	CONTENTS:
1	Objective
2	Introduction
3	Software tools
4	Circuit diagram
5	Procedure
6	Netlist
7	Netlist output
8	Microwind layout
9	Microwind layout simulation
10	Conclusion

Objective:

The objective of this CMOS project for a 2-bit x 2-bit multiplier is to create netlist and simulate a digital circuit that performs multiplication of two 2-bit binary numbers using CMOS (Complementary Metal-Oxide-Semiconductor) technology. The project aims to achieve the following:

- Netlist of the digital circuit describing the connectivity and characteristics of the CMOS components, including all the transistors, voltage supplies and other components used.
- Simulation of the circuit netlist in any SPICE software.
- Physical layout of the digital circuit routing the CMOS components on the silicon substrate according to the design specifications and constraints in the Microwind software.
- Input and output waveforms for verification of the layout design to ensure proper connectivity, spacing, and adherence to design rules and constraints.

Introduction:

The project explores the realm of CMOS (Complementary Metal-Oxide-Semiconductor) technology by designing and implementing a 2-bit x 2-bit multiplier circuit. CMOS technology is widely used in integrated circuits (ICs) due to its low power consumption, high noise immunity, and scalability, making it ideal for various digital applications. A multiplier is a fundamental building block in digital circuit design, responsible for performing the multiplication operation between two binary numbers. In this project, we will leverage CMOS technology to develop a compact and efficient multiplier circuit capable of computing the product of two 2-bit binary numbers accurately. This project offers a great opportunity to deepen the understanding of CMOS technology and netlist making aspect of digital circuits besides critical thinking and problem solving skills.

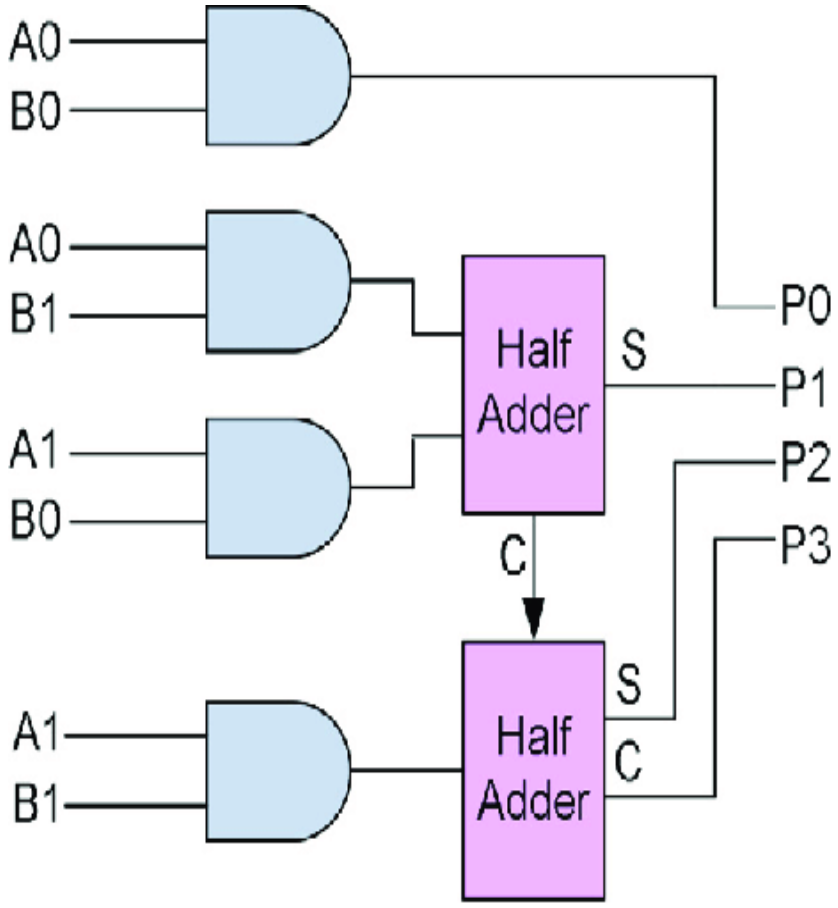
Software tools used:

The project makes use of the following software tools:

- A text editor software like notepad for writing the netlist
- A SPICE “Simulation Program with Integrated Circuit Emphasis” software for running and simulating the netlist and giving input and output waveforms to verify the functionality of the multiplier circuit.

- An integrated EDA (Electronic design automation) software like MircoWind for designing and simulating the circuits at layout level.

Circuit diagram:



Here, A1A0 and B1B0 are two 2-bit binary numbers that are to be multiplied and P3P2P1P0 is the product of them. This circuit makes use of 6 AND gates and 2 XOR gates as seen above.

Procedure:

Writing a netlist in Notepad involves following a specific syntax to define the components, connections, and characteristics of an electronic circuit. Here's a very short procedure:

- Open Notepad: Launch Notepad on your computer.
- Define Components: Use text-based commands to define the components of your circuit. For example, use "M" for transistors, "R" for resistors, and "C" for capacitors.

- Specify Connections: Define the connections between components using node numbers or labels. Use "+" and "-" to denote the positive and negative terminals of components.
- Set Component Parameters: Specify the parameters of each component, such as resistance values, transistor models, and capacitor capacitances. Use the appropriate syntax for each component type.
- Save the Netlist: Save the text file with a “.net” or “.cir” extension to indicate that it's a netlist file. Choose a descriptive filename that reflects the contents of the circuit.
- Review the netlist syntax to ensure correctness and adherence to the desired circuit specifications. Correct any errors or inconsistencies as needed.
- Use in Simulation: Once the netlist is written and saved, it can be used as input for circuit simulation software to analyse the behaviour of the electronic circuit.

