-ganita and dhuli-karma is used for samkhyana, higher arithmetic. Later the part of ganita dealing with algebra named as bija-ganita, bija-ganita is viewed separately from pati-ganita remain preserved by scholars in coming centuries.

If we talk about the evolution of Hindu numerals than Yajurveda Samhita is one of the early evidence as it mention the list of numerals like Eka (1), dasa (10), sata (100), sahasra (1000), ayuta (10,000), niyuta (100,000), prayuta (1,000,000), arbuda (10,000,000), nyarbuda (100,000,000), samudra (1,000,000,000), Madhya (10,000,000,000), anta (100,000,000,000), parardha (1,000,000,000,000) even the same list is also mentioned in Taittiriya Samhita [7]. There are multiple evidences which show that in early stages numbers are represented in full words but signs are used for smaller numbers. Inscription of Asoka contemplate an evidence to state that the people at that time use script namely Brahmi and Kharosthi. The Brahmi numerals are found in almost all over india during the period of King Asoka (300 BC) and is considered as the pure hindu invention. Brahmi regarded as the mother of Narari numerals which is currently used as hindi numerals for mathematics.

One of the important sutra from the vedic literature which can be considered as the basic concept behind the development of binary computer machine is Pingala chandah-sutra [8]. Which provide the oldest evidence for the use of zero by Pingala (before 200 BC) as mention in the sloke 1.3. In this sutra the solution for finding the total number of arrangements of two things in n places is identified. According to Pingala chandah-sutra, "write two when the number is halved and write zero when one is subtracted. And in the separate column do double, when zero and square when halved." Let us see an example where $n = 7$, which is representing the number



द्विरर्धे रूपे शून्यम्। द्विः शून्ये तावदर्धे तद्गुणितम्।

Figure 1.3: Sloke for Pingala Sutra

of places. According to sutra as the value of n is odd that is it can't be absolutely divisible by two so subtract unity that is one form n, write the value of n that is six in column A and write zero for the operation performed in column B. As now the value of n is 6 which is absolutely divisible by two so divide n by two which is represented as halved in the sutra, write the remaining value of n in column A and two for the operation in column B respectively. Now repeat the procedure till the value of n is not equal to zero. Now starting with the last digit in column B which is zero, in this case, we take unity in the last row of column C and double it that will be two. Moving to the second last row, the value of the last row of column C will be either doubled and square according to the value of B in that particular row as shown in Table 1.1. And the process will continue for all the values of column B, after completing the procedure the value of the first row of column C represent the total number of the arrangement of two things in seven places which is 2^7 in this case.

Table 1.1: Pingala chandah-sutra

	A	**B**	**C**
n = 7			
Subtract 1	6	0	2^7
Halved	3	2	2^6
Subtract 1	2	0	2^3
Halved	1	2	2^2
Subtract 1	0	0	2

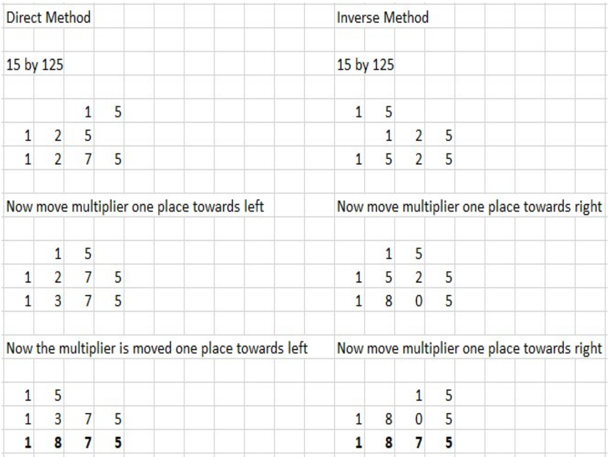
There are several other manuscripts where the evidence of the use of zero has been found such as Bakhshiali Manuscript (200 AD) and Panca-siddhantika (505 AD) where zero is mentioned at several places.

Various scholars from the Vedic period provide various methods of multiplication and also did commentary on various already developed methods. Here some of the methods are highlighted which were mentioned by various scholars named Arayabhata

I, Bhaskara I, Brahmagupta, Sridhara, Mahavira, Arayabhata II, Sripati, Narayana, Bhaskara II, Ganesha in their commentary.

1.1.1 Kapata – sandhi

Kapata means “door” and sandhi means “junction” so, the name represents the closing of the door. This is one of the most common methods mentioned by Sridhara, Arayabhata II, Sripati, Mahavira, Bhaskara II, and Narayana. According to Sripati this method state that “Placing the multiplicand below the multiplier as in the junction of two doors multiply successively by moving the multiplier in direct and inverse order” [1] as shown in the Fig. 1.4



Direct Method				Inverse Method			
15 by 125				15 by 125			
		1	5	1	5		
1	2	5			1	2	5
1	2	7	5	1	5	2	5
Now move multiplier one place towards left				Now move multiplier one place towards right			
	1	5			1	5	
1	2	7	5	1	5	2	5
1	3	7	5	1	8	0	5
Now the multiplier is moved one place towards left				Now move multiplier one place towards right			
1	5					1	5
1	3	7	5	1	8	0	5
1	**8**	**7**	**5**	**1**	**8**	**7**	**5**

Figure 1.4: Kapata - sandhi method

1.1.2 Gelosia method

The method is mention in “Ganita – manjari” also appear in Ganesha commentary on lilavati [9]. In this method, a bigger rectangle/square is divided into multiple small squares depending on the number of digits in the multiplier and multiplicands, and each square is divided diagonally. If each multiplier and multiplicand is of two digits than the square will be divided into four equal parts and each part is separated diagonally. As shown in the example below where the multiplier is 125 and multiplicand is 15 so the square is divided accordingly, then the multiplier is written as the title of each column starting from the leftmost side. And multiplicand is written at the end of each row, and each box is departed diagonally as shown in Fig 1.5. Now the first digit that is 1 of the multiplicand is multiplied with each digit of the multiplier (1, 2, 5) separately and resultant is written in the second half of the corresponding box of the multiplicand and first half of the box will contain generated carry if any. The same procedure is done with another digit of multiplicand that is 5 in this case. Finally, the sum of diagonal digits is carried out to get the final result.

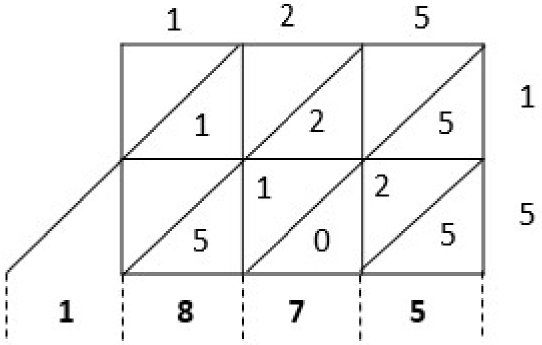


Figure 1.5: Gelosia method

1.1.3 Vajrabhyasa method

The method in [10] mention as a crosswise or zigzag multiplication, many Hindu scholars such as Mahavira, Sridhara, Sripati has referred the method in their work in just simple words. Ganesha has also mentioned the multiplication method and considers it as one of the fantastic methods which require traditional knowledge. In the case of Vajrabhyasa [2] after placing multiplier below multiplicand, multiply unity with the unity of both and store the result, then unity with tens and tens with unity adding them, place the result separately and so on to get the final result. As shown in Fig. 1.6 where the multiplier is 125 and multiplicand in 015, the underbar digits represent the digits to be mention in the final result and the overbar digit represents the carry for the next operation. So, starting from the first operation were the unity of both multiplier and multiplicand are multiplied resulting 25 so 5 will be the last digit of the final answer and 2 will be the carry for a result of the next operation. And with likewise operation final result will be 01875.

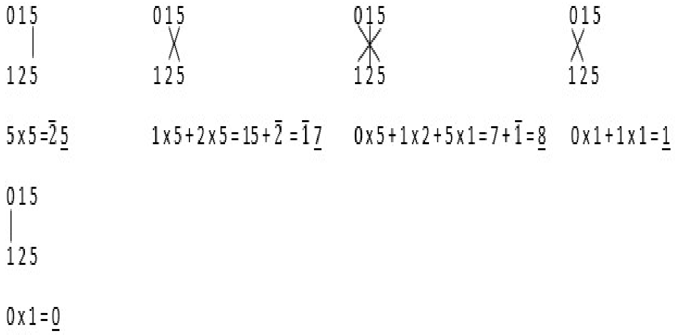
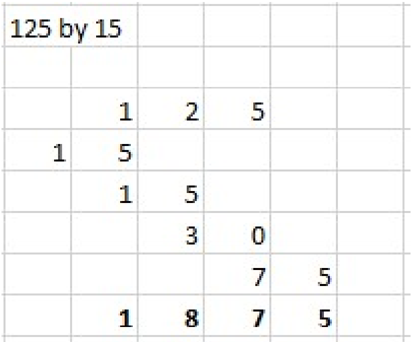

015
125
5x5=$\bar{2}\underline{5}$
015
125
1x5+2x5=15+$\bar{2}$=$\bar{1}\underline{7}$
015
125
0x5+1x2+5x1=7+$\bar{1}$=$\underline{8}$
015
125
0x1+1x1=$\underline{1}$
015
125
0x1=$\underline{0}$

Figure 1.6: Vajrabhyasa method

1.1.4 Sthana – Khanda

The method has been mention in the work after 628 AD. Bhaskara II defines the method as “Multiply separately by the places of figures and add together”[11]. Numerous scholars made the commentary on the method, an example of the same is shown in the Fig 1.7 where multiplicand 15 is multiplied with each digit of multiplier that is 125 and is added together for the final result that is 1875.

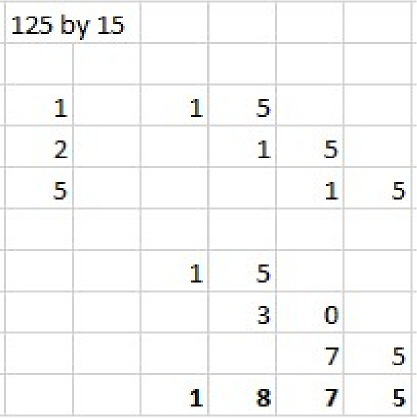


125 by 15					
	1	2	5		
1	5				
	1	5			
		3	0		
			7	5	
	1	**8**	**7**	**5**	

Figure 1.7: Sthana - Khanda method

1.1.5 Gomutrika method

The method has been described by Brahmagupta, both Sthana – Khanda and Gomutrika methods almost resemble with the way of multiplication of present days. Both methods can be applied for doing multiplication on paper which separates them from the category of pati-ganita. Example is shown in Fig 1.8



125 by 15					
1		1	5		
2			1	5	
5				1	5
		1	5		
			3	0	
				7	5
		1	**8**	**7**	**5**

Figure 1.8: Gonutrika method

1.1.6 Ista – Ganana method

The procedure is noticed in every Hindu work and discussed by many Hindu scholars. The method requires the addition and subtraction of any assumed number. As shown in the example below: 125 by 15

125 x 15 = 125(15 + 5) - 125 x 5

125 x 15 = 2500 – 625

125 x 15 = 1875

Or

125 x 15 = 125(15 – 5) + 125 x 5

125 x 15 = 1250 + 625

125 x 15 = 1875

Today also multiplication operation is the concern of various researchers for reducing the number of steps required to carry out the multiplication or reducing CPU cycles for the computation of the multiplication operation. As the multiplication and division operation requires more CPU utilization [12], so it always remains an important topic for the researchers to provide the algorithm which requires less utilization of CPU as compared to the current method. Many researchers proposed various algorithm and multiplexer circuit based on mainly two Vedic multiplication method that is *urdhva tiryakbhyam* and *nikhilam* sutra [13, 14, 15, 16, 17, 18, 19]. These algorithms and the based multiplexer is totally based on urdhva tiryakbhyam and nikhilam sutra without any modification or any novel approach to reduce the number of CPU cycles for performing the multiplication computation. Inspired from various Vedic multiplication methods discussed above and after learning about the Vedic multiplexer circuits proposed by various researchers, a totally novel algorithm for multiplication is developed and the multiplexer circuit is designed based on the proposed multiplication algorithm. Proposed work is compared with the combinatorial multiplier and Wallace-tree implementation of 8x8 multiplier [20] and it has been found that proposed circuitry is having much lesser gate count and gate delay as compared to the combinatorial and Wallace-tree based multiplier. Furthermore, the proposed work is compared with Vedic multiplexers based on urdhva tiryakbhyam sutra [21] and nikhilam sutra [16], found that the calculated delay of the proposed circuit is much less. The algorithm includes the development of sets and equations, the work is extended for the multiplication of binary numbers for current binary machines. And the circuit of the multiplexer is designed based on the algorithm for the multiplication of two binary numerals.

1.2 Ternary(3-state) Logic

Gottfried Leibniz was a German mathematician, after adopting the Indian numerals from 0-9 he convicted that only 0 and 1 are needed to do any mathematical calculations. That means with 0 and 1 all the required mathematical operations can be performed, this calculation was called Binary Number System. He published his work in 1703 AD in an article named Explanation de I' Arithmetic binary. This is how the binary number system came into existence in 1703 AD which is long back before its actual application in digital computers near about 1940's with the invention of transistors moreover we can say with the existence and use of silicon. But the theory of arithmetic remains same as given by Gottfried Leibniz in 1679AD.

So, today's computers are based on the binary number system, which works on two-state (0/1). Basically the logical bit 0 & 1 represents current/no current and in term of voltage it is 0V & 5V [22]. Where 0V is not an absolute 0V but near to 0V and lesser than the threshold voltage of the transistor. From the beginning, the speed of computation or performance of binary machines remains one of the major concerns for numerous researchers. As the speed of computation of the data actually depends on how fast the transition took place that is clock speed, which is at its limit nowadays [23]. In the case of binary machines data is represented and processed in binary form. The binary representation of data requires long strands of 0 and 1 or a large number of bits, which obviously require more processing time and high storage. Moreover, a large number of physical devices are required by the binary computers [24] which results in the overhead of many interconnections [25, 26] and finally bigger chip size & delay. With the concern of some major issues in current binary machines such as performance that is the processing of data (bits), complex circuits, plenty of components, high storage, large number of interconnections, bigger chip, and high delay area results in the introduction to Multi-state Logic. Here, multi-state and multi-valued logic is used interchangeably, we can define multi-state logic as increasing the state of machine from 2-state (binary) to 3-state (ternary) [27] or more states(fuzzy) [28].

Here, the work has been done on ternary logic (radix - 3), ternary logic includes three logical states instead of two that are 0, 1, and 2 to overcome the issues of current binary computer machines. Ternary logic-based computational circuits can enhance the processing speed of data as compared to the machine that works on two states. The number of bits required to represent the data in radix 3 is lesser as compare to the radix 2 so less processing time would be required to process the ternary data. And less storage to store ternary data, which results in higher performance of the ternary logic machine as compare to binary machines. Ternary logic based circuits require less number of electronic components, lesser interconnections overhead and finally smaller chip area [27]. Due to these advantages of ternary over binary logic, ternary logic attracted the interest of many researchers, and many circuits are proposed based on Multivalued Logic (MVL), circuits based on more than 2 logic states. Multiple circuits based on current (i-MVL) and voltages (v-MVL) are proposed for MVL [29, 30, 31, 32]. In the past 20 years, many circuits are proposed based on ternary logic using complementary metal-oxide semiconductor (CMOS) transistors and carbon nanotube FETs. Most of the author working on ternary logic has given the circuit design of ternary inverter circuit and classify it in three different circuits that are simple ternary inverter (STI), positive ternary inverter (PTI) and negative ternary inverter (NTI) [33, 29, 34, 35, 36, 37, 38] using high load resistors, variable resistors or variable voltage threshold of transistors and considered as the building block of any circuit based on ternary logic. Many circuits of ternary adders are also proposed in last few decades using CMOS [33, 38, 39], MOSFETs [37] and CNTFETs

[29, 40].

Already proposed ternary circuits by various authors have used three different inverter circuits that is STI, NTI, and PTI for ternary inverter logic which can not be considered as the extension of almost perfect binary logic. Furthermore this logic of three different inverter circuits for ternary logic can never be extended for higher radix like 4 or 5 state logic. So, it is required that circuits design for ternary logic should be logically based on binary or we can say as the extension of binary logic, moreover having scope to be further extended for higher radix. Ternary circuits proposed by various authors as mention above use high load resistors, variable resistors or variable voltage threshold of transistors to design the ternary circuits for dragging out the required results. Which can not be considered as base work for extending the logic to a higher number of states? The use of transistors with the different threshold values and high load resistors results in the increase of complexity and power consumption of the circuit.

So, for contributing to the field of multi-state logic, the truth table and circuits of various operators based on ternary logic are proposed. That will work on three different logic or three different levels of voltages represented by logical values 0, 1, and 2. Proposed circuits can be considered as the extension of the almost perfect binary system and can further be extended to higher radix(more than 3). Truth table and circuit diagram for *t*-NOT gate,*p*-NAND gate, *s*-NAND gate, *p*-NOR gate, *s*-NOR gate, adder, decoder based on ternary logic are proposed, where binary AND and OR gate are considered as MIN and MAX gate for ternary circuits. A novel algorithm for ternary addition based on ternary logic is proposed and which can be applicable for machines with more logical states if developed in the future. Here the term bit is used for representing any one of the logical states among three states

(0, 1, and 2) of ternary logic which is the same as in case of binary logic in which either 0 or 1 is considered a bit. The proposed circuits for ternary logic are based on the concept of considering a different range of voltage for representing three different logical states of 0, 1, and 2 as shown in table 1.2.

Table 1.2: Voltage representation of the logical states of ternary logic, where v is voltage

State Logic	**Voltage Range**
0	$0 < v <= 1$
1	$1 < v <= 3$
2	$3 < v <= 5$

1.3 Motivation

1.3.1 Multiplication Algorithm

Many researchers proposed multiple multiplexer circuits for current binary computer machines based on Vedic multiplication methods, but almost every researcher used only two Vedic methods namely *urdhva triyakbhyam* & *nikhilam* sutra. And the designed binary multiplexers are the exact depiction of these two Vedic sutras with almost no modification and none of the novel algorithms is derived based on Vedic literature in the past few decades. As the Vedic literature is comprised of a various untouched and simpler method of multiplication that motivates the need for a multiplexer with simpler circuit and high efficiency. Thus, a novel multiplication algorithm based on the study of multiple methods mention in Vedic literature is proposed and a binary multiplexer circuit is also designed referring to the algorithm.

1.3.2 Ternary operators & Adder

As the current binary computer machine work on base 2 that is 0 & 1, so the data is represented in a long series of 0s and 1s which consume more processing time and storage. Moreover, binary circuits require a large number of electronic components which causes interconnection overhead and high delay, which collectively cause more power consumption. All these drawbacks of binary machine strengthen the requirement of Multivalued logic-based machine, which actually work on more than two logical states. So, a novel algorithm is proposed for the addition of ternary bits where ternary logic works on three logical states that are {0, 1, 2} instead of two states as in the case of binary. Truth tables of various ternary gates are presented and circuits are designed on the basis of these truth tables. Moreover, a circuit of the adder is designed for performing addition of ternary bits.

The rest of the chapters are organized as follows. In Chapter 2 related work of multiplication operation and multivalued logic is presented. Chapter 3 describes the proposed algorithm for multiplication and the binary multiplexer circuit based on the algorithm. Chapter 4 comprises of proposed circuit diagrams for inverter circuit, NAND & NOR gate based on ternary logic. The circuit for ternary decoder and adder is also designed and a novel ternary addition algorithm is proposed for the addition of ternary bits. Chapter 5 contains the results of the proposed work. And finally, the conclusion and future scope are presented.

CHAPTER 2
LITERATURE REVIEW

This chapter is broadly categorized into two sections namely *Multiplication Operation & Multivalues Logic*. The first section comprises of the detailed study and related work carried out for the development of a novel approach for performing multiplication operation inspired from the Vedic literature and to design a binary multiplexer circuit based on the algorithm. Similarly, another section consists of related work which is studied for the development of various electronic circuits and algorithm for addition based on ternary logic.

2.1 Multiplication Operation

It is very difficult to find the starting point of ganita, but according to various commentaries and translation done for vedas, Vedanga Jyotisa [1, 2, 3] can be consider as as one of the early evidence that mention about ganita. Vedanga, considered as the mathematics with when join together with jyotisa that is astronomy. Then Vedanga Jyotisa is consider on the top position in all the sciences same the way crests is situated at the head of peacocks and gems on the heads of snakes is mention in the sloke. Vedanga Jyotisa is considered as one of the oldest text whose actual author is not know, has given the concept of luni-solar calendar by providing the accurate length of days, month and years according to the motion of the sun, the moon and the planets. Also provide the rules for relatively calculating the length of day and night. Has provide the actual length of shortest and the longest day, 12 and 18 muhurtas, according to [4] muhurtas is consider as one-third of the whole day. Different types of ganita like finger mathematics, mental arithmetic and higher mathematics have also been mentioned in Buddhist literature [2]. The higher mathematics considered as pati-ganita (procedural mathematics, algorithms) and bija-ganita (mathematics of algebra) both are regarded separately by Sridharacharya [41] in Trisatika (Patiganitasara), Patiganita, Bijaganita, Navasati, and Brhatpati. Similarly, the Yajurveda Samhita [7] is considered as the early source of hindu numerals providing the list of hindu numerals of large numbers. The same list of hindu numerals were seen in taittiriya, maitrayani and kathaka Samhita. Datta et al. [2] discussed about various Vedic method of *gunana* the common hindu name for multiplication, in various Vedic literature the term *gunana, vadh* also used for multuplication which actually mean killing or destroying. The author has tried to find out the beginning of muliplication by discussing about various historical literature that mention about the multiplication operation like word *abyasa* in the work *sulba (800 b.c.)* used for both addition and multiplication, Aryabhata I (yr. 499), Brahmagupta (yr. 628), Sridhara and many later studies have been used the word *hanana*, *parasparakrtam* used in Bakhshali Manuscript. Define the multiplication as ”the process of repetitive addition of multiplicand as many times equal to the multiplicator” mention in *Aryabhatiya & Lilavati* where the term *gunya* used for multiplicator and *gunaka* for multi-plicand the product of two is termed as *gunana-phala.* The author has mention the various Vedic methods discussed by early age mathematicians like *gomutrika, khanda, bheda & ista* by Brahmagupta, *kapata-sandhi, tastha, rupa-vibhaga & sthana-vibhaga* by Sridhara, Mahavira, Bhaskara II, *gelosia* by Ganesa etc. Various distinct old Hindu methods of multiplication from the pages of Vedic litrature discussed in detail which includes *kapata-sandhi, gelosia, tastha, sthana-khanda, gomutrika, istagunana.* Sharma et al. [42] discussed about the brahma sphuta siddhanta which mention the various methods of twenty arithmetic operations like addition, subtraction, multiplication, division, square, square-root, cube, cube-root, five standard forms of fractions, the rule of three, etc. The authors have discussed particularly four methods of multiplication gomutrika, khanda, bheda and ista. even the most common and known method kapata-sandhi has not been discussed by the author in his work. Gomutrika and sthana-khanda method of multiplication are consider to be resemble with each other, khanda method is explained in two different ways by breaking the multiplier into various parts. And it is also mention that ista gunana multiplication method is gone out of india and adopted by Arabs and in Europe. Colebrooke et al. [10] have discussed eight operations in arithmetic methods mentioned in LILAVATI, treatise of Bhaskaracharya. The authors have mentioned multiple examples based on the various method for performing these eight arithmetic operations. According to the author, LILAVATI mention the various method of multiplication which include *rupa-vibhaga* dividing the multiplicator into parts, *sthana-vibhaga* considering each digit of multiplicator separately and *ganesha* in which product goes along the compartments. In [43], the work provided the rules for solving problems involving linear equations, indeterminate equations of the second degree, arithmetical progressions, quadratic equations, approximate evaluations of square roots, complex series, problems of type $x(1-a_1)(1-a_2)\ldots.(1-a_n) = P$, the computation of the fineness of gold, income and expenditure, profit and loss. Various example of multiplication are mentioned including dvigunam, asta gunam, gunita jatam, gunita jata, anena gunitam jatam, phalam. The authors in [44], have discussed the Bija-Ganita of Bhaskara and also presented topics from Brahmagupta's Ganita and Bhaskara's Lilavati. In Arithmetic, various rules for performing fundamental operation, multiplication, division, squaring, square root, cubing, cube root, summation and vyutkalita are explained. Nowadays, many researcher are working on the utilization of various Vedic methods of arithmetic to propose architectures for adders, multiplexers to enhance the performance of the current binary machines. Basically binary machine work on 0 & 1 Leibniz et al. [45] in their work proposed the binary number system that is 0 and 1 and propose the method or performing all the basic mathematical operations such as addition, multiplication, subtraction and division based on binary number system. This theory prove to be the base for the development of binary computer machines after the invention of silicon. Vedic hindu mathematics also shows the evidences of binary number system in Pingala chandahsutra [8], which calculated the total number of arrangements of n things in m places. According to [12], the large percentage of CPU cycle utilization in performing multiplication and division operation. The authors have designed the multiplexer totally based on combinational logic for generating the product of two numbers. The basic idea for enhancing the speed of computation of multiplication process is to either reduce the number of summands or accelerating the process of formation/addition of summands. Thapliyal et al. [13] used the Vedic method named *Urdhva Tiryakbhyam* and proposed a design of multiplier and square architecture to enhance the speed of computation for performing the multiplication of two numbers and reducing the power consumption. The Vedic method is used for generating partial product in parallel and doing the summation of the partial products. The concept is based on generating the partial product at once and performing concurrent addition of all the partial products, the 4x4 multiplication procedure using *Urdhva Tiryakbhyam* is shown which can be generalized to nxn bit number, this parallelism reduce the time complexity for the operation. The author has used the Vedic method *Urdhva Tiryakbhyam* for calculating the square of a number. In [14], the author has design the multiplexer based on Vedic multiplication method called *urdhva triyakbhyam* to enhance the speed of computation and low the power consumption. Two Vedic methods namely *urdhva triyakbhyam* & *nikhilam sutra* has been discussed in detail. The design of multiplexer proposed in the paper based on *urdhva triyakbhyam* multiplication method shows the problem of high propagation delay in case of applying multiplication operation for large numbers. Proposed multiplexer work on the basis of generating all partial products and summing up in a single step. The work is compared with array multiplier on the basis of number of multiplication and addition operations. Dhillon et al. [15] proposed a reduced-bit multiplication algorithm using the *urdhva triyakbhyam*

