Bangladesh University of Engineering and Technology

Course No.: EEE454

Course Name: EEE 454 VLSI I Laboratory

Project Name: Design of ALU with Shifter

Submitted to:

Dr. A.B.M. Harun-ur Rashid
Professor,
Dept. of EEE, BUET.
Gobinda Saha
Lecturer,
Dept. of EEE, BUET.

Submitted By:

Shuvangkar Chandra Das

Student ID: 1106125

Niloy Saha

Student ID: 1106146

Table of Contents

List of Illustrations

Abstract:

The objective of this project is to design 4-bit ALU using CMOS gate ,Reduce the number of transistors ,Reduce the number of total IC / gate, Implement the 1-bit full adder circuit layout, Implement the shifter circuit layout, Minimize the total area of layout design .

Introduction:

An **arithmetic logic unit** (ALU) is a digital electronic circuit that performs arithmetic and bitwise logical operations on integer binary numbers[1]

ALU is getting smaller and more complex nowadays to enable the development of a more powerful but smaller computer. However there are a few limiting factors that slow down the development of smaller and more complex IC chip and they are IC fabrication technology, designer productivity, and design cost. The increasing demand for high-speed very large scale integration (VLSI) can be addressed at different design levels, such as the architectural, circuit, layout, and the process technology level. At the circuit design level, considerable potential for speed enhancement exists by means of proper choice of a logic style for implementing combinational circuits. This is because all the important parameters governing speed switching capacitance, transition activity, and short-circuit currents are strongly influenced by the chosen logic style. Depending on the application, the kind of circuit to be implemented, and the design technique used, different performance aspects become important. In the past, the parameters like low power dissipation, small area, and low cost were the major areas of concern, whereas speed considerations are now gaining the attention of the scientific community associated with VLSI design. Arithmetic Logical Unit is the very important subsystem in the digital system design. An Arithmetic logic Unit (ALU) is an integral part of a computer processor. It is a combinational logic unit that performs its arithmetic and logic operations. ALUs of various bit-widths are frequently required in very large-scale integrated circuits (VLSI) from processors to application specific integrated circuits (ASICs). ALU is getting smaller and more complex nowadays to enable the development of a more powerful but smaller computer. The demand for low power & high-speed processing has been increasing as a result of expanding computer and signal processing applications. Higher throughput arithmetic operations are important to achieve the desired performance in many real-time signals and image processing applications. The key arithmetic operations in such applications are multiplication, addition, division, and subtraction.

Major Parts of ALU

Arithmetic block:

This block is used to perform arithmetic operations such as addition, subtraction, and comparison. The core of the arithmetic block is an adder. In the architecture presented in Figure 4.1, the adder uses carry look-ahead and sum-select techniques.

• Logic block:

This block is used to perform simple bitwise logic operations such as AND (masking), OR and XOR, XNOR, NAND, NOT and etc. •

[1] https://en.wikipedia.org/

Multiplexers:

These blocks are used to select the appropriate inputs for the arithmetic and logic blocks. Usually, more than two buses arrive at the inputs of the ALU. Sometimes these multiplexers are used to perform some simple logic operations. The 2:1 MUX can be programmed to invert one of the operands (this can be used to execute a subtraction using just an added)

Problem Statement of our project:

This is the block diagram of the project where we have to design a 4bit Arithmetic logic Unit(ALU) and a Shifter unit.

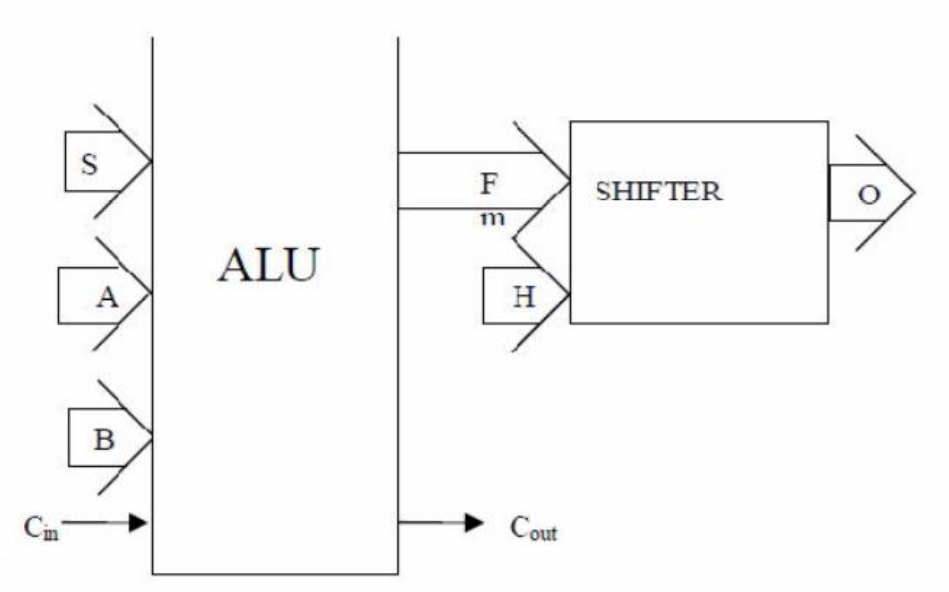


Figure 1: ALU with Shifter block Diagram

Our group no: 09. Our assigned work is listed below

Table : Functions of ALU

S3 S2 S1 S0	F	C_{out}	Description
0000	A	0	Transfer A
0001	A+B+1	1 IF (A+B)>2^4	Add A to B with Carry
0010	A-B-1	1 IF $A > B$	Subtract B from A with Borrow
0011	A+1	1 IF A>$2^4 - 1$	Increment A
0100	A	0	Transfer A
0101	A+B	1 IF (A+B)>$2^4 - 1$	Add A to B
0110	A-B	1 IF $A \geq B$	Subtract B from A
0111	A-1	1 IF A is not equal 0	Decrement of A
1000	A’	0	Complement of A
1010	A OR B	0	Bitwise OR

1100	A AND B	0	Bitwise AND
1110	A XOR B	0	Bitwise XOR

Table : Functions of Shifter

H_1	H_2	Operation	Function
0	0	O=F	Transfer (No Shift)
0	1	O=SHR(F)	Shift 1-bit Right
1	0	O=SHL(F)	Shift 1-bit Left
1	1	O=0	Transfer 0

Software :

We used Cadence Virtuoso to implement the project of 4bit ALU with Shifter

Step by step Implementation:

In this section, we will try to explain how we implemented the 4 bit ALU with Shifter. In the first section, we designed the basic gates and combinational circuit and then we used the basic gates and combinational circuit to implement the ALU & Shifter

Basic Gates & Combinational circuit

Inverter:

Here we designed a CMOS Inverter. The truth table for the Inverter is given below

Input	Output
A	Y
0	1
1	0

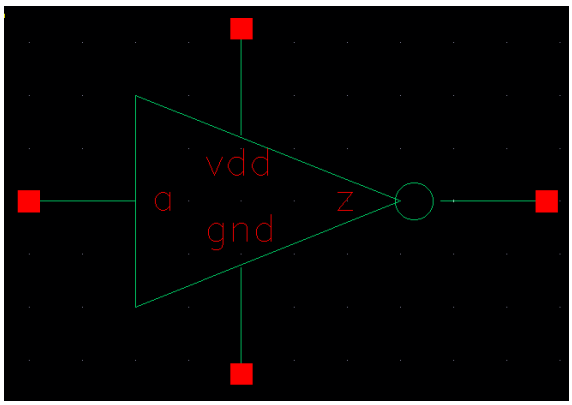


Figure 2: Truth table & Symbol of NOT gate

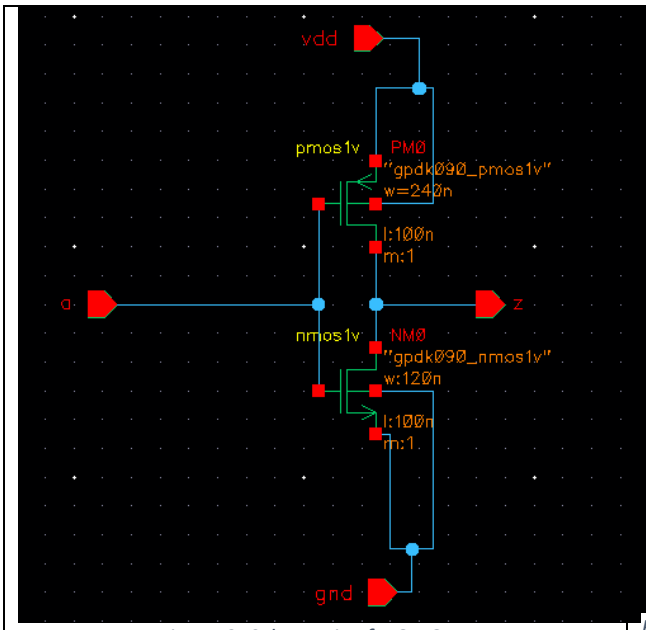


Figure 3: Schematic of NOT Gate

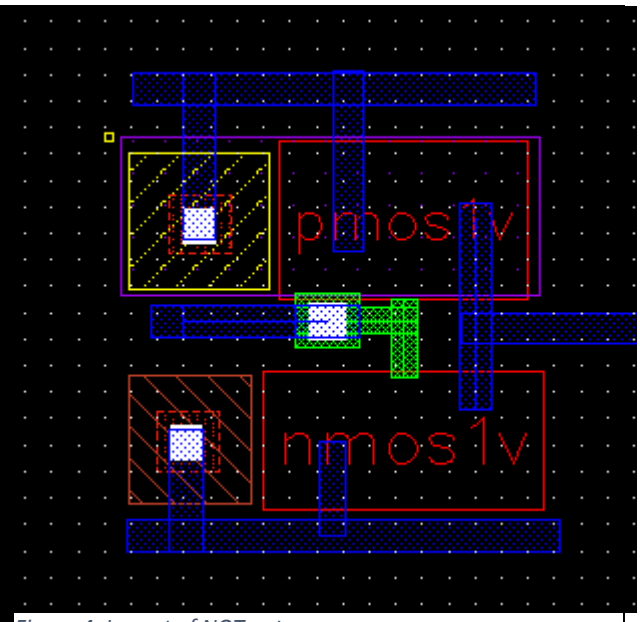


Figure 4: Layout of NOT gate

The following image is the layout of the Inverter.

NOR Gate:

Then we have designed a CMOS NOR gate.