Designing the layout of a circuit in Microwind involves following steps:

- Open Microwind: Launch the Microwind software on your computer.
- Create a New Project: Start a new project by selecting "File" > "New" from the menu. Choose the desired technology and parameters for your design.
- Place Components: Use the component palette to place electronic components (transistors, resistors, capacitors, etc.) onto the layout canvas. Arrange the components according to the desired circuit topology.
- Route Connections: Create metal interconnects to route connections between components. Use metal layers and vias to establish electrical connections between different layers of the layout.
- Ensure Proper Spacing: Ensure that there is adequate spacing between components and metal lines to avoid electrical interference and maintain signal integrity. Follow design rules and spacing constraints specified for the chosen technology.
- Verify Layout: Perform layout verification to check for errors, design rule violations, and connectivity issues. Use Microwind's built-in design rule check (DRC) and layout versus schematic (LVS) tools to ensure layout correctness.

- Optimize Layout: Optimize the layout design for factors such as area efficiency, power consumption, and signal integrity. Fine-tune layout parameters as needed to achieve optimal performance.
- Save Layout:
- Once the layout is complete and verified, save the layout file in the appropriate format. You can also export the layout for further analysis or fabrication.
- By following this procedure, you can design and implement the layout of your circuit in Microwind, ready for simulation or fabrication.

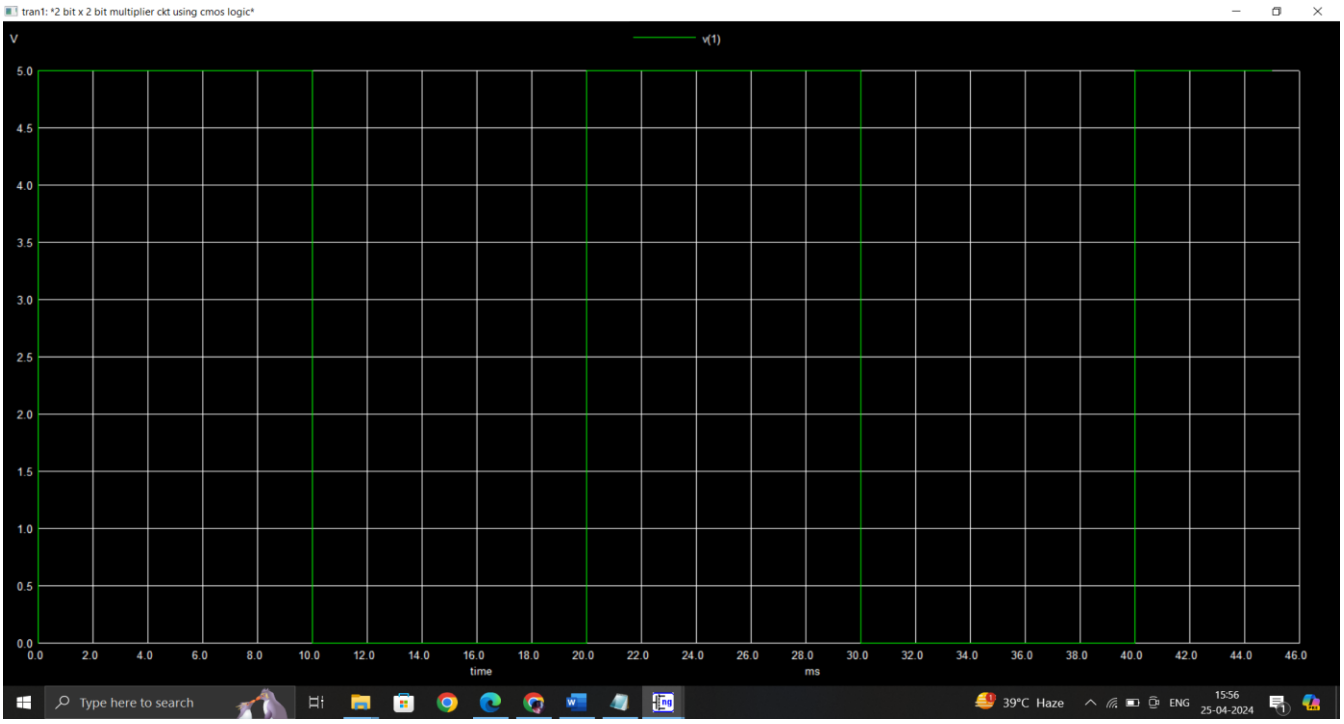
Netlist:

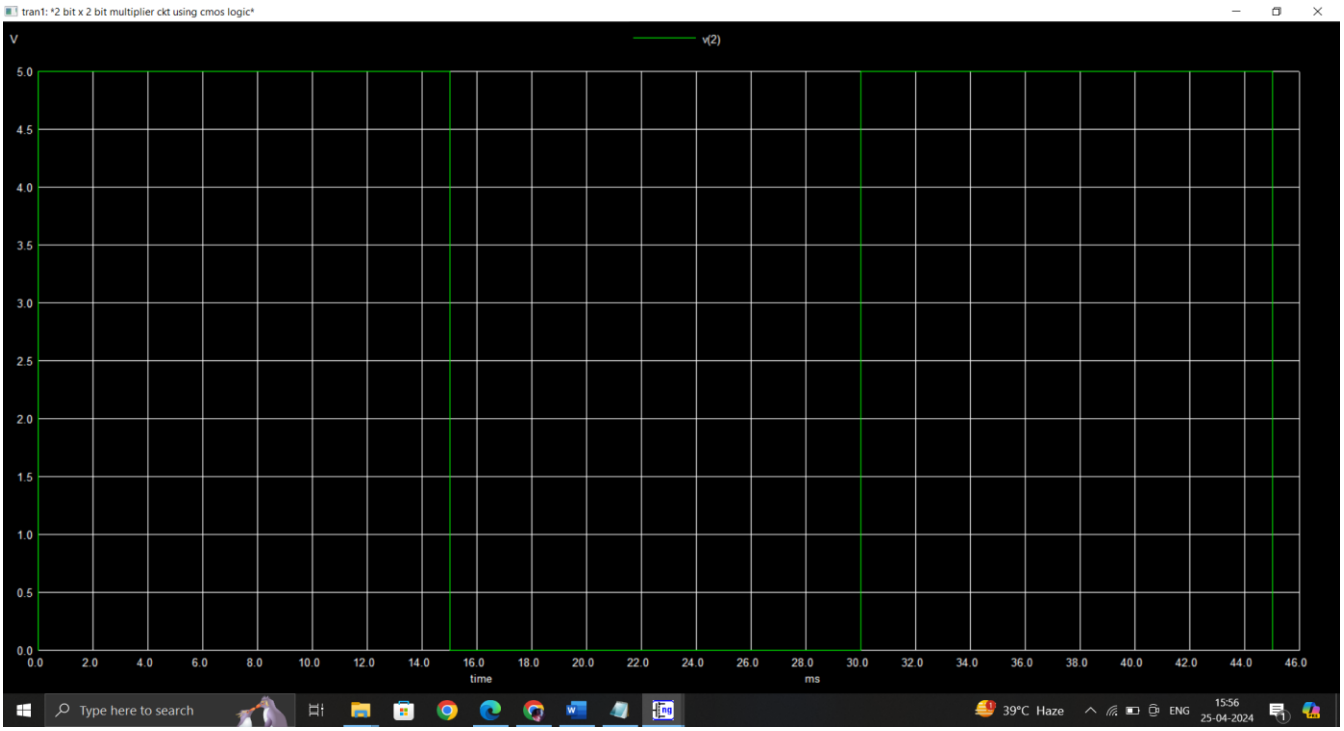
```
*2 bit x 2 bit multiplier ckt using cmos logic*
.model nmod nmos level=54 version=4.7
.model pmod pmos level=54 version=4.7
.subckt inv1 100 101 102 103 ;defining inverter subcircuit
M1 101 103 100 100 nmod w=100u l=10u ;D G S B is the order
M2 101 103 102 102 pmod w=100u l=10u
.ends
.subckt and1 104 105 106 107 108 109 110 ;defining and subcircuit
M3 107 105 104 104 nmod w=100u l=10u ;D G S B is the order
M4 108 106 107 107 nmod w=100u l=10u
M5 108 105 109 109 pmod w=100u l=10u
M6 108 106 109 109 pmod w=100u l=10u
xinv1 104 110 109 108 inv1          ;instantiating subckt
.ends
.subckt or1 111 112 113 114 115 116 117 ;defining or subcircuit
M7 114 112 111 111 nmod w=100u l=10u ;D G S B is the order
M8 114 113 111 111 nmod w=100u l=10u
M9 114 112 115 115 pmod w=100u l=10u
M10 115 113 116 116 pmod w=100u l=10u
xinv2 111 117 116 114 inv1 ;instantiating subckt
.ends
.subckt xor1 118 119 120 121 122 123 124 125 126 127 128 ;defining xor
subcircuit
xinv3 118 121 128 119 inv1
xinv4 118 122 128 120 inv1
M11 124 121 118 118 nmod w=100u l=10u ;D G S B is the order
```

