VIETNAM NATIONAL UNIVERSITY

HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY

EE4451: DIGITAL IC DESIGN LABORATORY 1-5

Instructor : **Dr. Tran Hoang Linh**

Teaching Assistant : **M. S. Bui Le Quoc Doanh**

M. S. Nguyen Phan Thien Phuc

Subject : Digital IC Design

Group : 02

Members : Phuong Thu Tan - 2051190

Nguyen Tung Duong - 2051104

Ho Chi Minh City, May 31st, 2024

TABLE OF CONTENT

PART I: LABORATORY

Laboratory 1: MOS Device Characterization

1.1 Experiment 1: Create layout for MOS devices. (PMOS and NMOS)

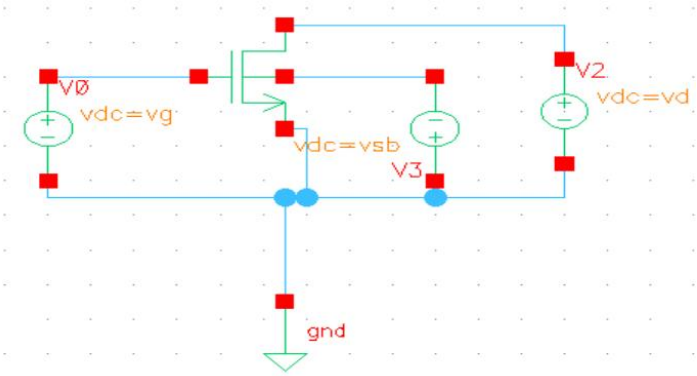


Figure 1.1*: NMOS Schematic from DeviceFreePDK45

MOS (Metal Oxide Semiconductor) transistors are a fundamental component in modern electronic devices. Their behavior can be described in terms of current-voltage (I-V) characteristics, which are essential for understanding their behavior in electrical circuits. The I-V curve of a MOS transistor is determined by the relationship between the drain current. Start checking the behavior of the MOS through the basic curve on the input output. Specifically, use the I-V characteristic to examine the active regions of the MOS and draw some technical conclusions. It is extremely necessary for IC design engineers to thoroughly understand how each MOS operates and the core of MOS installation. Through the cadence tool, we will conduct setup and measurements as described by the Schematic in Figure 1.1*.

First of all, we will work with the NMOS, we connect its terminals (G, D, S, B) to control the energy and operating signals to be more proactive in measurement. Thus, terminal G (Gate) will be connected to the VGS source, S (Source) will be connected to ground, D (Drain) will be attached to the VDS source. However, terminal B (Bulk), will normally be grounded to the ideal state of the MOS, we will also connect the power source to this terminal and set the voltage to approximately 0.

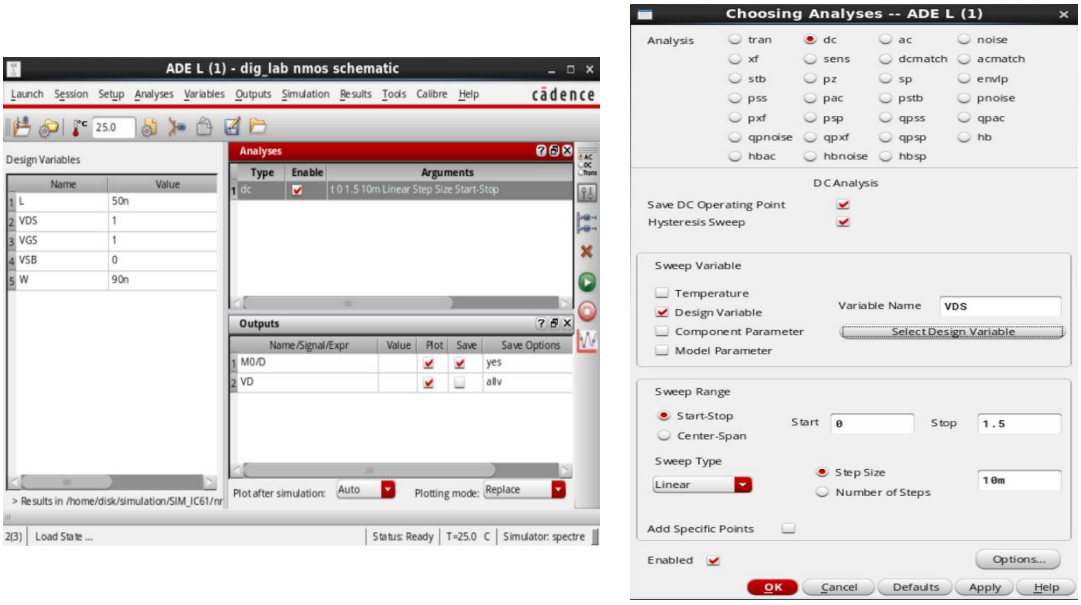


Figure 1.2 : *Setting parameter for characterizing when* $I_{DS} = f(V_{DS})$, $V_{GS} = 1V$

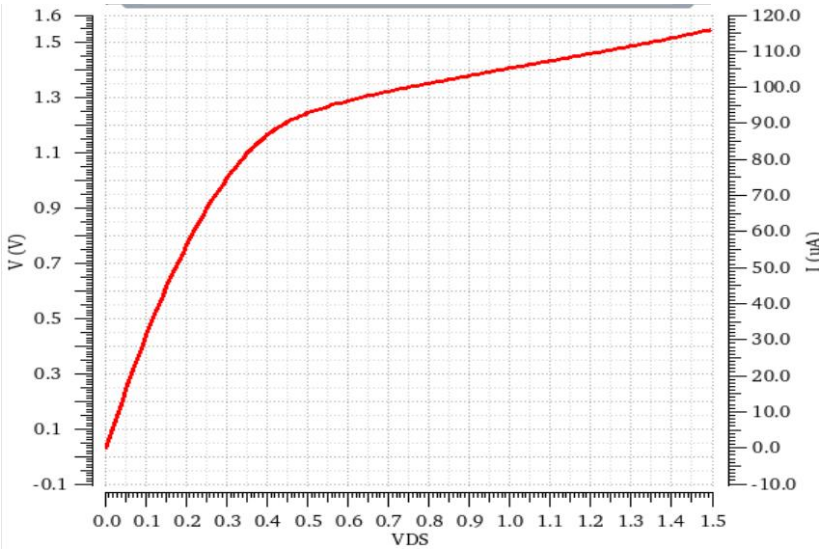


Figure 1.3 : *The simulation of* $I_{DS} = f(V_{DS})$, $V_{GS} = 1V$

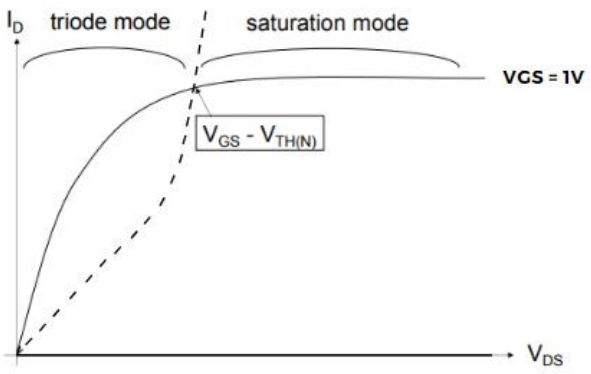


Figure 1.4 : *The sketch* $-$ *by* $-$ *hand of* $I_{DS} = f(V_{DS})$

In the triode active region of the MOS, simply described by the Drain-source V_{DS} being smaller than the Gate-Source VGS minus the threshold voltage (V_{TH}). After V_{DS} overcomes this threshold and reaches a certain result, that is, reaching the V_{GS} - V_{TH} level will lead the MOS to a new operating region called mode saturation.