& *nikhilam sutra*, both the Vedic multiplication methods are discussed in detail and also mention various other multiplication method from the Vedic literature. Author mentioned that the multiplier design on the basis of *urdhva triyakbhyam* is very much similar to the array multiplier and leads high carry propagation delay in multiplying large numbers, where as the multiplier designed on the basis of *nikhilam sutra* can over come the problem of carry propagation delay. And consider the multiplier based on *nikhilam sutra* as more efficient in performing multiplication among large numbers. In the proposed reduced-bit algorithm author basically use the process of right shift operation with both multiplicand and multiplier to remove the consecutive zeros from the least significant bits. The authors in [46], discussed that is conventional booths algorithm generates an extra partial product bit at least significant position of each row pf partial product that generate irregular array of partial product. This overhead of generating partial product bit caused increase in delay, area and power consumption. The author has proposed a modified booths algorithm which generated a regular shaped partial product array as compared with [47, 48] that reduced the overhead of partial product bit so that area, delay and power consumption also get reduced in modified booths algorithm. Pradhan et al. [16] explain the multiplication methods *urdhva triyakbhyam* & *nikhilam sutra* in detail and discussed about the 16x16 multiplexer architecture based on *urdhva triyakbhyam sutra*. The author modify the 16x16 multiplexer based on *urdhva triyakbhyam sutra* using *nikhilam sutra* for reducing the propagation delay. A new high speed architecture has been proposed in [18], for the multiplexer based on the Vedic method of multiplication nikhilam sutra. The proposed architecture of multiplexer is based on the concept of finding the complement(difference) of large operand with its nearest base. The multiplication and addition of the complement of two large operands was performed instead of multiplication of two large operands. The result of the architecture was also compared with the earlier multiplier architecture to show the better utilization and high speed of the binary machine. The author in the paper conclude that the proposed multiplication approach work more efficient when the operands are greater than the half of the base of the operands. James et al. [19] proposed a design of 2-bit multiplexer using *Urdhva Thirayakbhyam sutra* based on memristive threshold logic, where memristive threshold cell is designed using operational amplifier and CMOS. Where operational amplifier is used as voltage comparator which takes an input voltage and compare it with reference voltage and give the output voltage accordingly. The author concluded that the multiplexer based on Vedic method *Urdhva Thirayakbhyam sutra* using MTL will be more efficient for the multiplication of large numbers, require low power and reduced the chip area. The author in [49], proposed the multiplier based on *Urdhva-Tiryakbhyam* method using carry select adder and XOR gate. The basic mathematical calculation of *Urdhva-Tiryakbhyam* method is discussed in the paper and design of 16x16 multiplier is shown and conclude that the multiplier consume low power and enhance the speed of multiplication. The work is compared with [50] on the basis of latency, in which author has discussed two multiplexer design based in UCLSA (Uniform Carry Select Adder) & VCSLA (Variable Carry Select Adder) and compare both the design. The comparison conducted by author conclude that multiplexer based on UCLSA reduce the power, delay and area as compare to the multiplexer based on VCSLA. In [21], the importance of an algorithm and the feasible hardware for doing faster multiplication and addition in the field of Digital Signal Processing (DSP) has been discussed. The *Urdhva tiryakbhyam* considered as a fast method of multiplication and discussed in detail and the proposed algorithm is based on calculating the sum of partial product in parllel. The work is compared with the Booth Wallace multiplier on the basis of combinational delay. Also, the authors have mentioned the use of Vedic multiplication methods and their implementation on 8085 and 8086 microcontroller. The author in [51], proposed a design of multiplexer based on Vedic *Karatsuba* sutra and compared the multiplexer design with the already developed multiplexer based on Vedic *Nikhilam* sutra on the basis of efficiency. Mathematical theory is defined for both the methods in the article, *Karatsuba* sutra is discussed in detail with example but no such circuit design has been proposed based for multiplexer based on the Vedic method. In [52], multiplication operation is regarded as one of the important operation widely used in various fields like image processing, correlation, signal processing etc [53] so the importance of efficiency of multiplication operation is considered. The author has considered *Nikhilam sutra* [54] from the pages of of Vedic literature to enhance the performance of multiplication. No such multiplexer circuit is proposed based on the Vedic method, as the mention Vedic method for multiplication is based on addition so the article conclude that the multiplexer design on the basis of *Nikhilam sutra* would be more efficient then the existing. The author in [55], proposed a multiplication algorithm based on *Nikhilam sutra* used for binary multiplication. Special case of binary multiplication is carried out using Nikhilam algorithm when both the multiplier and multiplicand are same, on the basis of Nikhilam Squaring algorithm further multiplication is performed. Nikhilam algorithm can be used for the multiplication of number with certain range and for very large numbers Karatsuba algorithm [56] would be more efficient. In [53], multiplication is considered as the key operation that require high performance, to enhance the performance author proposed a multiplier based on the Vedic methods *Urdhva and Nikhilam* sutras. The mathematical model for both *Urdhva and Nikhilam* sutras has been explained, objective of the proposed work is to enhance the computation speed of multiplication operation on the basis faster addition [57]. Drawbacks of various multiplexers has been discussed like efficiency of serial multiplexers are lower than that of parallel multiplexer but at the same time parallel multiplexers(Wallace tree multiplier, Booth multiplier, Array multiplier etc) [57, 58] generate higher delay with high power consumption and require large number of logic gates.Summary of related work is shown in Table 2.1, where we noticed that most of the researcher have proposed the design for multiplexer for reducing the processing time of multiplication operation using mainly two Vedic methods, namely, urdhva tiryakbhyam sutra and nikhilam sutra.

Table 2.1: Summary of Related Work

Author & Year	Methodology Used	Proposed Work
Dhillon et al. [15], 2008	Urdhva tiryakbhyam and Nikhilam	Reduced bit multiplication algorithm and Multiplexer architecture
Jacobsohn, D [12], 1964	Purely combinational logics, using diode-transistor logic	Design of Multiplier
Kuang et al. [46], 2009	Enhance the conventional Modify Booth encoding	Reducing the area, delay and power consumption of conventional Modify Booth encoding
Pradhan et al. [18], 2014	Nikhilam sutra	Design of Multiplexer, for increasing the speed of multiplication operation.
Pradhan at al. [16], 2011	Urdhva tiryakbhyam sutra and nikhilam sutra	Architecture of 16x16 multiplexer using urdhva tiryakbhyam sutra and extends the existing 16x16 multiplexer based on nikhilam sutra
Pushpangadan at el. [21], 2014	Urdhva tiryakbhyam sutra	Discuss the importance of Vedic approach of multiplication in field of DSP and architecture based on 8085, 8086 microcontroller.
Saha et al. [17], 2014	Dhvajanka sutra	Transistor based architecture of division circuit, for reducing the propagation delay and power consumption.
Thapliyal et al. [13], 2004	Urdhva Tiryakbhyam sutra	Design of multiplexer to reduce the computation time for multiplication operation of two numbers.
Tiwari et al. [14], 2008	Urdhva Tiryakbhyam sutra	Discuss various Vedic multiplication methods and their importance. Extended the urdhva tiryakbhyam sutra for binary multiplication.

2.2 Multivalued Logic

Many researcher propose the concept of MVL to enhance the performance of the binary machine and overcome the flaws of existing binary computers. From the past few decades, researchers have proposed various circuit designs for logic gates based on the concept of ternary logic that is 3 logical states instead of two which can be an alternate for binary logic to enhance the speed of computation [59], to reduce the power consumption, delay and chip area. As the current binary circuitry require large number(plentiful) of electronic components [24], interconnection overhead [25, 26, 32, 60, 61, 37, 62] and large chip area [31, 63]. MVL can be considered as an increasing number of states in the current logic, which can be ternary or fuzzy. Many researchers consider fuzzy [23, 64, 28, 65] logic as infinite numbers of states in term of MVL, but in case of a mathematical model based machine base, 10 should be considered as a generic machine for computation of all kinds of arithmetic operations. Ternary inverter circuit proposed by many authors and considered as the building block for various other ternary circuits like a ternary adder, decoder, etc [30, 33, 36]. Numerous researchers consider the symmetry of sign conversion [66, 67, 68, 69] to introduce the concept of ternary arithmetic considering -1,0 and +1 as the 3-state logic. Somewhere the ternary bits also represented as N(-1), Z(0), and P(1), Parhami et al. [70] proposed the binary representation of balanced ternary numbers and each ternary digit is represented by at least two binary bits. And consider three states of ternary logic and N, Z, P instead of numeric. It also depicted the truth table for sum and carry of half adder based on 3-state logic using characters(N, Z, P). In initial state ternary logic is considered as an extension of 2-state that is binary logic to treat the ambiguous or uncertain state of a machine. Yamamoto et al. [71] stated that it is not always possible to represent the status of the machine in 2-states that is ON/OFF, the machine can be in the uncertain state also and this ambiguous state is considered as third state logic. This third state is represented by 1/2, the author categorizes ternary logic into two functions that is regular and the majority, where regular ternary logic is considered as a suitable way for treating the ambiguity, means when the state of the machine is not defined. Also mention about the use of B-ternary logic [72] function for the detection of hazards. The $\leq$ is considered as an ambiguity relation and defined as $0 \leq 1/2$ and $1/2 \leq 1$. In [27], the author also mentions ternary logic suitable for treating the ambiguous state where it is not possible to decide the state of the machine is 0/1 or ON/OFF or TRUE/FALSE. The author has considered the transition state from 0 to 1 or from 1 to 0 as the ambiguous state in which we can't define the state of the machine in either 0 or 1. And another example of an uncertain state is considered as the initial state of any circuit which can be in either of the state between 0 and 1 so, it is considered as an uncertain or ambiguous state by the author which can be treated with ternary logic. The main objective of considering the uncertain or ambiguous state of a circuit is to design a logic circuit with the properties of fault tolerance. And for that binary logic is extended for third uncertain state and it is considered as ternary logic. Allen et al. [24] have discussed the requirement of MVL over binary logic. Depicted about the drawbacks of binary machines which include the requirement of a large number of electronic components for the circuitry and mention that a binary model is not a suitable mathematical model for all integer values. The Boolean algebra based on the binary logic cant be used or extended for other higher base mathematical models so, the computers based on binary logic cant be considered as the perfect machine. Also mention about the concept of multivalued logic [73, 74, 75, 76] where the realization of the ternary switching circuit is proposed on the basis of diodes and transistors. And propose a switching theory based on the concept of ternary logic. Smith et al. [31] mention the problem of interconnections in the current binary machine which grows exponentially high with an increase in the number of components that grow the chip area. And stated the solution as the introduction to MVL for reducing the interconnection overhead in case of binary circuits. The author has provided the notations for MVL taking an example for base 4 circuits considering four logical states that are $0, 1, 2, 3$ and stated that the state varies in a circular manner. This means that every state in mention set $1, 2, 3, 0$ is always one logic higher than state $0, 1, 2, 3$. In [32], the author discusses the unacceptability of the binary logic as it doesn't fit with the mathematical decimal model which makes the requirement of MVL. The current binary machine works on the strings on 0's and 1's it is only because of the invention of diode based on silicon which works as a switch ON/OFF which is not according to the mathematical model based on decimal numerals. The author stated that machines based on higher radix exponentially reduce the wiring complexity and reduction in ternary logic gates. Also mention about the types of circuits for MVL that are voltage and current based and stated about the benefits of higher state logic in the reduction of interconnection cost. Choi et al. [23] propose the methodology for developing the circuits based on the concept of MVL for enhancing the performance or increasing the speed of computation. As in the case of a binary machine, the speed of computation depends on the speed of the clock pulse which is reaching its threshold. Also proposed the post algebra, adder circuit with truth table for higher radix. The author basically considers base four logic as sending two-bit on each wire in parallel and memory device storing two-bit at a time. Proposed the design of logic gates and flip flop based on base four and consider the representation of base four as $00, 01, 10, 11$ which depicts $0, 1, 2, 3$. The logic stated in the paper is basically binary model, it cant be considered as a separate logic based on higher radix, the author just proposed a model to take two binary bits at a time on each wire instead of one bit. In [77], author propose three operators that are α, β and γ for ternary logic where α is considered as AND operator which return minimum among the inputs β and γ as relative I_{max} and I_{min}. Circuits are proposed based on the npn transistors for the realization of α, β, and γ operators. It also proposed the clockwise shifter circuit which proved as the basic building block of ternary logic. The author has proposed the new operators based on post ternary logic, all operators are currently based to make the operator workable with both analog and digital circuits at high speed. Lin et al. [78] proposed the ternary gates based on carbon nano tubes FETs, consider ternary logic as a better alternative to replace the machines based on binary logic. Ternary logic is simpler and energy-efficient than the conventional binary machines also reduced the chip area and the interconnection. The author on the basis of numerous advantages of ternary logic over binary, proposed various ternary operators (inverters, NAND, NOR) using resistive load CNTFET where the ternary inverter requires three types of inverters namely negative ternary inverter (NTI), standard ternary inverter (STI) and positive ternary inverter (PTI). Also propose the ideology to use the ternary operator with the existing binary operators to achieve a higher speed of computation and reduction in the power consumption, on the basis of the ideology proposed the design of ternary adder and multiplexer. The variation in the supplied voltage is based on the variable threshold voltage of CNTFET based on its diameter. Balla et al. [37] have proposed the ternary logic family based on MOS transistor which includes inverter circuit, NAND, and NOR gates. The inverter circuit is considered as the basic building block of other ternary operators. On the basis of ternary operators, the design of circuits for arithmetic and memory units are proposed. The author proposed a new inverter circuit that is a general ternary inverter (GTI) based on the same concept of previously developed three different inverter circuits namely simple ternary inverter (STI) , positive ternary inverter (PTI) and negative ternary inverter (NTI) using MOS transistors. And compared with proposed inverter circuits [35, 79, 80] showing the advantage of the reduction in power dissipation with the proposed alternative circuit of the simple ternary inverter (STI). In [80], ternary algebra is proposed based on three logical values that is 0, 1 and 2 representing the different voltage level(low, intermediate & high). The author has not mentioned the actual voltage level distribution for representing the three logical states for ternary logic. Algebra is proposed for various previously developed operators of ternary logic in which simple ternary inverter (STI) , positive ternary inverter (PTI), negative ternary inverter (NTI), a forward diode (FD) and reverse diode (RD) are considered in the category of basic unary operators and TOR, TAND belong to operators accepting multiple inputs. Ternary inverters are considered as the basic circuit for building other ternary logic gates. A ternary inverter circuit is presented using COS/MOS by inserting resistors between the two complementary type transistors with common drain, one input, and three output representing three levels of voltage (negative, zero, and positive). The ternary inverter circuit realized as a combination of three types of ternary inverters. FD and RD, a simple diode with shunt resistance. DeMorgan's

Theorem for ternary logic is represented on the basis of three inverters(STI, NTI & PTI). Mouftah et al. [35] represent the ternary logic considering +1V as high, 0V as intermediate and +1V as high. On the basis of different voltage level the circuit for simple ternary inverter (STI) , positive ternary inverter (PTI), and negative ternary inverter (NTI) is proposed using two opposite MOS transistors (p-type and n-type) with common drain and one resistor. The load resistor is connected at the output with the fixed voltage and according to the voltage, the circuit behaves either STI, PTI or NTI. On the basis of three inverter circuit, separate ternary NOR gate that is simple TNOR, positive TNOR & negative TNOR has been realized. The author concludes that the proposed ternary operator consumes less power and can be used for building a ternary digital logic system. Heung et al. [33] have proposed a circuit for ternary inverter STI, NTI, and PTI to further reduce the power consumption and increase the speed of circuit using both enhancement and depletion MOS transistor (DECMOS [81]) without using any resister. The author has proposed a circuit for inverter, NAND & NOR considering as basic ternary operator and building block for ternary digital system. And considering high (+1V), intermediate (0V) and low (-1V) three different levels of voltages for representing the ternary logic (0, 1, 2), the article has not represented any range of actual voltage for representing the ternary logic. Ternary adder circuit is designed as an application of the developed ternary operators The work has been compared for power consumption with the inverter circuits [35, 82]. Mouftah et al. [83] have taken +4V, 0V, and -4V as high, middle, and low level to represent three different levels of voltage to represent 3-states of ternary logic. Proposed the design of ternary operators (inverter, NAND ad NOR) based on MOS transistors to simplify the already proposed ternary circuits and lower the power consumption. Inverter circuits are considered as the building block of other ternary logic gates, which actually consist of three different inverter circuits namely simple ternary inverter (STI), negative ternary inverter (NTI) and positive ternary inverter (PTI) comprising of two resisters and mos transistor each of p-channel & n-channel. On the basis of ternary inverter proposed the cheaper circuit of already proposed [82] Jk arithmetic and T-gate. Srivastava et al. [38] proposed the design of a ternary adder based on CMOS and consider inverter circuits that are simple ternary inverter (STI), negative ternary inverter (NTI) and positive ternary inverter (PTI) as a building block for the circuit. The author claims the implementation of inverter circuits without using as external resistance, but use the concept of length-to-width ratio [34] to provide the internal resistance to the transistor for fetching the required values suitable for the ternary inverter. On the basis of the ternary inverter circuits, further ternary NOR and NAND gate circuit has been designed. In [78], the author has proposed the circuit of the inverter (STI, PTI, and NTI) using carbon nanotubes FETs instead of CMOS without an external resistor. The author uses the concept of changing the threshold voltage of the transistor according to the diameter of CNT [84]. An external resistor is actually replaced by varying the internal threshold voltage of the transistors. It has been observed that many researchers propose the circuit of inverter based on the concept of the simple ternary inverter (STI), negative ternary inverter (NTI), and positive ternary inverter (PTI)., which are actually three different structures for the inverter circuit. Moreover, the authors in multiple research papers claim the removal of external resistance in the circuit, but they have used the variable resistance transistors instead of external resistance. Still, the circuit is having the overhead of variable threshold based transistors. The author in [61], discussed the multivalued I^L circuits based on current-mode, in which the current threshold is used for defining multivalued logic for a machine. But the multivalued machine based on current mode faces the problem of tolerance that makes it almost disappear. The multivalued ROMs have been designed based on the combination of transistors maintaining different threshold values for multivalued logic, but the varying threshold for different transistors increases the complexity of the circuit. Takagi et al. [60]

discuss the regular functions based on the ternary logic to deal with the ambiguity in the binary logic, where the initial and/or transient state of the logic is considered as an ambiguous state that is uncertain to define the finite state of a machine at that time. The Kleen regular ternary logic functions [27] discussed in which the uncertain state has been represented by 1/2 for representing the ambiguity in the existing binary system. The author proposed the multivalued regular function such as an extension of Kleen logic functions which has already been extended/modified by many researchers in the order [85, 86, 87, 88, 89]. In the proposed work [36] new circuit for ternary inverter circuit is defined using carbon nanotube FETs where the threshold voltage of the transistor is varied according to the diameter of the CNT for achieving the required results of previously developed ternary inverter circuit(STI, NTI & PTI). The author stated that the proposed design of the inverter circuits eliminates the requirement of large resistors [90] to have a different level of voltages for multi-valued logic. But in the proposed work author has derived the results of the ternary inverter logic by changing the threshold voltage level for different transistors on the basis of CNT diameter which increases the complexity of the circuit. Ternary logic denoted by 0, 1, and 2 representing false, undefined, and true, the author has not defined the actual voltage levels for ternary logical values. Phanindra et al. [40] proposed ternary adder circuits based on the ternary inverter circuits (STI , NTI & PTI) and derive the design of ternary NAND and NOR gate based on the inverter gate. The concept of CNTFETs is discussed in detail, with variable threshold voltages according to the diameter of the transistors [91]. The Standard Ternary Inverter (STI) is defined with truth table representing the three different voltage all in the range 0V - 1V where 0V depicts logic 0, 0.5V for logic 1, and 1V for logic 2. The author has not defined any range for ternary logic as logic 0 can not be taken as absolute 0V. On the basis of STI the circuits for ternary NAND, NOR gate are shown in the article. Static hazards in ternary combinational circuits are concerned in [92] where the author has developed the techniques for detecting and eliminating the hazards in circuits based on ternary logic. The article seems to be an extension of the theory proposed in [93, 94] about static hazards in binary logic based circuits. For representing the ternary logic the author has taken three logical values $0, 1/2, 1$, in which 0 & 1 is interpreted as the usual state of a machine the same as in binary and 1/2 as the intermediate state or the transient state. Dawley et al [26] prefer the MVL over the conventional binary logic on the basis of power consumption, interconnections, and chip area. The author considers the Quaternary Logic for MVL and represented the states as $0, 1, 2, 3$ the objective of the work is to reduce the power consumption and the burdensome of interconnections in the binary machines. The concept of split circuits is proposed which basically 1X4 decoder having one input line and four output lines, the author has not provided detailed circuits for the same. And no operators such as inverter circuits, NAND, NOR based on base 4has been proposed in the article. The above literature is summarized in Table 2.2, where it has been noticed that seperate three inverter circuits are proposed that is Simple Ternary Inverter (STI), Positive Ternary Inverter (PTI), and Negative Ternary Inverter (NTI). Which can not be considered as the extension of binary machine and no scope for further extension for higher radix. Based on same inverter circuit further operators are proposed for 3-state machine which don't maintain any pattern or common logic. Outputs are drawn on the basis of variable resistance (internal or external).

Table 2.2: Summary of Related Work

Author & Year	Methodology Used	Proposed Work
Phanindra et al. [40], 2016	COS/MOS by inserting resistors between the two complementary type transistors with common drain, one input and three output representing three level of voltage (negative, zero and positive).	Ternary inverters simple ternary inverter (STI) , positive ternary inverter (PTI), negative ternary inverter (NTI) are considered as the basic circuit for building other ternary logic gates.
Mouftah et al. [80], 1976	Two opposite MOS transistors (p-type and n-type). The load resistor is connected at the output with the fixed voltage	The circuit for simple ternary inverter (STI) , positive ternary inverter (PTI) and negative ternary inverter (NTI)
Balla et al. [37], 1984	MOS transistors	General ternary inverter (GTI) based on the same concept of three different inverter circuits namely simple ternary inverter (STI) , positive ternary inverter (PTI) and negative ternary inverter (NTI)
Mingoto et al.[30], 2006	DECMOS without using any resister.	Circuit for STI, NTI and PTI to further reduce the power consumption and increase the speed of circuit
Mouftah et al.[82], 1984	Taken +4V, 0V and -4V as high, middle and low level to represent three different levels of voltage to represent 3-states of ternary logic.	Proposed the design on ternary operators (inverter, NAND and NOR)
Zvonko G Vranesic and Kenneth C Smith.[32], 1974	Comparison of binary logic with MVL	Discuss the benefits of higher radix that is MVL over binary logic. Mention that the circuit based on the MVL require less wiring complexity and large reduction in the inter-connections
Lin et al.[29], 2009	carbon nanotube FETs	Design ternary logic inverters.
Srivastava et al.[38], 1996	CMOS technology.	Circuit for ternary adder based on inverter circuits.
Lin et al.[78], 2011	carbon nanotubes FETs instead of CMOS without any external resistor but author use the concept of changing threshold voltage of the transistor according to the diameter of CNT.	Design ternary logic gates and arithmetic circuit NAND, NOR gates based on old inverter circuits (STI, NTI and PTI)
Allen et al.[24], 1968	Depicted about the drawbacks of binary machines which include the Requirement of large number of electronic components for the circuitry.	Conclude that binary model is not a suitable mathematical model for all integer values.
Vranesic et al. [32], 1974	Decimal Number System	Binary logic as it doesn't fit with the mathematical decimal model which make the requirement of MVL
Choi et al.[23], 2013	Considering 2-bits at a time to operate.	Binary to four logic state based on binary bits.
Serran et al.[95], 1997	Current based circuit.	α ,β and γ for ternary logic where α is considered as AND operator which return minimum among the inputs β and γ as relative Imax and Imin.