Truth table for the NOR gate:

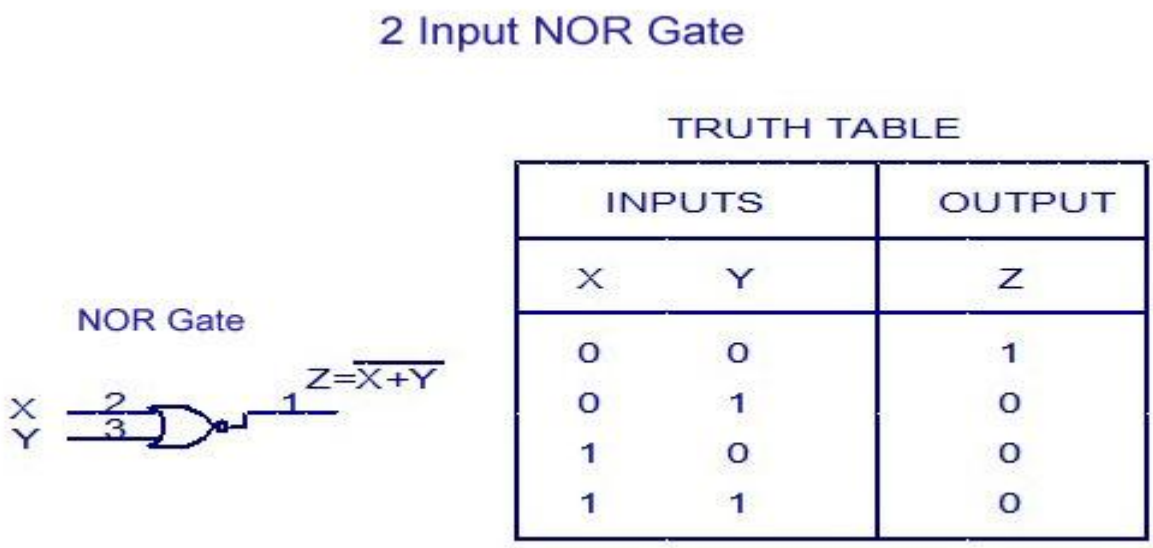


Figure 5: Symbol & Truth table for NOR gate

The schematic and symbol for the CMOS NOR Gate.

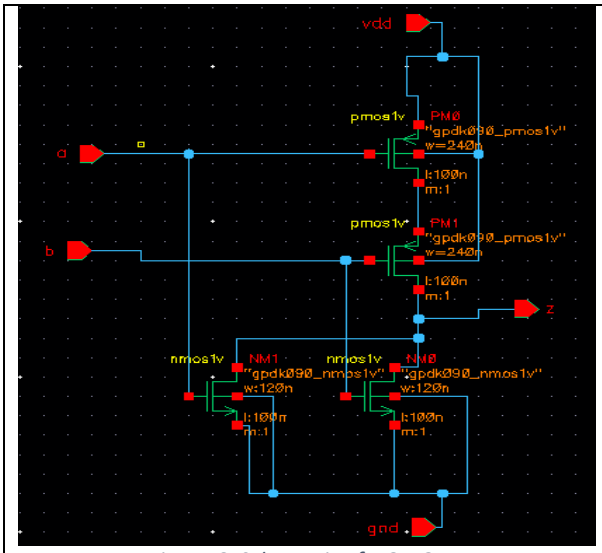


Figure 6: Schematic of NOR Gate

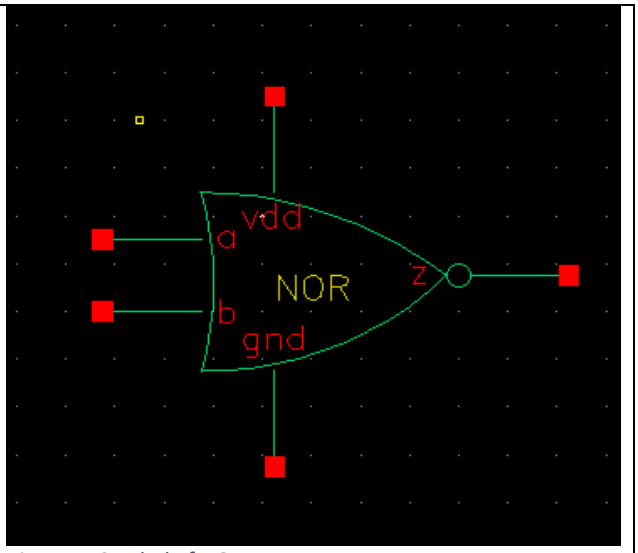
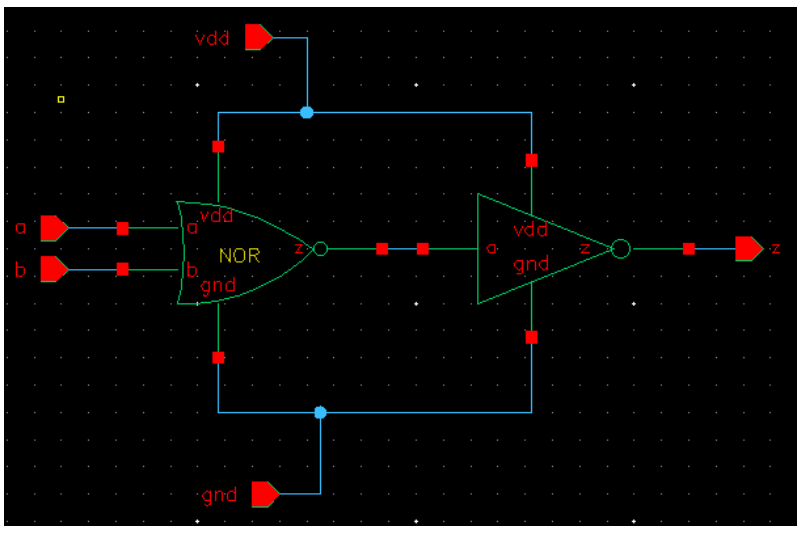


Figure 7: Symbol of NOR gate

Features	Measurements/ quantity
No of MOSFET	4
Size of Layout	1.5x1.8um^2

OR gate:

Using NOR gate & NOT gate we have designed OR gate



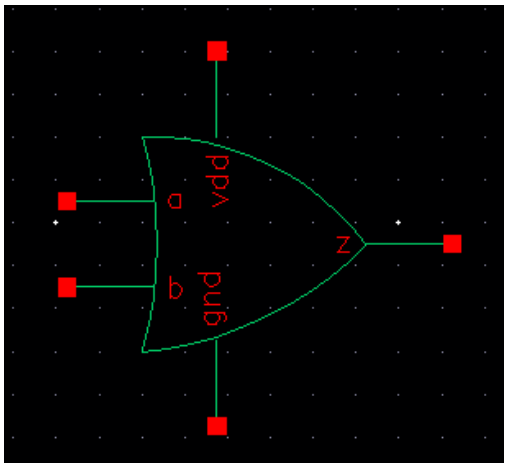


Figure 8: Schematic & Symbol of OR gate

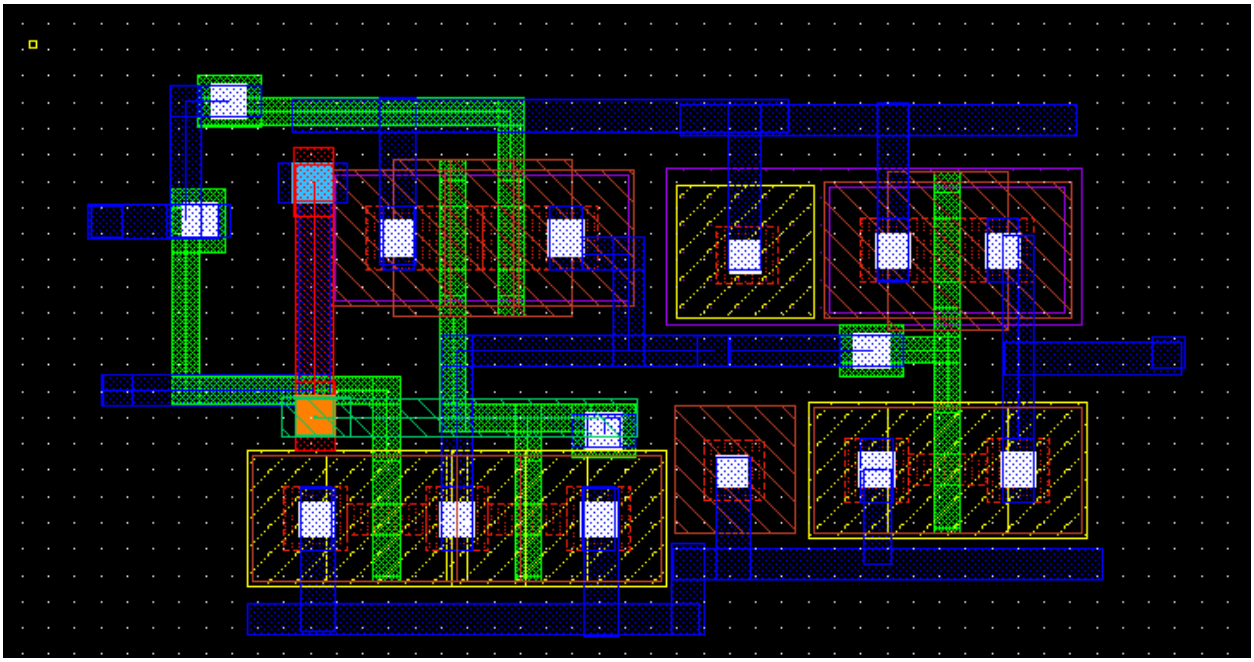


Figure 9: Layout of OR gate

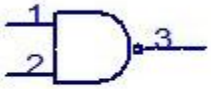
Features	Measurements/ quantity
No of MOSFET	6
Size of layout	3.5x2.07um^2

NAND:

In the below picture is the CMOS implementation of the NAND Gate. Truth table for the NAND gate is given below

2 Input NAND Gate

NAND Gate



TRUTH TABLE

INPUTS		OUTPUT
X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

Figure 10: Truth table for NAND Gate

In the NAND Gate implementation here we see only four MOSFET was used.

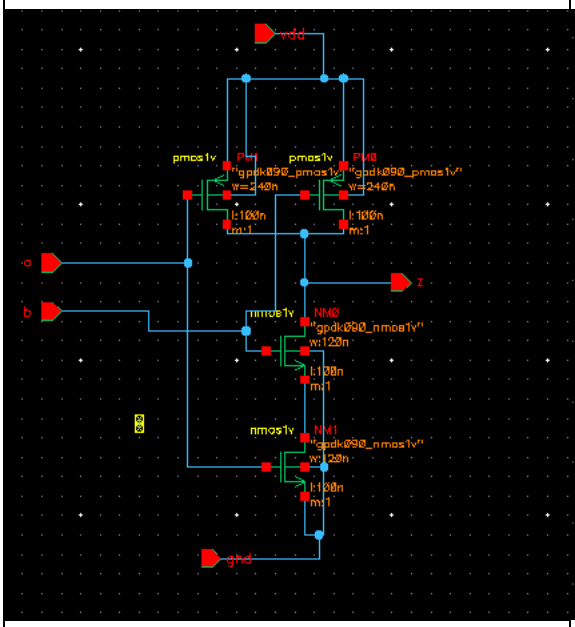


Figure 11: Schematic of NAND Gate

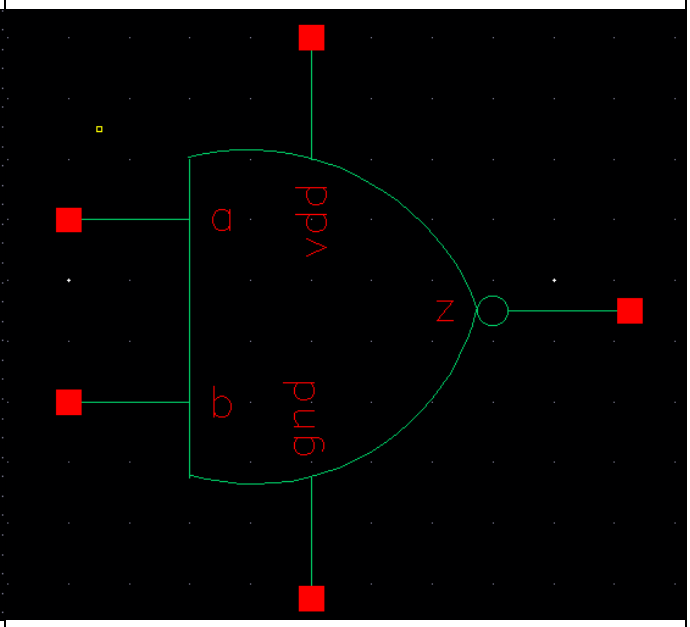


Figure 12: Symbol of NAND Gate

AND:

Using One NAND gate and One NOT gate an AND gate can be found. So here 6 MOSFETs were required to design this

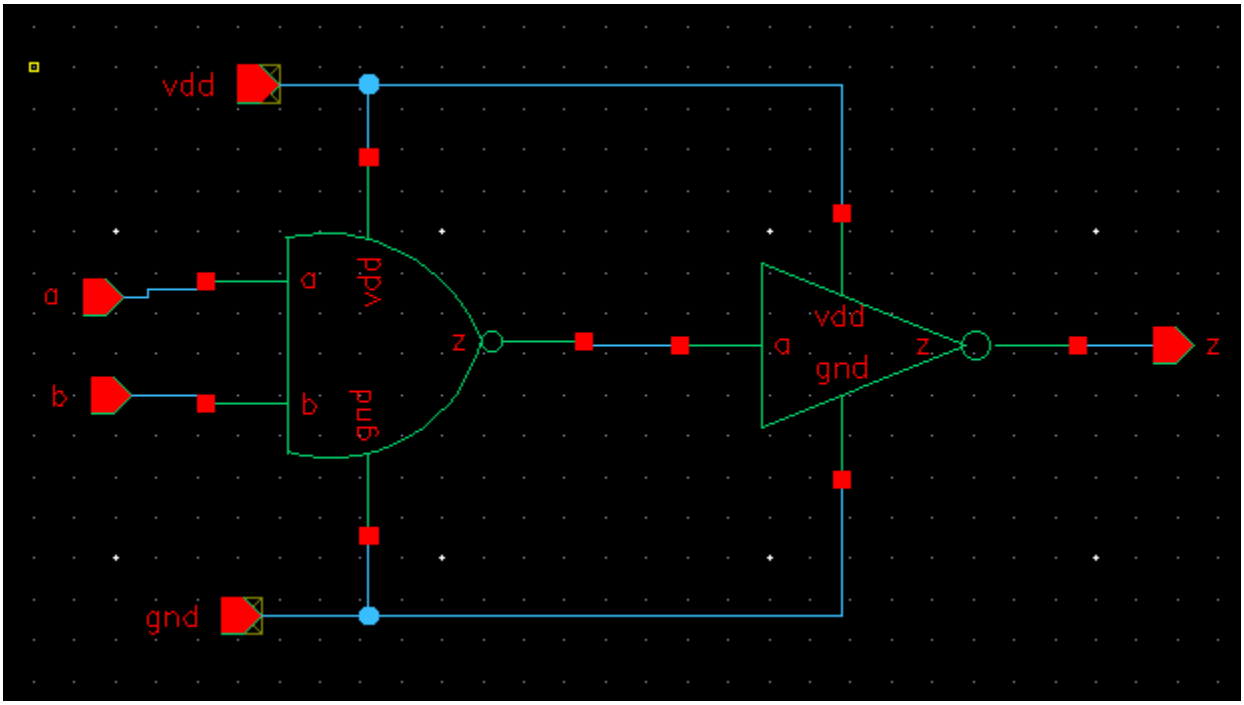


Figure 13: Schematic of AND Gate

Description	Size/quantity
No of MOSFET	6
Size of layout	2.09x2.3um^2

In the layout of the AND gate, we have tried to minimize the size. So here is the layout design of the AND Gate

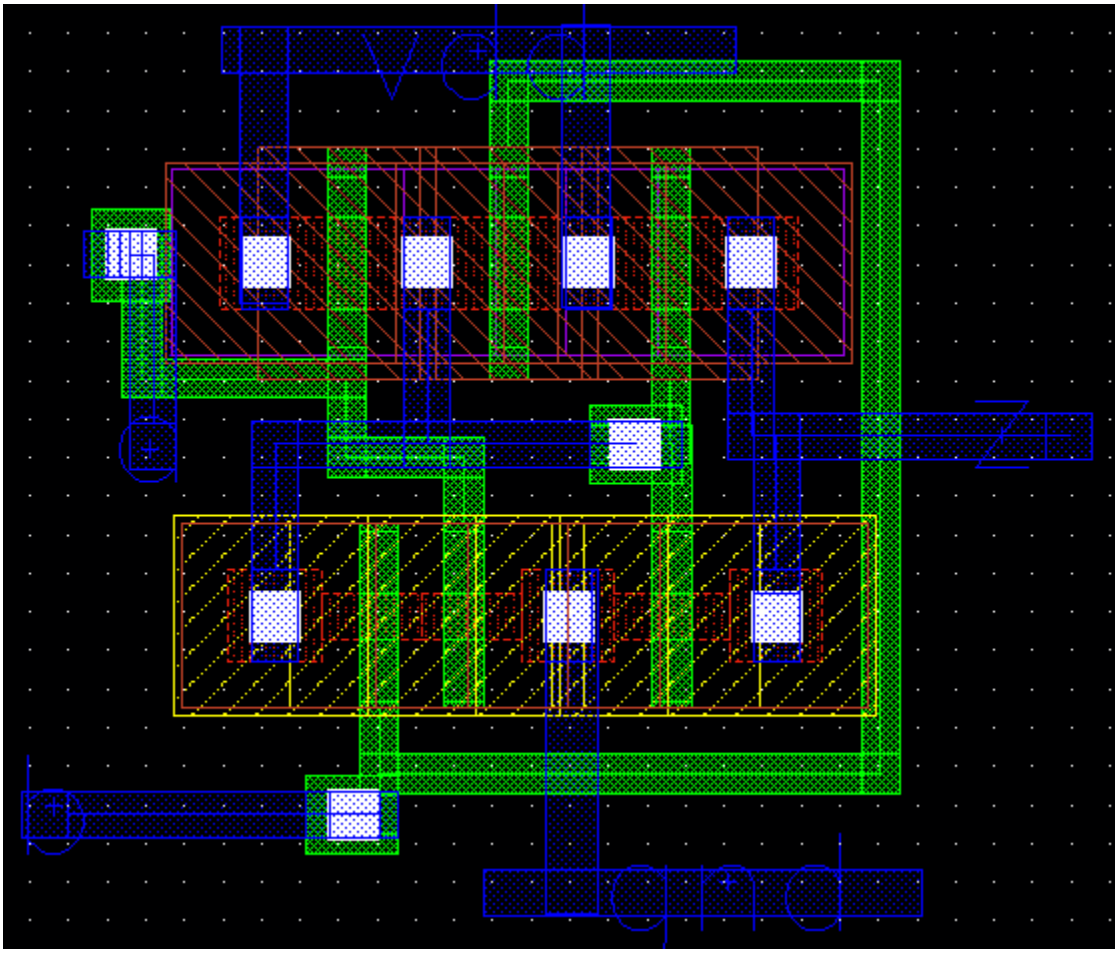


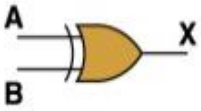
Figure 14: Layout of AND Gate

XOR Gate:

Boolean Expression

$X = A \oplus B$

Logic Diagram Symbol



Truth Table

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Figure 15: Truth Table for XOR Gate

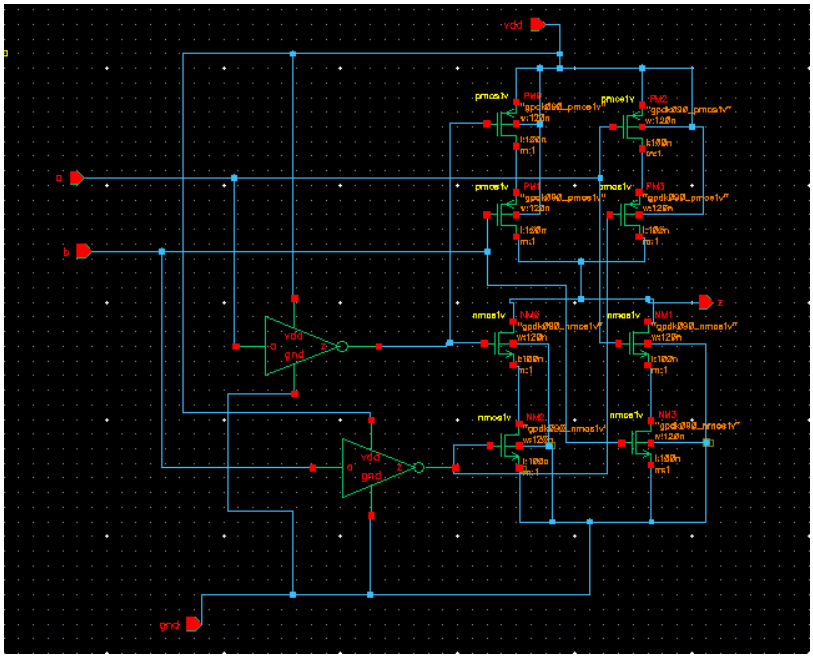


Figure 16: Schematic of XOR Gate

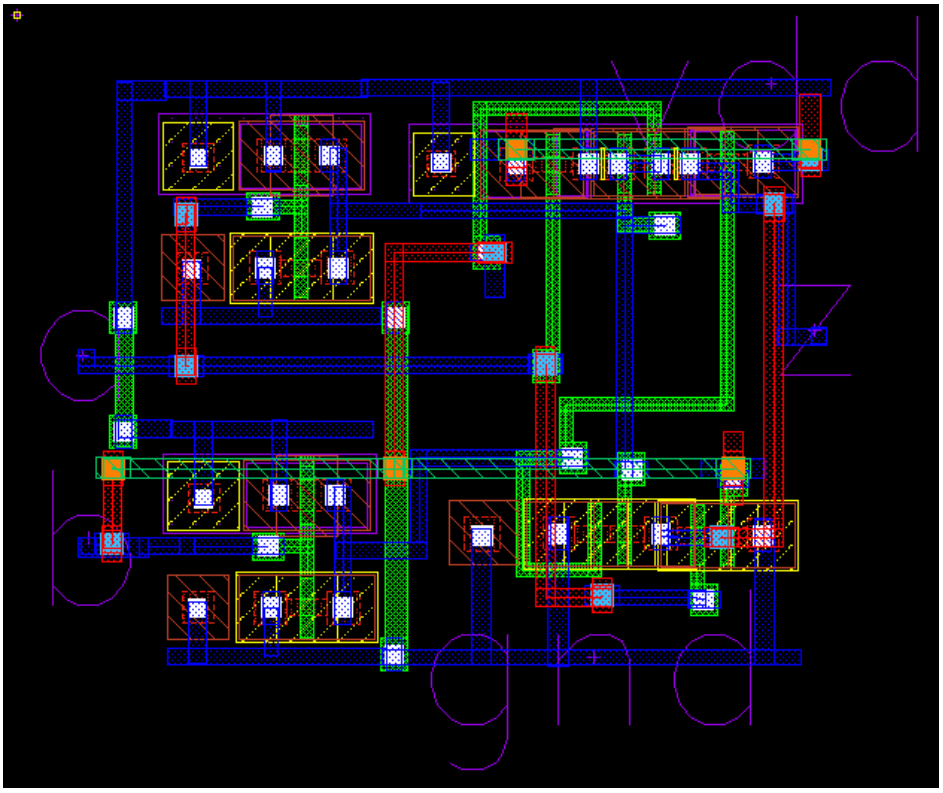


Figure 17: Layout of XOR Gate

Description	Size/quantity
No of MOSFET	12
Size of layout	5.3x4.3 (umxum)

2 to 1 MUX

This is the CMOS Pass gate transistor implementation of 2 to 1 Multiplexer. Here Number of MOSFETs used was 6.