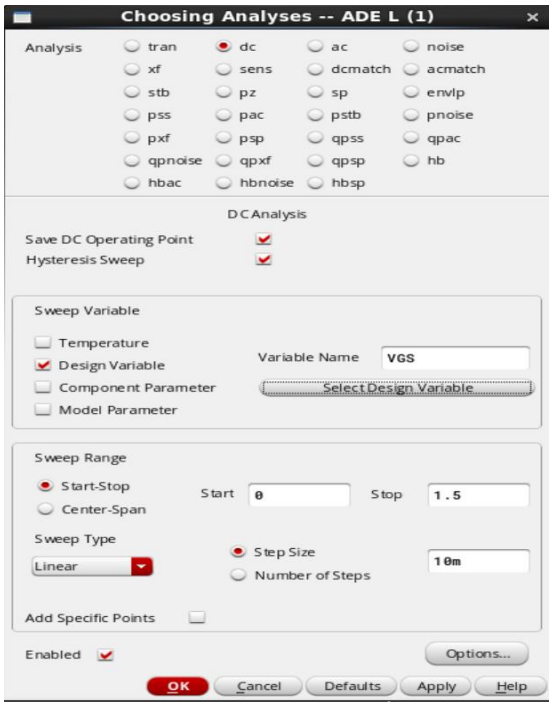


Figure 1.5 : *Setting parameter for characterizing when* $I_{DS} = f(V_{GS})$, $V_{DS} = 1V$

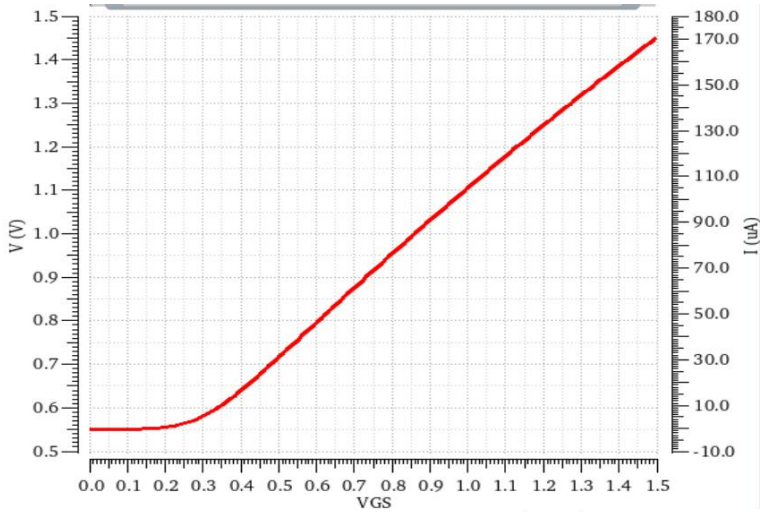


Figure 1.6 : *The simulation of* $I_{DS} = f(V_{GS})$, $V_{DS} = 1V$

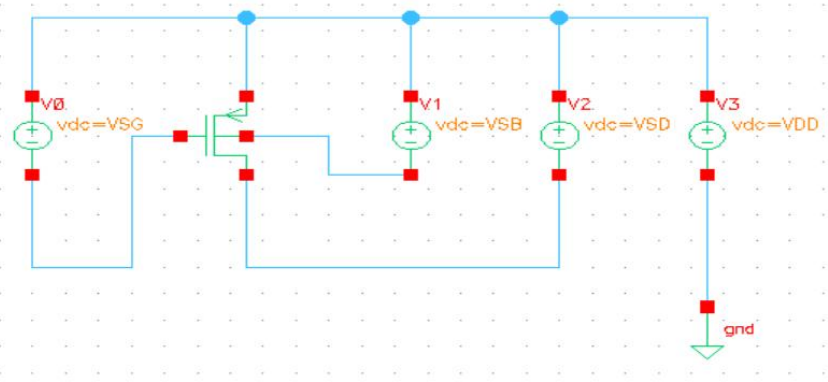


Figure 1.7: NMOS Schematic from DeviceFreePDK45*

In digital IC design, PMOS and NMOS transistors are crucial components used to create logic gates and circuits. PMOS transistors conduct when a low voltage is applied to their gate, while NMOS transistors conduct with a high gate voltage. By combining these transistors in complementary pairs, such as in CMOS technology, designers can create efficient and low-power logic gates. PMOS is typically used for logic 1 (high) levels, while NMOS is used for logic 0 (low) levels. This complementary pairing allows for minimal power consumption during operation, making CMOS technology widely used in digital ICs, ranging from microprocessors to memory chips.

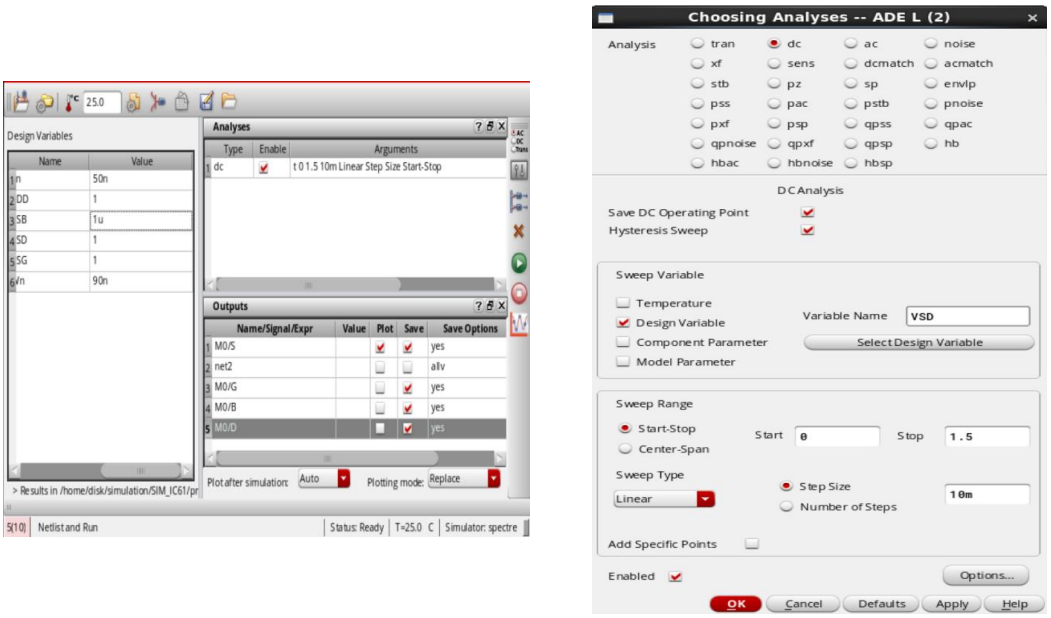


Figure 1.8 : *Setting parameter for characterizing when* $I_{SD} = f(V_{SD})$, $V_{SG} = 1V$

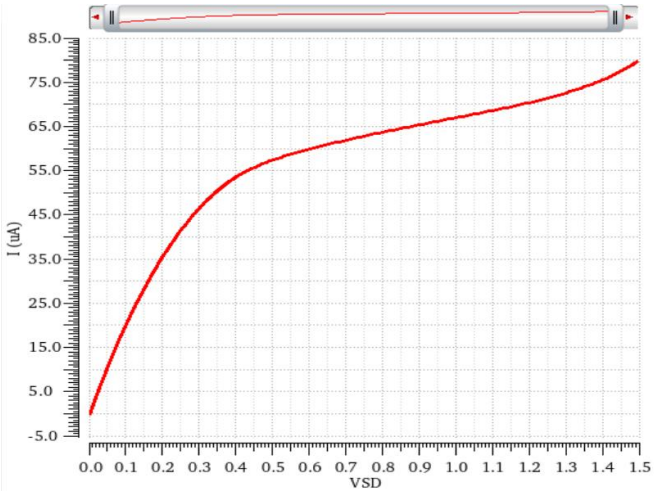


Figure 1.9 : *The simulation of* $I_{SD} = f(V_{SD})$, $V_{SG} = 1V$

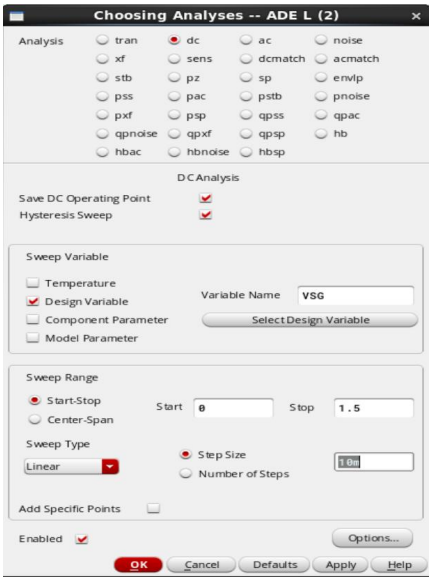


Figure 1.10 : *Setting parameter for characterizing when* $I_{SD} = f(V_{SG})$, $V_{SD} = 1V$