CHAPTER 3
AM-MULTIPLICATION ALGORITHM & BASED MULTIPLEXER

3.1 AM-Multiplication Algorithm

In the section, proposed AM-Multiplication Algorithm is discussed in detail which is motivated from vedic literature for decimal multiplication and extended for the multiplication of binary numbers. Algorithm can be used for the current computer machine based on binary number system. As multiplication operation is one of the important operation and take more time as compare with other arithmetic operations, so algorithm is proposed with concern of reducing processing time for performing multiplication operation for binary based machines. The work is compared with other multiplexers designed totally based on some method from vedic literature. Almost all authors just provide the multiplexer design referring mainly two methods that is *urdhva tiryakbhyam* and *nikhilam* sutra, no modifications or nor any other novel approach is derived from vast vedic literature. Circuit for multiplexer is also designed based on AM-Multiplication algorithm and compared with other multiplexer circuits which include multiplexers based on vedic methods and multiplexer used in current binary machine on the basis of number of components(gates) and gate delay. The algorithm basically consist of one main algorithm that is am-MULTIPLCATION and two sub algorithm MIN & MAX. The complete procedure of finding the final resultant after performing multiplication of two binary numbers, involve various steps that are depicted in the work flow as shown in Fig. 3.1. Which includes development of sets and equations required in finding the multiplication of two binary numbers.

In rest of the chapter, development of sets & equations and am-MULTIPLICATION algorithm is discussed in detail.

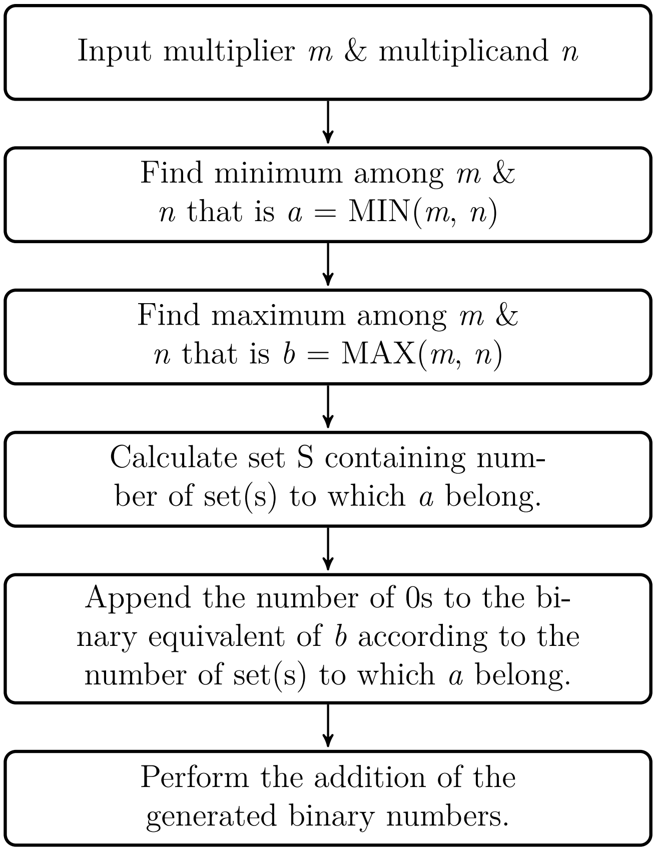

Input multiplier m & multiplicand n
Find minimum among m & n that is a = MIN(m, n)
Find maximum among m & n that is b = MAX(m, n)
Calculate set S containing number of set(s) to which a belong.
Append the number of 0s to the binary equivalent of b according to the number of set(s) to which a belong.
Perform the addition of the generated binary numbers.

Figure 3.1: Work Flow representing the basic steps involved the proposed am-Multiplication algorithm.

3.1.1 Development of Sets and Equations

The development of various sets are required in the proposed work so, firstly the sets are calculated manually then separate equation is developed for generating particular set of numbers and finally one general equation is developed for generating all set of numbers. So, firstly we calculated the required four sets S_0 to S_3 manually for performing the multiplication operation for small decimal numbers. The generated values of the following sets are shown below

$S_0 = \{1\}, \{3\}, \{5\}, \{7\}, \{9\}, \{11\}, \{13\}, \{15\}\ldots$

$S_1 = \{2, 3\}, \{6, 7\}, \{10, 11\}, \{14, 15\}\ldots$

$S_2 = \{4, 5, 6, 7\}, \{12, 13, 14, 15\}, \{20, 21, 22, 23\}\ldots$

$S_3 = \{8, 9, 10, 11, 12, 13, 14, 15\}, \{24, 25, 26, 27, 28, 29, 30, 31\}\ldots$

Set S_0 is basically consist of odd numbers starting from 1, with the missing starting number that is 0 and the gap of one number like 2 is missing between 1 and 3, similarly 4 is missing in 3 and 5 and so on. More specificity we can say set S_0 is the combination of various sets, each set is containing 2^0 value and 2^0 missing value between the every two immediate sets.

In the same way S_1 can be considered as the combination of various sets and each set is having 2^1 continuous numbers starting from set {2, 3} that is set {0, 1} is missing. And moreover each pair of sets is having a missing set of 2^1 continuous numbers like, {4, 5} is missing between the sets {2, 3} and {6, 7} numbers similarly {8, 9} is missing between {6, 7} and {10, 11} and so on.

Similarly set S_2 consisting of various sets and each set is having 2^2 continuous numbers starting from set {4, 5, 6, 7}, set {0, 1, 2, 3} is missing. And each two sets is having a missing set with 2^2 continuous numbers. As shown above in set S_2, {8, 9, 10, 11} is missing set between sets {4, 5, 6, 7} and {12, 13, 14, 15}, in the same way {16, 17, 18, 19} is missing set between {12, 13, 14, 15} and {20, 21, 22, 23}.

Alike above, set S_3 is started with the missing set of 2^3 continuous number from 0 to 7, therefore the first set of S_3 series is {8, 9, 10, 11, 12, 13, 14, 15}. Now similar to the above cases each pair of set is having a missing set of 2^3 continuous number such as {16, 17, 18, 19, 20, 21, 22, 23} is missing set of continuous numbers between the sets {8, 9, 10, 11, 12, 13, 14, 15} and {24, 25, 26, 27, 28, 29, 30, 31} and so on. As it can be seen that it is very tedious and time taking task to manually generate large number of required sets with long series of values. Therefore, different equations are developed for generating values of different sets.

The general equation(s) are shown below for finding the values in particular set, where O is the set of odd numbers.

O = {1, 3, 5, 7, 9, 11, 13...}

$S_0 = \{2n + 1, n \in N \cup \{0\}\}$

$S_1 = \{2^1 * i + j\}$ where $i \in O$ & $\forall\ i \in O$, we have $0 \leq j \leq 2^1 - 1$.

$S_2 = \{2^2 * i + j\}$ where $i \in O$ & $\forall\ i \in O$, we have $0 \leq j \leq 2^2 - 1$.

$S_3 = \{2^3 * i + j\}$ where $i \in O$ & $\forall\ i \in O$, we have $0 \leq j \leq 2^3 - 1$.

$S_4 = \{2^4 * i + j\}$ where $i \in O$ & $\forall\ i \in O$, we have $0 \leq j \leq 2^4 - 1$.

Still its not possible to develop separate equation for generating values of every set, finally a single general equation is developed for generating n number of sets. This single equation can be used for finding values of n number of sets like from S_0 to S_n.

$$S_n = \{2^n * i + j\}\ where\ n\ \in N \cup \{0\}\ i \in O\ \&\ \forall\ i \in O\ we\ have\ 0 \leq j \leq 2^n - 1. \quad (3.1)$$

In the above equation for finding all sets from S_0 to S_n where n can be any number from the set of natural number N including 0,i is any number from the set of odd numbers O and j lie between $0 \leq j \leq 2^n - 1$. For example if we want to find the values of set S_2, then the value of n will be 2 so for each value of i we consider all value of j from 0 to $2^2 - 1$ according to equation 3.1. After calculating the values for set S_2 according to the variables i and j we get following set.

$$S_2 = \{2^2*1+0, 2^2*1+1, 2^2*1+2, 2^2*1+3, 2^2*3+0, 2^2*3+1, 2^2*3+2, 2^2*3+3...\}$$

Similarly any number of sets can be generated from equation 3.1. According to the work flow shown in Fig. 3.1, now there is the question to find to how many set(s) minimum among multiplicand and multiplier will belong to, which can be calculated using proposed set of three equations as shown below (3.2 - 3.4).

$$x = 2^i * p\, where\, i \in N, \tag{3.2}$$

$$y = x - (x\%2^i), \tag{3.3}$$

$$p = y/2^i, \tag{3.4}$$

In equation 3.2, x is the minimum value among the multiplier and multiplicand and i is any natural number starting from 0. Once we have the value of x the value of p is calculated for i = 0 and if the value of p is any non-fractional odd number then it is conclude that x belong to set S_0. Else if the value of p is any non-fractional even number then it is considered that x does not belong to set S_0. And if the value of p is any fractional number then the value of y is calculated using equation 3.3 and after that the value of p is recalculated using equation 3.4, whose resultant would always be a non-fractional even or odd number. And on the basis of the value of p which is calculated using equation 3.4 it is decided whether x belong to S_0 or not. So the final conclusion is if the value of p is any non-fractional odd number than $x \in S_0$ and if it is any non-fractional even number than $x \notin S_0$. Similarly for checking whether x belong to set S_1 or not the value of p is calculated for i = 1 and if it is found that the value of p is any non-fractional odd number then we conclude that x belong to set S_1. Else in case the value of p is any non-fractional even number then x does not belong to set S_1. Apart from any non-fractional even or odd number if we get p as fractional number then y is calculated using equation 3.3 and on the basis of y we recalculate the value of p using equation 3.4 which will always be either a non-fractional even or odd number. On the basis of the above calculation if the recalculated value of p is non-fractional odd number than x considered to be in set S_1 else not.This process of finding total number of set(s) to which x belong continue till the value of x ¿= 2^i. After this process we have resultant set S containing the number of set(s) from $S_0\ to\ S_n$ where n is any natural number, to which x belong. The number x can belong to one or multiple set(s) but sets are designed in such a way that x always belong to at-least one set.

Here, one theorem is proposed which is utilized in the operation of performing the multiplication for the binary numbers.

Theorem. If we add 0 at the end of any number n of base m where $m \leq 10$ then the decimal equivalent of the resultant number will be m times the decimal equivalent of number n.

Proof. Let the number be n of base m that is n_m and the decimal equivalent of the number n is x_{10}. So, if we add 0 at the end of number n then the decimal equivalent of the number n will be $(m.x)_{10}$.

For example,

$$(111)_2 = (?)_{10}$$

$$1 * 2^2 + 1 * 2^1 + 1 * 2^0 = (7)_{10}$$

Now, after appending 0 at the end of the binary number the decimal equivalent of the binary number will be $(14)_{10}$ as shown below.

$$(1110)_2 = (14)_{10}$$

$$1 * 2^3 + 1 * 2^2 + 1 * 2^1 + 0 * 2^0 = (14)_{10}$$

And 14 is two times of the decimal equivalent of $(111)_2$ that is 7.

Let's take another example

$(13)_4 = (?)_{10}$

$1 * 4^1 + 3 * 4^0 = (7)_{10}$

Now, after appending 0 at the end of the number that is 13 the decimal equivalent of the same will be $(28)_{10}$ as shown below.

$(130)_4 = (?)_{10}$

$1 * 4^2 + 3 * 4^1 + 0 * 4^0 = (28)_{10}$

And 28 is four times of the decimal equivalent of $(13)_4$ that is 7. The above two example shows that if 0 is appended at the end of any number $(n)_m$ than the resultant number will be $(nxm)_m$.

3.1.2 am-MULITPLICATION Algorithm

The algorithm 1 that is am-MULTIPLICATION is main algorithm for performing the multiplication operation, take two decimal numbers as parameter that is m and n, where m is multiplicand and n is multiplier. The main objective of the algorithm is to find the set S which contains the number of set(s) to which minimum among m and n belong to. As shown in line 1 the procedure MIN as shown in algorithm 2 is called that return the minimum number among m & n which get stored in a. MIN algorithm takes two parameter that is m and n, return the minimum value among the passed parameters. In line 2 variable b store the maximum value among m & n which is return by the MAX algorithm 3. And it is checked in line 3 if any of the value among a and b is 0 then algorithm will return 0 as shown in line 4. If both a and b are having non-zero numbers then starting from $i = 0$ the value of p is calculated as shown in line 5 and 6. According to line 7 and 8 if the value of p is any odd number it is considered that a belong to the set S_0 and the set S is updated from NULL to S_0. And if the value of p is any even number then we consider that a does not belong to the set S_0 and set S will not be update and remain NULL as shown in line 9 and 10. Further if value of p is any fractional number then in that case the value of p

is recalculated according to the equations 3.3 and 3.4, further recheck the value of p is either even or odd to decide whether the value of a belong to set S_0 or not as shown in line 11, 12, 13, and 14. Now the value of i is incremented by 1 and checked whether the value of a is still greater or equal to 2^i as shown in line 15 and 16. If the condition mention in line 16 is true then same procedure is followed again for $i = 1$ from line 6 to 14 to check whether the value of a belong to set S_1 or not. And if the condition is false then as mention in line 18 resultant set S get returned which contains all the set(s) to which a belong.

Algorithm 1 am-MULTIPLICATION(m, n)

Require: Multiplier m and multiplicand n as parameter
Ensure: Find the number of set(s) to which the minimum among m and n belong.

```
 1: $a = \textbf{MIN}(m, n)$
 2: $b = \textbf{MAX}(m, n)$
 3: if $a == 0 \; || \; b == 0$
 4:         return 0
 5: $i = 0$
 6: $p = a \; / \; 2^i$
 7: if $p \in O$        // $O$ is set of all odd numbers
 8:     $S = \phi \cup S_i$        // update set $S$
 9: else if $p \in E$        // $E$ is set of all even numbers
10:     $S \neq \phi \cup S_i$        // don't update Set $S$
11: else if $p \in f$        // $f$ is any fractional number
12:     $y = a - (a \; \% \; 2^i)$
13:     $p = y \; / \; 2^i$
14:     goto step 6
15: $i = i + 1$
16: if $2^i \leq a$
17:     goto step 5
18: else exit
```

As shown below MIN algorithm 2 takes two parameter that is multiplier m and multiplicand n and return the minimum value among m and n. In case both the values m & n are equal then the algorithm will return m.

MAX algorithm 3 takes two parameter multiplier m and multiplicand n, return the maximum number among them.

Algorithm 2 MIN(m, n)

```
Require: Multiplier $m$ and multiplicand $n$ as parameter
Ensure: Find the minimum among $m$ and $n$.
If $m \leq n$
    return $m$
else
    return $n$
```

Algorithm 3 MAX(m, n)

```
Require: Multiplier $m$ and multiplicand $n$ as parameter
Ensure: Find the maximum among $m$ and $n$.
If $m > n$
    return $m$
else
    return $n$
```

After applying the am-MIULTIPLICATION algorithm on multiplier m and multiplicand n we get the set S which contains the total number of set(s) among S_0 to S_n where n is any natural number, to which a belong. Where a store the value return by MIN algorithm 2 as shown in line 1 of algorithm 1. Now we append the number of 0s according to the set S in the binary equivalent of maximum number among multiplier m and multiplicand n that si b return by an MAX algorithm 3 . The binary addition of all the generated numbers according to set S is performed to get the final multiplication result.

For better understanding lets take an example for performing the multiplication of two unsigned whole numbers and getting the binary resultant. Let the numbers are $m = 5$ and $n = 9$. Now, As shown in algorithm 1 procedure MIN 2 return the minimum among m & n that is $a = 5$. Now, according to the algorithm 1 we find the number of set(s) to which $a = 5$ belong as shown below.

$a = 5, b = 9, i = 0, S = NULL$

$p = 5\ /\ 2^0,\ p = 5$

$p \in \{odd\ \ numbers\}$, update $S = \{S_0\}$

As value of p is an odd number it means that $a \in \{S_0\}$ so we update the set S. Now we increment the value of i by one.

$i = i\ +\ 1,\ i = 1$

$p = 5\ /\ 2^1,\ p \in f$

As the value of p is any fractional number so we recalculate the value of p.

$y = 5 - (5\ \%\ 2^1),\ y = 4$

$p = 4\ /\ 2^1,\ p = 2$

$p \in \{even\ \ numbers\}$, dont update $S\ =\ \{S_0\}$

As the value of p is even number it means that $a \notin \{S_0\}$ so the set S will not be updated. Now increment the value of i by one.

$i = i\ +\ 1,\ i = 2$

$p = 5\ /\ 2^2,\ p \in f$

Again as the value of p is any fractional number so the value of p will be recalculated.

$y = 5 - (5\ \%\ 2^2),\ y = 4$

$p = 4\ /\ 2^2,\ p = 1$

$p \in \{odd\ \ numbers\}$, update $S\ =\ \{S_0, S_2\}$

As value of p belong to set of odd number it signifies that $a \in \{S_2\}$ so update the set S and increment the value of i by one.

$i = i\ +\ 1,\ i = 3$

Now $2^3 > 5$, so the process stop here.

And we get the final set $S\ =\ \{S_0, S_2\}$

According to the final set S, $a \in \{S_0, S_2\}$ append the number of 0s in binary equivalent of $b = 9$ that is 1001.

As a belong to set S_0 & S_2 so as per S_0 no 0 will be appended in the binary equivalent of $(9)_{10}$ that is 1001 and we get one binary equivalent number as 1001. Now according to set S_2 two 0s will be appended in the binary equivalent of $(9)_{10}$ so another binary number will be 100100. In general the number of set(s) to which a belong, each set would generate a binary equivalent according to it. So in this case we have two binary numbers that is 1001 & 100100.

Now we perform the binary addition of 1001 and 100100 which is equal to 101101, that is binary equivalent of the product of input numbers. To check the result, after converting the resultant binary number 101101 into the decimal we get $(45)_{10}$ which is actually the product of input numbers m & n.

3.1.3 Circuit Diagram for Binary Multiplexer based on am-MULTIPLICATION Algorithm

In this section, binary multiplexer circuit is designed based on am-MULTIPLICATION algorithm 1 in which once we get the set S which contain number of set(s) to which a belong. Where a is the minimum number return by MIN algorithm 2 among the parameter of multiplier and multiplicand. According to the number of set(s) in set S we upend the number of 0s to the binary equivalent of b, where b is the maximum value among the multiplier and multiplicand return by MAX algorithm 3. So we get number of binary constants which are equal to the number of set(s) in set S. Now the bits are provided in parallel to the serial shift registers as shown in Fig. 3.2 and the output of the serial shift registers are provided as input to the carry look-ahead full adder as shown in Fig. 3.3 to get the final result. In carry look-ahead full adder, carry generated with the current state is represented by CN and $C(N-1)$ represents the carry generated from the previous state. Therefore, $C(N-1)$ initially considered as 0 and the value of $C(N)$ will be equal to the value of $C(N-1)$ for the summation of the next two bits.

The carry look-ahead full adder accept three inputs, two bits for addition as inputs and third input is the carry generated(if any) from the addition of previous two bits represented by $C(N-1)$ as shown in the Fig. 3.3 where CN is representing the carry generated in the current state.

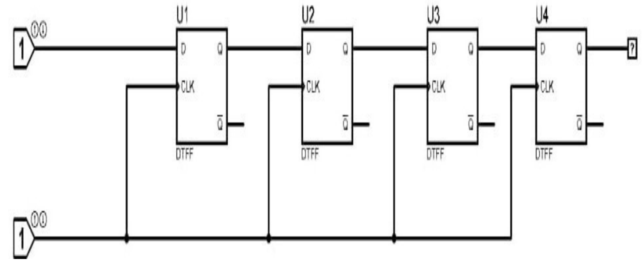


Figure 3.2: Shift Register to shift the binary number bit by bit.

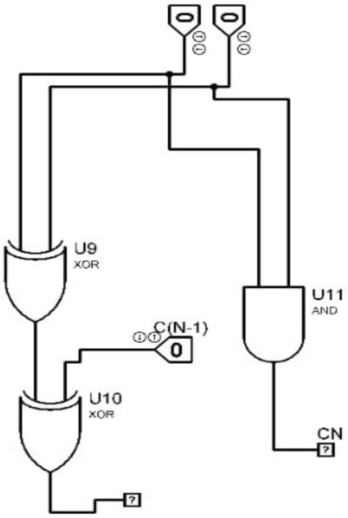


Figure 3.3: Carry Look-ahead Full Adder for the summation of generated carry and the input received form shift registers.

In Fig. 3.4 complete circuit for binary multiplexer based on am-MULTIPLICATION algorithm is shown in which binary input is given to the serial shift registers. And the output of the sift register is passed as input to the carry look-ahead full adder for bit by bit addition including carry bit generated in previous state represented by $C(N-1)$ and CN is representing the carry generated in addition of current bits and previous carry bit.

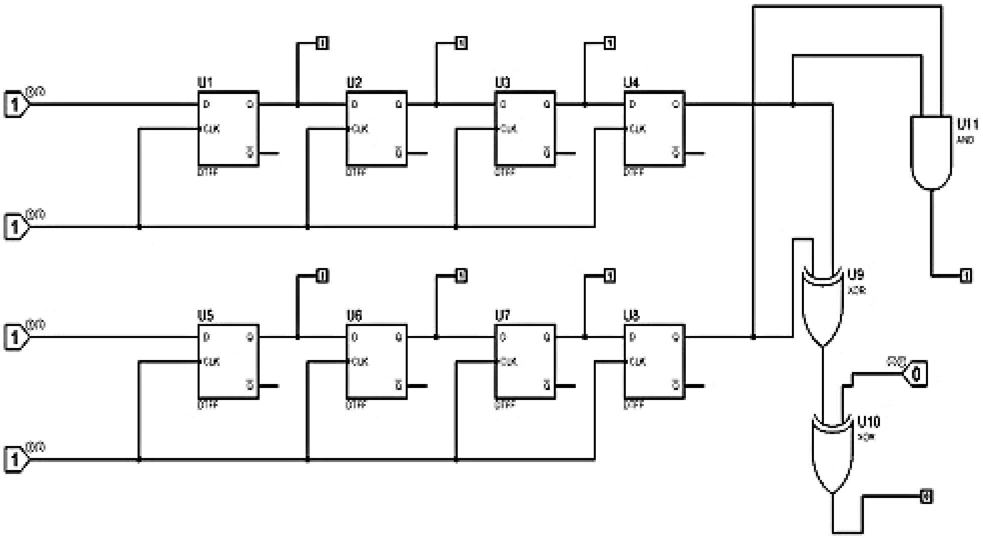


Figure 3.4: Circuit based on am-MULTIPLICATION algorithm in which the output of the shift registers and the generated carry of previous state act as input to carry look-ahead adder for binary addition.

CHAPTER 4
MULTISTATE LOGIC (3-STATE)

Current machines are based on the concept of 2-states that is binary logic, which means that state of a machine can be in either 0 or 1, ON or OFF state. In term of voltage 0 and 1 represents 0 and 5 volt respectively. From decades, processing speed of the binary machine remains one of the major challenge. As the performance of the 2-state machine depends on the state transition that is how fast transition occur from one state to another, ultimately the performance of the binary machine depends on the clock speed, now which is on it limit. So to further enhance the performance of binary machine and overcome some other major issues of 2-state machines which include plentiful of electronic devices, large number of interconnections and large chip area, all these factor introduce the concept of Multivalued Logic(MVL). Increasing the states of machine from 2-state(binary) to 3-state(ternary) or more can be consider as the concept of multi-stat or multivalued logic. Ternary logic attract the interest of many researchers to overcome the gaps of binary machines. Due to many advantages of ternary machine over binary machine many researcher propose circuits for ternary logic, but still there is no system which is widely in use based on ternary logic. In past 20years various MVL circuits are proposed which can be categorized into two category, v-MVL and i-MVL which means circuits based on voltage and current logic. Almost every researchers consider inverter circuit as the building block of any ternary circuit, and broadly classify it into three type Simple Ternary Inverter (STI), Positive Ternary Inverter (PTI) and Negative Ternary Inverter (NTI) based on MOS transistors. Many designs on ternary adders have been proposed in last three decades based on the logic of STI, NTI and PTI the functionality of ternary inverter circuit can be consider as similar to binary inverter circuit that is binary NOT gate. But if we see the truth table of binary NOT gate as shown in Table 4, we come up with the conclusion that whenever NOT gate get 0 as input it will give 1 as output and vice-versa. The logic of state transition in case of NOT gate remain same for any case and maintain the symmetry in the functionality for any input that is for input 0 it give output 1 and for input 1 it give output 0.