Description	Size/quantity
No of MOSFET	6
Size of layout	5.3x4.3 um^2

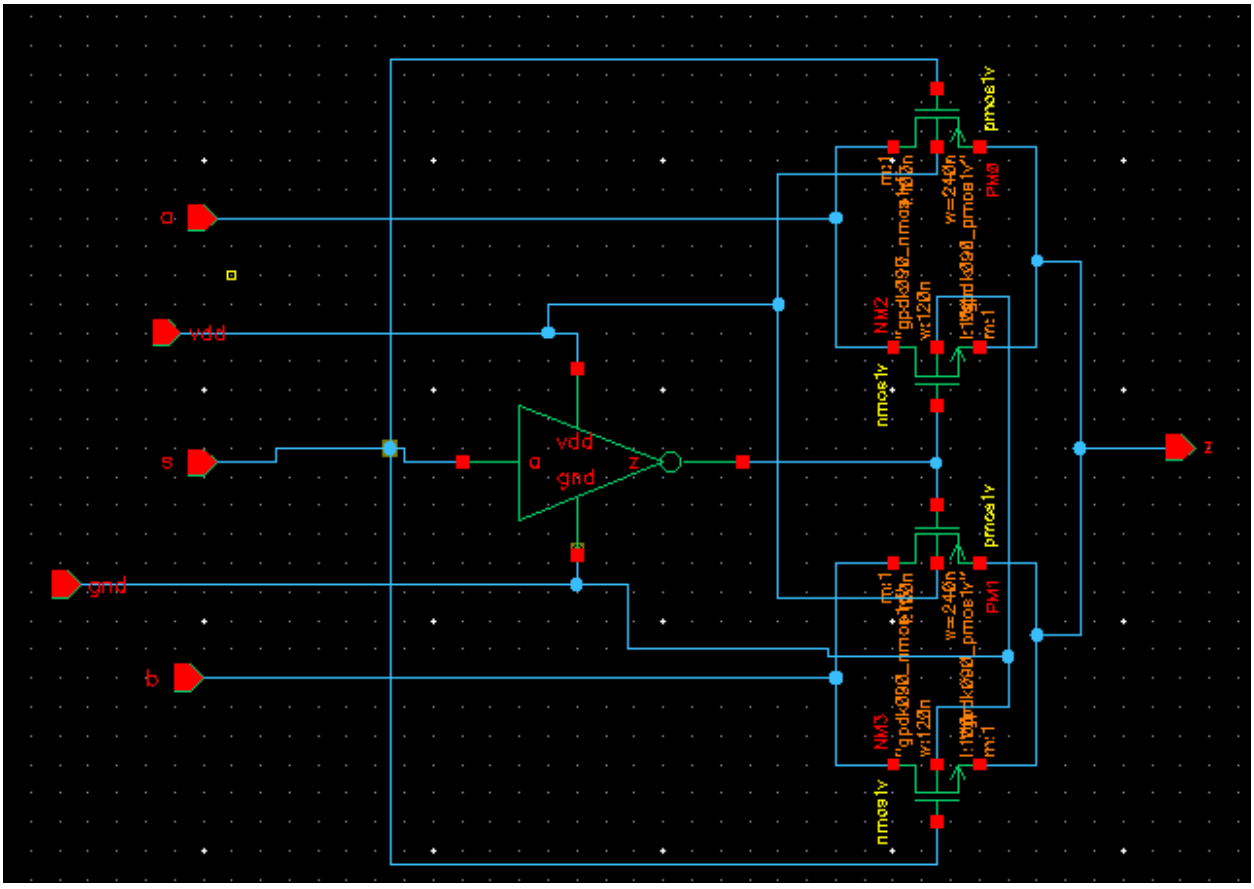


Figure 18: Schematic of 2 to 1 MUX

4 to 1 MUX

Using 3 2-to-1 MUX we designed 4-to-1 Multiplexer.

Description	Size/quantity
No of MOSFET	18

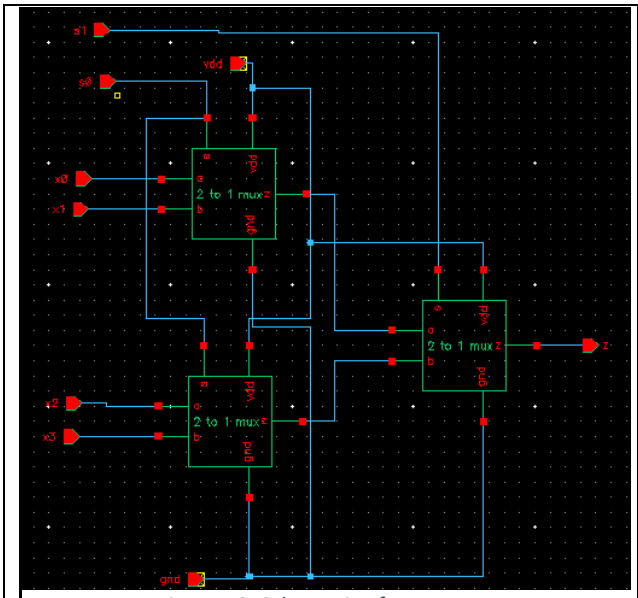


Figure 19: Schematic of 4 to 1 MUX

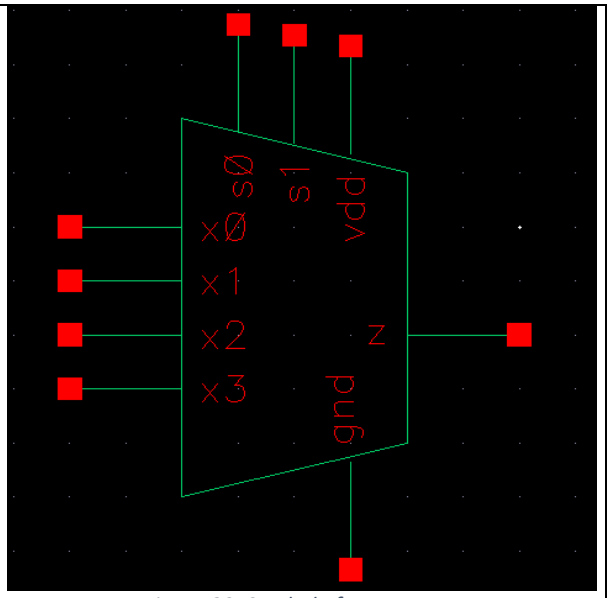


Figure 20: Symbol of 4 to 1 MUX

Quad 2 to 1 MUX:

As this is a 4 bit ALU. So we need to pass 4 bits at a time. For this reason we must have Quad 2-to-1 MUX to pass 4 bits at a time. Here is the implementation of quad 2-to-1 MUX .

Number of MOSFETs required: 24.

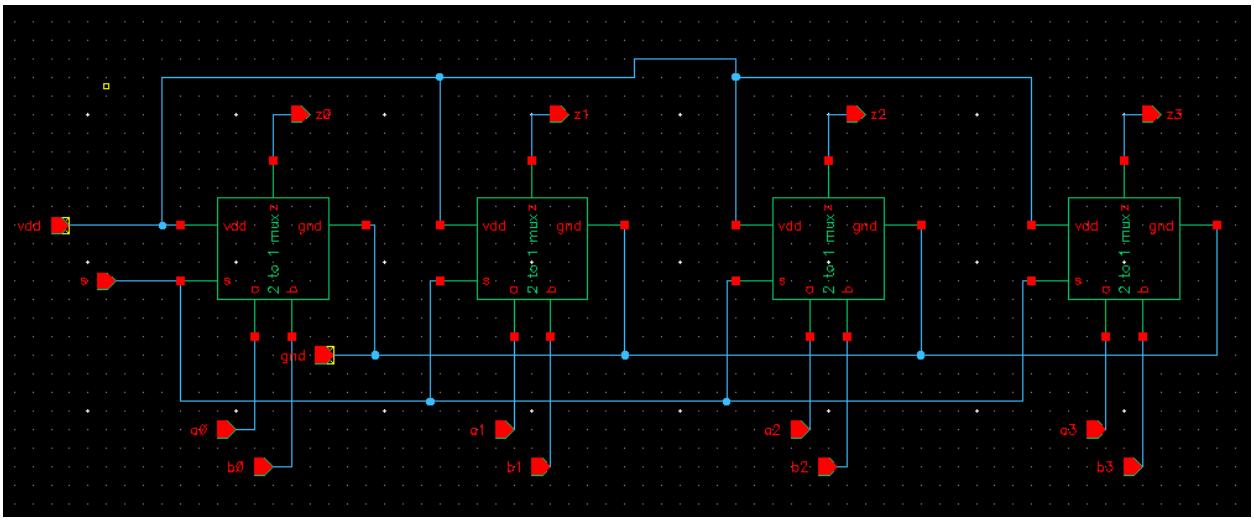


Figure 21: Shcematic of Quad 2 to 1 MUX

Description	Size/quantity
No of MOSFET	24

Full adder

For arithmetic operation, we need a full adder. Which can take carry input and produce carry output. So the truth table for Full adder is

Input			Output	
A	B	Cin	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

In the implementation of the Full adder, we needed two XOR gate, two and Gates, one Or gates.