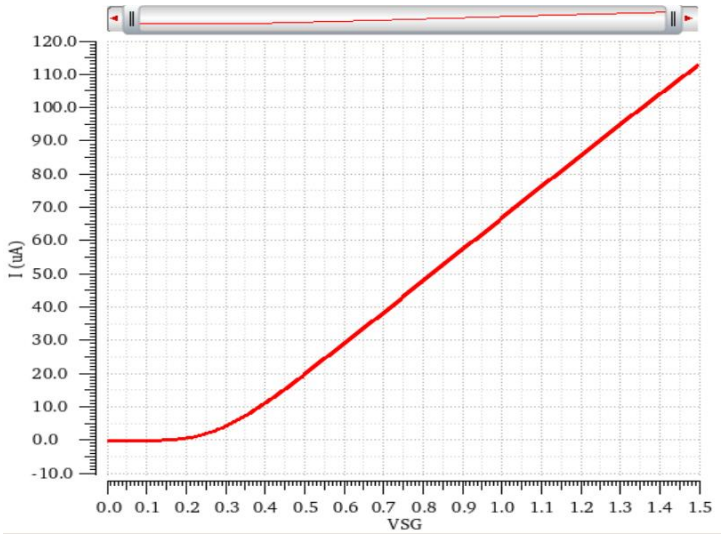
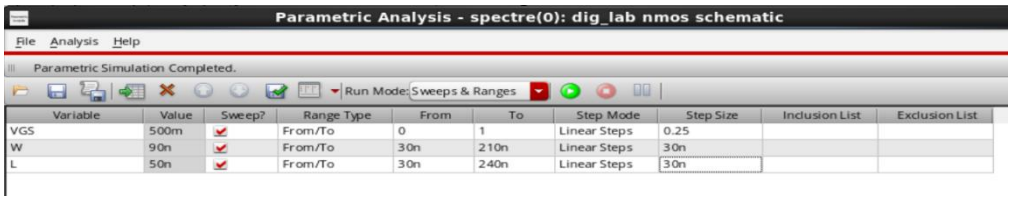


Figure 1.11 : *The simulation of* $I_{SD} = f(V_{SG})$, $V_{SD} = 1V$

1.2 Experiment 2: Varying VGS, and device's size effect on IV characteristics

In this second experiment, the report will clarify the influence of VGS and transistor size on the IV characteristic of the circuit signal. To practice on Cadence, the experiment will be conducted through Parametric Analysis in Cadence's ADE-L to create more variable measurement units in the same analysis.



Variable	Value	Sweep?	Range Type	From	To	Step Mode	Step Size	Inclusion List	Exclusion List
VGS	500m	✓	From/To	0	1	Linear Steps	0.25		
W	90n	✓	From/To	30n	210n	Linear Steps	30n		
L	50n	✓	From/To	30n	240n	Linear Steps	30n		

Figure 1.12 : *Setup parameter for Parametric Analysis for varying Gate Voltage and transistor's size*

As soon as we change the magnitude of VGS on the NMOS behavior measurement circuit, we can determine that the ID will simultaneously change with each different VGS applied.

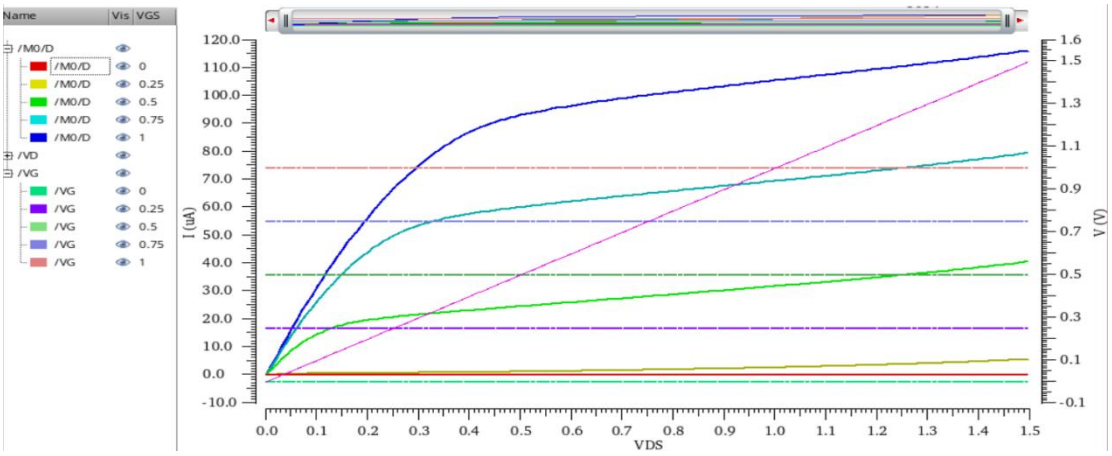


Figure 1.13 : $I_{DS} = f(V_{GS})$ *at* $V_{GS} = \{0,\ 0.25,\ 0.5,\ 0.75,\ 1.0\}\ V$

Specifically at Figure 1.13, when VG = 1V, ID (dark blue trace) is showing a saturation state in the signal region reaching 100-120uA when referenced on the ascending scan direction of VDS [0-1.5V]; Meanwhile, when the VGS level is reduced to half (0.5V), this same value is immediately found in the ID (green) state of 22-40uA. This also happens when VG gradually decreases, resulting in a decreasing ID value on the VDS domain.

Comment: The value of ID depends a lot on the magnitude value of VGS, proportional to the magnitude of VG, evidenced by the fact that when VG decreases, ID decreases and vice versa.

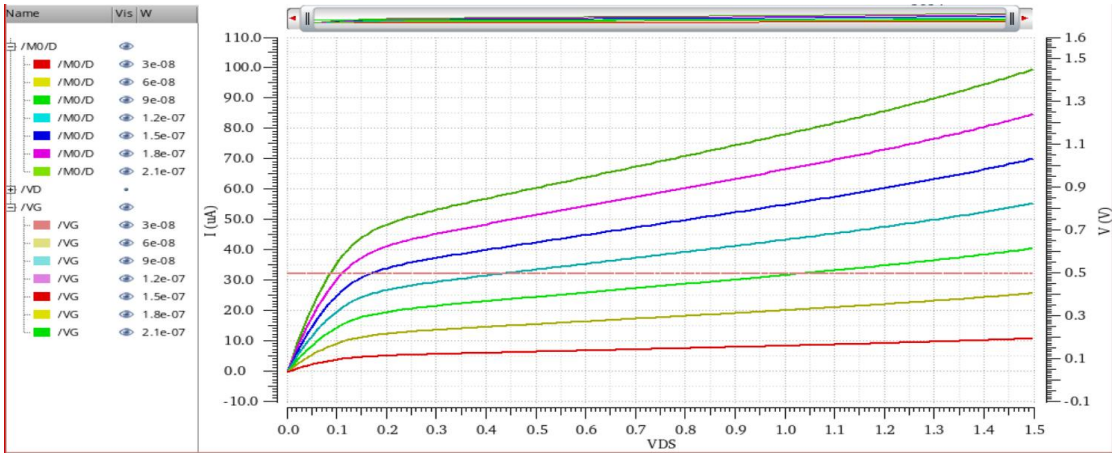


Figure 1.14 : $I_{DS} = f(V_{DS})$ *at* $W = \{90,\ 180,\ 270,\ 360\}\ nm$

Do the same with the size of the Transistor, when we gradually increase the Width of NMOS, immediately the ID according to the domain reference of VDS gradually decreases. Specifically, when gradually increasing the Width of NMOS in steps of 90nm from 90-360nm, we see that the ID corresponding to VDS gradually increases in the saturation region from 5-100uA.

Comment: The value of ID depends a lot on the size of NMOS, proportional to the magnitude of Width, evidenced by the fact that when VG decreases, W decreases and vice versa.

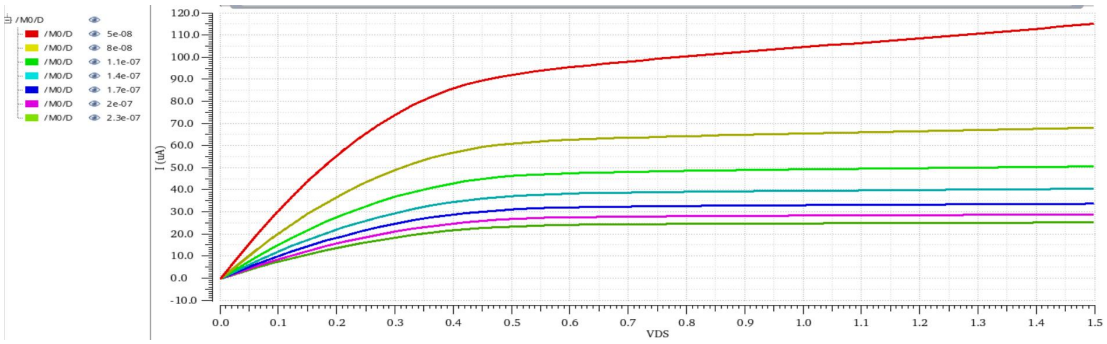


Figure 1.15 : $I_{DS} = f(V_{DS})$ *at* $L\ = \{50,\ 80,\ 110,\ 140,\ 170,\ 200,\ 240\}\ nm$

Finally, let's increase L on the changing area of ID according to VDS, we can see that the current on the Drain gate of NMOS gradually decreases. This is exactly the opposite of the two experiments above, but it reflects quite accurately the influence of transistor size on the NMOS input current. Specifically, when gradually increasing the

L value on the 90nm step, we see that the more gradually we increase, the more we see the decreasing level of ID. Specifically, when gradually increasing the Length of NMOS in steps of 30nm from 50-240nm, we see that the ID corresponding to VDS gradually decreases in the saturation region from 115 to 25uA.