Table 4.1: Truth Table of binary NOT Gate

a	$\bar{a}$
0	1
1	0

Now considering the truth table for previously developed ternary inverter circuit design as shown in Table 4, three different ternary inverter logic has been developed that is STI, NTI & PTI. The results of inverter operator are drawn either by using transistors with different threshold voltage or load resisters. Considering binary inverter whose functionality is to transfer the state of machine from low to high that is 0 to 1 or viceversa, as binary machine work on two logical state (0 & 1). Now for designing ternary inverter such as an extension of the binary inverter for three logical states instead of two. That is ternary inverter basically transfer the state of machine from one state to another according to the current state of the machine. Such as if the state of the machine is 0(LOW) than the ternary inverter transfer the state to 1, and with further inversion change the state to 2(HIGH). To design ternary inverter circuit on the basis of the above logic, doesn't require three separate circuits like STI, NTI & PTI as proposed by various researchers in their work. Single inverter circuit can be design to solve the purpose, as mention in this work. Further more in future the inverter circuit based on above logic can also be extended for the system based on more higher radix. That is why these gates are further extended to get NAND, NOR and many other logic gates with perfect logic and state transition. Keeping that in mind the ternary machine should be design such as an extension of binary machine, similar state transition with 3 states instead of 2. So that ternary machine can be further extended for more states in future.

Table 4.2: Truth Table of Ternary Inverter Gate

x	**STI**	**PTI**	**NTI**
0	2	2	2
1	1	2	0
2	0	0	0

In this work on the basis of binary machine, various operators based on ternary logic has been proposed which are discussed in the upcoming sections of this chapter. Which includes the development of truth table for ternary inverter gate that is t-NOT gate, which can be considered as the basic building block for other gates. As in case of ternary logic there are three transition states instead of two so, NAND and NOR are classify in two main categories that is *p* and *s*. The truth table for p-NAND, s-NAND, p-NOR, s-NOR, ternary decoder and ternary Adder is proposed. And based on the truth table implementable circuit is design for t-NOT gate, p-NAND, s-NAND, p-NOR, s-NOR, ternary decoder and ternary Adder. And finally a novel ternary addition algorithm is proposed, same algorithm can be used for future machines based on higher radix that is which work on multiple levels of voltage instead of two.

4.1 t-NOT gate (inverter circuit)

In case of binary, basically inverter circuit invert the logic from high that is 1 to low that is 0 and viceversa in circular manner as binary machines are having only two logical states that is 0(low) & 1(high). So, binary NOT gate takes the the machine from any one of the state to another state that is if the initial state is 0 than the output state will be 1 and if the initial state is 1 than the output state will be 0. More specifically, if the initial state of machine is 0 then with the single transition

(NOT operation) the output state of machine become 1, and if one more transiotion is applied on the current state of the machine than the machine get back to its initial state that is 0 as shown in the Fig. 4.1. Now, it can be concluded that in case of binary machine, two transition is required to take the machine to its initial state. In general in case of inverter logic the total number of transition equal to the number of states take the machine to its initial state. But as in case of ternary there are three logical states that is 0, 1, and 2 on the basic of three different level of voltages respectively. So, ternary inverter circuit basically invert the state of machine from one level of voltage to another level of voltage instead of high to low or low to high.

Here *t* in t-NOT stands for ternary which means three. The working of the proposed t-NOT gate is just an extension of binary NOT gate the only difference is in ternary there are three logical states instead of two. So, t-NOT gate basically invert the logic from one state to another as shown in the Fig. 4.2 representing the state transition diagram of ternary inverter (t-NOT gate). If the initial state of machine is 0 that is low then the ternary inverter take the machine to state 1. And if the state of machine is 1 then ternary inverter take it to logic 2 that is the high state of ternary machine. If the initial state of machine is high that is 2 then the ternary inverter circuit take the machine to logic 0 (low state). On the basis of same logic, further inverter circuits can be designed for future machine which will work on higher radix that is more logical states or different level of voltages as shown in Fig. 4.3 & 4.4, the state transition diagram for inverter circuits for the machines based on 4 and 5 logical state.

On the basis of the state transition diagram of t-NOT gate the truth table for the same is shown in Table 4.1, t-NOT gate can consider as the base for designing various gates for ternary machine.

The t-NOT gate basically take the machine from initial state to very next state in

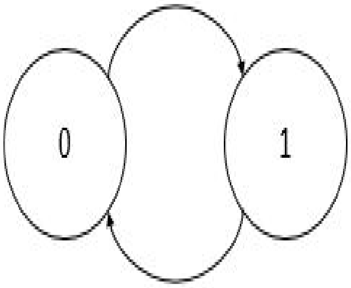


Figure 4.1: State Transition Diagram for Binary NOT gate

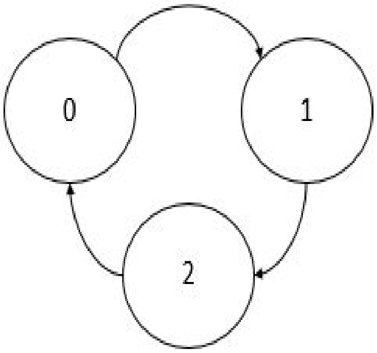


Figure 4.2: State Transition Diagram for Ternary NOT gate

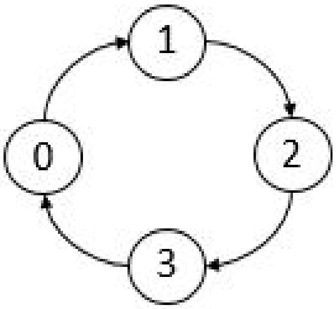


Figure 4.3: State Transition Diagram for NOT gate for 4-state machine

Table 4.3: Truth Table t-NOT Gate

a	**a'**
0	1
1	2
2	0

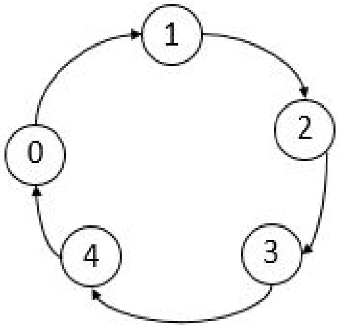


Figure 4.4: State Transition Diagram for NOT gate for 5-state machine

the direction of arrow as shown in Fig. 4.2 and initial state can be any state that is either 0, 1, or 2. For example if the initial state of machine is 0 than t-NOT gate takes the machine to state 1 that is intermediate state between high and low. Similarly if the initial state of machine is 1 then the output of t-NOT gate will be 2. Finally if the initial state is 2 than t-NOT gate will give output 0.

As in case of binary, if the initial state of machine is 0(low) than the binary inversion change the state of machine to high. But in case of ternary to change the state of machine from 0(low) to 2(high), two transitions are carried out represented by primary(p) & secondary(s) transition as shown in the truth table 4.1 of t-NOT gate with a' and a" respectively. Here we are considering it as complete transition, when a machine from initial state as low(0) transit to the final state as high(2). To achieve complete transition, pair of t-NOT gates are connected in such a way that the output of one t-NOT gate is the input of another t-NOT gate. Proposed symbol for t-NOT gate is shown in Fig. 4.5 where a is representing the initial state of the machine and a' as the immediate next state of initial state according to the state transition diagram of t-NOT gate shown in Fig. 4.2.

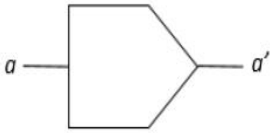


Figure 4.5: Proposed symbol for Ternary NOT gate

4.1.1 Circuit diagram of t-NOT gate

Proposed circuit for t-NOT gate is comprises of 741 op-amp, 7432 IC (OR) and 7408 IC (AND) as shown in Fig. 4.6, where 741 op-amp is the operational amplifier which can be use for various purpose, here 741 op-amp used as a comparator. The pin configuration of 741 op-amp is shown in Fig. 4.7 according to which the input voltage VIN is connected to pin 3 (non-inverting voltage), pin 2(inverting voltage) is connected with the reference voltage VR where reference voltage is the voltage with which VIN is compared to get the output. Pin 7 and 4 is connected with the +VCC and -VCC to specify the range(upper and lower limit) of voltage and from pin 6 we get the output voltage VOUT. The basic functionality of 741 op-amp is to compare the input voltage VIN connected at pin 3 with the reference voltage VR connected with pin 2 and if the input voltage is greater than reference voltage then +VCC(upper limit) passed as output on pin 6. Else, if the input voltage is lesser than the reference voltage then we get -VCC(lower limit) as the output at pin 6 as shown in Table 4.1.1.

Table 4.4: Working of 741 op-amp

VIN	**VOUT**
>VR	+VCC
<VR	-VCC

Apart from the standard functionality of 741 op-amp, various readings have been taken in the lab with the input voltage VIN is greater than the upper limit of the voltage that is +VCC and it is observed that in this case we get the output voltage VOUT as +VCC as shown in Table 4.5 and some of the reading are shown in Table 4.6 when +VCC is constant 5V. The same concept is used in the designing and working

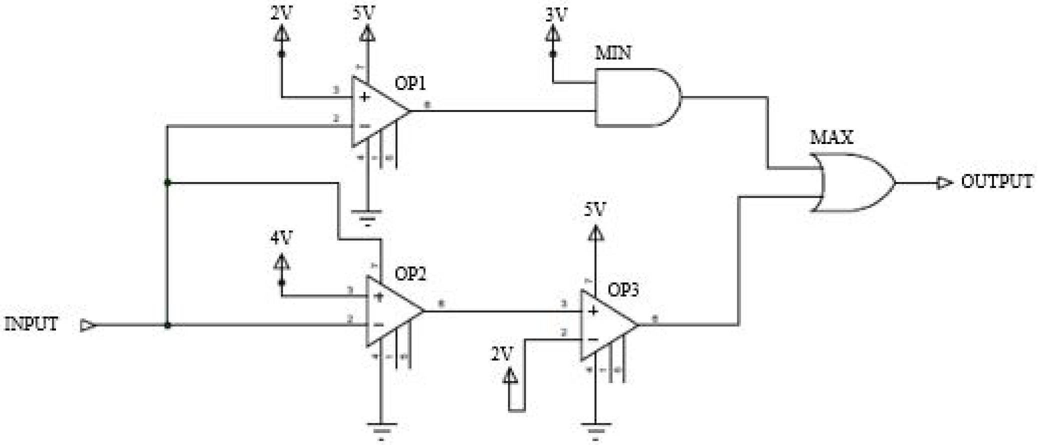


Figure 4.6: Circuit for t-NOT gate.

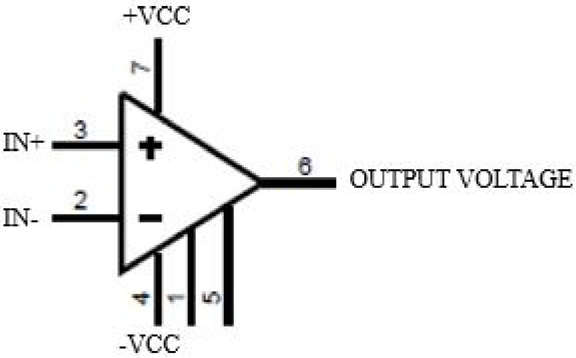


Figure 4.7: Circuit for 741 op-amp.

of the ternary logic based circuits.

Table 4.5: Working of 741 op-amp when VIN is greater than +VCC

VIN	**VOUT**
>+VCC	+VCC

Table 4.6: Reading of 741 op-amp when VIN is greater than +VCC

VIN	**VOUT**
+6.37	+4.26
+7.54	+4.33
+8.51	+4.47
+9.27	+4.54

And 7432 IC (OR) & 7408 IC (AND) are used as MAX and MIN gate for ternary logic. As 7432 IC takes two or more input voltages and return the maximum voltage among the input voltages as output. Similarly, 7408 IC takes two or more input voltages and return the minimum voltage among the input voltages as output as shown in the Table 4.7.

Table 4.7: Reading of 7432 IC & 7408 IC returning maximum and minimum voltage among the supplied inputs.

7432 IC			7408 IC		
VIN1	VIN2	VOUT	VIN1	VIN2	VOUT
5V	3V	+4.30V	5V	3V	+2.42V

According to the Fig. 4.6 the circuit of t-NOT gate comprises of three 741 op-amp, one 7432 IC and 7408 IC that is MAX and MIN gate respectively. Now here the connections of various components are explained in detail. As per the circuit diagram of t-NOT gate, first 741 op-amp (OP1) is having a constant voltage of 5V at pin number 7 that is +VCC, 2V at pin number 3, pin number 4 that is -VCC is connected to ground and pin number 2 (VR) is connected to input voltage (VIN). Similarly, in second 741 op-amp (OP2) pin number 7(+VCC) is connected to the input voltage (VIN), constant voltage of 4V is applied to pin number 3, pin number 2 is connected with the input voltage (VIN) and pin 4 is connected with ground. In third 741 op-amp (OP3) pin number 4 is connected with ground, pin number 2 is connected with constant 2V, pin number 7 is connected with the constant 5V and pin number 3 is connected with the output pin that is 6 of OP2. One input pin of 7408 IC is having a constant voltage of 3V and another input pin is connected with the output pin number 6 of OP1. One input pin of 7432 IC is connected with output pin of 7408 IC and another input pin is connected with output pin number 6 of OP3. And final resultant voltage VOUT is received from output pin of 7432 IC.

4.2 p-NAND & s-NAND gate

Considering binary NAND gate which can be defined as binary NOT gate followed by binary AND gate that is the output of AND gate is connected with the input of the NOT gate. Based on the same logic ternary NAND is proposed, but as in case of binary logic only one NOT gate is required to take the state from 0 to 1 that is from low to high voltage as shown in the state transition diagram of binary NOT gate in Fig. 4.1. But this is not the case in ternary inverter logic, in ternary two transitions are required to take the state from 0 to 2 that is from low to high voltage. So these two transitions of t-NOT gate can be categorized as primary(p) transition which take the state from 0 to 1 and the secondary (s) transition which take the state from 1 to 2 that is complete inversion from low voltage to high voltage. On the basis of this concept ternary NAND gate is also classify into two type of gates that are p-NAND and s-NAND the truth table of p-NAND & s-NAND gate in shown in Table. 4.8.

Table 4.8: Truth Table for Ternary p-NAND and s-NAND gate

a	b	MIN	p-NAND	s-NAND
0	0	0	1	2
0	1	0	1	2
0	2	0	1	2
1	0	0	1	2
1	1	1	2	0
1	2	1	2	0
2	0	0	1	2
2	1	1	2	0
2	2	2	0	1

According to the corresponding truth table the symbolic diagram for p-NAND and s-NAND is shown in Fig. 4.8 & 4.9. In case of p-NAND gate, one t-NOT gate is followed by MIN gate (binary AND) that is the output of the MIN gate is connected with the input of t-NOT gate. Where as in case of s-NAND gate, pair of t-NAND gate is followed by MIN gate that is output of the MIN gate is connected with the input of first t-NOT gate and the output of first t-NOT gate is connected with the input of second t-NOT gate.

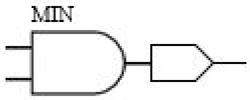


Figure 4.8: Symbolic diagram for p-NAND gate

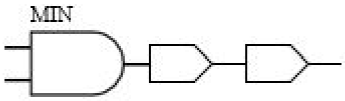


Figure 4.9: Symbolic diagram for s-NAND gate

The elaborated circuit of p-NAND and s-NAND gate is shown in Fig. 4.10 and 4.11 in which MIN gate is accepting two inputs (input 1 and input 2) and the output of MIN gate is connected with the pin 2 (VR) of op-amp (OP2) of the t-NOT gate. The components and connections of t-NOT gate is already explained in section 4.1.1.

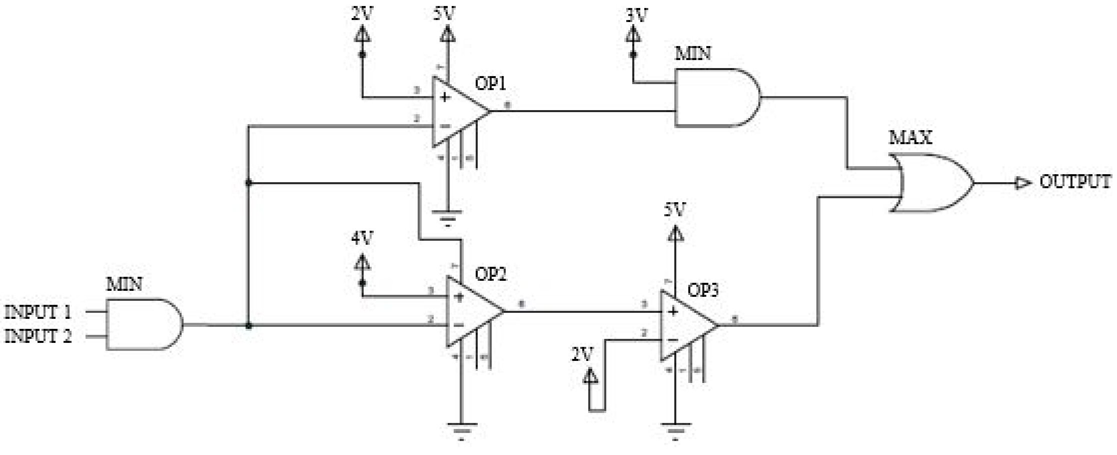


Figure 4.10: Circuit diagram for p-NAND gate

As shown in the truth table for the above operator that basic binary AND gate is acting as a MIN gate in ternary logic and the same concept can be extended for

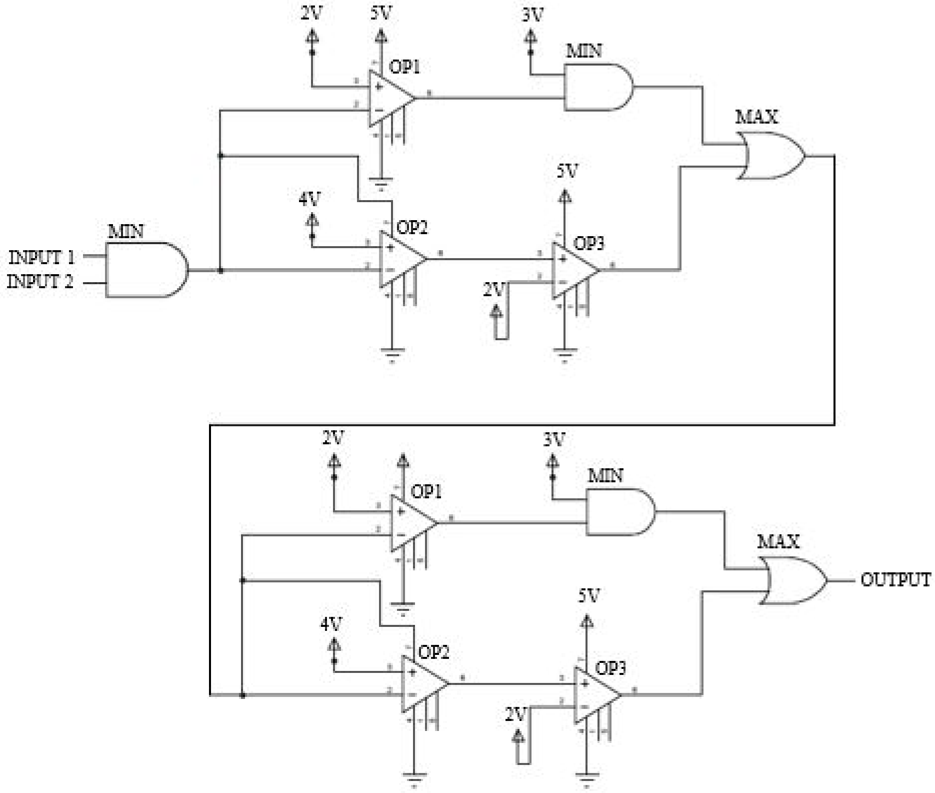


Figure 4.11: Circuit diagram for s-NAND gate

higher radix. As the binary AND gate return the minimum voltage as output among the supplied voltages as inputs.

4.3 p-NOR & s-NOR gate

As discussed in the previous subsection 4.2 about p-NAND and s-NAND in the same way ternary NOR gate is the extension of binary NOR gate. As binary NOR gate is a binary NOT gate followed by binary OR gate that is the output of OR gate is connected with the input of the NOT gate. Based on the same concept ternary NOR gate is proposed which is classify in two types that is p-NOR and s-NOR. In p-NOR, t-NOT gate is followed by MAX gate (binary OR gate) that is output of MAX gate is connected with the input of t-NOT gate as shown in the symbolic diagram in Fig. 4.12. In s-NOR, pair of t-NOT gate is followed by the MAX gate that is the output of the MAX gate is connected with the input of first t-NOT gate and the output of first t-NOT gate is further connected with the input of second t-NOT gate as shown in the symbolic diagram in Fig. 4.13. The truth table for p-NOR and s-NOR gate is shown in Table 4.9, basic OR gate is acting as a MAX gate in ternary logic and the same concept can be extended for higher radix. As the binary OR gate return the maximum voltage as output among the supplied voltages as input.

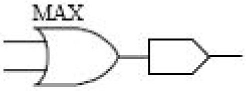


Figure 4.12: Symbolic diagram for p-NOR gate

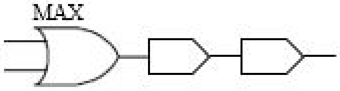


Figure 4.13: Symbolic diagram for s-NOR gate

Table 4.9: Truth Table for Ternary p-NOR and s-NOR gate

a	b	MAX	p-NOR	s-NOR
0	0	0	1	2
0	1	1	2	0
0	2	2	0	1
1	0	1	2	0
1	1	1	2	0
1	2	2	0	1
2	0	2	0	1
2	1	2	0	1
2	2	2	0	1

The elaborated circuit of p-NOR and s-NOR gate is shown in Fig. 4.14 & 4.15 in which MAX gate is having two inputs that is input 1 & input 2, output of the MAX gate is connected with the reference voltage(VR) at pin 2 of op-amp(OP2) of t-NOT gate. The connections and the components of t-NOT gate is already discussed in section 4.1.1.

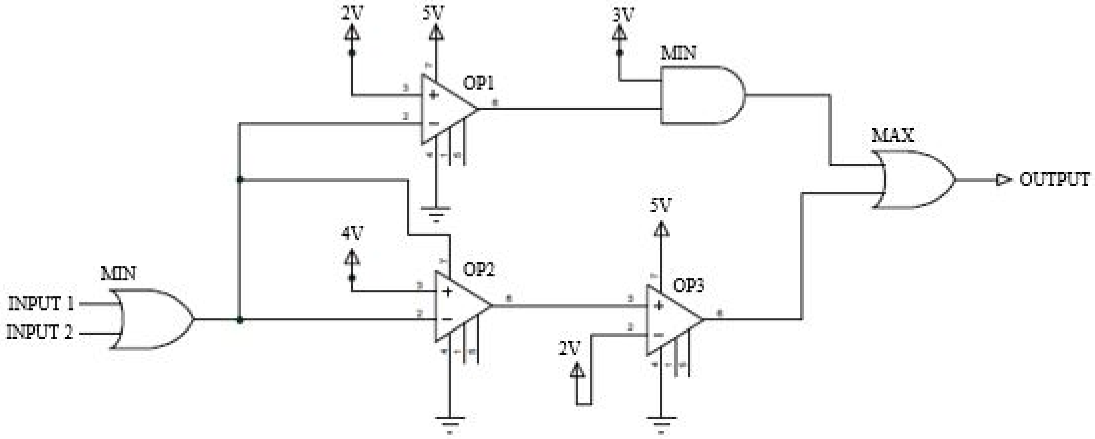


Figure 4.14: Circuit diagram for p-NOR gate

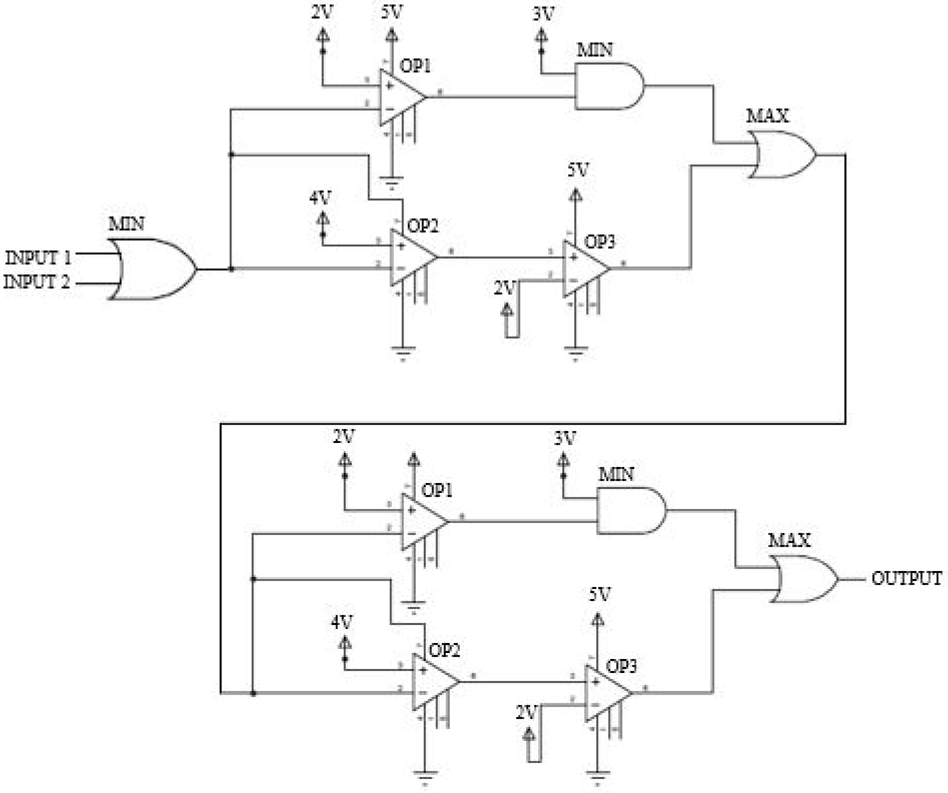


Figure 4.15: Circuit diagram for s-NOR gate

4.4 Ternary Decoder

In case of binary decoder, it is having n input lines and 2^n output lines that is according to the inputs one output line get enabled at a time. The truth table for

2:4 binary decoder is shown in Table 4.10 where according to the input a and b one output line from D_0 *to* D_3 get enabled at a time and rest of the output lines remain disable.