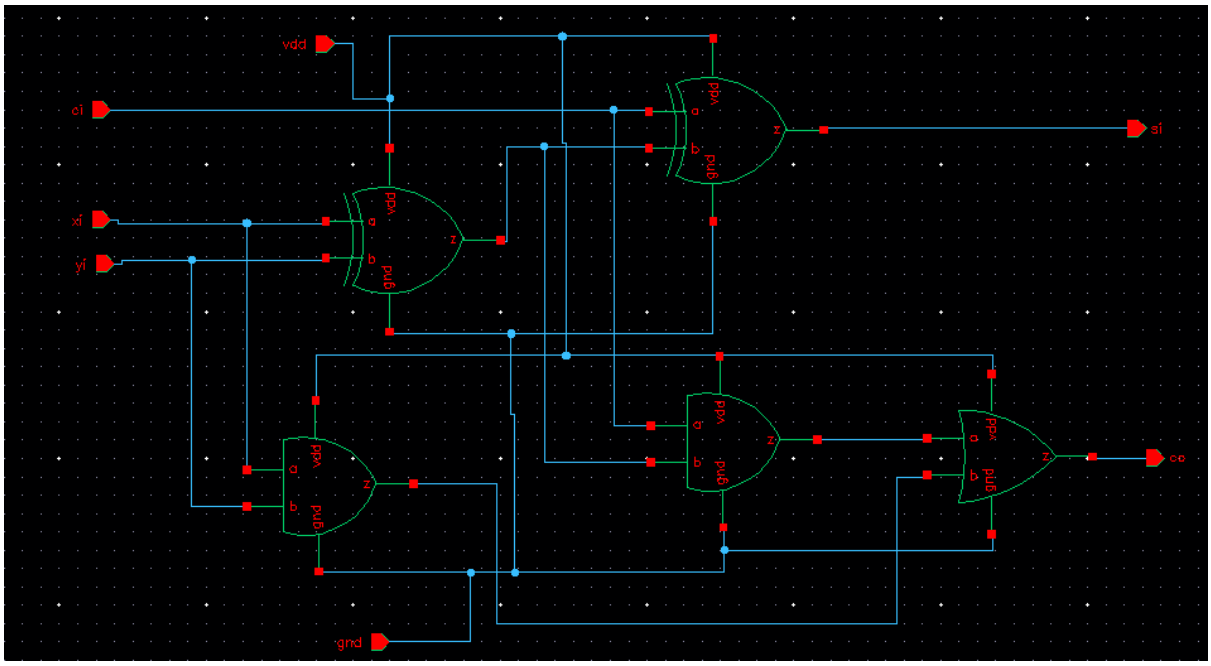


Figure 22: Schematic of Full Adder

Description	Size/quantity
No of MOSFET	42
Size of layout	13.7x9 um^2

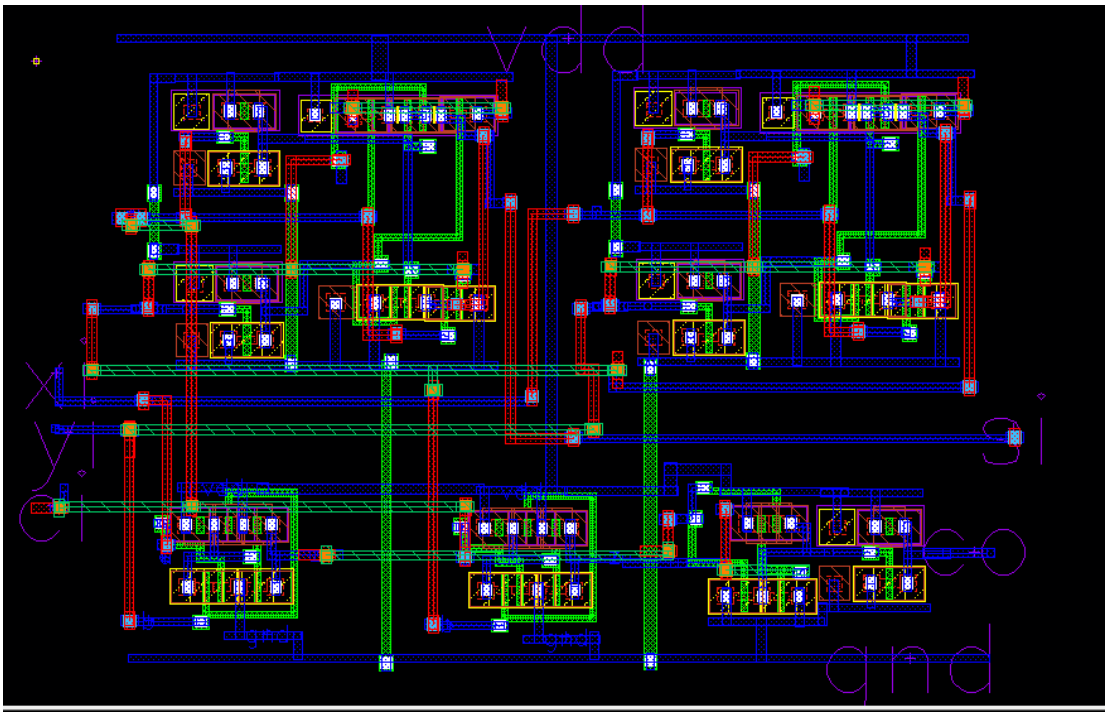


Figure 23: Layout of Full Adder

4 bit Ripple Carry adder:

For 4 bit operation, we need 4-bit full adder. In this ripple carry adder carry out of every step is passed to the next stage. This can also perform subtract operation.

Description	Size/quantity
No of MOSFET	216
Size of layout	

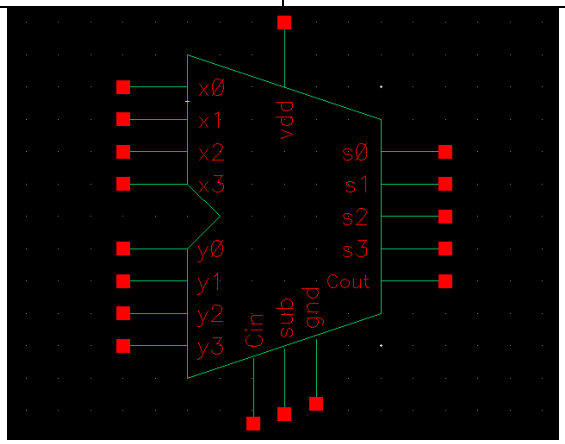


Figure 24: Symbol of 4 bit ripple carry adder

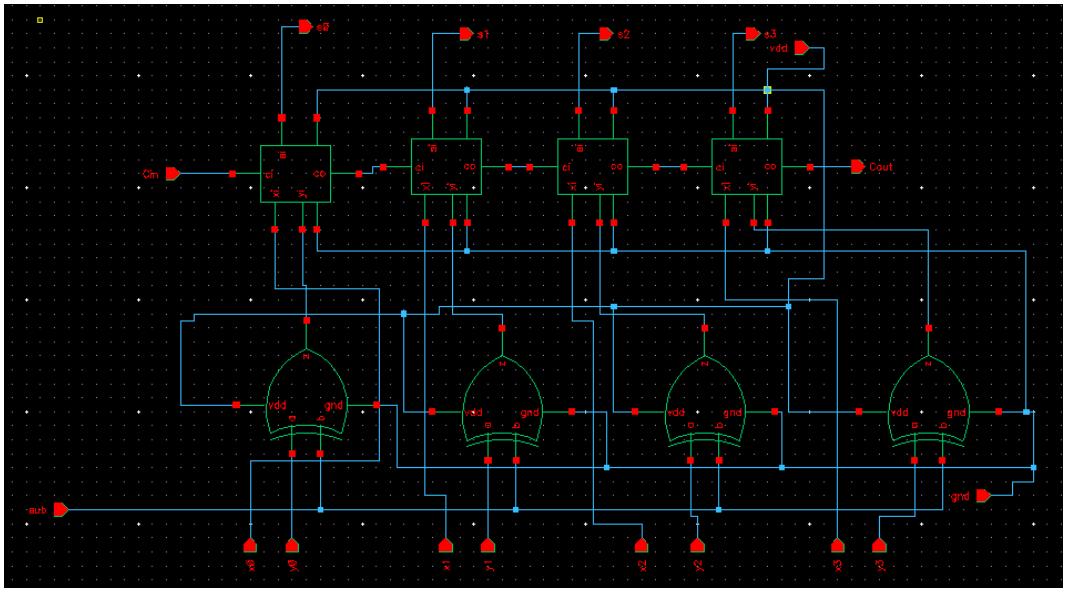


Figure 25: Schematic of 4-bit ripple carry adder

ALU design:

For the design the convince to implement the design I subdivided into the system into few parts;

1. Adder/Subtract function design
2. Transfer function design
3. Increment/Decrement Function
4. Carry input function design

Adder/Subtract function design

4-bit ripple carry can perform addition and subtraction function. But for this, I have to Design the decoder when to add and when to subtract. There are two pins in the 4bit Ripple carry adders such as **Cin** and **Sub.** Here is the work function of this pin.

Cin	Sub	Function
0	0	Addition
0	1	Subtraction with Borrow
1	0	Addition with Carry Input
1	1	Just Subtraction

So before designing Karunaugh map sort out the addition and subtraction op-code

OP Code	Function	Addition/ Subtraction	Value of sub pin
0001	A+B+1	Addition	sub=0
0011	A+1		
0101	A+B		
0010	A-B-1	Subtraction	sub=1
0110	A-B		
0111	A-1		

So karnaugh Map for the Sub Function

S2|S1S0	00	01	11	10
0	d	0	0	1
1	0	0	1	1

$$sub = S_0\bar{S_1} + S_1S_2$$

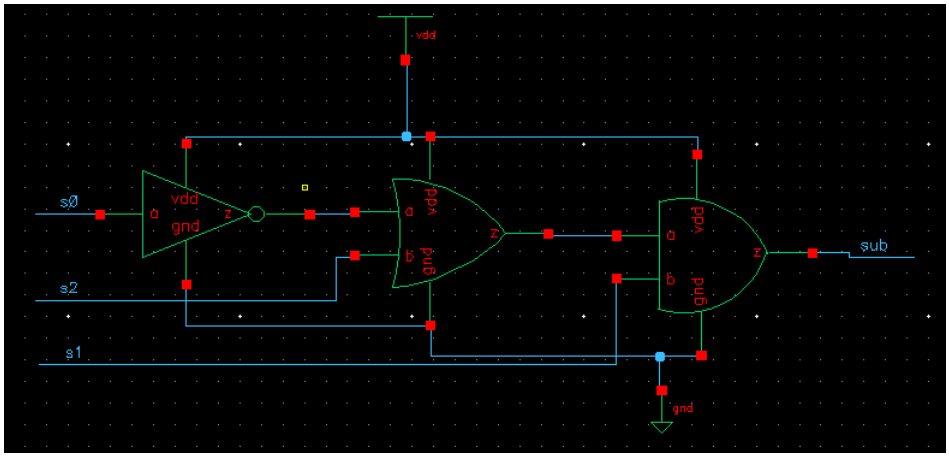


Figure 26: Adder-Subtractor decoder

Increment/Decrement function Selection:

We implement this using quad 2 to 1 MUX. So for this Selection **pin, S** have to design.