Comment: The value of ID depends a lot on the size of NMOS, inversely proportional to the magnitude of L, evidenced by the fact that when VG decreases, ID increases and vice versa.

This makes sense since, when VGS is raised to a specific point, the NMOS channel opens up and becomes active, facilitating significant flow from the Source through the Drain. The trials mentioned above will also make it clear that when VGS increases, the channel's limit expands in comparison to earlier VGS.

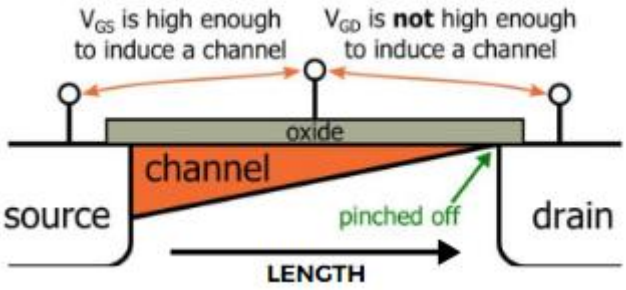


Figure 1.16 : *Schematic view of a surface – channel*

Furthermore, enlarging the width of the transistor will enlarge the channel's depth, which will inevitably result in a greater rise in the current flowing from the source to the drain. Ports are connected by channels, therefore when the channel's length from S to D stays the same but its depth grows, naturally, there will be more flow through at both ends. The W value of NMOS is likewise equivalent to the channel's depth. Conversely, a significant increase in Length in NMOS results in an indirect increase in the distance between Source and Drain in the transistor; nevertheless, it should be noted that VGS determines the channel's length. Thus, if VGS stays constant, the state of NMOS will saturate sooner and the current through the NMOS Drain will decrease when L rises to a sufficiently high level.

Additionally, analyzing according to the formula of IDS, we have

$$I_{\mathrm{Dsat}} = \frac{W}{2L} C_{\mathrm{i}} \mu \left(V_{\mathrm{g}} - V_{\mathrm{th}} \right)^2$$

Obviously, W and Vgs are proportional to ID in saturation mode, whereas L increases would lead to the decrease of ID.

1.3 Experiment 3: Second order effect

Second-order effects in NMOS transistors include velocity saturation, channel length modulation, drain-induced barrier lowering, body effect, gate-induced drain leakage, and hot carrier injection. These effects affect device performance beyond fundamental MOSFET behavior, making it crucial for reliable, high-performance integrated circuit design as device dimensions decrease. In this experiment, we will analysis deeply in body effect (VBS_back gate effect) to ID_VDS characteristics.

Setting up VBS on Parametric set on Parametric Analysis in Cadence of VBS [0.1, 0.55, 0.9], testing IDS on VDS domain as experiments shown above. It is known that in all the above experiments, we almost always try to bring this quantity to as low a level as possible, or equal to 0.

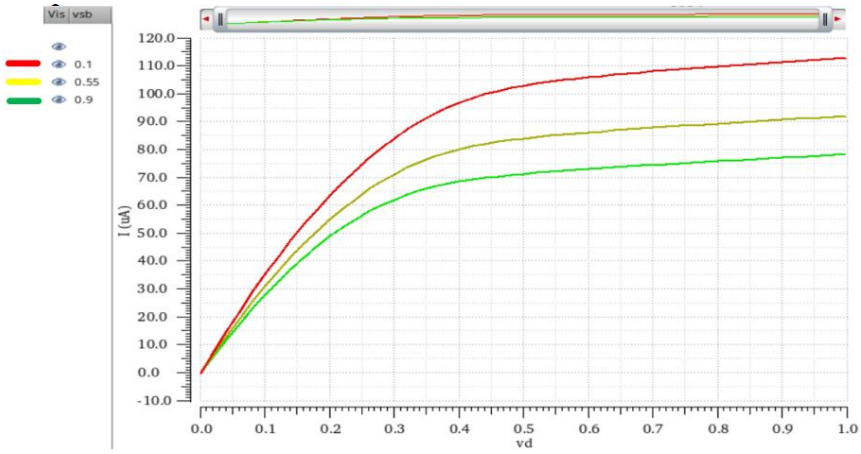


Figure 1.16 : $I_{DS} = f(V_{DS})$ *at* $V_{BS} = \{0.1, 0.55, 0.9\}V$

Thus, after getting the results of 3 signals at VBS values - 0.1, 0.55 and 0.9, we have the following comments about the slope of the ID signal on VDS:

- Slope gradually increases as the number of VBS decreases, meaning the value of VBS will immediately greatly affect trend of increasing the saturation of IDS.
- The value of VBS immediately affects the threshold voltage value of the voltage. When VBS increases, VT in NMOS also increases.
- The relationship between the quantities VBS, VDS and VGS is extremely tight, further emphasized by the behavior of the circuit.

Comment: As the body-to-source voltage (VBS) rises, the body effect raises the threshold voltage. As a result, the connection among IDS, VGS, and VDS is altered.

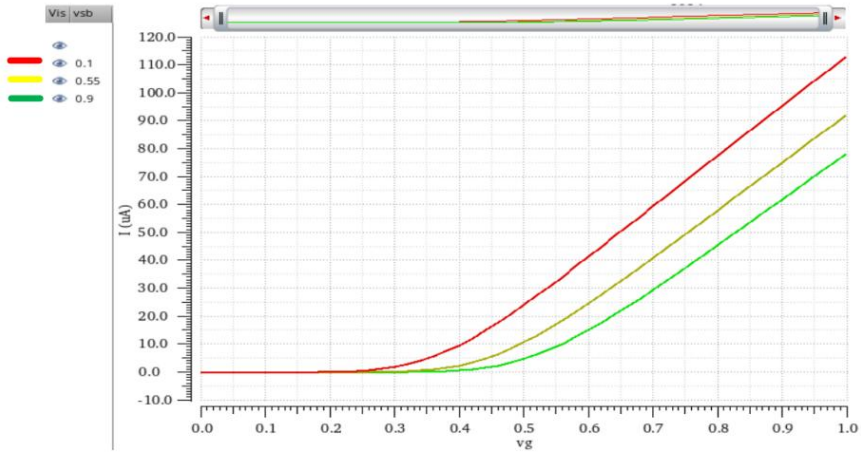


Figure 1.17 : $I_{DS} = f(V_{GS})$ *at* $V_{BS} = \{0.1, 0.55, 0.9\}V$

1.4 Experiment 4: Create layout for MOS devices. (PMOS and NMOS)

In order to layout an NMOS transistor, Cadence must first define the active region, then identify the source and drain diffusion zones, draw a polysilicon gate, connect contacts or vias, and use Layout Versus Schematic (LVS) and Design Rule Checking (DRC) to confirm the layouts. For manufacturing to be accurate and reliable, design criteria must be followed, and layout verification procedures are carried out to make sure the circuit design satisfies requirements. The FreePDK45 layout library is used to implement NMOS, the layout needs to be extremely tight by DRC and LVS.

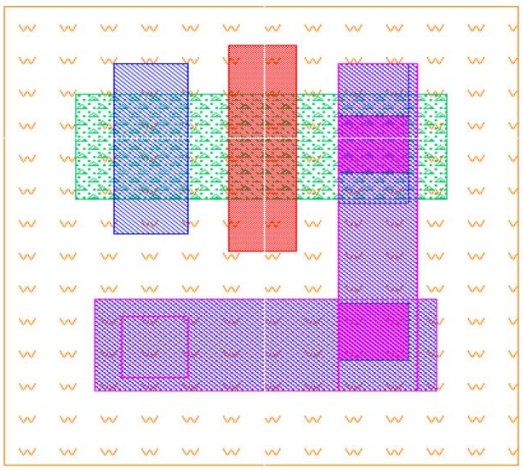


Figure 1.18 : *Layout of NMOS*

Setting up the layout for NMOS and PMOS right from the moment you create the library is extremely necessary to make the layout work for more complex circuits easier. Regarding the layout, we will need to set up a set of n-implants representing the Width of NMOS, in this case we have W = 90nm, so the size of n-implants in the layout must be represented at 0.9. In addition, for the n-implant to work, there must be an Active coating overlapping the n-implant. Here the NMOS will need to be

connected to ground to exhibit strong zero conduction characteristics. NMOS will operate only when located in the p-well region (orange layer), while the red line laid out in the image is the poly layer that separates the D and S terminals of NMOS. The width of this poly layer is also the specified NMOS L size, in this case 50nm.