Table 4.10: Truth table for 2:4 binary decoder

a	**b**	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

The concept of binary decoder is extended for designing ternary decoder. Here the ternary decoder is designed with one input line and three output line. And on the basis of voltage supplied on the input line, one among the three of output line get enabled and rest two remain disable. The working of ternary decoder will be clear with the help of truth table as shown in Table 5.5 in which depending on the input, one output line among three get enabled. If the input is 0 then X_0 line get enabled similarly for input 1 and 2 output line X_1 & X_2 get enabled respectively and rest of the lines remain disable.

Table 4.11: Truth table for ternary decoder

X	X_0	X_1	X_2
0	2	0	0
1	0	2	0
2	0	0	2

According to the above truth table, circuit has been designed for ternary decoder as shown in Fig. 4.16. The circuit comprises of 741 op-amp, t-NOT gate and 7432 IC where the circuit of t-NOT gate and the functionality of 741 op-amp is discussed in section 4.1.1. In the circuit, symbol X is representing the applied input VIN which is connected as input to t-NOT 1, 7432 IC that is MAX gate as of its one input and pin 3 of op-amp 2. The output of t-NOT 1 is applied as input to t-NOT 2 and the output of t-NOT 2 is connected with pin 3 of OP-AMP 1 and as another input of MAX gate. Pin number 2(VR) of OP-AMP 1 is connected with the constant voltage of 4V, pin 7(+VCC) is connected with the constant voltage of 5V, pin 4 (-VCC) connected to ground and pin 6 that is output pin is providing the first output line of ternary decoder that is X_0. The output of the MAX gate is connected with the input of t-NOT 3 and the output of t-NOT 3 is providing the second output line X_1. Similarly pin 6 of OP-AMP 2 is providing the third output line X_2 of ternary decoder, where as pin number 7 is connected with constant 5V, pin number 4 is connected with ground and pin number 2 is connected with constant 4V.

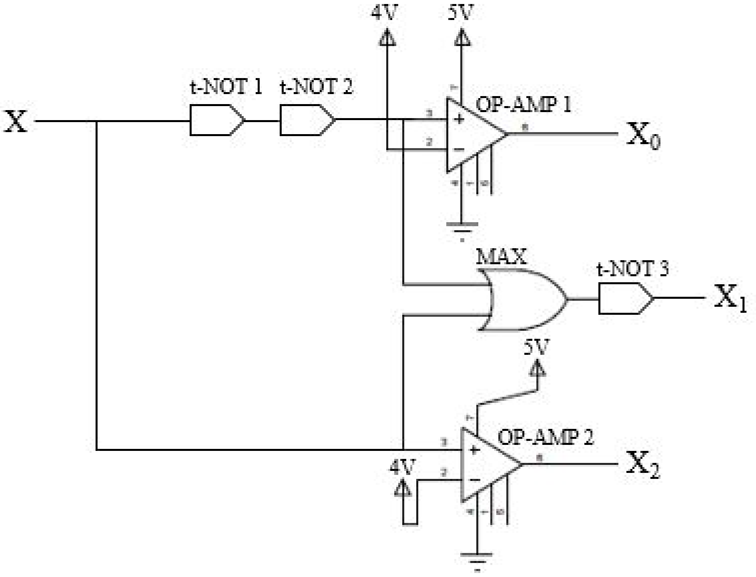


Figure 4.16: Circuit diagram for Ternary Decoder

4.5 Ternary Addition Algorithm

Before discussing about the algorithm lets see the Table. 4.13 representing the three bit ternary representation of first nineteen decimal numbers that is from 0 to 18 and the Table. 4.12 representing the truth table for addition of ternary bit and the generated carry. As we can see in the Table. 4.12 carry would be generated only if any one of the bit among two bits is 2 and another bit is non-zero bit, based on this an addition algorithm is proposed which can also be extended with minor modification for the addition of numbers with more higher radix for future machines which will work on more level of voltages.

An addition algorithm 4 is proposed for addition of ternary numbers using stack of size one for holding generated carry. The input of the algorithm is two ternary numbers as shown in equation (4.1) where a_i & b_i is representing the ternary numbers in which each bit belong to set {0, 1, 2}. And $i \in \{0, 1, 2...n-1\}$ so for $i = 0$, a_0 and b_0 refer the first bit of the ternary numbers same as for $i = 1$, a_1 and b_1 representing the second bit of ternary numbers and so on.

$$a_i \ \& \ b_i \ \in \ \{0,\ 1,\ 2\} \ where \ 0 \ <= \ i \ <= \ n-1 \tag{4.1}$$

Table 4.12: Truth Table for performing Ternary Addition

A	**B**	**S**	**K**
0	0	0	0
0	1	1	0
0	2	2	0
1	0	1	0
1	1	2	0
1	2	0	1
2	0	2	0
2	1	0	1
2	2	1	1

According to algorithm 4 initially stack is NULL or 0 as shown in line 1 and the size of stack is one that is it can hold one bit at a time. Now considering each bit of two ternary numbers to be added from $i = 0$ to $i = n-1$, initially for $i = 0$ bit a_0 and b_0 is considered. As shown in line 3 condition is checked for a_0 and b_0 that is if any of the bit among two is equal to 2 and another bit is non-zero in that case carry K would be generated. And if stack is NULL then addition operation is performed and the sum of a_0 and b_0 is stored in c_0 as shown in line 4 & 5 and generated carry K is pushed into the stack (line 6). Now if stack is not NULL in that case previous carry is

Table 4.13: 3-bit Ternary equivalent of first eighteen natural numbers of base 10

Decimal Number	**a**	**b**	**c**
0	0	0	0
1	0	0	1
2	0	0	2
3	0	1	0
4	0	1	1
5	0	1	2
6	0	2	0
7	0	2	1
8	0	2	2
9	1	0	0
10	1	0	1
11	1	0	2
12	1	1	0
13	1	1	1
14	1	1	2
15	1	2	0
16	1	2	1
17	1	2	2
18	2	0	0

popped from the stack and added with a_0 and b_0, generated carry pushed into stack as shown in line 8, 9 & 10. And if condition mention in line 3 does not satisfied then in that case carry bit would never be generated and the sum of bits a_0 and b_0 get stored in c_0. After performing first iteration for $i = 0$ the value of i get incremented by one, $i = 1$ and the same procedure is performed for second bit of ternary numbers that is a_1 and b_1. If any bit among a_1 and b_1 is 2 and another bit is non-zero than in that case carry bit would be generated and get stored into the stack if initially stack is NULL. Otherwise the bit is popped from the stack and added with the two bits a_1 and b_1, generated carry pushed into stack. If the initial condition as mention in line 3 is false then simple addition of two bits a_1 and b_1 is performed and stored in c_1, no carry would be generated in this case. Similarly, same procedure is carried for rest of the bits in two ternary inputs and after all iterations from $i = 0 \ \ to \ \ n - 1$ the status of stack is checked as shown in line 16 that is if the stack is not NULL then stack is popped and popped bit is positioned at c_n. And if the status of stack is NULL then no pop operation is performed and in either of the case the final string of bits generated by the addition of two ternary input strings represented by C is returned by the algorithm as shown in line 19.

Algorithm 4 Ternary Addition

Require: Ternary numbers a_n and b_n to be added
Ensure: Addition of two input ternary numbers.

```
 1: Stk = NULL //Stack is initially empty
 2:     for i = 0 to $n - 1$
 3:         if ( $a_i$ == 2 && $b_i$ != 0) || ($a_i$ != 0 && $b_i$ == 2 )
 4:           if Stk == NULL
 5:                   $c_i = a_i + b_i$
 6:                   push(k) //push generated carry into stack
 7:           else
 8:                   pop(k) //pop previously stored carry from the stack
 9:                   $c_i = a_i + b_i + k$
10:                   push(k)
11:           end if
12:         else
13:                   $c_i = a_i + b_i$
14:         end if
15:     end if
16:         if Stk != NULL
17:                 pop(k)
18:                 $c_n = k$
19: return $C$ //Resultant ternary sum of input numbers.
```

The time complexity of the proposed algorithm of Ternary Addition depends on the length of the string of two inputs to be added. On that basis we can calculate the best, average and the worst case of addition operation. The best case of the algorithm can be considered, if the length of the inputs to be added is 1 in that case the time complexity of the algorithm will be $O(1)$. And the worst case will be when the length of the inputs to be added is n, so the time complexity for the worst case of the algorithm will be $\Omega(n)$. And the average case time complexity of the algorithm can be considered as $\Theta(logn)$ as shown in the Table 4.14 the best, average and worst case for the time complexity of the proposed algorithm 4 for performing ternary addition.

Table 4.14: Time complexity of the proposed algorithm

Best case	$O(1)$
Average case	$\Theta(logn)$
Worst case	$\Omega(n)$

4.6 Ternary Adder

Proposed circuit of ternary adder is an extension of binary half adder, the truth table of binary half adder as shown in Table. 4.15. As binary number system work on only two bits that is 0 and 1 so only four different combinations are required to make the truth table for binary half adder. So, simple circuits comprising of binary XOR and AND gate is required where XOR gate is used for computing sum(S) of two input bits and AND gate is used for computing carry(C) as shown in Fig. 4.17.

Table 4.15: Binary Half Adder

a	**b**	**S**	**C**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

As ternary adder work on three logic bits that is 0, 1 and 2 so there will be 9 different combinations in the truth table for ternary adder as shown in the Table. 4.16 where A and B are the input bits, S and K representing the sum of two input bits and generated carry(if any) respectively. On the basis of the truth table separate circuit is designed for computing sum S and carry K as shown in Fig. 4.18 & 4.19 respectively.

The circuit diagram of ternary adder for computing sum S is comprises of two MAX gate (7432IC), two MIN gate (7408IC), four t-NOT gate and ternary decoder.

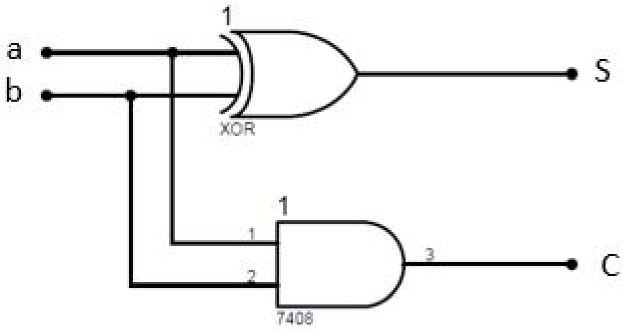


Figure 4.17: Circuit diagram for Binary Half Adder

Having two inputs A and B, where A is supplied as input to ternary decoder. As per the Table. 4.16, if the input A is 0 then the output line X_0 of the ternary decoder remain enable to carry the input A and the remaining output line X_1 and X_2 remain disable. The output line X_0 is attached as one of the input to MAX 1 gate and the output of the MAX 1 is connected as input of MAX 2 gate, and another input of MAX 1 is input B. And in case when the input A is 1 than the output lines X_0 and X_2 of ternary decoder remain disable where as the output line X_1 carry intput A. The output line X_1 is connected as input to the t-NOT 1 whose output is attached to one of the input line of gate MIN 1. Another input of MIN 1 is connected with output of t-NOT 2 whose input is B, the output of MIN 1 is attached to one of the input to MAX 2. Now when the input A is 2 then the output line X_2 of the ternary adder remain enable to carry the input A and the remaining output line X_0 and X_1 remain disable. The output line X_2 is connected with one of the input of gate MIN 2 and its another input is connected with the output of t-NOT 4, the input of t-NOT 4 is connected with the output of t-NOT 3 and input of t-NOT 3 is B. The output of MIN 2 is supplied as the input of MAX 2 and the output of MAX 2 provide the sum S of input bit A and B.

The circuit diagram for the computation of carry K is shown in Fig. 4.19 which

Table 4.16: Truth Table for Ternary Adder

A	B	S	C
0	0	0	0
0	1	1	0
0	2	2	0
1	0	1	0
1	1	2	0
1	2	0	1
2	0	2	0
2	1	0	1
2	2	1	1

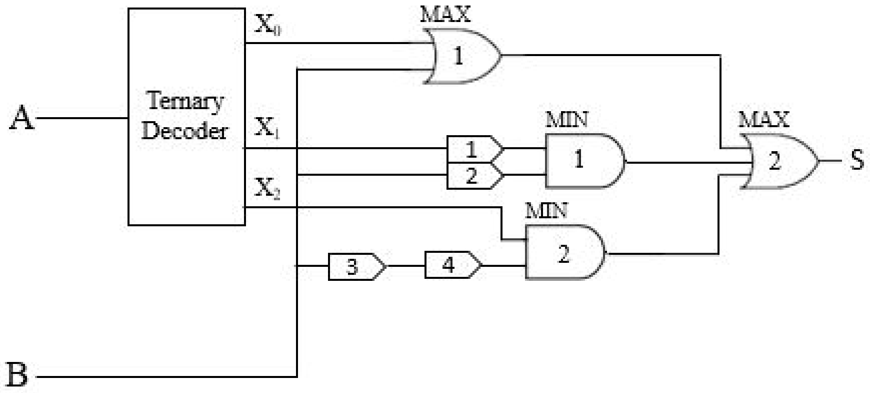


Figure 4.18: Circuit diagram for Ternary Adder for computation of sum S

consist of four t-NOT gate, two MAX gate (7432IC), five MIN gate (7408IC) and a ternary decoder. The circuit takes two inputs A & B, A is connected with the input of ternary decoder and on the basis of truth table 4.16 if input A is 0 then in that case no carry would be generated so output line X_0 of ternary decoder remain disable always. And for input A = 1 in that case output line X_1 of ternary decoder remain enable and rest of the two output lines that is X_0 and X_1 remain disable. The output line X_1 is attached with one input of MIN 1 gate, where as the output of MIN 1 is connected to one input of MAX 1 and another input of MIN 1 is connected with the output of MIN 3 gate whose one input is connected with the output of t-NOT 2 and another input is B. The input of t-NOT 2 is connected with the output of t-NOT 1 which is having input B. When the input A is 2 then the output line X_2 of ternary decoder remain enable and line X_0 and X_1 remain disable, the output line X_2 is connected with one input of MIN 2 and another input of MIN 2 is connected with the output of MAX 4 gate. Where one input of MIN 4 is B and another input is the output of MAX 2, one input of MAX 2 is connected with the output of MIN 5 and another input is connected with the output of t-NOT 4. The input of t-NOT 4 is the output of t-NOT 3 and the input of t-NOT 3 is B where as the output of t-NOT 3 is also connected with one input of gate MAX 5 whose another input is connected with B. Finally resultant carry K is generated as the output of MAX 1 gate.

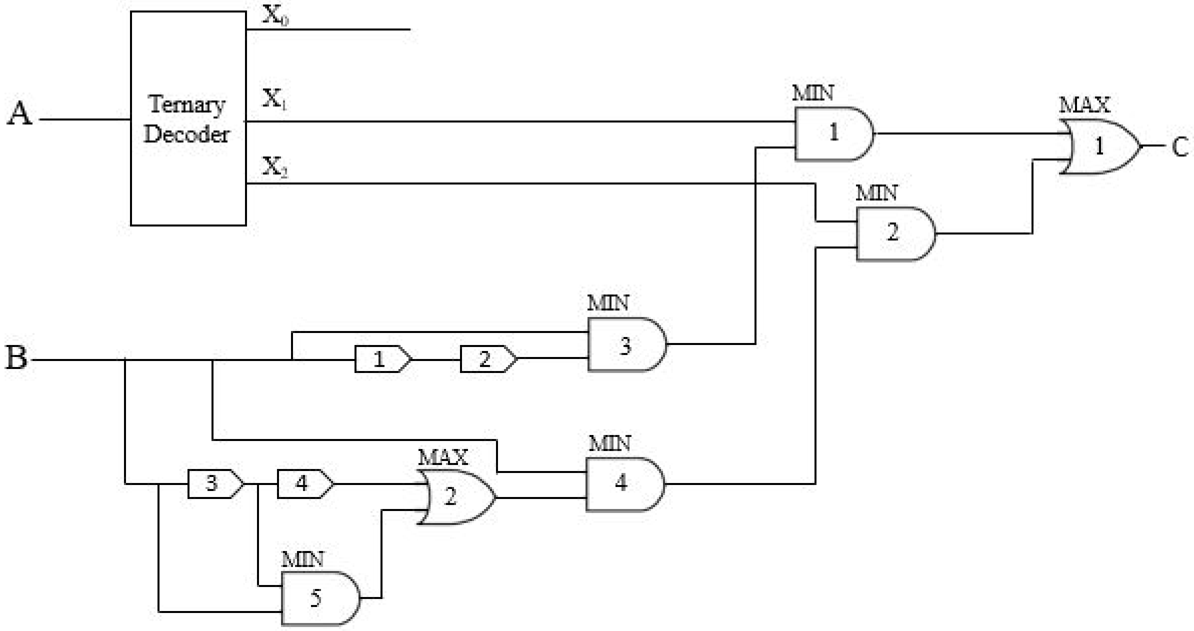


Figure 4.19: Circuit diagram for Ternary Adder for computation of carry C

CHAPTER 5
RESULTS AND DISCUSSION

5.1 am-MULTIPLICATION Algorithm

In this section, we are considering two unsigned whole numbers for multiplication according to the proposed am-MULTIPLICATION algorithm 1 discussed in Section 3.1. Let the numbers are $m = 3$ and $n = 9$. Now, according to the algorithm MIN 2 and MAX 3 return minimum and maximum among m, n that is $a = 3$ and $b = 9$. Next, following the am-MULTIPLICATION 1 to calculate the multiplication of m and n in which we calculate the value of p to find the number of set(s) to which $a = 3$ belong as shown below.

$a = 3, b = 9, i = 0$

$p = 3 \ / \ 2^0, \ p = 3$

$p \in \{odd \ \ numbers\}$, update $S = \{S_0\}$

$i = i \ + \ 1, \ i = 1$

$p = 3 \ / \ 2^1, \ p \in f$

$y = 3 - (3 \ \% \ 2^1), \ y = 2$

$p = 2 \ / \ 2^1, \ p = 1$

$p \in \{odd \ \ numbers\}$, update $S = \{S_0, S_1\}$

Now $2^2 > 3$

Final set $S = \{S_0, S_1\}$

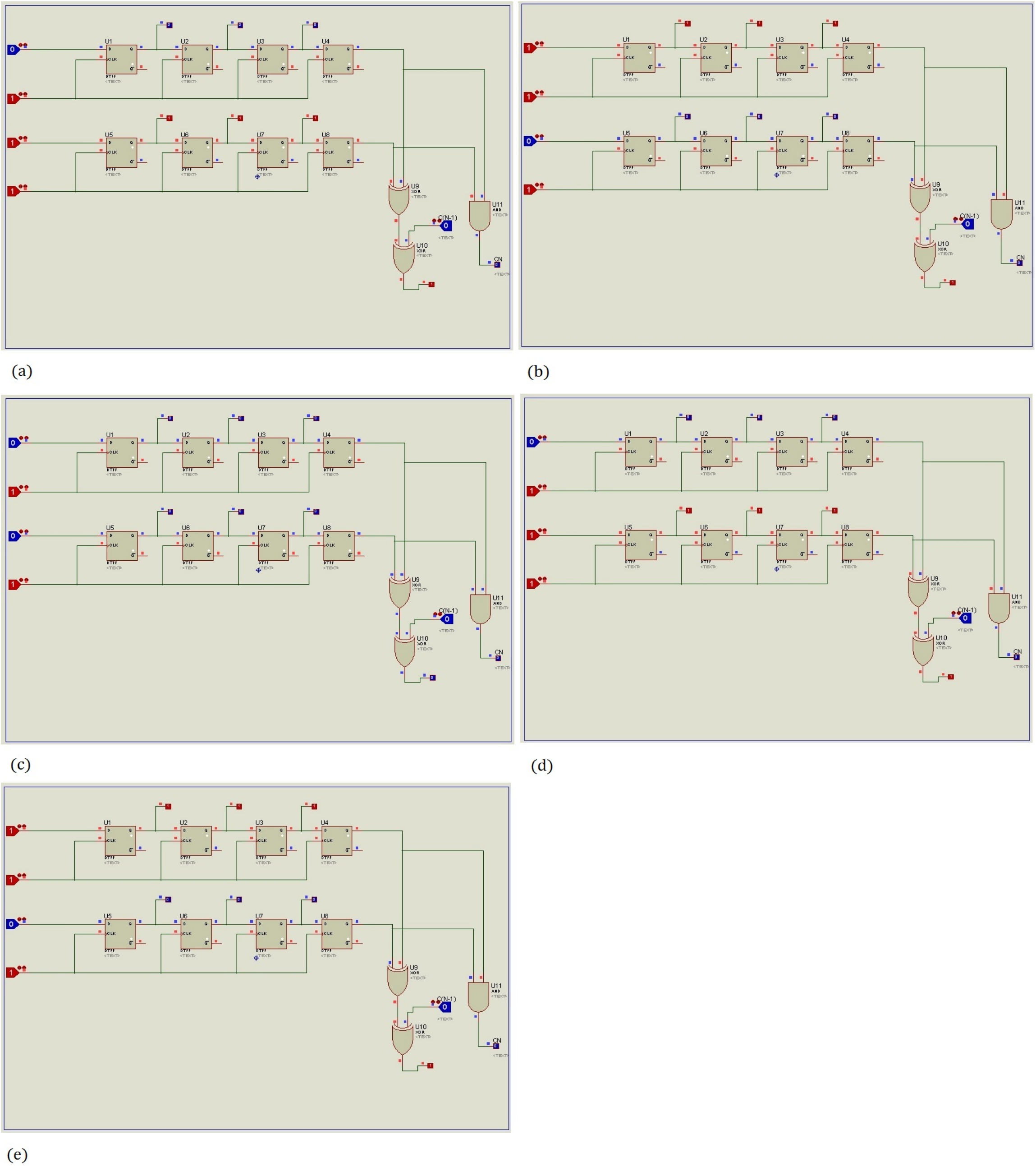
Append the number of 0s according to S in binary equivalent of $b = 9$ that is 1001 to binary number would be generated that is, $x = 1001$ and $y = 10010$

And now after performing the binary addition of x and y the resultant will be $r = 11011$ which is equivalent to $(27)_{10}$ that is the multiplication result of a & b.

The above multiplication operation is simulate over the binary multiplexer circuit designed on the basis of proposed multiplication algorithm using simulation software proteus-8 for the multiplication of $a = 3$ and $b = 9$ after finding out the set $S = \{S_0, S_1\}$ to which $a = 3$ belong to. Next, the binary numbers based on the set $S = \{S_0, S_1\}$, i.e., $x = 1001$ and $y = 10010$ are simulate on the circuit for getting the final result $r = 11011$. As shown in the Fig. 5.1 where (a) The output of the first and second shift register that is the last bit of y and x which are 0,1 act as input to carry look-ahead full adder so the output will be sum = 1 and carry for the current state $CN = 0$ where the $C(N-1)$ initially remain 0. (b) Now the next two bits that is 1 and 0 from the sift register is provided to the carry look-ahead adder with $C(N-1) = 0$, so we get sum = 1 and $CN = 0$. Here $C(N-1)$ is 0 which is equivalent to the CN of the previous state. (c) Similarly, next bits that is 0 and 0 will be the output of the sift register which will act as input to the carry look-ahead adder with $C(N-1) = 0$ to get the sum = 0 and $CN = 0$. (d) And now for next two bits, 0, 1 and $C(N-1) = 0$ we get sum = 1 and $CN = 0$. (e) For last pair of bits 1,0 and $C(N-1) = 0$ we get sum = 1 and $CN = 0$. In this way after keeping the track of all the sum values, the result will be $r = 11011$ which is equivalent to the $(27)_{10}$.