OP Code	Function	B/1 Select
0001	A+B+1	MUX will select B
0010	A-B-1	
0101	A+B	
00110	A-B	
0011	A+1	MUX will select 1
0111	A-1	

So Karnaugh map for the Selection pin **s**

S2|S1S0	00	01	11	10
0	0	0	1	0
1	0	0	1	0

MUX selection S= $S_1 S_0$

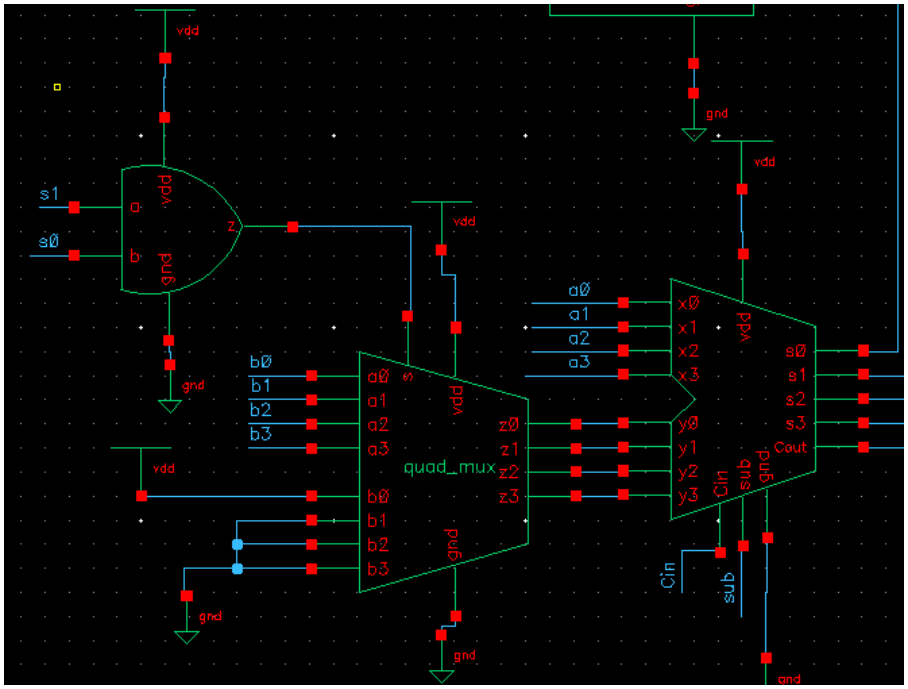


Figure 27: Increment/ decrement decoder

Carry In Function:

OP Code	Function	Cin Value
0001	A+B+1	For these Function Cin value 1 .
0110	A-B	
0111	A-1	
Rest of the functions		Cin=0

So Karnaugh Map for the Cin selection pin

S2|S1S0	00	01	11	10

0	1	0	0	0
1	0	0	1	1

$$Cin = S_1S_2 + S_0\overline{S_1S_2}$$

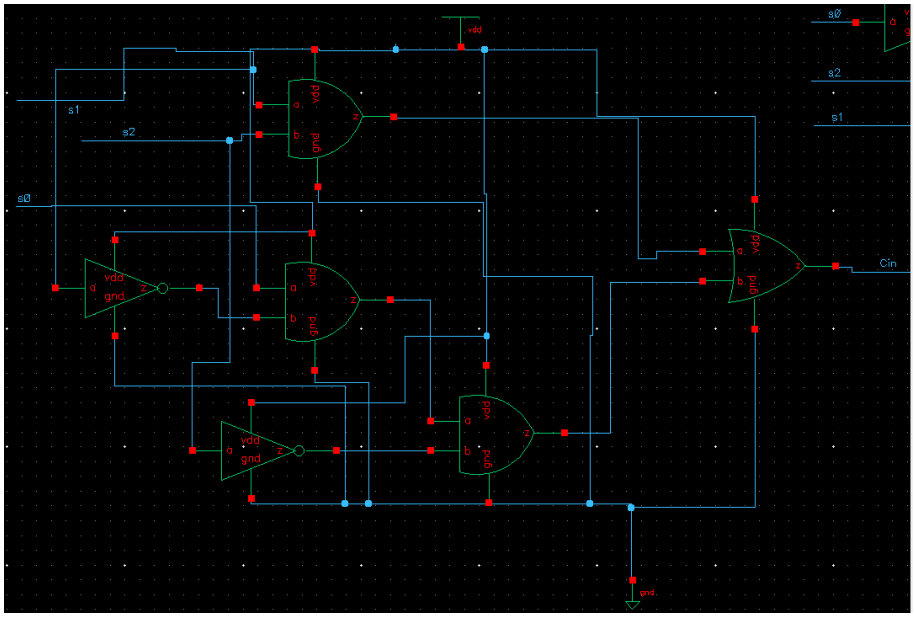


Figure 28: Carry in function decoder

Logic Function design

There are four logic functions in our problem statement.

OP Code	Functions
1000	A’
1010	A OR B
1100	A AND B
1110	A XOR

So to implement logic functions I need to design quad logic gates So that it can perform 4-bit operation at a time.

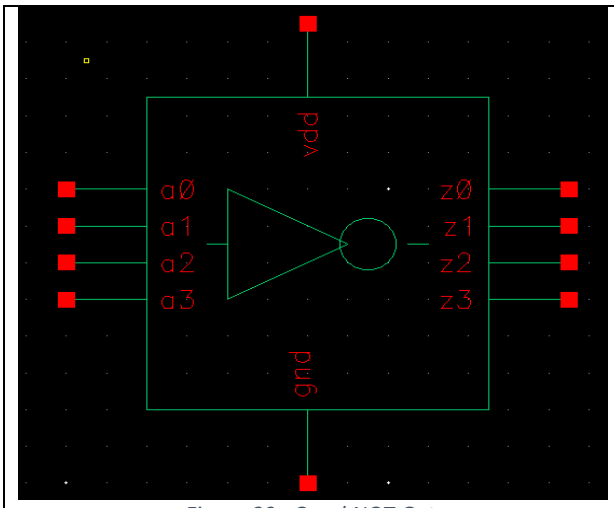


Figure 29: Quad NOT Gate

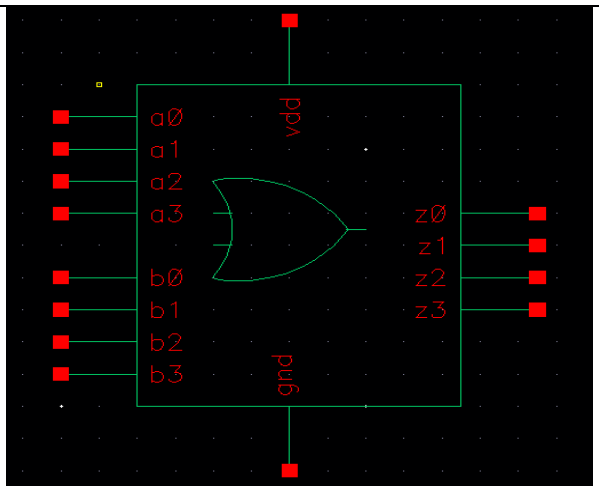


Figure 30: Quad OR Gate

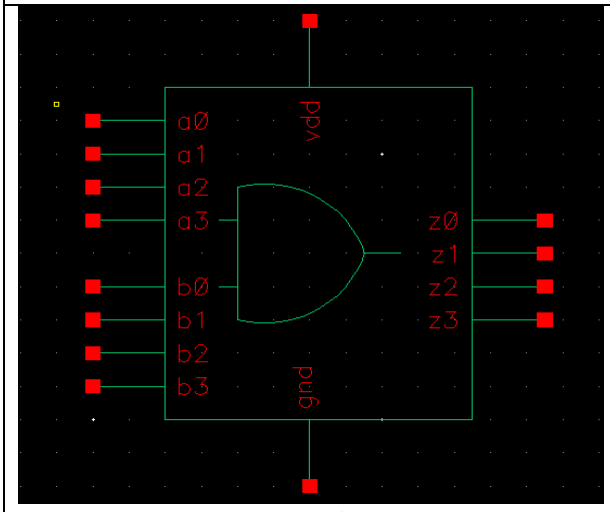


Figure 31: Quad AND Gate

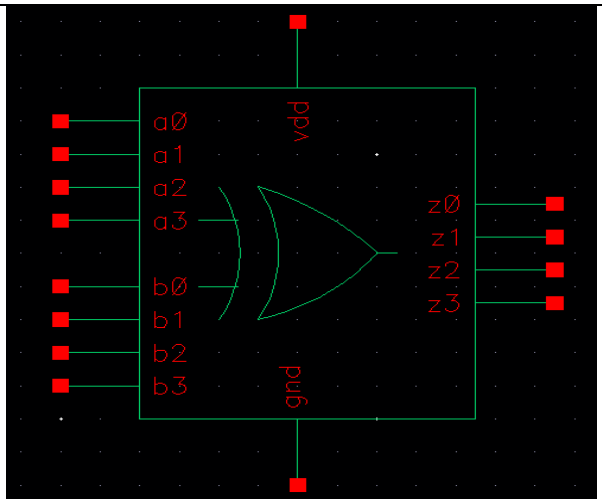


Figure 32: Quad XOR Gate

Then I have designed a 16 to 4 Multiplexer using quad 2 to 1 MUX .

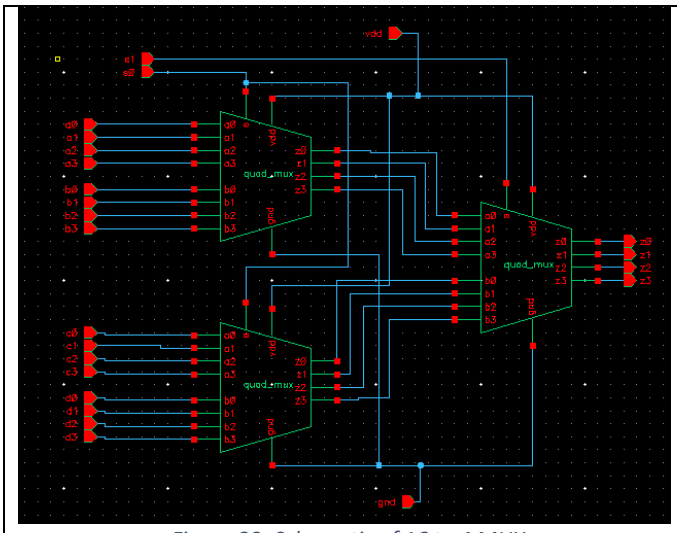


Figure 33: Schematic of 16 to 4 MUX

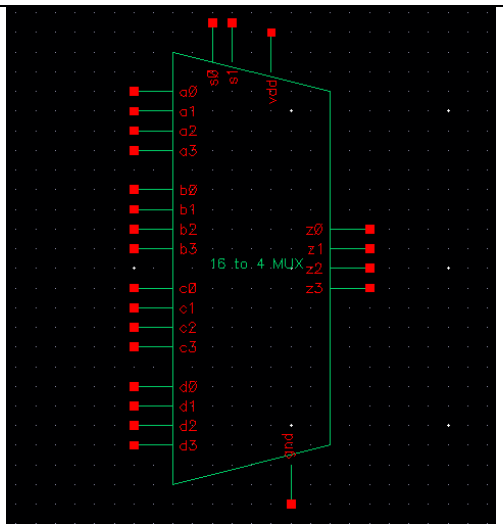


Figure 34: Symbol of 16 to 4 MUX

Then the final logic function will be

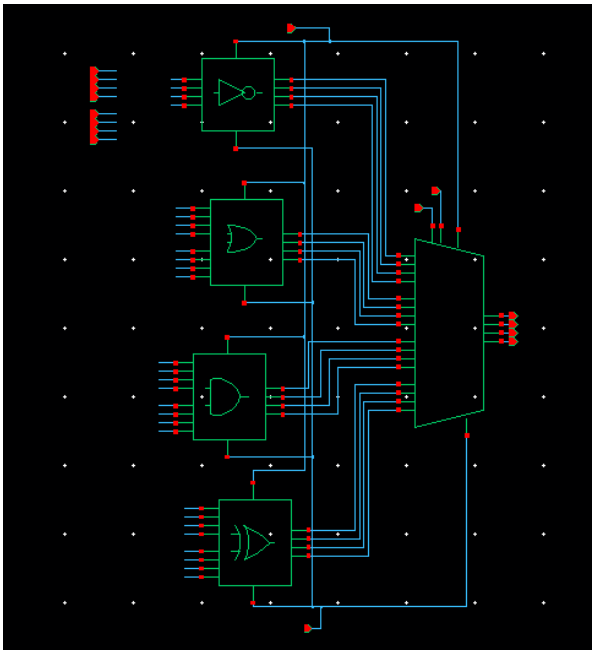


Figure 35: Schematic of Logic functions

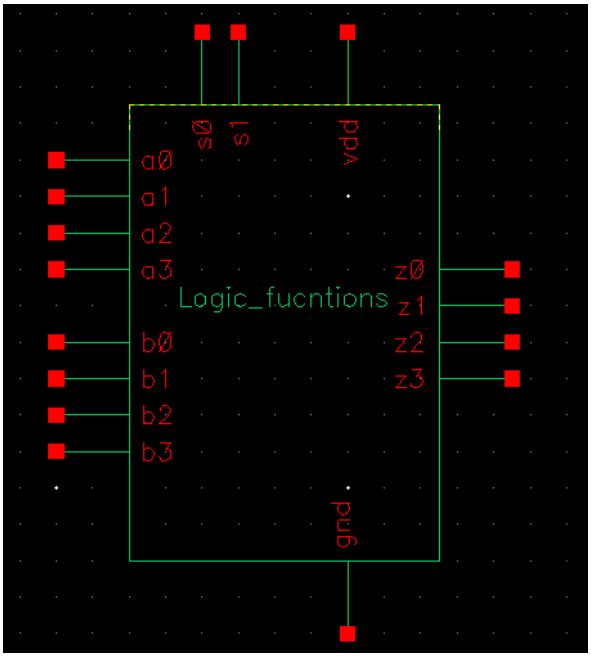


Figure 36: Symbol of Logic Function

Shifter design:

Another part of our project is a Shifter which can shift right, left transfer and transferred to zero.