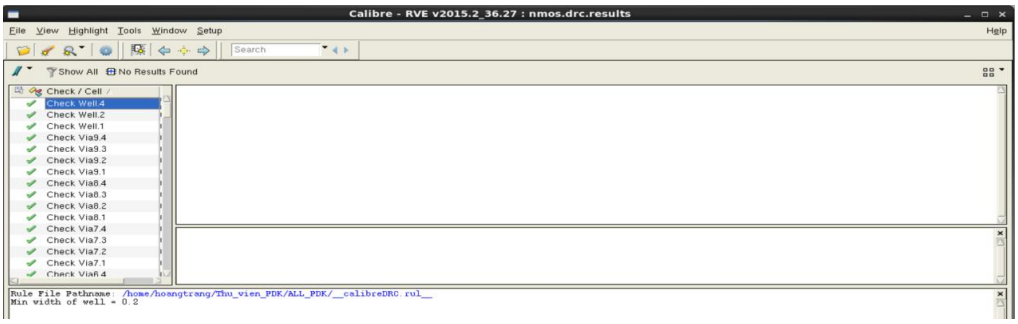


Figure 1.19 : *Design Rule Check of NMOS*

Check the DRC and LVS for the layout until there are no Errors on the DRC, and the wires and ports of the layout are shown to be similar to the Schematic.

Figure 1.20 : *Layout Versus Schematic of NMOS*

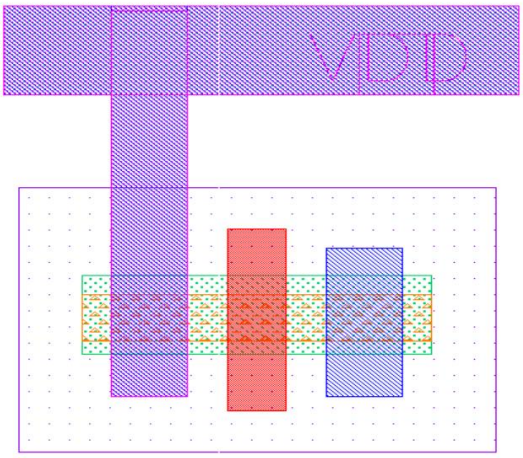


Figure 1.21 : *Layout of PMOS*

1.5 Experiment 5: Review MOS transistos's operations

The components of a MOSFET (Metal-Oxide-Semiconductor Field-Effect Transistor) are a gate electrode composed of metal or heavily doped polysilicon, a channel between the source and drain regions doped to form n-type or p-type areas, and a thin insulating layer of silicon dioxide (SiO_2) above the channel. The conductivity of the

channel is regulated by the gate when a voltage is applied, allowing current to flow between the source and drain. Gate-to-source (Cgs), gate-to-drain (Cgd), gate-to-body (Cgb), and drain-to-source (Cds) capacitances are examples of parasitic capacitances in a MOSFET. These result from the gate's physical layout, its overlap with the source and drain regions, the connections between other areas, and other factors. In high-frequency applications in particular, these parasitics can affect the device's speed, power consumption, and overall performance.

For capacitance, the impedance is 1/jcw. It is open and has no effect on the circuit because it is regarded as infinite at low frequencies. On the other hand, as the frequency rises, the circuit's capacitance acts like an impedance, which might alter the behavior of our transistor by restricting its speed. Therefore, when operating at high frequencies, the transistor has limitations.

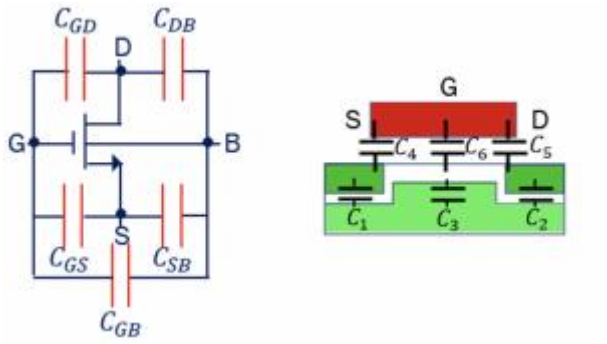


Figure 1.22 : *NMOS cross – section view*

Question: Which is the largest capacitance? Explain.

The circuit's highest capacitance in the saturation zone is **CGS**. To determine the input capacitance of the transistor gate, which is equal to Cin=CGS and operates in the saturation region. When compared to the gate-drain capacitance CGD, which is equal to WCov, CGS has a greater value.

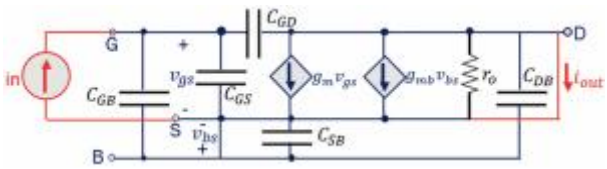


Figure 1.23 : *A high – frequency small – signal model*

In an NMOS transistor operating in the small-signal regime, the largest capacitance is typically the gate-to-channel capacitance (C_gc), which is often approximated as the gate-to-source capacitance (C_gs) in low-frequency or DC analysis. This capacitance is significant because it encompasses the overlap capacitance between the gate and the

source region, as well as the capacitance due to the gate control over the inversion layer (channel). When the NMOS transistor is in saturation, the gate-to-drain capacitance (C_gd) also contributes but is usually smaller due to the Miller effect being more prominent at high frequencies. Therefore, C_gs tends to be the dominant capacitance in small-signal analysis at low frequencies. The CGS can be calculated as:

$$C_{GS} = 2/3 * WLC_{OX} + WC_{OV}$$

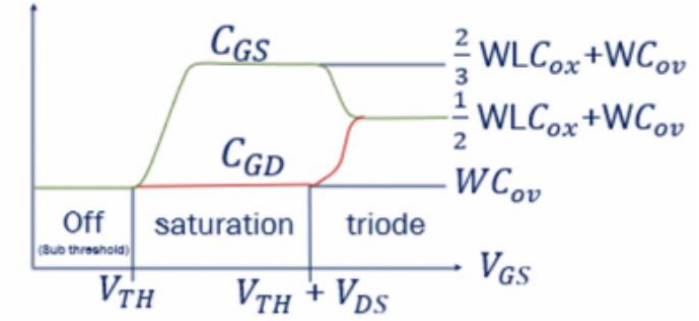


Figure 1.24 : *A high – frequency small – signal model*

Question: When the accuracy is important, explain what is LOV and write an expression of capacitance of COV.

For NMOS transistors to have precise threshold voltage control, less leakage, lower parasitic capacitance, and improved device performance, Length of Overlap (LOV) is an essential design parameter. To ensure full coverage even amid manufacturing differences, it entails the purposeful extension of the polysilicon gate over the source and drain regions. In sophisticated and compact semiconductor technologies, LOV plays a crucial role in guaranteeing dependable switching properties and uniform performance among various transistors.

The overlap capacitance COV in an NMOS transistor is a parasitic capacitance that occurs due to the physical overlap between the gate electrode and the source/drain regions. This capacitance can be expressed as:

$$C_{OV} = e_{ox}/t_{ox} \text{ or } C_{OV} = C_{ox} * (W * L_{OV})$$

This expression indicates that the overlap capacitance COV is directly proportional to both the width of the transistor and the length of the overlap. It contributes to the total gate capacitance and affects the transistor's switching speed and power consumption, making it an important factor in the design and optimization of NMOS transistors, especially in high-frequency and high-performance circuits.

Question: Write expression for Cgs, Cgd, Csb, Cdb.