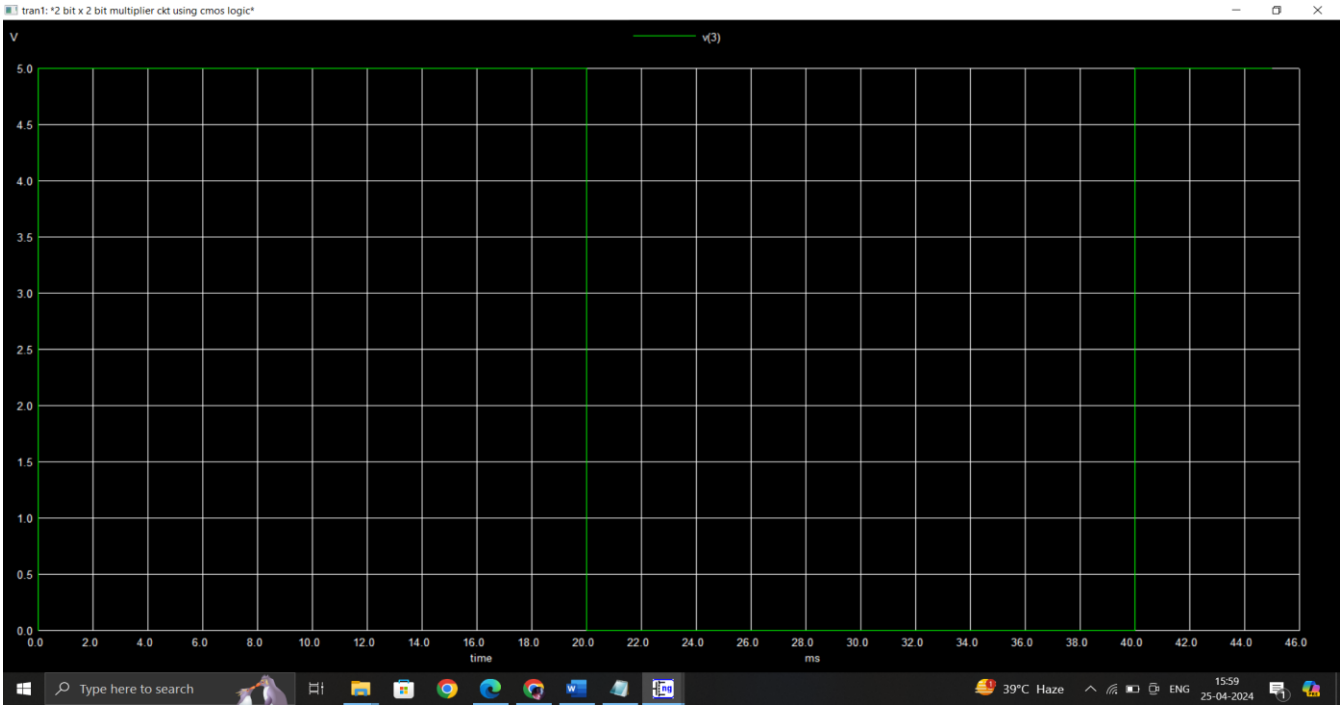
```
M12 125 122 124 124 nmod w=100u l=10u
M13 123 119 118 118 nmod w=100u l=10u
M14 125 120 123 123 nmod w=100u l=10u
M15 125 121 127 127 pmod w=100u l=10u
M16 127 120 128 128 pmod w=100u l=10u
M17 125 119 126 126 pmod w=100u l=10u
M18 126 122 128 128 pmod w=100u l=10u
.ends
x1and 0 1 2 5 6 7 8 and1 ;a0 and b0 output p0 at 8
x2and 0 2 3 9 10 11 12 and1 ;a1 and b0 output at 12
x3and 0 1 4 13 14 15 16 and1 ;a0 and b1 output at 16
x4xor 0 12 16 17 18 19 20 21 22 23 24 xor1 ;a1b0 xor a0b1 output p1 at 21
x5and 0 12 16 25 26 27 28 and1 ;a1b0 and a0b1 output at 28
x6and 0 3 4 29 30 31 32 and1 ;a1 and b1 output at 32
x7xor 0 28 32 33 34 35 36 37 38 39 40 xor1 ;a1b1 xor c1 output p2 at 37
x8and 0 28 32 41 42 43 44 and1 ;a1b1 and c1 output p3 at 44
vd1 7 0 5
vd2 11 0 5
vd3 15 0 5
vd4 24 0 5
vd5 27 0 5
vd6 31 0 5
vd7 40 0 5
vd8 43 0 5
va0 1 0 pulse(0 5 0 5ns 5ns 10m 20m)
vb0 2 0 pulse(0 5 0 5ns 5ns 15m 30m)
va1 3 0 pulse(0 5 0 5ns 5ns 20m 40m)
vb1 4 0 pulse(0 5 0 5ns 5ns 25m 50m)
.tran 0.1m 60m
.control
run
plot v(1) ;a0
plot v(2) ;b0
plot v(3) ;a1
plot v(4) ;b1
plot v(8) ;p0
plot v(21) ;p1
plot v(37) ;p2
plot v(44) ;p3
```