In Fig.5.1 red and blue LED representing 1 and 0 respectively.

The comparison of the proposed circuit has been done with existing techniques. For example, in [20], the authors have shown the circuit for 8x8 combinatorial multiplier and Wallace-tree implementation of 8x8 multiplier, to ease the calculation of the speed of multiplication in terms of gate delays and gate count. The total number of (AND/OR) gates count was 624 including 56 binary adders for 8x8 combinatorial multiplier and a gate delay of 57 was recorded. For Wallace-tree implementation of 8x8 multiplier, the total gate count was 564 gates which include 22 binary adders and 23 gate delays. The proposed multiplexer circuit based on AM-Multiplication algo-



(a)

(b)

(c)

(d)

(e)

Figure 5.1: Results on Proteus 8 (a) The output of carry look-ahead adder Sum = 1 for y = 0, x = 1 generated from shift registers and carry C(N-1) = 0 (b) Now, Sum = 1 for y = 1, x = 0 and C(N-1) = 0 (c) Sum = 0 for y = 0, x = 0 and C(N-1) = 0 (d) Sum = 1 for y = 0, x = 1 and C(N-1) = 0 (e) Sum = 1 for y = 1, x = 0 and C(N-1) = 0.

rithm for 8x8 multiplier requires 8 carry look-ahead adders and 8 shift register each carry look-ahead adder comprises of 30 (AND/OR) gates where each shift register is the combination of 4 D flip-flop. The four flip flops are used to maintain the stability of the bit propagation. However, only one D flip-flop can be used to propagate a single bit with each clock pulse. And each D flip-flop requires 4 gates, so the total number of gates will be 272 gates and 9 gate delays where 1 gate delay is required for the generation of partial product. The gate delay and gate count for the proposed circuitry are much lesser as compare to the combinatorial and Wallace-tree based multiplier as shown in the Table 5.1.

Table 5.1: Gate count and gate delay for three types of 8x8 multiplexers.

Multiplexer	Gate count	Gate delay
Combinatorial	624	57
Wallace-tree	564	23
am-Multiplication(Proposed)	**272**	**9**

In [21], the authors provided the result of delays in modified booth Wallace multiplier and proposed Vedic multiplexer based urdhva tiryakbhyam sutra simulated on VHDL. The result comparison for the delay is calculated with the proposed circuit based on AM-Multiplication algorithm according to VHDL synthesis report as shown in Table 5.2. Similarly, the authors in [16] have proposed a 16x16 Vedic multiplexer based on urdhva tiryakbhyam and Nikhilam Sutra. The results were reported using VHDL synthesis. We have calculated the delay of our proposed circuit using VHDL synthesis which is much lesser as compared to the results of Vedic multiplexer based on urdhva tiryakbhyam and Nikhilam Sutra as presented in Table 5.3.

Table 5.2: Calculated delays in ns based on VHDL Synthesis report for 8x8 multiplexers.

Multiplexer	Gate delay
Modified booth Wallace	15.815
Vedic	15.685
am-Multiplication (Proposed)	**8.653**

Table 5.3: Calculated delays in ns based on VHDL Synthesis report for 16x16 multiplexers.

Multiplexer	Gate delay
urdhva tiryakbhyam sutra	41.751
Nikhilam Sutra	33.729
am-Multiplication (Proposed)	**16.797**

5.2 Ternary inverter circuit

The connections of various components used in t-NOT gate have already been discussed in section 4.1.1, nowhere in this section, we will discuss the working of the proposed inverting circuit that is a t-NOT gate as shown in Fig 4.6. The t-NOT gate circuit is working on the basis of the table as shown in Table. 5.4 which is showing the input and output voltages of various components of the t-NOT gate circuit to understand the functionality of the circuit.

When the supplied voltage that is VIN is 0V(which is not actually 0V in reference to the Table. 1.2) then pin P2 of op-amp 1(OP1) and P2 & P7 of op-amp 2(OP2) receive 0V as input. As in OP1 pin P3 is having a constant voltage of 2V, P7 is connected with 5V and P4 is ground so according to the functionality of op-amp 741 mention in subsection 4.1.1 pin P6 get the output voltage of 5V. Similarly, in OP2 pin P3 is connected with 4V, P4 is ground so the output at pin P6 will be 0V. Now as the output of OP1 at P6 is connected with pin P2 of MIN gate and P1 of MIN gate is connected with 3V so the output of MIN gate will be 3V which is passed as input to pin P1 of MAX gate. The output pin P6 of OP2 is connected with pin P3 of op-amp 3(OP3), P2 & P7 of OP3 is connected with 2V and 5V respectively and P4 is ground so P6 of OP3 will get the output 0V which is connected with input pin P2 of MAX gate. So, the final output voltage VOUT will be 3V as the MAX gate will return the maximum voltage among the two input voltages 3V and 0V at pin P1 and P2.

Now when the input voltage VIN is 3V then the directly connected pin P2 of OP1 and P2 & P7 of OP2 also receive 3V as input. The pin P3 of OP1 is connected with 2V, P7 is with 5V and P4 is ground so the output pin P6 of OP2 will get 0V. In a similar way, pin P3 of OP2 is connected with 4V, P4 is ground so at output pin P6 we get 3V. The output of OP1 is connected with one of the input pin P2 of MIN gate and another input pin P1 of MIN gate is connected with 3V as the MIN gate return the minimum voltage among the supplied input voltages so MIN gate returns 0V. The output of MIN gate in turn connected with one of the input pin P1 of MAX gate. And the output pin P6 of OP2 is connected with pin P3 of OP3, P2 & P7 of OP3 is connected with 2V and 5V respectively and P4 is ground so P6 of OP3 will get the output 5V, this output pin P6 of OP3 is connected with the second input pin P2 of MAX gate. Now as pin P1 and P2 of MAX gate is having voltages 0V and 5V respectively so final output voltage VOUT will be 5V.

Now we will see the distribution of voltages in the circuit when the supplied input voltage VIN = 5V, pin P2 of OP1 and P2 & P7 of OP2 receive 5V as input. As in OP1 pin P3 is connected with 2V, P7 with 5V and P4 is ground so the output we get on pin P6 of OP1 will be 0V. This output of OP1 is pass to one of the input pin of MIN gate pin P2 in this case and another input pin P1 is connected with 3V, so the output return by MIN gate will be 0V. This output of MIN gate is connected with one of the input line (P1) of MAX gate. In OP2 pin P3 is connected with 4V and pin P4 is ground so at output pin P6 voltage will be 0V. The output pin P6 of OP2 is connected with pin P3 of OP3 and P2 & P7 of OP3 is connected with 2V and 5V respectively whereas P4 is ground so the output of OP3 will be 0V at pin P6 which is connected to input pin P2 of MAX gate. So the final output VOUT return by MAX gate will be 0V as both the input pin P1 and P2 of MAX gate are 0V.

Final results of the ternary inverter circuit in term of ternary logic bits is shown in Table. 5.2, this ternary inverter circuits is an extension of the binary inverter circuit and the same logic can be further extended for machine based on higher radix. It can

Table 5.4: Voltage at each component in circuit of t-NOT Gate

	OP-AMP 1					OP-AMP 2					OP-AMP 3					MIN		MAX		
VIN	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P1	P2	P1	P2	VOUT
0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0

be considered as the replacement of three different inverter circuits that are simple ternary inverter (STI), positive ternary inverter (PTI), and negative ternary inverter (NTI). The t-NOT gate can be used as a building block of various different circuits based on ternary logic and for higher radix as well.

Table 5.5: Truth Table t-NOT Gate

a	**a’**
0	1
1	2
2	0

5.3 p-NAND & s-NAND gate

Basically, p-NAND & s-NAND gate is based on the already developed concept of the binary NAND gate, in the p-NAND gate, a t-NOT gate is followed by MIN gate (binary AND gate) that is the output of the MIN gate is connected with the input of t-NOT gate as shown in Fig. 4.8. And in the case of the s-NAND gate, two t-NOT gates are followed by MIN gate as shown in Fig. 5.2 that is the output of the MIN gate is connected with the input of t-NOT gate 1 and the output of t-NOT gate 1 in turn connected with the input of t-NOT gate 2 and output of t-NOT gate 2 gives the final output of s-NAND gate. The truth table for the p-NAND & s-NAND gate is shown in Table. 4.8 in section 4.2 where the working and connections between various components of p-NAND & s-NAND gate circuits are discussed. This section mentions the input and output voltages of each component in the circuits of p-NAND & s-NAND gate.

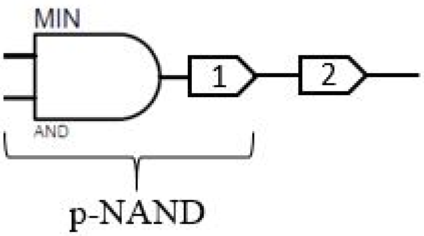


Figure 5.2: Symbolic diagram for s-NAND gate

In the case of p-NAND gate, Table 5.6 shows the input and the output of voltage across the various components of p-NAND as per the circuit shown in Fig. 5.3.

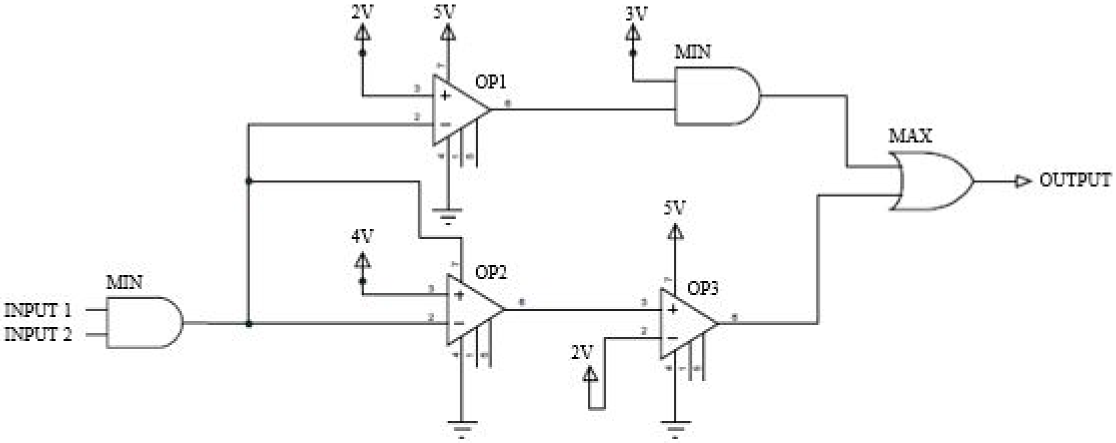


Figure 5.3: Circuit diagram for p-NAND gate

That is when any of the inputs among INPUT 1(VIN 1) and INPUT 2(VIN 2) supplied to MIN 1 gate is 0V then as the gate MIN 1 return the output as minimum voltage among the supplied inputs so the output voltage of MIN 1 will be 0V. The output of the MIN 1 is connected with the input of the t-NOT gate that is the output of MIN 1 is supplied as the input to the t-NOT gate. The t-NOT gate for input 0V, the voltage distribution among the various components of the t-NOT gate is according to Table 5.4 as discussed in section 5.2 so the output of the t-NOT gate will be 3V. That is in case of the p-NAND gate if any of the input among VIN 1 and VIN 2 is 0V then the output of the p-NAND operator will be 3V.

Now we consider another case when both the inputs VIN 1 and VIN 2 are non-zero and any one of the inputs among VIN 1 and VIN 2 is 3V, then MIN 1 return 3V. As in ternary logic, three voltage level is considered that is 0V, 3V, and 5V, non-zero voltages are 3V and 5V so the minimum among the two non-zero input voltages in which any one of the voltage is 3V will always be 3V. The output of MIN 1 that is 3V is applied as the input to the t-NOT gate. So, the t-NOT gate will return 5V as output it means that the final output of the p-NAND gate will be 5V when both the inputs VIN 1 and VIN 2 are non-zero and any one of the input is 3V.

Finally, when both the input VIN 1 and VIN 2 is 5V then the MIN 1 will return the output of 5V which in turn supplied as input to the t-NOT gate. The t-NOT gate for input 5V will give the output 0V. So, when both the input voltages of the p-NAND gate are 5V then in that case the output of the p-NAND operator will be 0V.

Table 5.6: Voltage distribution in the circuit of p-NAND gate

MIN 1		OP-AMP 1					OP-AMP 2					OP-AMP 3					MIN 2		MAX		
VIN 1	VIN 2	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P1	P2	P1	P2	VOUT
0	0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
0	3	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
0	5	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
3	0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
3	3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
3	5	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
5	0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
5	3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
5	5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0

Now considering the s-NAND gate, the electronic circuit of the s-NAND gate is shown in Fig. 4.11 and Table 5.7 show the distribution of voltage among the various components of the s-NAND gate circuit. The s-NAND gate is a p-NAND gate mounted with a t-NOT gate as shown in Fig. 5.2, that is the output voltage of p-NAND gate as shown in Table 5.6 is pass through a t-NOT gate to get the output of s-NAND gate as shown in Table 5.7, in which the input voltage of s-NAND gate that is VIN is the output voltage (VOUT) of the p-NAND gate. This input voltage VIN is supplied as input to the t-NOT 2 gate shown in Fig. 5.2 to get the final output voltage VOUT of s-NAND gate.

According to the Table 5.7 when the VIN (VOUT of p-NAND gate) is 3V is passed as input to the t-NOT gate then the output voltage VOUT will be 5V for every VIN = 3V, according to the functionality of t-NOT gate discussed in section 5.2. The Table 5.7 also shows the voltage distribution in t-NOT gate as per the input voltage VIN. Now when the input voltage VIN is 5V which is supplied as input to t-NOT gate then input and output voltages of each component of the t-NOT gate is according to the table 5.7 and output voltage VOUT will be 0V for each 5V input voltage. And when VIN = 0V supplied as input to t-NOT gate then according to the voltage at various components of t-NOT gate as shown in the table 5.7 the output voltage VOUT will be equal to 3V.

Table 5.7: Voltage distribution in the circuit of s-NAND gate

	OP-AMP 1					OP-AMP 2					OP-AMP 3					MIN		MAX		
VIN	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P1	P2	P1	P2	VOUT
3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3

5.4 p-NOR & s-NOR gate

The p-NOR & s-NOR gate can be considered as an extension of binary NOR gate as in case of binary NOR gate, binary OR gate is mounted with binary NOT gate. In the same way for p-NOR & s-NOR gate MAX gate is mounted with one and two t-NOT gates respectively. In the p-NOR gate, the t-NOT gate is followed by the MAX gate as shown in Fig. 4.12 that is the output of the MAX gate is connected with the input pin of the t-NOT gate. And in the case of the s-NOR gate, the p-NOR gate is mounted with the t-NOT gate as shown in Fig. 5.5 in which the output of the p-NOR gate is connected with the input of the t-NOT gate. The p-NOR & s-NOR gate work according to the truth table as shown in Table 4.9, elaborated working and the connections between various components in the circuit of p-NOR & s-NOR gate already discussed in section 4.3. In this section, the voltage distribution among the various components in the circuit of p-NOR & s-NOR gate is discussed.

In the p-NOR gate, whose circuit is shown in Fig. 5.4 in which the t-NOT gate is followed by the MAX gate and the flow of voltage across various components of the circuit is shown in Table 5.8. The p-NOR gate accepts two inputs that are INPUT 1 (VIN 1) & INPUT 2 (VIN 2) and give single OUTPUT(VOUT).

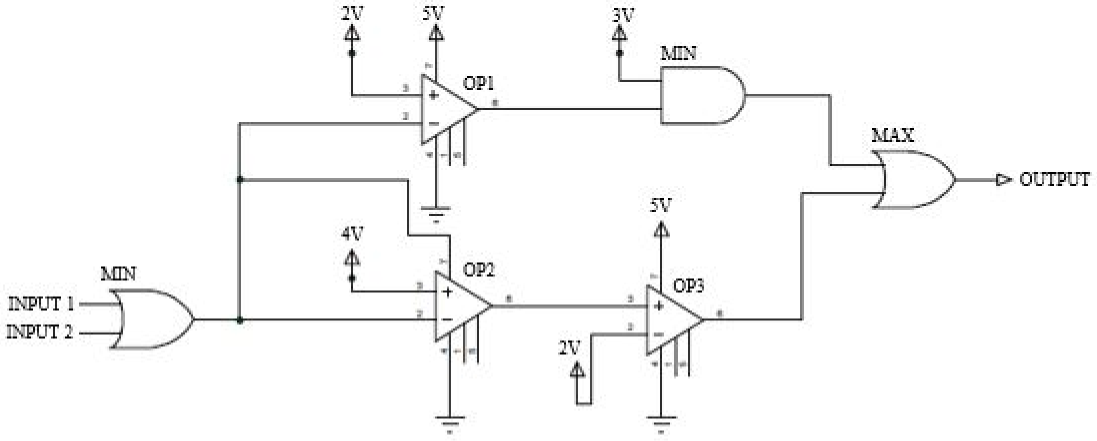


Figure 5.4: Circuit diagram for p-NOR gate

According to Table 5.8 when both the inputs VIN 1 & VIN 2 are 0V then the MAX gate gives output 0V as the MAX gate returns the maximum voltage out of the supplied input voltages. The output of the MAX gate is given as input to the t-NOT gate, the input and the output voltages of the various components of the t-NOT gate is shown in Table 5.8 so for input 0V the t-NOT gate will give output voltage as 3V as final output VOUT of the p-NOR gate. The working of the t-NOT gate is already discussed in section 5.2.

When one input VIN 1 = 0V and another input VIN 2 = 3V or vice-versa then, in that case, the MAX gate always returns 3V as output which is given as input to the t-NOT gate. So final output VOUT will be 5V as the t-NOT gate for input 3V will give output as 5V. And if both the inputs that are VIN 1 & VIN 2 equal to 3V in that case also MAX gate give output 3V which is passed as input to the t-NOT gate to get the final output VOUT = 5V.

Now for VIN 1 = 0V and VIN 2 = 5V or vice-versa the MAX gate will give output as 5V which in turn passes as input to the t-NOT gate to get the final output voltage VOUT = 0V. Similarly, for VIN 1 = VIN 2 = 5V in that case also MAX gate will return the maximum voltage form the supplied input voltages which are 5V. As the output of the MAX gate is connected with the input of the t-NOT gate, and the t-NOT gate will give the final output VOUT as 0V for input 5V. Now considering the case when both the inputs VIN 1 & VIN 2 are any non-zero unequal voltages then also MAX gate will return the maximum voltage which will be 5V and again the t-NOT gate will give the final output VOUT as 0V.

Table 5.8: Voltage distribution in the circuit of p-NOR gate

		OP-AMP 1					OP-AMP 2					OP-AMP 3					MIN		MAX 2			
VIN 1	VIN 2	MAX 1	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P1	P2	P1	P2	VOUT
0	0	0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
0	3	3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
0	5	5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
3	0	3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
3	3	3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
3	5	5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
5	0	5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
5	3	5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
5	5	5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0

In s-NOR gate, MAX gate is mounted with two t-NOT gate such as one following the another or we can also say that output of p-NOR gate is connected with the input of t-NOT gate depicted in Fig. 5.5, whose output give the final output voltage VOUT.

As shown in the Table 5.9 below where VIN is the output voltage VOUT of the p-NOR gate according to the Table 5.8 supplied as input to the t-NOT gate to get the final output of s-NOR gate. Table 5.9 shows the voltage at each pin in the t-NOT gate according to the supplied input VIN which is the output of the p-NOR gate to

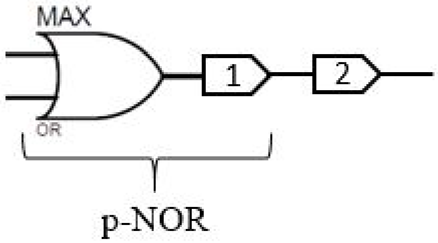


Figure 5.5: Symbolic diagram for s-NOR gate

get the final output VOUT. So, for every input voltage VIN = 3V, supplied as input to t-NOT gate the final output VOUT will be 5V in the same way for input VIN = 5V and VIN = 0V the output VOUT will be 0V and 3V respectively.

Table 5.9: Voltage distribution in the circuit of s-NOR gate

	OP-AMP 1					OP-AMP 2					OP-AMP 3					MIN		MAX		
VIN	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P2	P3	P4	P6	P7	P1	P2	P1	P2	VOUT
3	3	2	0	0	5	3	4	0	3	3	2	3	0	5	5	3	0	0	5	5
5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
5	5	2	0	0	5	5	4	0	0	5	2	0	0	0	5	3	0	0	0	0
0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3
0	0	2	0	5	5	0	4	0	0	0	2	0	0	0	5	3	5	3	0	3

5.5 Ternary Decoder

A ternary decoder can be considered as an extension of binary decoder, basically, the binary decoder has n input and 2^n output lines in which a single output line remain enable at a time according to the input. Similarly, the ternary decoder has one input line and three output lines, one output line remains enabled at a time. The circuit diagram for the ternary decoder is shown in Fig. 4.16 and the corresponding truth table is shown in Table. 5.5 according to which when X = 0V than the line X_0 will get enable that will decode that the supplied input is 0V and the rest of the two lines remain disabled. In the same way, if X = 3V than the line X_1 will get enable to depict the supplied input is 3V and the rest of the two lines remain disabled. And if

X = 5V than the output line X_2 will get enable and the rest of the two lines remain disabled (inactive).

Table 5.10: Truth Table Ternary Decoder

X	X_0	X_1	X_2
0	5	0	0
3	0	5	0
5	0	0	5

The electronic components and the connections between the components are already discussed in section 4.4, in this section voltage distribution across the circuit, is considered as shown in Table 5.11. Now according to the circuit diagram of the ternary decoder as shown in Fig. 4.16. For input X = 0V, which is supplied as input to t-NOT 1 gate, pin P3 of OP-AMP 2 and pin P2 of MAX gate, the output of t-NOT 1 for input 0V will be 3V which in turn is connected with the input of t-NOT 2 gate. The output of t-NOT 2 that is 5V is supplied as the input to pin P1 of MAX gate and P3 of OP-AMP 1, whereas the other pins of OP-AMP 1 are as follows pin P2 is connected with a voltage source of 4V, P4 is ground, P7 is connected with 5V so the output pin P6 will give 5V to the output line X_0. And as the pin P1 and P2 of the MAX gate get the input voltages as 0V and 5V so the MAX gate gives the output 5V which is passed from the t-NOT 3 gate to get 0V at output line X_1. Finally, OP-AMP 2 which receives the input 0V at pin P3 and having 4V at P2, P4 is ground, P7 is connected with 5V gives the output 0V at pin P6 connected with output line X_2. So, for X = 0V output line X_0 remain active and rest of the two output lines X_1 & X_2 remain inactive. Similarly, when the input X = 3V, the t-NOT gate 1, pin P3 of OP-AMP 2, P2 of MAX gate receives input as 3V so t-NOT 2 give input as 0V to pin P1 of MAX gate and P3 of OP-AMP 1 which disable the output line $X_0 = 0$. The output of the MAX gate which is 3V connected with the input of t-NOT 3 to give 5V at output line X_1. And output line X_2 also remain disabled as pin P6 of OP-AMP 3 gives 0V as output. Now in the same way for X = 5V, the output pin P6 of OP-AMP

3 gives the output as 5V to enable the output line X_3, and the rest of the two output lines remain disabled.

Table 5.11: Ternary Decoder

			OP-AMP 1					MAX			OP-AMP 2								
X	t-NOT 1	t-NOT 2	P2	P3	P4	P6	P7	P1	P2	P3	P2	P3	P4	P6	P7	t-NOT 3	X0	X1	X2
0	0	3	4	5	0	5	5	5	0	5	4	0	0	0	5	5	5	0	0
3	3	5	4	0	0	0	5	0	3	3	4	3	0	0	5	3	0	5	0
5	5	0	4	3	0	0	5	3	5	5	4	5	0	5	5	5	0	0	5

5.6 Ternary Adder

Separate circuits are designed for calculating sum of two ternary bits and generated carry(if any) based on the truth Table 4.16, working and connections between the electronic components used in the circuit has already been discussed in section 4.6. The truth table for ternary adder can be classify into three different tables according to input which remain constant for three different values of another input as shown in the Tables 5.12, 5.13 & 5.14. In Table 5.12 input A is 0V and input B vary from 0V to 5V similarly in Table 5.13 and 5.14 input A remain constant as 3V & 5V respectively where as input B vary from 0V to 5V.