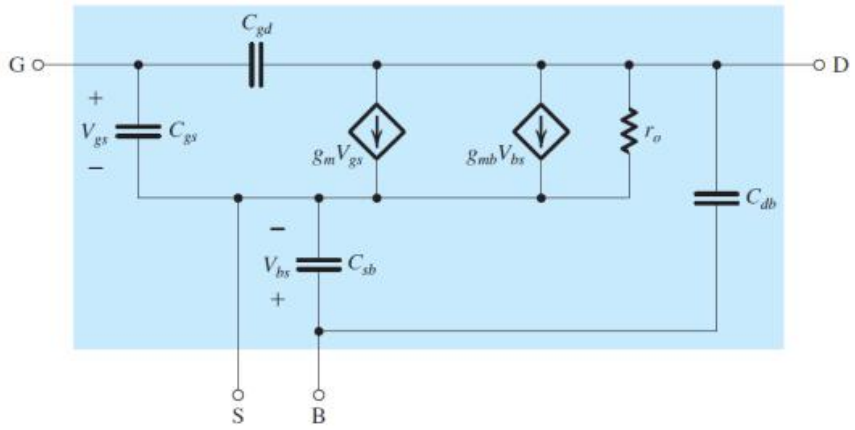


Figure 1.25 : *MOSFET high – frequency model*

Parasitic capacitances in a MOSFET, including CGS, CGD, CSB, and CDB , are inherent to the device structure and influence its behavior in various ways. CGS represents the capacitance between the gate and source terminals, impacting input impedance and high-frequency response. CGD affects voltage gain and stability due to the Miller effect, arising from the capacitance between the gate and drain terminals. Csb and Cdb denote capacitances between the source/drain terminals and the substrate, respectively, influencing biasing conditions, switching speed, and transient response. Understanding and accurately modeling these parasitic capacitances are essential for designing MOSFET-based circuits with optimal performance and reliability across different operating conditions and frequency ranges.

$$C_{GS} = 2/3 * WLC_{ox} + WL_{OV}C_{ox} \qquad C_{SB} = \frac{C_{sb0}}{\sqrt{((1 + |V_{SB}|/V_0)}}$$

$$C_{GD} = WL_{OV}C_{ox} \qquad C_{DB} = \frac{C_{db0}}{\sqrt{((1 + |V_{DB}|/V_0)}}$$

Table 1.1 : *Model parameters expression*

Laboratory 2: Digital Logic Circuit

2.1 Experiment 1: Logic gates using CMOS technology

2.1.1 Schematic

A	Y
0	1
1	0

Table 2.1 : *Truth table of inverter*

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Table 2.2 : *Truth table of* $NAND2$

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Table 2.3 : *Truth table of* $NOR2$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

Table 2.4 : *Truth table of* $EX - OR2$

A truth table is a tabular representation of a logic gate's output for every possible combination of inputs. When the input is 0 and the output is 1, the inverter's output is the logical NOT of the input. When both inputs are 1, a 2-input NAND gate outputs 0; otherwise, it outputs 1. On the other hand, a 2-input NOR gate outputs 0 otherwise and only outputs 1 when both inputs are 0. When two inputs are distinct (one is 0 and the other is 1), an exclusive OR gate, also known as a 2-input XOR gate, outputs 1; when the inputs are the same (both 0 or both 1), it outputs 0. For the purpose of understanding and developing digital circuits, these truth tables are essential.

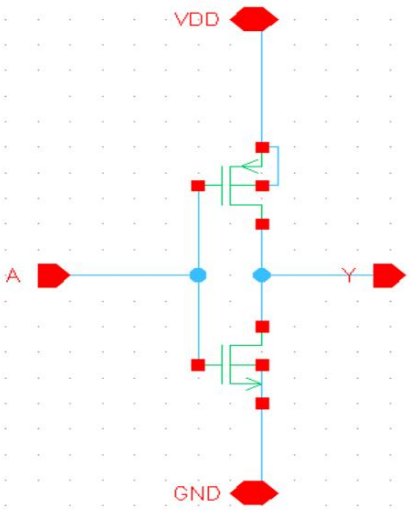


Figure 2.1 : *Schematic of INV*

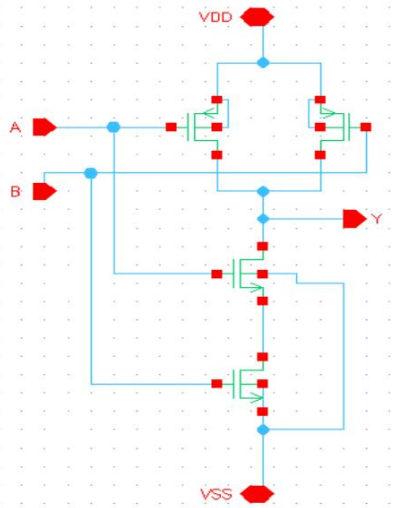


Figure 2.2 : *Schematic of NAND2*

To achieve the truth table output value for each gate, we need to understand the nature of strong 1 conduction of PMOS and strong 0 conduction of NMOS. Here, with the INV set up to connect both MOS as shown in Figure 2.1, we understand that when input A = 1, the circuit will be in pull down state, meaning it conducts 0 from NMOS and the output value will give the value of 0 (Y=0), vice versa. Similar but more complicated is the NAND2 gate, we can see that to achieve the truth table value for the 2 inputs we need to connect it in parallel for PMOS and in series for NMOS. This is explained that when A = 1 and B = 1, the PMOS will turn off, the circuit enters the PDN (pull down) excited state, Y will conduct 0 through series NMOS, this is also reasonable when 1 NAND 1 = 0.

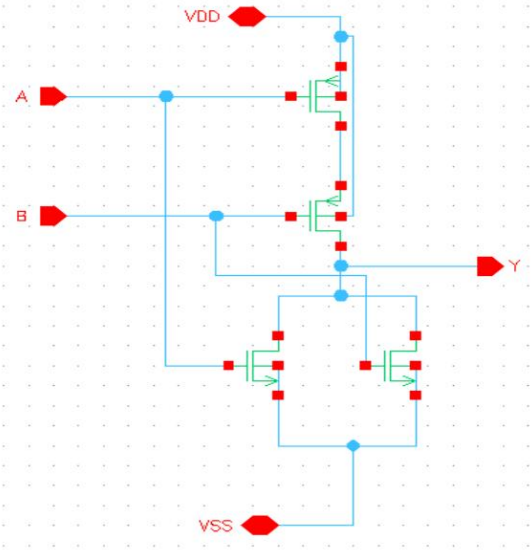


Figure 2.3 : *Schematic of NOR2*

With NOR gate, we will see that the behavior here is almost the opposite of NAND2. Then, the PUN (pull up) state of the circuit is only turned on if and only when both A and B are = 0, this will lead to PMOS A and B both conducting, Y leading the output state is 1. More is explained when 0 NOR 0 = 1. If one of A and B is not set to 0, the circuit will naturally return to PDN state, where any NMOS will lead 0 to output Y.

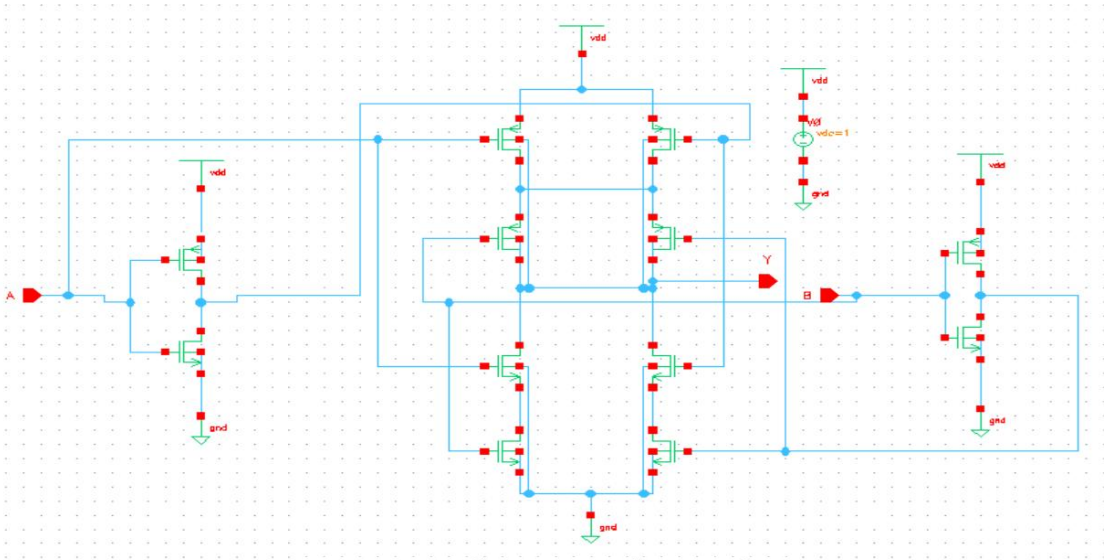


Figure 2.4 : *Schematic of* $EX - OR2$

With EX-OR2 pin, it is a more complicated matter, when the circuit reaches the PUN (pull up) state only when both A and B are equal. For example, when A = 1 and B also = 1, the inverter of both A and B will reach 0, disabling the NMOS conduction in the lower branch and letting the PMOS upper branch conduct Y to 1, combined with the external INV. output to achieve the expression A XOR B and the output reaches 0. This can be clearly seen on the truth table when 1 each other, the circuit will fall into PDN (pull down) state, leading to Y = 0 and going through the inverter to return the final output value = 1.