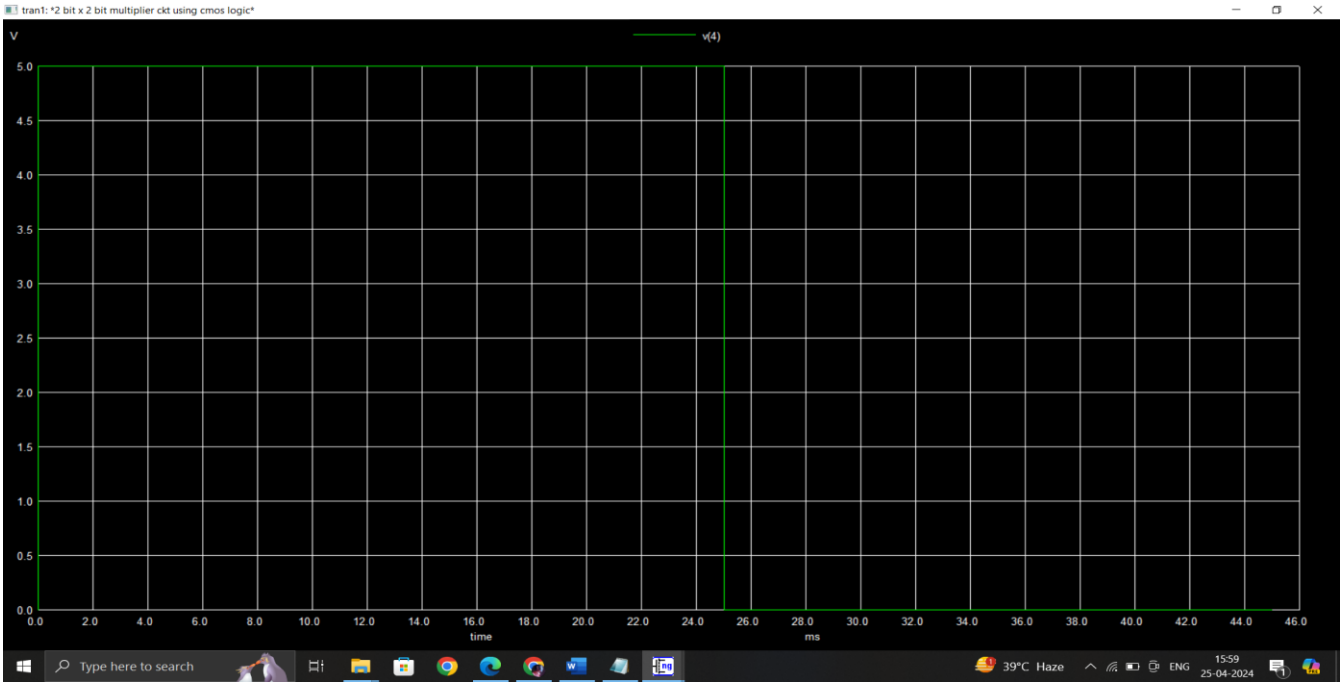
.endc
.end

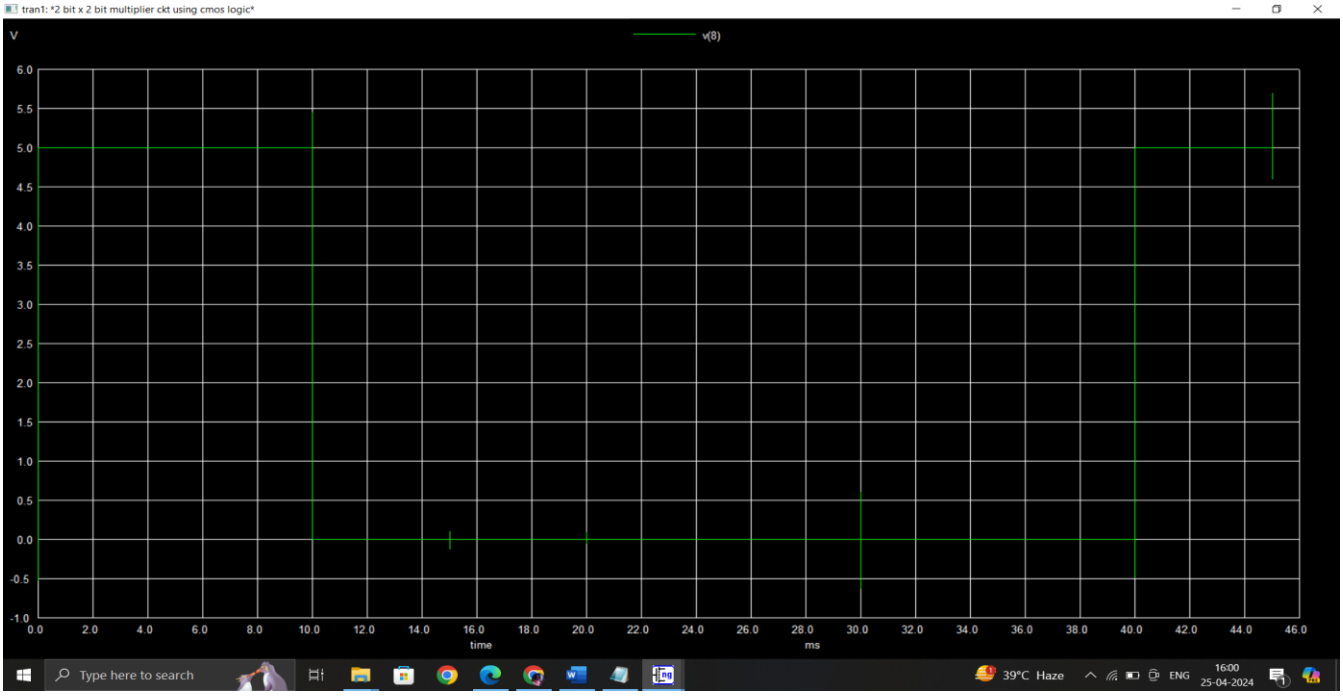
Netlist output:

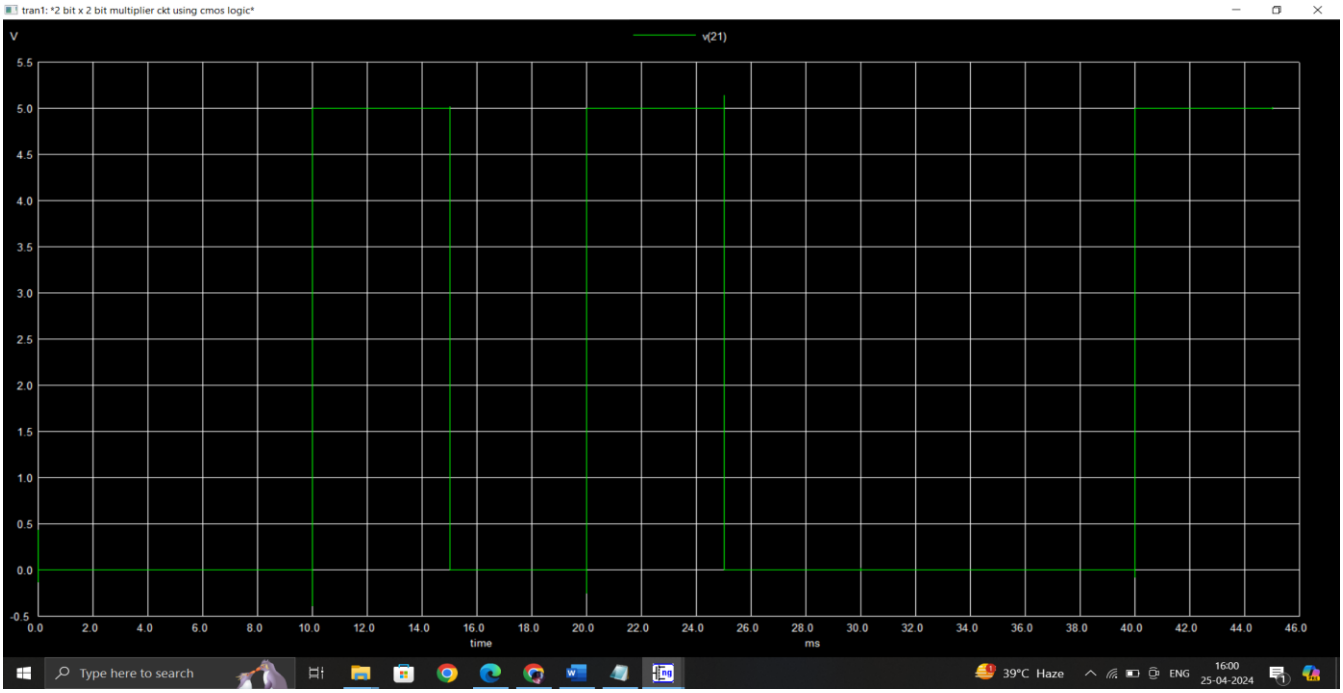


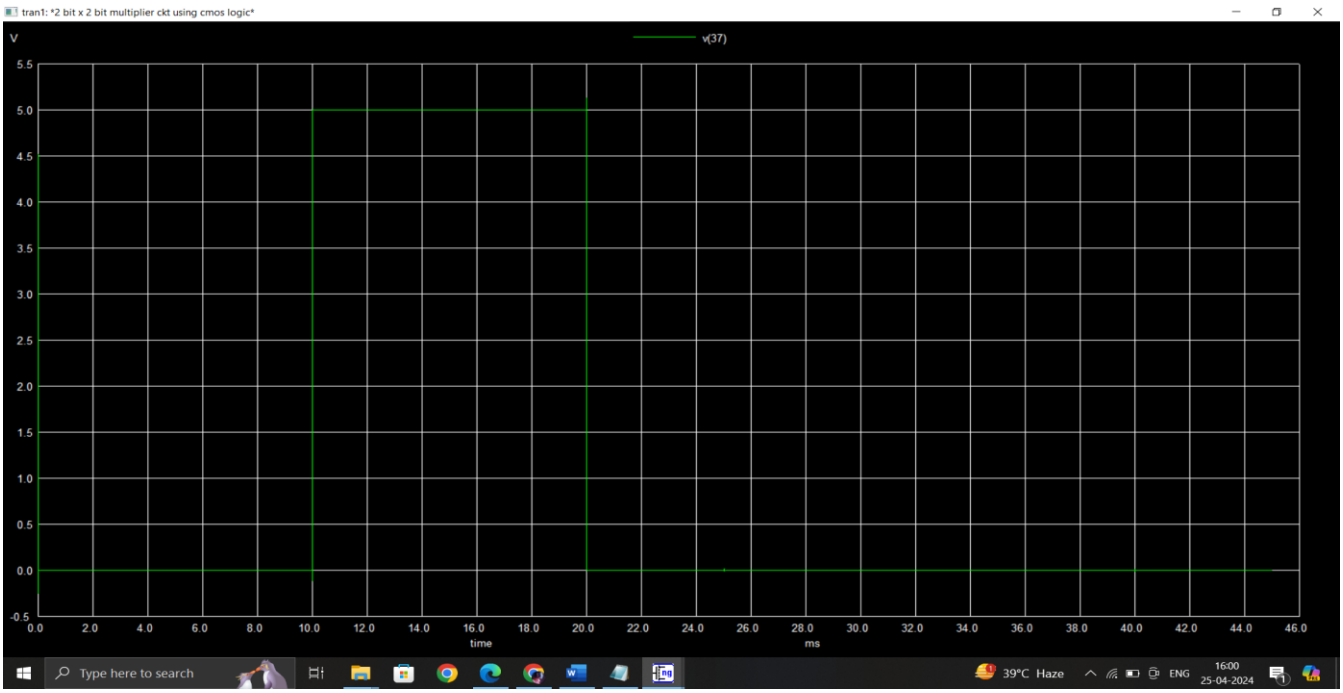


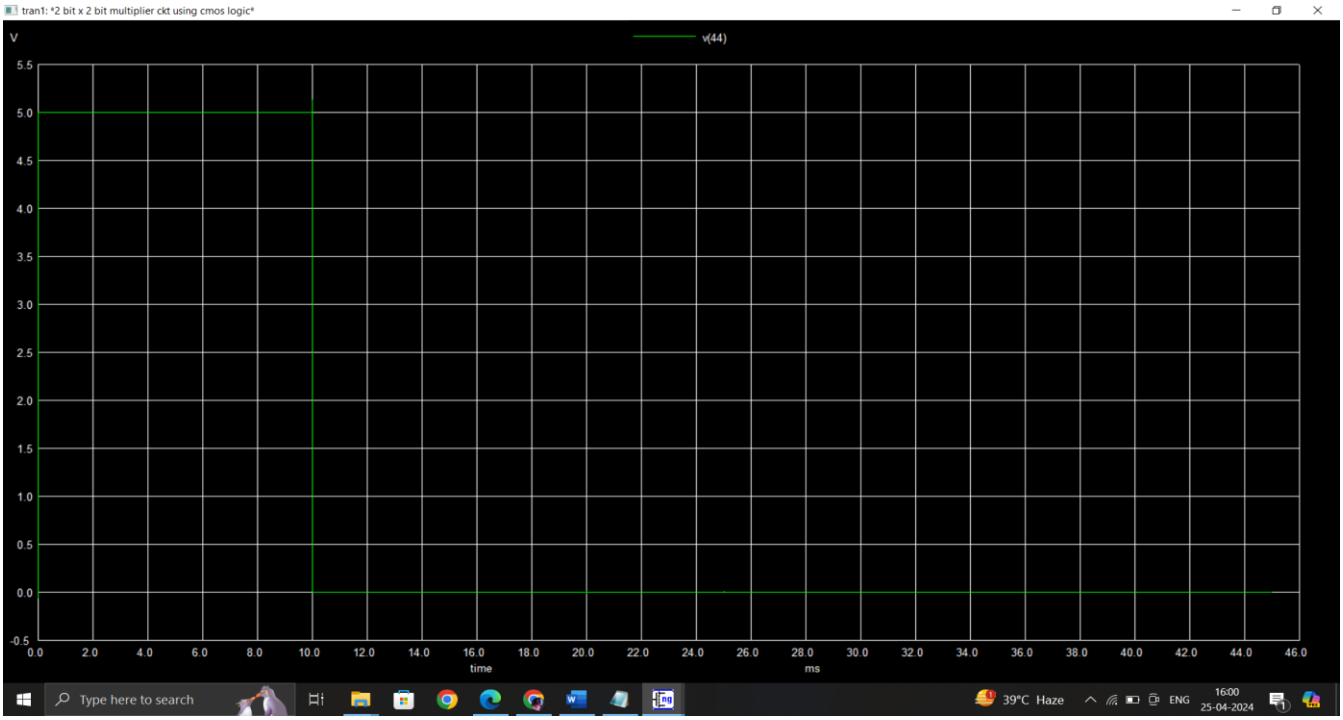

tran1: *2 bit x 2 bit multiplier ckt using cmos logic*
V
v(3)
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0 2.0 4.0 6.0 8.0 10.0 12.0 14.0 16.0 18.0 20.0 22.0 24.0 26.0 28.0 30.0 32.0 34.0 36.0 38.0 40.0 42.0 44.0 46.0
time
ms
Type here to search
39°C Haze
ENG
15:59
25-04-2024


tran1: *2 bit x 2 bit multiplier ckt using cmos logic*
V
v(4)