According to the circuit for calculating sum S as shown in Fig. 5.6 in which when the input A is 0V then the output line X_0 of ternary decoder remains enabled to carry the input and rest of the two output lines of ternary decoder remain disabled. Table 5.12 is representing the voltage across the circuit when the input A = 0V and another input B vary from 0V to 5V. As A = 0V so the output line X_0 of ternary decoder remains active and the rest of the output lines remain disabled represented by 0. Now for input B = 0V, input pin P1 and P2 of MAX 1 gate receive 0V & 0V respectively, return the maximum voltage among the supplied inputs which will be 0V in this case. The output of MAX 1 is connected with one of the input P1 of the MAX 2 gate and the rest of the input lines of MAX 2 gate that is P2 & P3 remains disabled can be considered as 0. So the output of the MAX 2 gate which is providing the resultant S of supplied inputs A and B will be 0V. Now considering B = 3V then MAX 1 gives the output as 3V which is connected with the input pin P1 of MAX 2 gate so output S will be 3V. And for input B = 5V MAX 1 gate return 5V as output which is supplied to input pin P1 of MAX 2 gate to get the final output S = 5V.

Table 5.12: Voltage across the circuit when the output line X_0 of ternary decoder is enabled (E)

			MAX 1		MAX 2			
A	B	X_0	P1	P2	P1	P2	P3	S
0	0	E	0	0	0	0	0	0
0	3	E	0	3	3	0	0	3
0	5	E	0	5	5	0	0	5

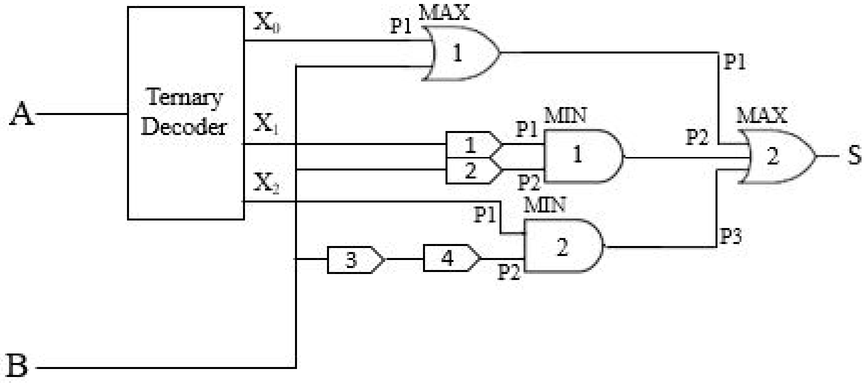


Figure 5.6: Circuit of ternary adder for calculating sum S

According to Table 5.13 which is representing the voltage across the circuit when input A = 3V for every B from 0V to 5V, that is only output line X_1 of ternary decoder remain enabled and rest of the output lines remain disabled. When the input B = 0V then the input pins P1 and P2 of MIN 1 receive 5V and 3V respectively as the input A and input B pass through t-NOT 1 & t-NOT 2 gate before connecting with the input pins of MIN 1 gate. The output pin of MIN 1 is connected with the input pin P2 of MAX 2 so the output of MIN 1 is passed as one of the inputs to MAX 2 and like the rest of the input pins, P2 and P3 remain disabled so S = 3V. And when input B = 3V then the output of both the t-NOT gates that is t-NOT 1 & t-NOT 2 will be 5V. And both the input pins P1 and P2 of MIN 1 receive 5V each and give 5V to pin P2 of MAX 2 so S also will be 5V. Now for B = 5V gate t-NOT 1 gives the output 5V whereas t-NOT 2 gives 0V, both output voltages of t-NOT gates are passed as input to MIN 1 which return the minimum voltage among the supplied inputs so MIN 1 gives 0V as input to P2 of MAX 2, therefore, we get S = 0V as the rest of the input pins of MAX 2 gate will remain 0V.

Table 5.13: Voltage across the circuit when the output line X_1 of ternary decoder is enabled (E)

					MIN 1		MAX 2			
A	B	X1	t-NOT 1	t-NOT 2	P1	P2	P1	P2	P3	S
3	0	E	3	0	5	3	0	3	0	3
3	3	E	3	3	5	5	0	5	0	5
3	5	E	3	5	5	0	0	5	0	0

Now considering Table 5.14 in which input A remains 5V and input B varies from 0V to 5V, so the output line X_2 of ternary decoder remain enable and rest of the output lines remain disabled. For B = 0V, both the input pins P1 & P2 of MIN 2 gate receive 5V as pin P1 is directly connected with output line X_2 of a ternary decoder which carries input A. And input B pass through two t-NOT gates that are t-NOT 3 & t-NOT 4 before connecting to input pin P2 of MIN 2 gate so MIN 2 gives the output voltage as 5V which is connected with the input pin P3 of the MAX 2 gate. The MAX 2 gate returns the maximum voltage among the supplied input voltages, as input pin P1 & P2 of MAX 2 gate remains disabled so we get output S as 5V. And when the input B = 3V, which passes through t-NOT 3 gate and the output of t-NOT 3 that is 5V further pass from t-NOT 4 gate which gives the output voltage as 0V. So, MIN 2 gate get the input voltages at pin P1 & P2 as 5V and 0V respectively to give output as 0V which is further connected with pin P3 of MAX 2 gate to give output S = 0V. Finally, when input B = 5V in that case t-NOT 3 gives the output 0V which is connected as an input to t-NOT 4 gate therefore t-NOT 4 gives the output as 3V. The output of the t-NOT 4 gate is supplied as input to pin P2 of MIN 2 gate which is having 5V at pin P1 therefore MIN 2 gate return the minimum voltage that is 3V to pin P3 of MAX 2 gate to get the final output S = 3V.

Table 5.14: Voltage across the circuit when the output line X_2 of ternary decoder is enabled (E)

					MIN 1		MAX 2			
A	B	X2	t-NOT 3	t-NOT 4	P1	P2	P1	P2	P3	S
5	0	E	0	3	5	5	0	0	5	5
5	3	E	3	5	5	0	0	0	0	0
5	5	E	5	0	5	3	0	0	3	3

A separate circuit is designed for calculating the generated carry with input A and B based on the truth table 4.12 in which when input A is 0 in that case for any value of input B varying from 0 to 2, no carry would be generated. So, the circuit is designed for input A as 3V & 5V and each input, B varying from 0V to 5V. Circuit is shown in Fig. 5.7 in which there are two inputs A & B and the output C that is generated carry (if any) at the output of MAX 1 gate. When the input A is 0V in that case no carry would be generated for any value of input B so output line X_0 of the ternary decoder will never be required in the circuit.

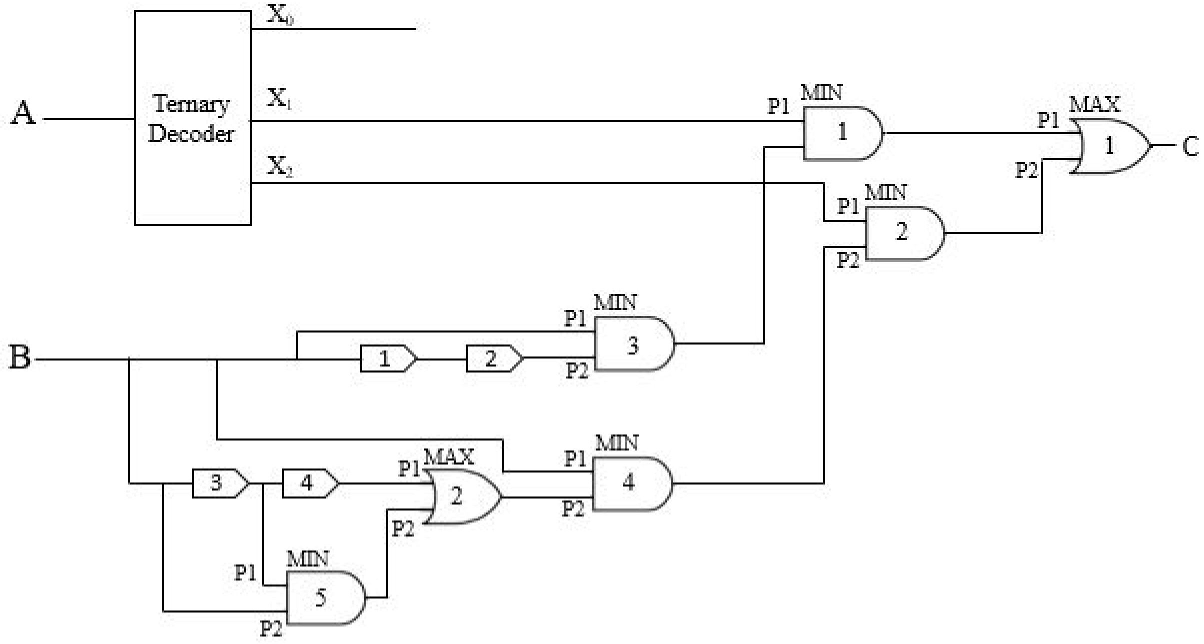


Figure 5.7: Circuit of ternary adder for calculating calculating carry C

With input A as 3V for every input B varying from 0V to 5V, the voltage distribution across the circuit is shown in Table. 5.15 in which the output line X_1 of ternary decoder remain enabled to carry the input A so the input pin P1 of MIN 1 gate will always remain 3V. Now when the value of B = 0V then P1 of MIN 3 gate will also be 0V as it is directly connected with the input B, and P2 of MIN 3 gate get 5V as it is followed by two t-NOT gates that is t-NOT 1 & t-NOT 2 before connecting with input B. As the output of MIN 3 gate is connected with input pin P2 of MIN 1 gate so the output of MIN 1 gate will be 0V which is connected to the input pin P1 of the MAX 1 gate. As the MAX 1 gate return the maximum voltage among the supplied inputs P1 & P2 so no carry would be generated in this case. And another input pin P2 of MAX 1 gate is connected with the output of MIN 2 gate whose input pin P1 is connected with the output line X_2 of a ternary decoder which always remains disabled which is 0 in a case when input A = 3V. Because the MAX 1 gate returns the maximum voltage as output among the supplied input voltage at pin P1 & P2, therefore, no carry would be generated in this case that is C = 0. For input B = 3V, as the input B is directly connected with the t-NOT 1 gate and P1 of MIN 3 gate so t-NOT 1 gate gives the output voltage as 5V which is supplied as input to t-NOT 2 gate whose output is connected with the input pin P2 of MIN 3 gate. So MIN 3 gate gives the output as 0V connected with the input pin P2 of MIN 1 gate, as MIN gate return the minimum voltage among the supplied inputs, MAX 1 gate get 0V at input pin P1 and finally in this case also carry C remain 0. Similarly for input B = 5V, the input pins P1 and P2 of MIN 3 gate will get 5V & 3V respectively which in turn gives the output 3V connected to the input pin P2 of MIN 1 gate. The output voltage 3V of MIN 1 gate is supplied to input pin P1 of MAX 1 gate which gives output that is generated carry C = 3V.

Table 5.15: Voltage across the circuit for calculating carry C when the output line X_1 of ternary decoder is enabled (E)

					MIN 3		MIN 1		MAX 1		
A	B	X1	t-NOT 1	t-NOT 2	P1	P2	P1	P2	P1	P2	C
3	0	E	0	3	0	5	3	0	0	0	0
3	3	E	3	5	3	0	3	0	0	0	0
3	5	E	5	0	5	3	3	3	3	0	3

The Table. 5.16 shows the voltage across the components of the circuit when the input A = 5V and the input B vary from 0V to 5V, in this case, the output line X_2 of ternary decoder, only remain enabled to carry supplied input A. Now for B = 0V as the input B is directly connected with the input pin P1 of MIN 4 gate, t-NOT 3 gate and P2 of MIN 5 gate, so all the pins receive 0V as input. The pin P1 of MIN

5 gate gets 3V as it is connected with the output of the t-NOT 3 gate and the same voltage is supplied as input to the t-NOT 4 gate so the MIN 5 gate gives output 0V. The input pins P1 & P2 of MAX 2 gate receive the input voltage as 5V which is the output of t-NOT 4 gate and 0V that is the output of MIN 5 gate respectively, the output voltage 5V of MAX 2 gate is supplied as input to MIN 4 gate through pin P2. The output of the MIN 4 gate that is 0V is passed to input pin P2 of the MIN 2 gate and the output of the MIN 2 gate which will be 0V is further supplied to input pin P2 of MAX 1 gate. As another input pin P1 of the MAX 1 gate is connected with the output of the MIN 1 gate which will be 0V is this case as the input pin P1 of MIN 1 gate is connected with the disabled output line X_1 of a ternary decoder. So, MAX 1 gate gives the output as 0V which means no carry would be generated in this case. Similarly when the input B is 3V then the input pin P1 & P2 of MIN 5 gate receive the input voltage is 5V and 3V respectively and give output as 3V which is supplied to pin P2 of MAX 2 gate whereas pin P1 of MAX 2 gate get the voltage as 0V which is the output of t-NOT 3 gate. The MAX 2 gate gives the output as 3V which is supplied to input pin P2 of MIN 4 gate whose pin P1 is directly connected with input B, therefore, MIN 4 gate gives the output voltage as 3V which in turns supplied to input pin P2 of MIN 2 gate and as the pin P1 of MIN 2 gate is connected with enabled output line X_2 of ternary decoder so MIN 2 gate give the output as 3V. This output voltage 3V of the MIN 2 gate is supplied to pin P2 of MAX 1 gate which gives the output as 3V representing the generated carry.

In the same way for input B = 5V, MIN 5 gate gives the output as 0V, provided to pin P2 of MAX 2 gate, whereas pin P1 of MAX 2 gate gets the input as 3V so MAX 2 gate return 3V as output. The output of the MAX 2 gate passed to input pin P2 of MIN 4 gate, as the pin P1 having voltage 5V therefore, MIN 4 gate gives the output voltage as 3V, connected to pin P2 of MIN 2 gate. Therefore MIN 2 gate return 3V as output that is minimum voltage among the supplied input at pin P1 &

P2, where pin P1 connected with output line X_2 of ternary decoder, and the output of MIN 2 gate provided to the input pin P2 of MAX 1 gate. Therefore MAX 2 gate return 3V as generated carry after receiving 0 at pin P1 and 3V at pin P2.

Table 5.16: Voltage across the circuit when the output line X_2 of ternary decoder is enabled (E)

					MIN 5		MAX 2		MIN 4		MIN 2		MAX 1		
A	B	X2	t-NOT 3	t-NOT 4	P1	P2	P1	P2	P1	P2	P1	P2	P1	P2	C
5	0	E	0	3	3	0	5	0	0	5	5	0	0	0	0
5	3	E	3	5	5	3	0	3	3	3	5	3	0	3	3
5	5	E	5	0	0	5	3	0	5	3	5	3	0	3	3

5.7 Ternary Addition Algorithm and MIN & MAX gate Logic

In this section, example of ternary addition is presented based on Ternary Addition Algorithm 4 as discussed in Section 4.5. And experimental results are shown based on proteus-8 for using binary AND and OR gate as MIN and MAX gate for ternary logic or even fuzzy logic. Where two batteries of different voltages (3V and 5V) are connected to the inputs of binary AND and OR gate, the voltage across the LED connected at the output of both the gates are checked using a voltmeter. And with the reading of the voltmeter at the output, it is proved that binary AND and OR gates return the maximum and minimum value among the supplied voltages. So, there is no requirement for any special hardware for higher radix. Even many researchers have proposed the hardware design for MIN and MAX gates based on higher radix (MVL).

Figure 5.8 shows that 3V and 0.7V batteries are connected at the input of the OR gate, where voltmeter across the LED(ON) connected at the output shows that binary OR gate gives +2.42V. That is OR gate returns the maximum voltage among the input voltages. In a similar way Figure 5.9 in which 5V and 0.5V batteries are connected as the input of binary AND gate. And with the reading of voltmeter(0.00V) connected across the LED(OFF) at the output prove that binary AND gate return the minimum voltage among the applied input voltages.

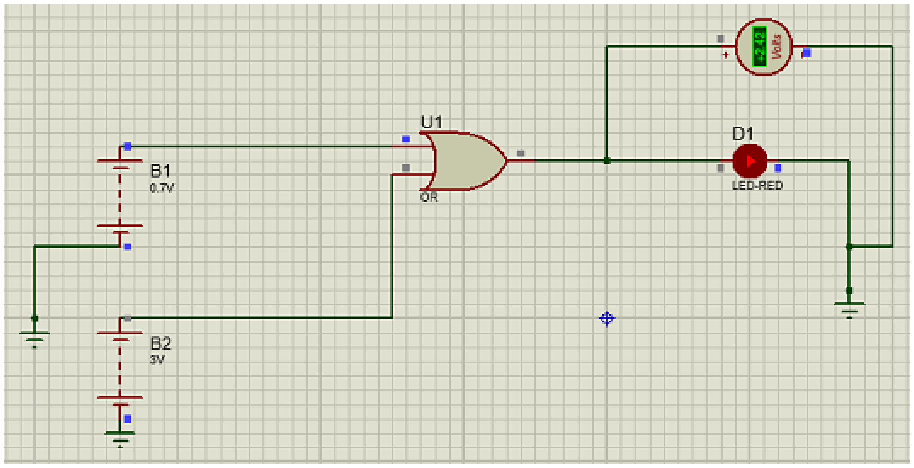


Figure 5.8: Binary OR gate return maximum among the applied inputs voltages

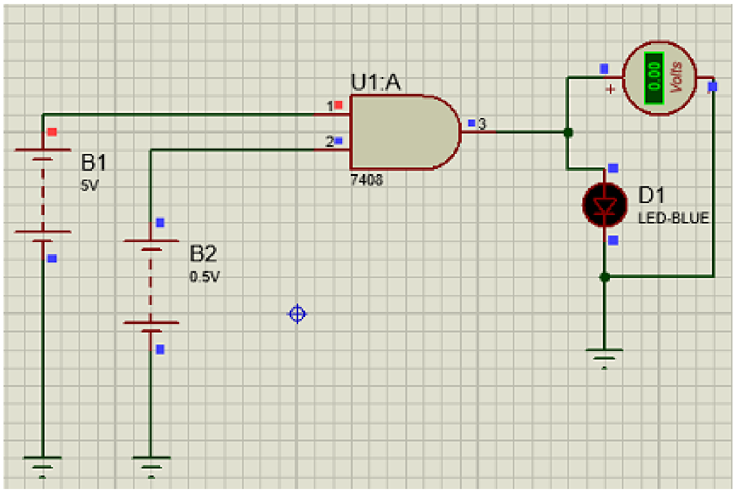


Figure 5.9: Binary AND gate return minimum among the applied inputs voltages

Now, we will see ternary addition example based on the proposed algorithm. Let us, we have two ternary numbers as

$a = 12$ and $b = 21$, considering $a_0 = 2, a_1 = 1$ and $b_0 = 1, b_1 = 2$.

$c_0 = a_0 + b_0$, carry k = 1 will be generated

$c_0 = 0$.

push(k), into the stack

Now addition of another two bits that is a_1 & b_1 is performed

$c_1 = a_1 + b_1$, carry k = 1 will be generated

$c_1 = 0$.

as stack $\neq$ NULL

so, pop(k), retrieve previously generated carry with sum of a_0 and b_0 and add the carry k in the sum of next two bits

$c_1 = c_1 + k$ so, $c_1 = 1$

Now push(k), k is generated carry with addition of a_1 and b_1

As there is no further bits to be added so check the status of stack, still stack $\neq$ NULL

so, consider another pair of bits by our own that is $a_2 = 0$ and $b_2 = 0$ perform addition

$c_2 = a_2 + b_2$ so, $c_2 = 0$

Now pop(k), and add the carry $c_2 = c_2 + k$ so, $c_2 = 1$

finally, return $C = c_2c_1c_0$ that is $(110)_3$

To verify the results consider the decimal equivalent of a and b that is equal to 5 and 7 respectively. And the decimal equivalent of the result obtained by ternary addition of a and b that is 110 according to the proposed algorithm is $c = 12$, which is equal to the sum of a & b that is 5 and 7.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

A novel multiplication algorithm is proposed based on the study of various vedic multiplication methods, a binary multiplexer circuit is design on the basis of the algorithm. The algorithm basically work on the basis of generic equation to develop multiple set of numbers represented as $S = \{\{S_0\}, \{S_1\}, \{S_2\}...\{S_n\}\}$ where $n \in N$. And set of equations to check whether to how many sets the minimum number among the multiplier & multiplicand belong to. Accordingly 0's are appended in the binary representation of maximum number among among the multiplier & multiplicand, finally addition of the generated binary numbers is performed. The circuit of the binary multiplexer based in the algorithm is simple, require less number of gates which in turn result in lowering the interconnection overhead, delay and power consumption. In future algorithm can be extended for the multiplication of fractional numerals.

To contribute in the field of Multivalued Logic, circuits are designed for various operators based on ternary logic where ternary system work on three logical bits that is {0, 1, 2} or we can say on three level of voltages instead of two as in current binary system. Ternary inverter circuit (t-NOT gate) is designed which is considered as a building block of ternary system and ternary NAND, ternary NOR gate and ternary decoder circuits are proposed based on the truth tables. A ternary adder circuit is also design for performing the addition of ternary bits, the circuit is basically comprises of two separate circuits one for the computation of sum and another for generated carry. Finally a ternary addition algorithm is also proposed using a stack of size one for performing the addition of two ternary bits at a time. As the decimal number system is globally accepted system for performing any mathematical calculation, so if the computer system is developed which would also work on decimal number system that is base 10 instead of base 2 as in case of current binary machine. Then the system could be considered as perfect computing machine with highest performance. And this gap between the current computer system and a perfect computer machine, provide a wide future scope in the field of Multivalued Logic.

REFERENCES

[1] P. Holay, “Vedic astronomy,” *Shri Babasaheb Apte Smarak Samitee, Nagpur*, 1994.

[2] B. Datta and A. N. Singh, *History of Hindu mathematics.* Asia Publishing House; Bombay, 1935.

[3] S. R. 2004, *Ak krishnaswami ayyangar’s works on the history of indian mathematics.*

[4] M. A. A. 1915, *A history of Sanskrit literature.* D. Appleton, vol. 3.

[5] E. B. Cowell and R. A. Neil, *Divyavadana*, 1886.

[6] D. B. 1932, “The science of the sulba,” *Calcutta Univ. Calcutta*,

[7] A. Weber, *The history of Indian literature.* Routledge, 2013.

[8] R Sridharan, “Sanskrit prosody, pigala sūtras and binary arithmetic,” in *Contributions to the History of Indian Mathematics*, Springer, 2005, pp. 33–62.

[9] E. Boag, “Lattice multiplication,” *BSHM Bulletin*, vol. 22, no. 3, pp. 182–184, 2007.

[10] H. T. Colebrooke, “Algebra with arithmetic and mensuration from the sanskrit of brahmegupta and bhaskara (london: 1817),” *Google Scholar*,

[11] T. J. 1816, *Lilawati: Or a treatise on arithmetic or geometry by bhascara acharya.*

[12] J. D. 1964, “A suggestion for a fast multiplier,” *IEEE Trans. on Electron. Comput.*, vol. 6, no. EC-13, p. 754,

[13] T. H and A. H. R. 2004, “A time-area-power efficient multiplier and square architecture based on ancient indian vedic mathematics.,” in *ESA/VLSI*, pp. 434–439.

[14] K. C. M. Tiwari H D Gankhuyag G and C. Y. B. 2008, “Multiplier design based on ancient indian vedic mathematics,” in *SoC Des. Conf., 2008. ISOCC’08. Int.*, IEEE, vol. 2, pp. II–65.

[15] D. H. S and M. A. 2008, “A reduced-bit multiplication algorithm for digital arithmetic,” *Int. J. of Comput. and Math. Sci.*, vol. 2, no. 2,

[16] P. R. Pradhan M and S. S. K. 2011, “Speed comparison of 16x16 vedic multipliers,” *Int. J. Comput. Appl. (0975–8887)*, vol. 21, no. 6,

[17] B. P. Saha P Kumar D and D. A. 2014, “Vedic division methodology for high-speed very large scale integration applications,” *J. Eng.*, vol. 2014, no. 2, pp. 51–59,

[18] P. M and P. R. 2014, “High speed multiplier using nikhilam sutra algorithm of vedic mathematics,” *Int. J. Electron.*, vol. 101, no. 3, pp. 300–307,

[19] K. G. G and S. S. K. 2015, “Implementation of a high speed multiplier for high-performance and low power applications,” in *VLSI Des. Test (VDAT), 2015 19th Int. Symp.*, IEEE, pp. 1–4.

[20] W. S, “High-speed monolithic multipliers for real-time digital signal processing,” *Computer*, no. 10, pp. 19–29, 1978.

[21] I. R.S. D. Pushpangadan R Sukumaran V and S. V. 2009, “High speed vedic multiplier for digital signal processors,” *IETE J. Res.*, vol. 55, no. 6, pp. 282–286,

[22] R. J. Tocci, *Digital Systems: principles and applications.* Pearson Education India, 1996.