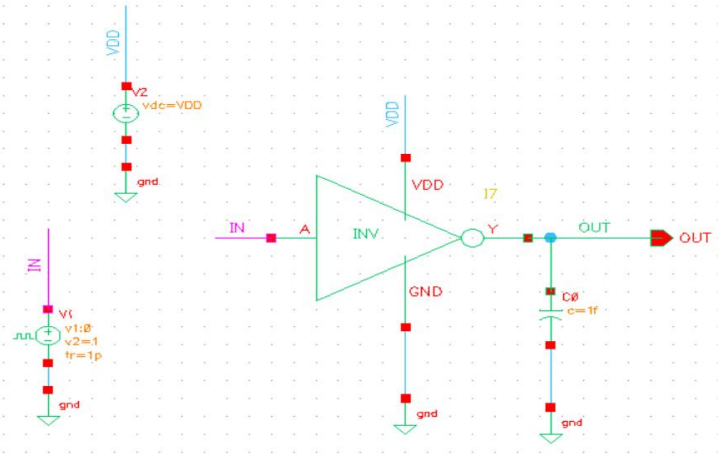


Figure 2.5 : *Testbench of INV*

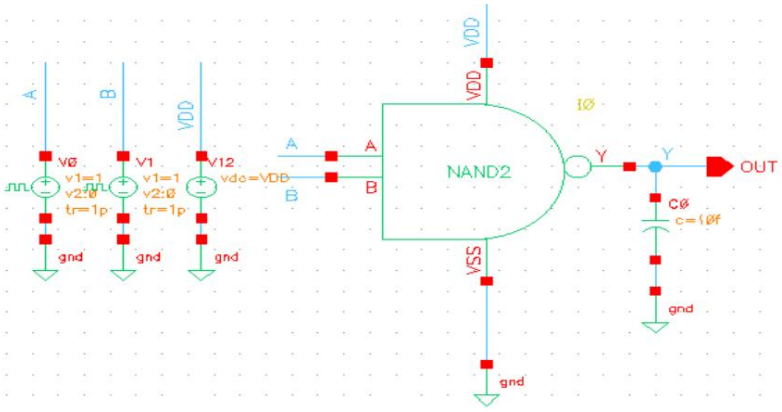


Figure 2.6 : *Testbench of NAND2*

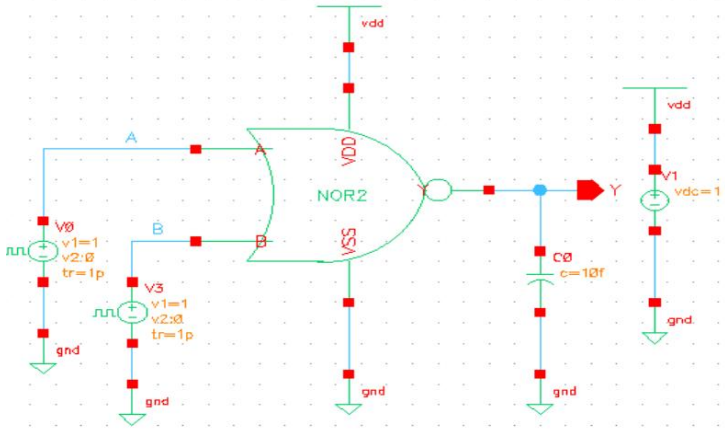


Figure 2.7 : *Testbench of NOR2*

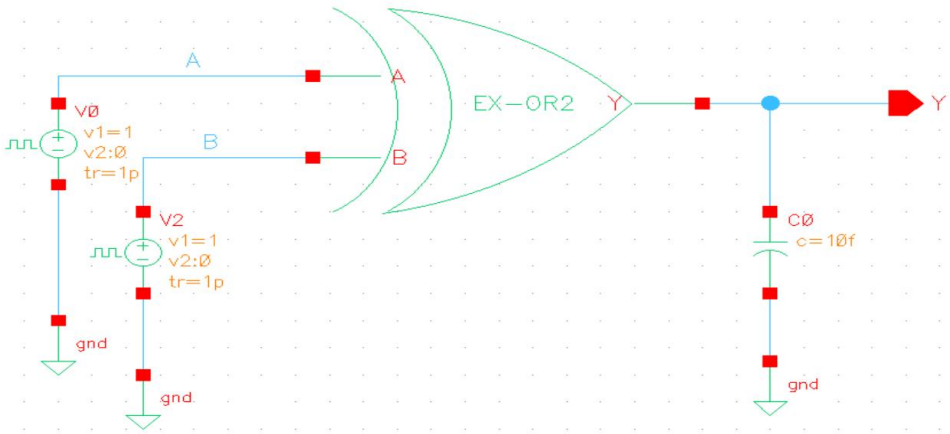


Figure 2.8 : *Testbench of EX − OR2*

Set up testbench and simulate them via DC and AC analysis to prove that the theoretical performance values reach the same magnitude in reality.

2.1.2 DC Analysis Simulation

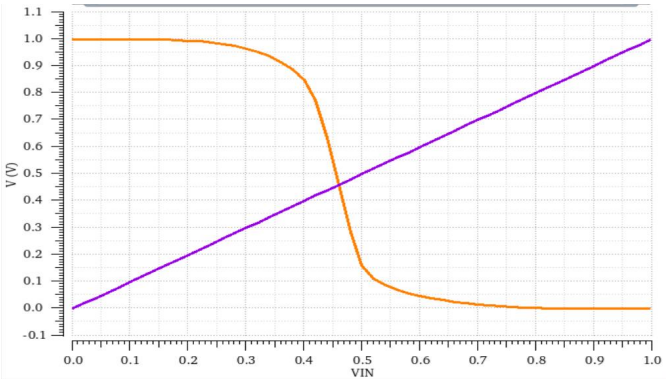


Figure 2.9 : V_{out} *and* V_{in} *DC analysis of INV*

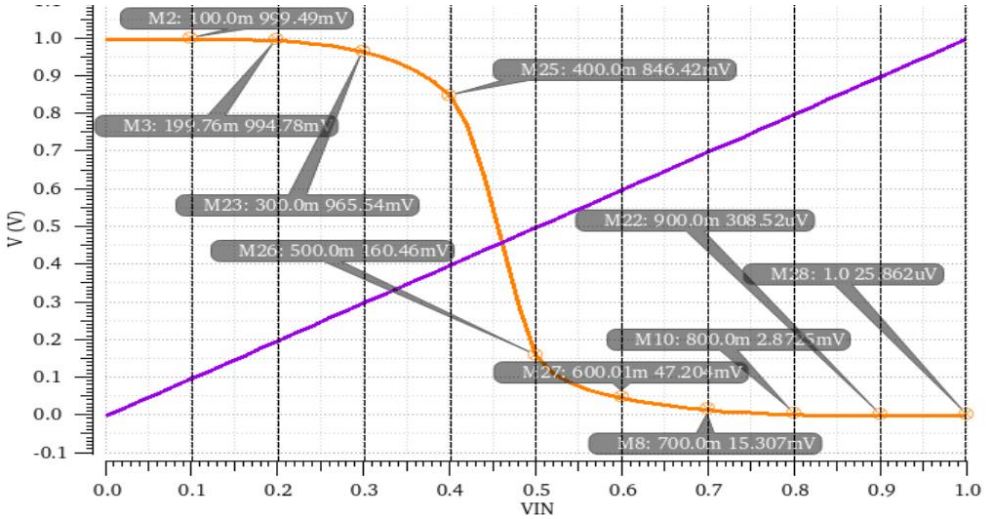


Figure 2.10 : *Voltage Transfer Curve of CMOS INV*

V_{IN} (V)	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
V_{OUT} (V)	1	0.999	0.995	0.967	0.852	0.175	0.048	0.015	0.003	0.001	0

Table 2.5 : *Output voltage values at various values of* V_{in} *with* $0.1V$ *step*