We used 4 to 1 MUX to design the Shifter.

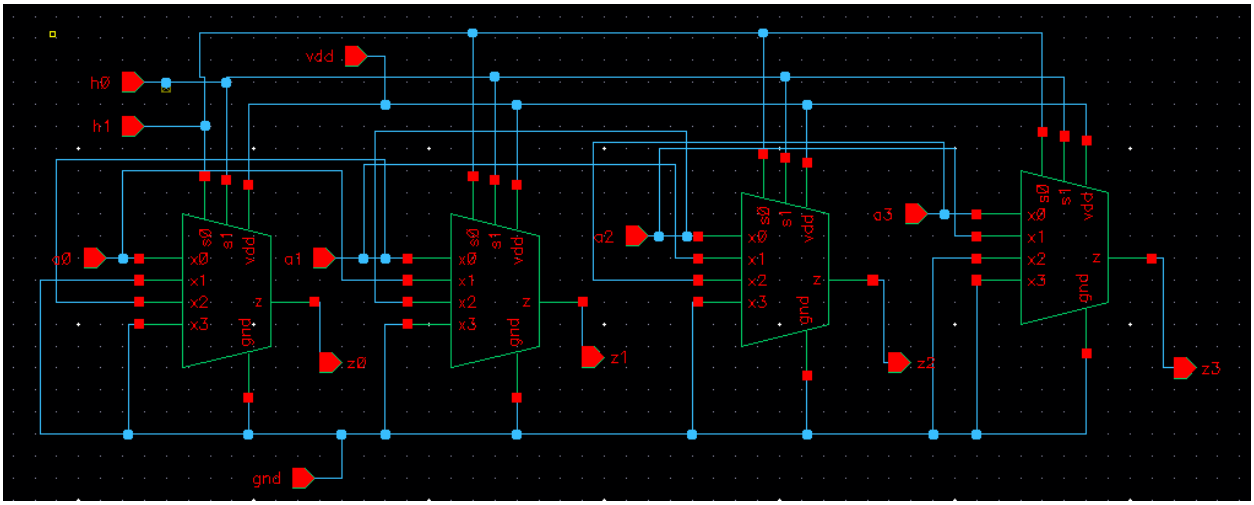


Figure 37: Schematic of 4 bit Shifter

ALU with shifter:

So our final schematic of the ALU with Shifter look like

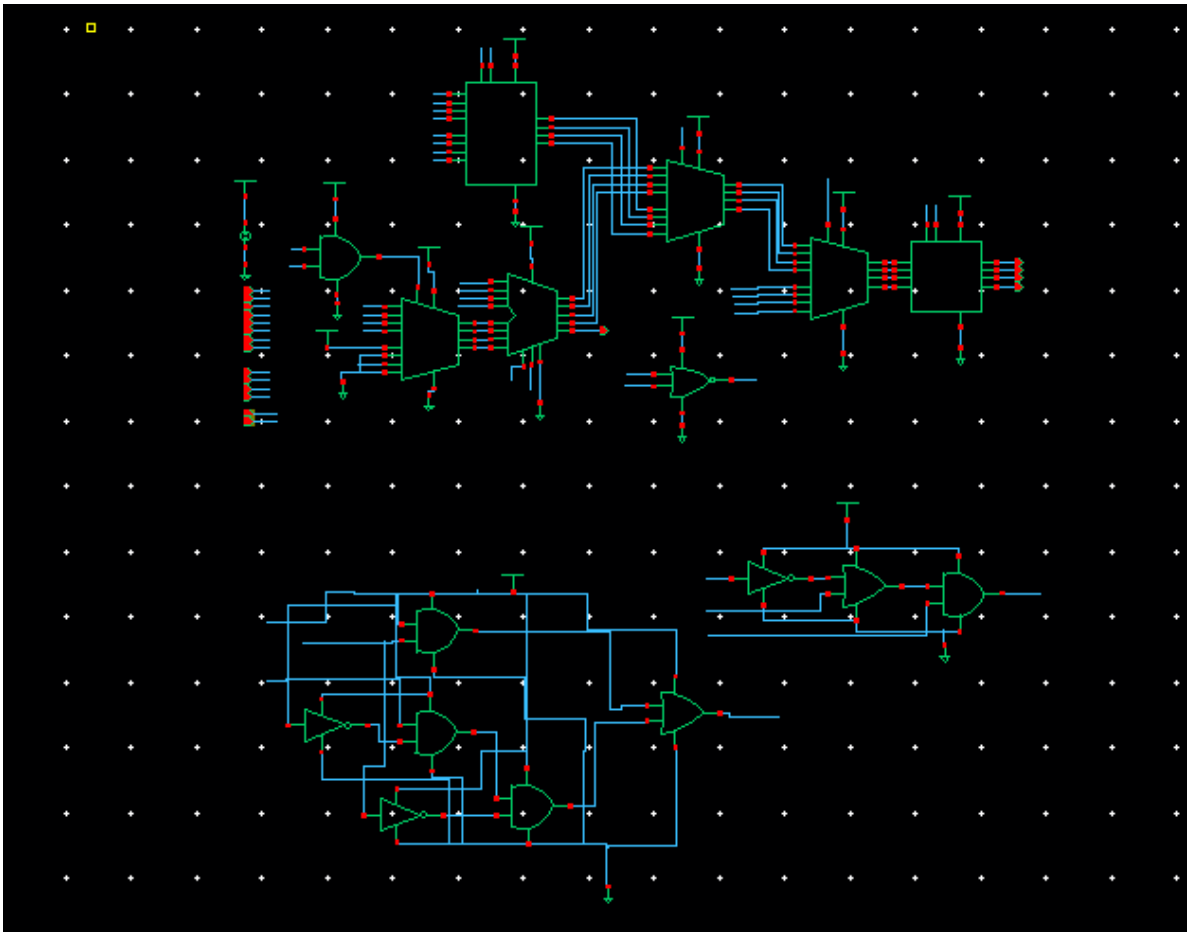


Figure 38: Schematic of ALU with Shifter

Output Results and Analysis

No shift:

We set **A=0111**(decimal 7) and **B=0011**(decimal 3) and the output result is given below

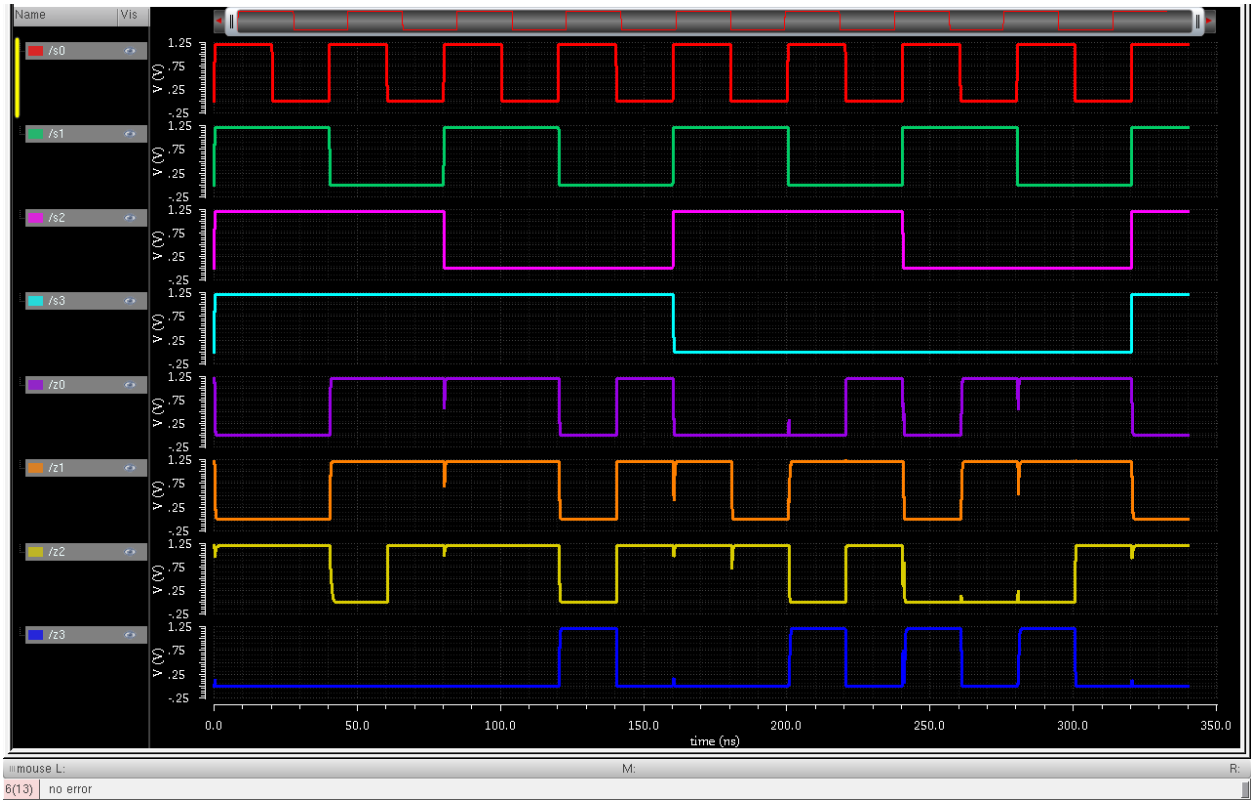


Figure 39: Output result(NO Shifted)

S3 S2 S1 S0	F		Output Result
0000	A	A=0111	0111
0001	A+B+1	B=0011	
0010	A-B-1		
0011	A+1		
0100	A		
0101	A+B		
0110	A-B		
0111	A-1		
1000	A’		
1010	A OR B		
1100	A AND B		

1110	A XOR B		

Left shift:

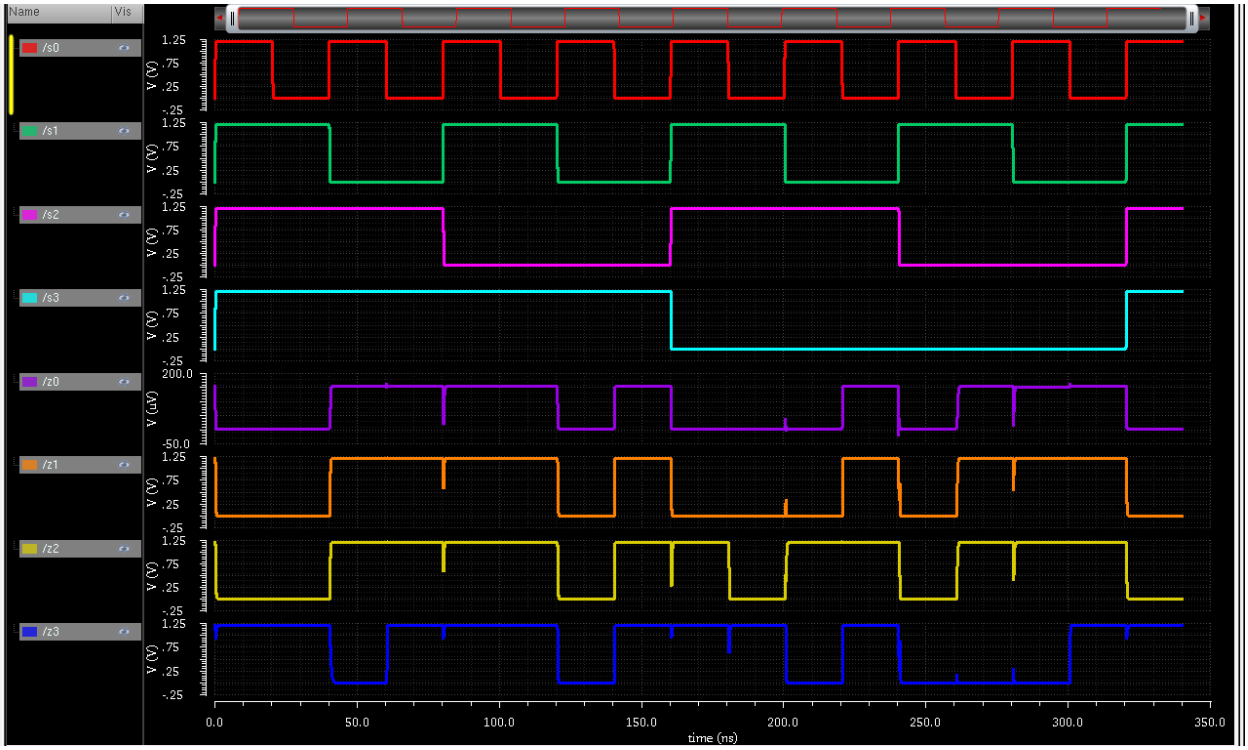


Figure 40: Output result(left Shifted)

S3 S2 S1 S0	F		Output Result
0000	A	A=0111	1110
0001	A+B+1	B=0011	
0010	A-B-1		
0011	A+1		
0100	A		
0101	A+B		
0110	A-B		
0111	A-1		
1000	A’		
1010	A OR B		
1100	A AND B		
1110	A XOR B		

Right Shift:

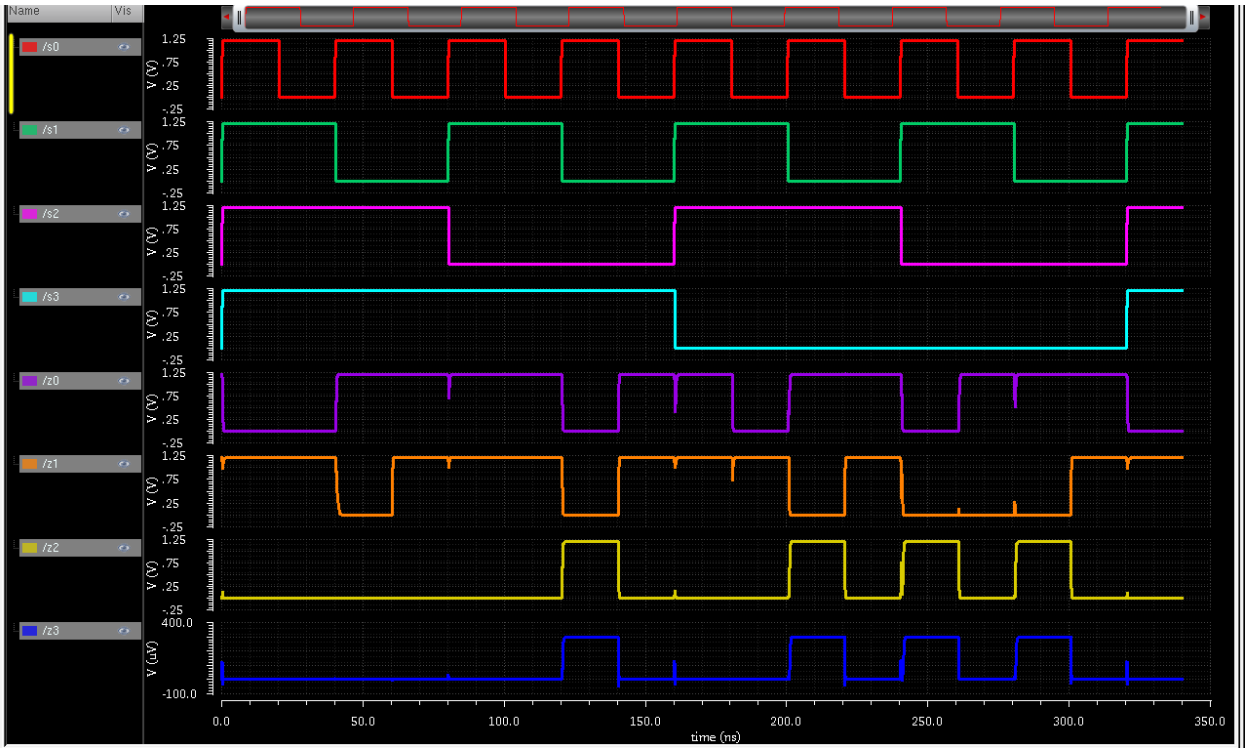


Figure 41: Output result(Right Shifted)

S3 S2 S1 S0	F		Output Result
0000	A	A=0111	0011
0001	A+B+1	B=0011	
0010	A-B-1		
0011	A+1		
0100	A		
0101	A+B		
0110	A-B		
0111	A-1		
1000	A’		
1010	A OR B		
1100	A AND B		
1110	A XOR B		

Transfer Zero:

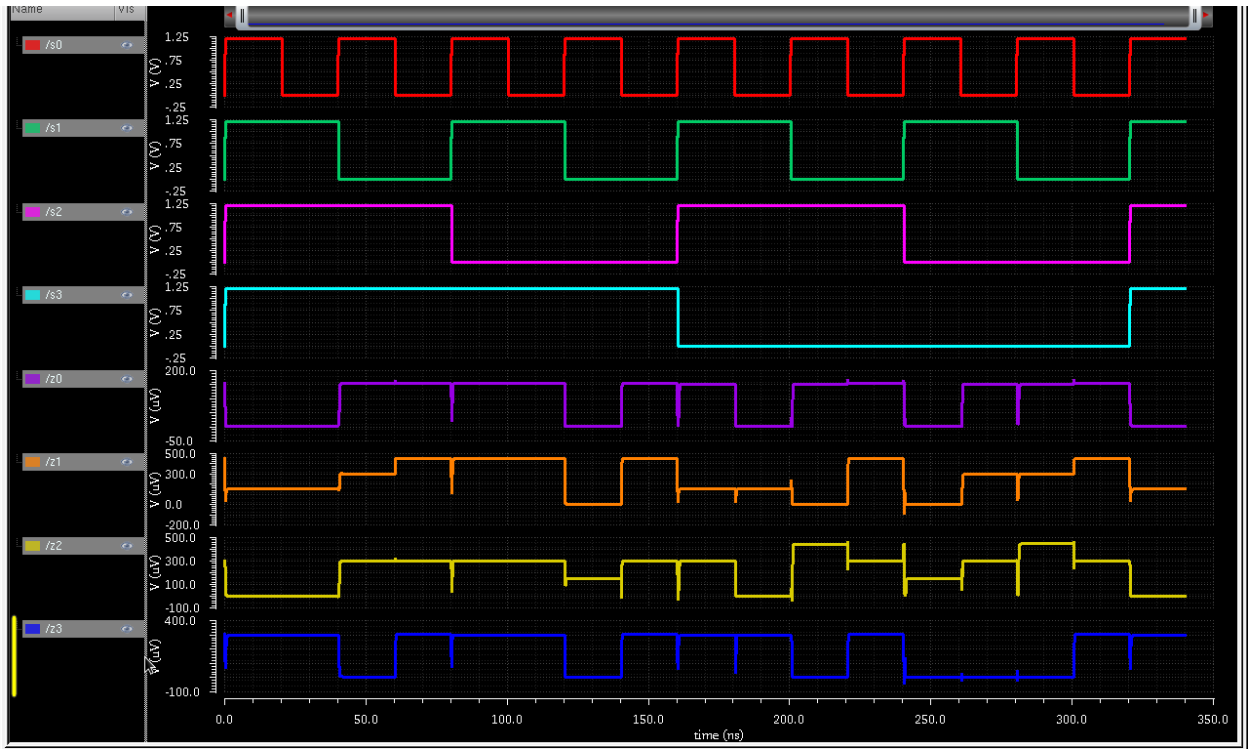


Figure 42: Zero Transfer

Total No of MOSFET:

The Chart shows MOSFET required to design different components of the ALU with Shifter Project

Name of the Elements	No of MOSFETs
Inverter	2
NOR	4
OR	6
AND	6
XOR	12
2 to 1 MUX	6
4 to 1 MUX	18
Quad 2 to 1 MUX	24
1 bit Full adder	42
4 bit ripple carry adder	216
Adder/subtract Selector	14
Increment/Decrement Selector	6
Carry In Selector	28
4 bit Logic gates	104
16 to 4 MUX	72
Logic Functions	176

Transfer function Selector	14
Shifter	72

Now we will calculate total no of MOSFET required to design the project

Name of the components	**No of components**	**No of MOSFET**
Logic Function	1	176
Quad 2 to 1 MUX	3	24x3
4 bit ripple carry adder	1	216
Shifter	1	72
Carry In Selector	1	28
Adder/subtract Selector	1	14
Transfer function Selector	1	14
Increment/Decrement Selector	1	6
Total No of MOSFET		**598**

Design Features:

Discussion

1. Faced problem using display and grid snapping. Sometimes wire does not connect
2. It was very difficult to design the layout of because there were many wires the , so got confused which wire to be connected
3. Our total number of MOSFET required is 598 & number of MOSFET may be reduced.
4. WE have tried to minimize size of the layout

Conclusion

Cadence Virtuoso® Schematic and Layout helped us to learn how to design the small scale Integrated circuit. This is the begging of VLSI. We have also learned how ALU and Shifter design. The journey of the design process of ALU was amazing. We have seen how low level to high-level design approach works