[23] B. Choi, “Advancing from two to four valued logic circuits,” in *Industrial Technology (ICIT), 2013 IEEE International Conference on*, IEEE, 2013, pp. 1057–1062.

[24] C. M. Allen and D. D. Givone, “A minimization technique for multiple-valued logic systems,” *IEEE Transactions on Computers*, no. 2, pp. 182–184, 1968.

[25] R. W. Keyes, “The evolution of digital electronics towards vlsi,” *IEEE Journal of Solid-State Circuits*, vol. 14, no. 2, pp. 193–201, 1979.

[26] S. S. Dawley and P. A. Gajbhiye, “Design and comparative analysis of binary and quaternary logic circuits,” in *Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), World Conference on*, IEEE, 2016, pp. 1–6.

[27] M. Mukaidono, “Regular ternary logic functions? ternary logic functions suitable for treating ambiguity,” *IEEE transactions on computers*, no. 2, pp. 179–183, 1986.

[28] T. Araki, H. Tatsumi, M. Mukaidono, and F. Yamamoto, "Minimization of incompletely specified regular ternary logic functions and its application to fuzzy switching functions," in *ismvl*, IEEE, 1998, p. 289.

[29] S. Lin, Y.-B. Kim, and F. Lombardi, "Cntfet-based design of ternary logic gates and arithmetic circuits," *IEEE transactions on nanotechnology*, vol. 10, no. 2, pp. 217–225, 2009.

[30] C. Mingoto, "A quaternary half-adder using current-mode operation with bipolar transistors," in *36th International Symposium on Multiple-Valued Logic (ISMVL'06)*, IEEE, 2006, pp. 15–15.

[31] K. C. Smith, "The prospects for multivalued logic: A technology and applications view," *IEEE Transactions on Computers*, no. 9, pp. 619–634, 1981.

[32] Z. G. Vranesic and K. C. Smith, "Engineering aspects of multi-valued logic systems," *Computer*, vol. 7, no. 9, pp. 34–41, 1974.

[33] A. Heung and H. Mouftah, "Depletion/enhancement cmos for a lower power family of three-valued logic circuits," *IEEE Journal of Solid-State Circuits*, vol. 20, no. 2, pp. 609–616, 1985.

[34] H. Mouftah and A. Garba, "Vlsi implementation of a 5-trit full adder," in *IEE Proceedings G-Electronic Circuits and Systems*, IET, vol. 131, 1984, pp. 214–220.

[35] H. Mouftah and K. Smith, "Injected voltage low-power cmos for 3-valued logic," in *IEE Proceedings G-Electronic Circuits and Systems*, IET, vol. 129, 1982, pp. 270–272.

[36] S. Lin, Y.-B. Kim, and F. Lombardi, "A novel cntfet-based ternary logic gate design," in *2009 52nd IEEE International Midwest Symposium on Circuits and Systems*, IEEE, 2009, pp. 435–438.

[37] P. C. Balla and A. Antoniou, "Low power dissipation mos ternary logic family," *IEEE Journal of Solid-State Circuits*, vol. 19, no. 5, pp. 739–749, 1984.

[38] A Srivastava and K Venkatapathy, "Design and implementation of a low power ternary full adder," *VLSI design*, vol. 4, no. 1, pp. 75–81, 1996.

[39] T. Rajashekhara and I.-S. Chen, "A fast adder design using signed-digit numbers and ternary logic," in *IEEE Technical Conference on Southern Tier*, IEEE, 1990, pp. 187–194.

[40] L. Phanindra, M. Rajath, V Rakesh, and K. V. Patel, "A novel design and implementation of multi-valued logic arithmetic full adder circuit using cntfet," in *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT)*, IEEE, 2016, pp. 563–568.

[41] S. K. S. 1950, "On sridhara's rational solution of nx$\hat{2}$+ 1= y$\hat{2}$," *Ganita*, vol. 1, pp. 1–12,

[42] S. A.R. S. 1966, "Shri brahmagupta viracita brahma-sphuta siddhanta," *Vol. I, Indian Inst. of Astron. and Sanskrit Res., New Delhi, Kuttakadhyya*, pp. 64–65,

[43] S. S. S. P and J. U. 1979, *The bakhshali manuscript. an ancient treatise of indian arithmetic.*

[44] R. M. 1912, "The ganita-sāra-sangraha of mahāvīrācārya," *Bull. Amer. Math. Soc*,

[45] L. G, *1703. explication de l'arithmétique binaire [explanation of binary arithmetic]; gerhardt, mathematical writings.*

[46] W. J.-P. Kuang S-R and G. C.-Y. 2009, "Modified booth multipliers with a regular partial product array," *IEEE Trans. Circuits Syst. II: Express Briefs*, vol. 56, no. 5, pp. 404–408,

[47] W.-C. Yeh and C.-W. Jen, "High-speed booth encoded parallel multiplier design," *IEEE transactions on computers*, vol. 49, no. 7, pp. 692–701, 2000.

[48] J.-Y. Kang and J.-L. Gaudiot, "A simple high-speed multiplier design," *IEEE Transactions on computers*, vol. 55, no. 10, pp. 1253–1258, 2006.

[49] J. S and C. G, "High speed 16- bit vedic multiplier using modified carry select adder," *An International Journal of Engineering Sciences*, vol. 17, no. 10, pp. 303–309, 2016.

[50] S Ravi, A. Patel, M. Shabaz, P. M. Chaniyara, and H. M. Kittur, "Design of low-power multiplier using ucsla technique," in *Artificial Intelligence and Evolutionary Algorithms in Engineering Systems*, Springer, 2015, pp. 119–126.

[51] S. Mishra and M. Pradhan, "Synthesis comparison of karatsuba multiplierusing polynomial multiplication, vedic multiplier and classical multiplier," *International journal of computer applications*, vol. 41, no. 9, pp. 13–17, 2012.

[52] C. Eyupoglu, "Investigation of the performance of nikhilam multiplication algorithm," *Procedia-Social and Behavioral Sciences*, vol. 195, pp. 1959–1965, 2015.

[53] S. Patil, D. Manjunatha, and D. Kiran, “Design of speed and power efficient multipliers using vedic mathematics with vlsi implementation,” in *2014 International Conference on Advances in Electronics Computers and Communications*, IEEE, 2014, pp. 1–6.

[54] S. B. K. Tirtha and V. S. Agrawala, *Vedic mathematics*. Motilal Banarsidass Publ., 1992, vol. 10.

[55] S. P. Dwivedi, “An efficient multiplication algorithm using nikhilam method,” 2013.

[56] R. P. Brent and P. Zimmermann, *Modern computer arithmetic*. Cambridge University Press, 2010, vol. 18.

[57] S. Nivas and N Kayalvizhi, “Implementation of power efficient vedic multiplier,” *International Journal of Computer Applications*, vol. 43, no. 16, pp. 21–24, 2012.

[58] M Pradhan and R Panda, “Design and implementation of vedic multiplier,” *AMSE Journal, Series D, Computer Science and Statistics, France*, vol. 15, no. 2, pp. 1–19, 2010.

[59] E. Shahrom and S. A. Hosseini, “A new low power multiplexer based ternary multiplier using cntfets,” *AEU-International Journal of Electronics and Communications*, vol. 93, pp. 191–207, 2018.

[60] N. Takagi and K. Nakashima, “Discrete interval truth values logic and its application,” *IEEE Transactions on Computers*, vol. 49, no. 3, pp. 219–229, 2000.

[61] D. Etiemble, “On the performance of multivalued integrated circuits: Past, present and future,” *IEICE Transactions on Electronics*, vol. 76, no. 3, pp. 364–371, 1993.

[62] S. K. Sahoo, G. Akhilesh, R. Sahoo, and M. Muglikar, “High-performance ternary adder using cntfet,” *IEEE Transactions on Nanotechnology*, vol. 16, no. 3, pp. 368–374, 2017.

[63] B. Hayes, “Computing science: Third base,” *American scientist*, vol. 89, no. 6, pp. 490–494, 2001.

[64] P. N. Marinos, “Fuzzy logic and its application to switching systems,” *IEEE Transactions on Computers*, vol. 100, no. 4, pp. 343–348, 1969.

[65] J. Berman and M. Mukaidono, “Enumerating fuzzy switching functions and free kleene algebras,” *Computers & mathematics with applications*, vol. 10, no. 1, pp. 25–35, 1984.

[66] J. Łukasiewicz, “On three-valued logic,” *The Polish Review*, pp. 43–44, 1968.

[67] E. L. Post, “Introduction to a general theory of elementary propositions,” *American journal of mathematics*, vol. 43, no. 3, pp. 163–185, 1921.

[68] G. Frieder, A Fong, and C. Chow, “A balancedternary computer,” in *Conference Record of the 1973 International Symposium on Multiple-valued Logic*, 1973, pp. 68–88.

[69] I. Halpern and M. Yoeli, “Ternary arithmetic unit,” in *Proceedings of the Institution of Electrical Engineers*, IET, vol. 115, 1968, pp. 1385–1388.

[70] B. Parhami and M. McKeown, “Arithmetic with binary-encoded balanced ternary numbers,” in *Signals, Systems and Computers, 2013 Asilomar Conference on*, IEEE, 2013, pp. 1130–1133.

[71] Y. Yamamoto and M. Mukaidono, “Meaningful special classes of ternary logic functions-regular ternary logic functions and ternary majority functions,” *IEEE transactions on computers*, vol. 37, no. 7, pp. 799–806, 1988.

[72] M. Mukaidono, “The b-ternary logic and its applications to the detection of hazards in combinational switching circuits,” in *Proceedings of the eighth international symposium on Multiple-valued logic*, IEEE Computer Society Press, 1978, pp. 269–275.

[73] R. Berlin, “Synthesis of n-valued switching circuits,” *IRE Transactions on Electronic Computers*, no. 1, pp. 52–56, 1958.

[74] C. Lee and W. Chen, “Several-valued combinational switching circuits,” *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, vol. 75, no. 3, pp. 278–283, 1956.

[75] O Lowenschuss, “Non-binary switching theory, 1958 ire nat’l,” in *Convention Record*.

[76] M. Yoeli and G Rosenfeld, “Logical design of ternary switching circuits,” *IEEE Transactions on Electronic Computers*, no. 1, pp. 19–29, 1965.

[77] N. Serran, A. M. Jorge, and J. S. Dias, “A proposal for the implementation of ternary digital circuits,” *microelectronics Journal*, vol. 28, no. 5, pp. 533–541, 1997.

[78] S. Lin, Y.-B. Kim, and F. Lombardi, “Cntfet-based design of ternary logic gates and arithmetic circuits,” *IEEE transactions on nanotechnology*, vol. 10, no. 2, pp. 217–225, 2011.

[79] H. Mouftah and I. Jordan, "Integrated circuits for ternary logic," in *Proc. 1974 Int. Symp. Multiple-Valued Logic*, 1974, pp. 285–302.

[80] H. Mouftah, "A study on the implementation of three-valued logic," in *Proceedings of the sixth international symposium on Multiple-valued logic*, IEEE Computer Society Press, 1976, pp. 123–126.

[81] A. Heung and H. Mouftah, "Decmos—a low power family of three-valued logic circuits for vlsi implementation," in *Proc. ISMVL-84*, 1984, pp. 120–124.

[82] H. Mouftah, A. Heung, and L. Wong, "Qtc-1 a cmos ternary computer," in *Proc. ISMVL-84*, 1984, pp. 125–132.

[83] H. Mouftah and K. Smith, "Design and implementation of three-valued logic systems with mos integrated circuits," in *IEE Proceedings G-Electronic Circuits and Systems*, IET, vol. 127, 1980, pp. 165–168.

[84] J. Deng and H.-S. P. Wong, "A compact spice model for carbon-nanotube field-effect transistors including nonidealities and its application—part i: Model of the intrinsic channel region," *IEEE Transactions on Electron Devices*, vol. 54, no. 12, pp. 3186–3194, 2007.

[85] R. C. Lee and C.-L. Chang, "Some properties of fuzzy logic," *Information and Control*, vol. 19, no. 5, pp. 417–431, 1971.

[86] A. Kandel and J. M. Francioni, "On the properties and applications of fuzzy-valued switching functions," *IEEE Transactions on Computers*, no. 11, pp. 986–994, 1980.

[87] Y. Hata, K. Nakashima, and K. Yamato, "Some fundamental properties of multiple-valued kleenean functions and determination of their logic formulas," *IEEE transactions on computers*, vol. 42, no. 8, pp. 950–961, 1993.

[88] N. Takagi, K. Nakashima, and M. Mukaidono, "A necessary and sufficient condition for kleenean functions," *IEICE TRANSACTIONS on Information and Systems*, vol. 79, no. 11, pp. 1511–1517, 1996.

[89] N. Takagi, H. Kikuchi, K. Nakashima, and M. Mukadiono, "Identification of incompletely specified multiple-valued kleenean functions," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 28, no. 5, pp. 637–647, 1998.

[90] A. Raychowdhury and K. Roy, "Carbon-nanotube-based voltage-mode multiple-valued logic design," *IEEE Transactions on Nanotechnology*, vol. 4, no. 2, pp. 168–179, 2005.

[91] K. You and K. Nepal, "Design of a ternary static memory cell using carbon nanotube-based transistors," *Micro & Nano Letters*, vol. 6, no. 6, pp. 381–385, 2011.

[92] M. Yoeli and S. Rinon, "Application of ternary algebra to the study of static hazards," *Journal of the ACM (JACM)*, vol. 11, no. 1, pp. 84–97, 1964.

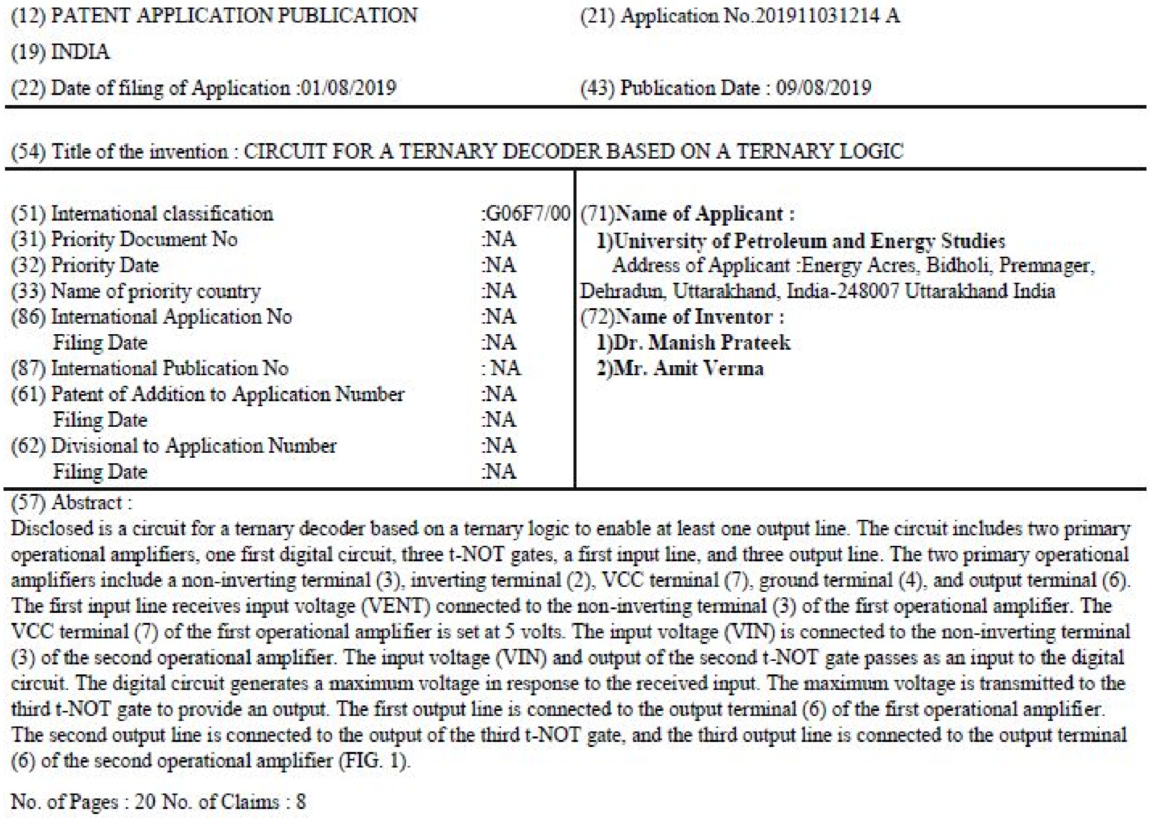
[93] D. A. Huffman, "The design and use of hazard-free switching networks," *Journal of the ACM (JACM)*, vol. 4, no. 1, pp. 47–62, 1957.

[94] E. McCluskey, "Transients in combinational logic circuits," *Redundancy Technique for Computing System*, vol. 9, 1962.

[95] N. Serran, A. M. Jorge, and J. S. Dias, "A proposal for the implementation of ternary digital. circuits,"

PUBLICATIONS

The circuits for ternary decoder is published in The Patent Office Journal No. 32/2019 Dated 09/08/2019 as shown in the Fig. 6.1, ternary NAND and NOR gate in The Patent Office Journal No. 36/2019 Dated 06/09/2019 as shown in the Fig. 6.2.



(12) PATENT APPLICATION PUBLICATION (21) Application No.201911031214 A

(19) INDIA

(22) Date of filing of Application :01/08/2019 (43) Publication Date : 09/08/2019

(54) Title of the invention : CIRCUIT FOR A TERNARY DECODER BASED ON A TERNARY LOGIC

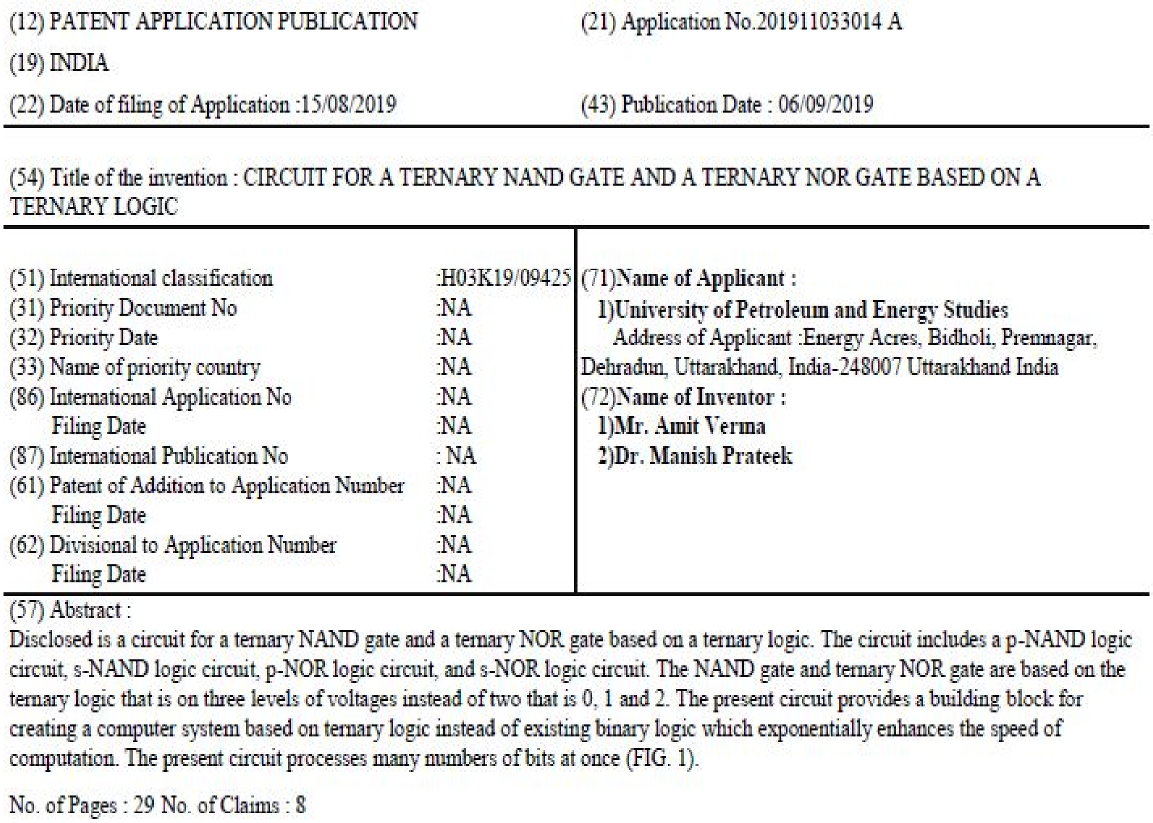
(51) International classification	:G06F7/00	(71)**Name of Applicant :**
(31) Priority Document No	:NA	**1)University of Petroleum and Energy Studies**
(32) Priority Date	:NA	Address of Applicant :Energy Acres, Bidholi, Premnager,
(33) Name of priority country	:NA	Dehradun, Uttarakhand, India-248007 Uttarakhand India
(86) International Application No	:NA	(72)**Name of Inventor :**
Filing Date	:NA	**1)Dr. Manish Prateek**
(87) International Publication No	: NA	**2)Mr. Amit Verma**
(61) Patent of Addition to Application Number	:NA	
Filing Date	:NA	
(62) Divisional to Application Number	:NA	
Filing Date	:NA	

(57) Abstract :

Disclosed is a circuit for a ternary decoder based on a ternary logic to enable at least one output line. The circuit includes two primary operational amplifiers, one first digital circuit, three t-NOT gates, a first input line, and three output line. The two primary operational amplifiers include a non-inverting terminal (3), inverting terminal (2), VCC terminal (7), ground terminal (4), and output terminal (6). The first input line receives input voltage (VENT) connected to the non-inverting terminal (3) of the first operational amplifier. The VCC terminal (7) of the first operational amplifier is set at 5 volts. The input voltage (VIN) is connected to the non-inverting terminal (3) of the second operational amplifier. The input voltage (VIN) and output of the second t-NOT gate passes as an input to the digital circuit. The digital circuit generates a maximum voltage in response to the received input. The maximum voltage is transmitted to the third t-NOT gate to provide an output. The first output line is connected to the output terminal (6) of the first operational amplifier. The second output line is connected to the output of the third t-NOT gate, and the third output line is connected to the output terminal (6) of the second operational amplifier (FIG. 1).

No. of Pages : 20 No. of Claims : 8

Figure 6.1: Circuit for a Ternary Decoder



(12) PATENT APPLICATION PUBLICATION

(21) Application No.201911033014 A

(19) INDIA

(22) Date of filing of Application :15/08/2019

(43) Publication Date : 06/09/2019

(54) Title of the invention : CIRCUIT FOR A TERNARY NAND GATE AND A TERNARY NOR GATE BASED ON A TERNARY LOGIC

(51) International classification	:H03K19/09425
(31) Priority Document No	:NA
(32) Priority Date	:NA
(33) Name of priority country	:NA
(86) International Application No	:NA
Filing Date	:NA
(87) International Publication No	: NA
(61) Patent of Addition to Application Number	:NA
Filing Date	:NA
(62) Divisional to Application Number	:NA
Filing Date	:NA

(71)Name of Applicant :
1)University of Petroleum and Energy Studies
Address of Applicant :Energy Acres, Bidholi, Premnagar, Dehradun, Uttarakhand, India-248007 Uttarakhand India
(72)Name of Inventor :
1)Mr. Amit Verma
2)Dr. Manish Prateek

(57) Abstract :
Disclosed is a circuit for a ternary NAND gate and a ternary NOR gate based on a ternary logic. The circuit includes a p-NAND logic circuit, s-NAND logic circuit, p-NOR logic circuit, and s-NOR logic circuit. The NAND gate and ternary NOR gate are based on the ternary logic that is on three levels of voltages instead of two that is 0, 1 and 2. The present circuit provides a building block for creating a computer system based on ternary logic instead of existing binary logic which exponentially enhances the speed of computation. The present circuit processes many numbers of bits at once (FIG. 1).

No. of Pages : 29 No. of Claims : 8

Figure 6.2: Circuit for a Ternary NAND and NOR gate

The work is published in following journals under SCOPUS indexed.

1. *T-NOT Gate : A Novel Circuit based on Ternary Logic* published in *International Journal of Innovative Technology and Exploring Engineering Volume-8 Issue-4, February 2019.*

2. *Am-Multiplication: A Novel Multiplication Algorithm Based Binary Multiplexer* published in *International Journal of Recent Technology and Engineering* Volume-7, Issue-6, March 2019

3. *Required 3rd State: A Novel Ternary Addition Algorithm* published in *Jour of Adv Research in Dynamical & Control Systems* Vol. 10, 13-Special Issue, 2018

Thesis

ORIGINALITY REPORT

1%	2%	0%	0%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.scribd.com Internet Source	1%

Exclude quotes	Off	Exclude matches	< 1%
Exclude bibliography	Off		