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0.0 2.0 4.0 6.0 8.0 10.0 12.0 14.0 16.0 18.0 20.0 22.0 24.0 26.0 28.0 30.0 32.0 34.0 36.0 38.0 40.0 42.0 44.0 46.0
time
ms
Type here to search
39°C Haze
ENG
15:59
25-04-2024

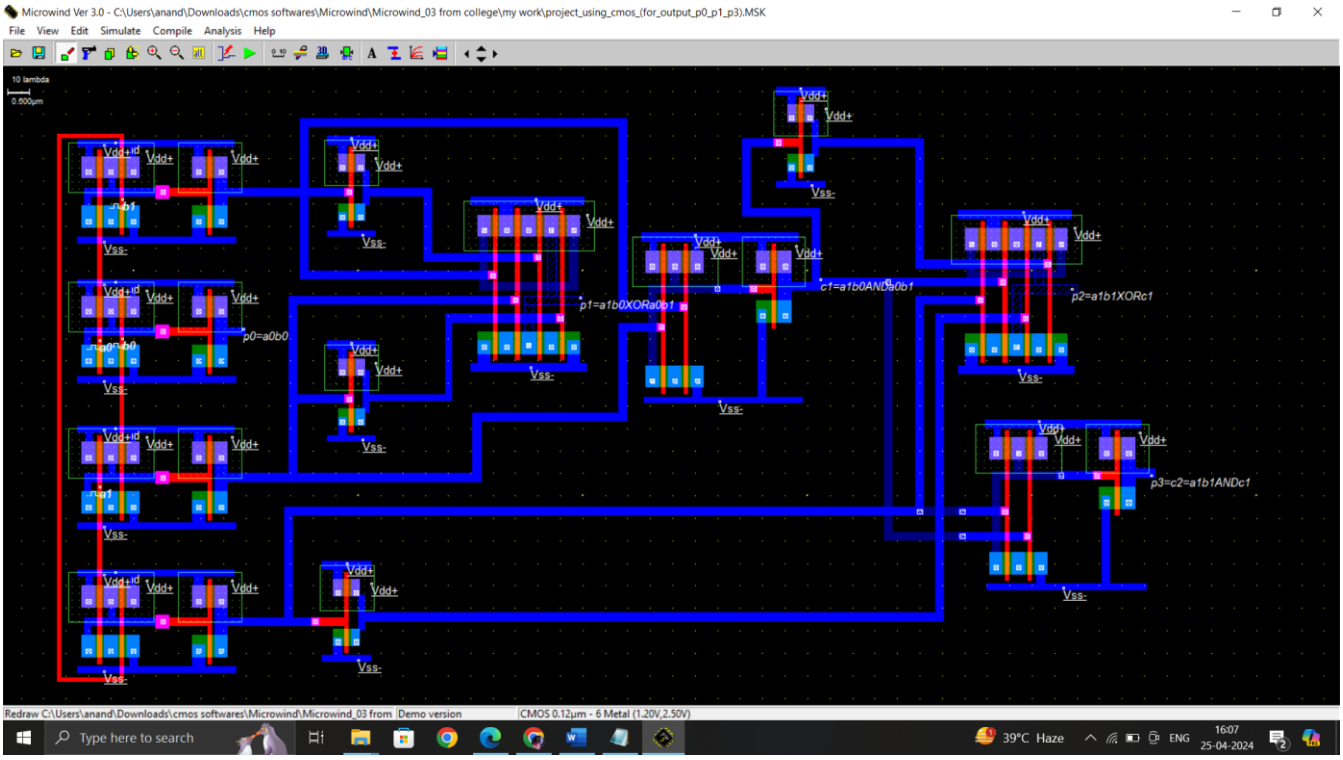

tran1: *2 bit x 2 bit multiplier ckt using cmos logic*
V
v(8)
6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
-0.5
-1.0
0.0 2.0 4.0 6.0 8.0 10.0 12.0 14.0 16.0 18.0 20.0 22.0 24.0 26.0 28.0 30.0 32.0 34.0 36.0 38.0 40.0 42.0 44.0 46.0
time
ms
Type here to search
39°C Haze
ENG
16:00
25-04-2024


tran1: *2 bit x 2 bit multiplier ckt using cmos logic*
V
v(21)
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
-0.5
0.0 2.0 4.0 6.0 8.0 10.0 12.0 14.0 16.0 18.0 20.0 22.0 24.0 26.0 28.0 30.0 32.0 34.0 36.0 38.0 40.0 42.0 44.0 46.0
time
ms
Type here to search
39°C Haze
ENG
16:00
25-04-2024

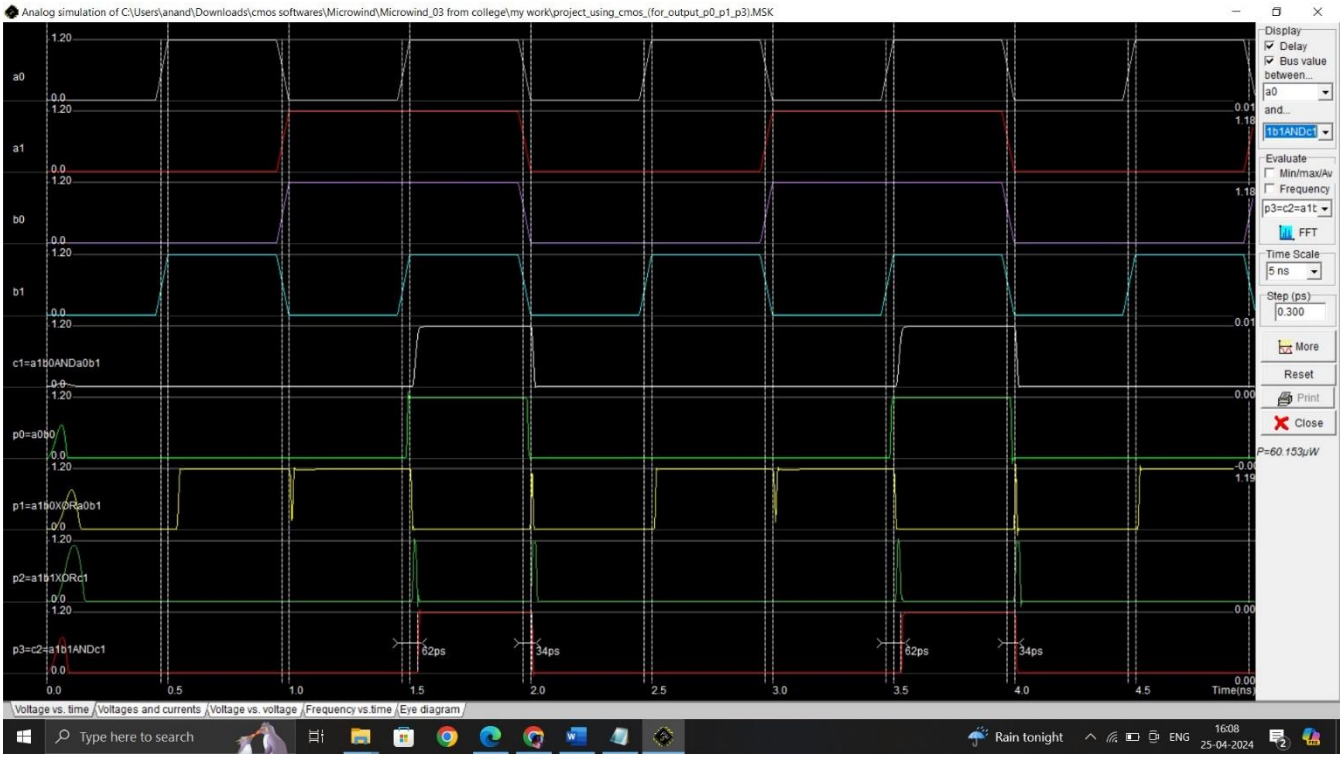

tran1: *2 bit x 2 bit multiplier ckt using cmos logic*
V
v(37)
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
-0.5
0.0 2.0 4.0 6.0 8.0 10.0 12.0 14.0 16.0 18.0 20.0 22.0 24.0 26.0 28.0 30.0 32.0 34.0 36.0 38.0 40.0 42.0 44.0 46.0
time
ms
Type here to search
39°C Haze
ENG
16:00
25-04-2024


tran1: *2 bit x 2 bit multiplier ckt using cmos logic*
V
v(44)
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
-0.5
0.0 2.0 4.0 6.0 8.0 10.0 12.0 14.0 16.0 18.0 20.0 22.0 24.0 26.0 28.0 30.0 32.0 34.0 36.0 38.0 40.0 42.0 44.0 46.0
time
ms
Type here to search
39°C Haze
ENG
16:00
25-04-2024

Microwind layout:



Microwind layout simulation:



Hence, we have successfully made a working netlist and layout design of a two-bit by two-bit multiplier circuit and also verified both of them by simulating the netlist and layout design to obtain the set of inputs and corresponding outputs.