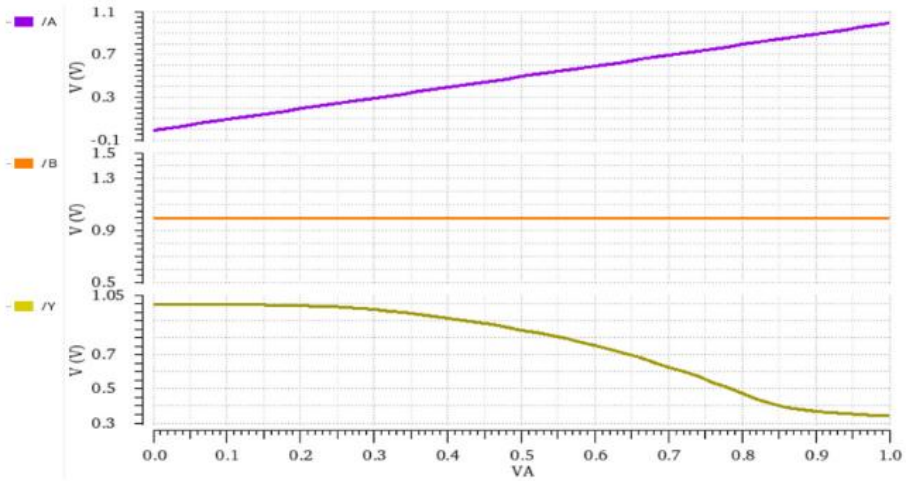


Figure 2.11 : A , B, & Y *DC analysis of EXOR2 when sweeping* V_A

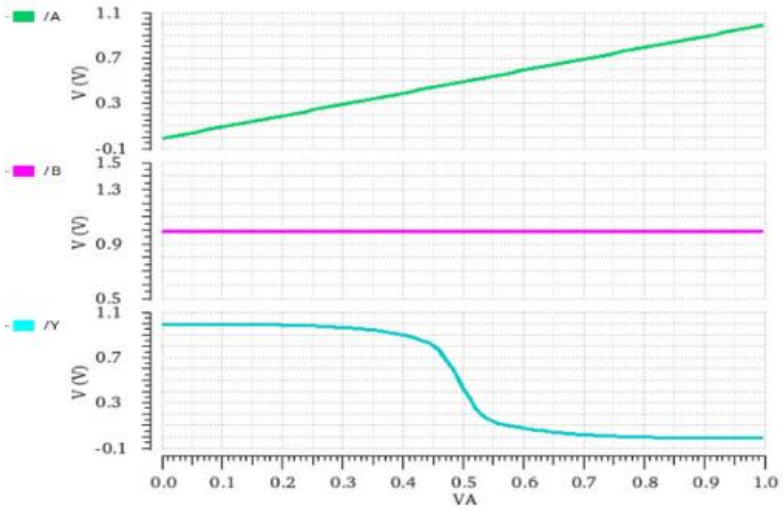


Figure 2.12 : *A , B, & Y DC analysis of NAND2 when sweeping* V_A

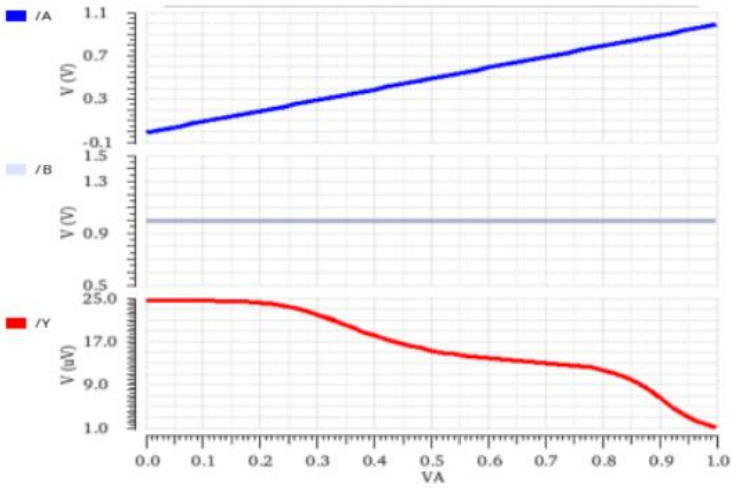


Figure 2.13 : *A , B, & Y DC analysis of NOR2 when sweeping* V_A

2.1.3 Transient

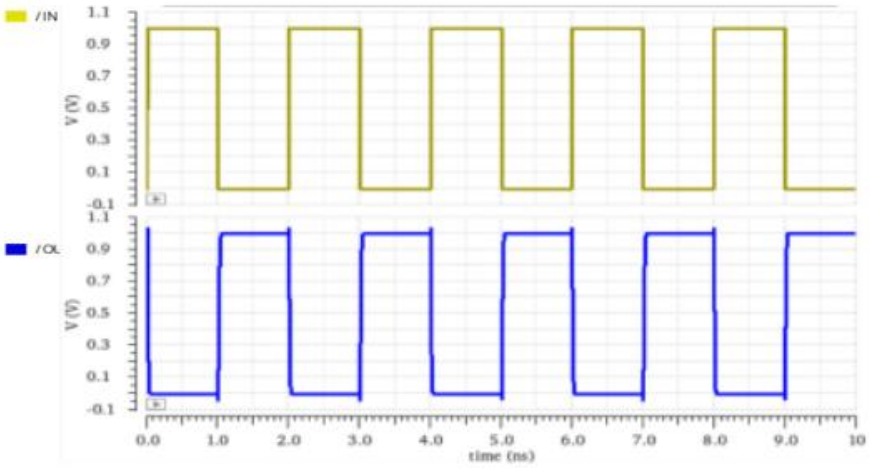


Figure 2.14 : *Transient of INV*

When simulating the transient for the inverter, we see opposite signals from Vin and Vout, this is achieving the initial desire set out in the truth table.

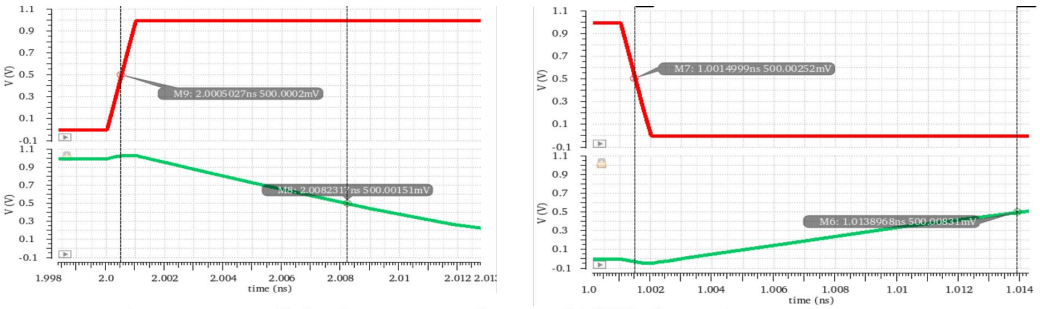


Figure 2.15 : *Calculate* t_{rise} *and* t_{fall} *of INV through simulation*

Through Cadence's ADEXL Calculator tool, execute the riseTime, fallTime and delay functions to measure the rise time of the Vout signal faster. In addition, to check the results of the following functions, we can calculate directly on the output voltage signal as follows. With tfall, we compare Vin at VDD/2 until Vout = Vout/2 as shown in *Figure 2.15*. Continue doing so to get each measurement table in INV, NAND2, NOR2 and XOR2.

Parameter	Measure
t_{rise} – *Rising time* (10% – 90%) (*s*)	13.86*p*
t_{fall} – *Falling time* (90% – 10%) (*s*)	12.12*p*
t_{pdr} – *Rising propagation delay* (90% – 50%) (*s*)	12.8*p*
t_{pdf} – *Falling propagation delay* (10% – 50%) (*s*)	8.171*p*
t_{pd} – *Average propagation delay* (50% – 50%) (*s*)	12.4*p*
Power consumption (*W*)	774.8*n*

Table 2.6 : *Measurement results of INV*

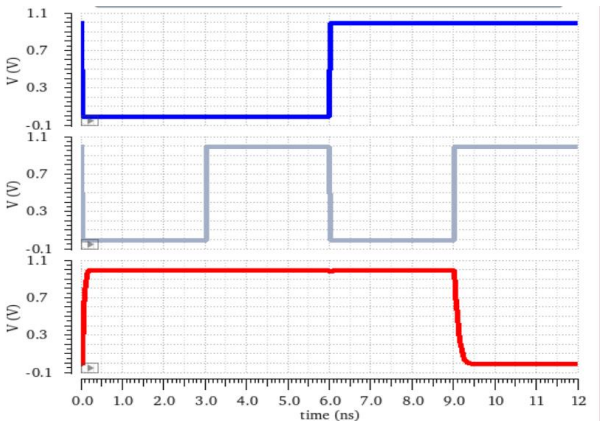


Figure 2.15 : *Transient of NAND2*

Parameter	Measure
t_{rise} – *Rising time* (10% – 90%) (s)	148.9p
t_{fall} – *Falling time* (90% – 10%) (s)	106.4p
t_{pdr} – *Rising propagation delay* (90% – 50%) (s)	37.96p
t_{pdf} – *Falling propagation delay* (10% – 50%) (s)	37.96p
t_{pd} – *Average propagation delay* (50% – 50%) (s)	87.33p
Power consumption (W)	1,292p

Table 2.7 : *Measurement results of NAND2*

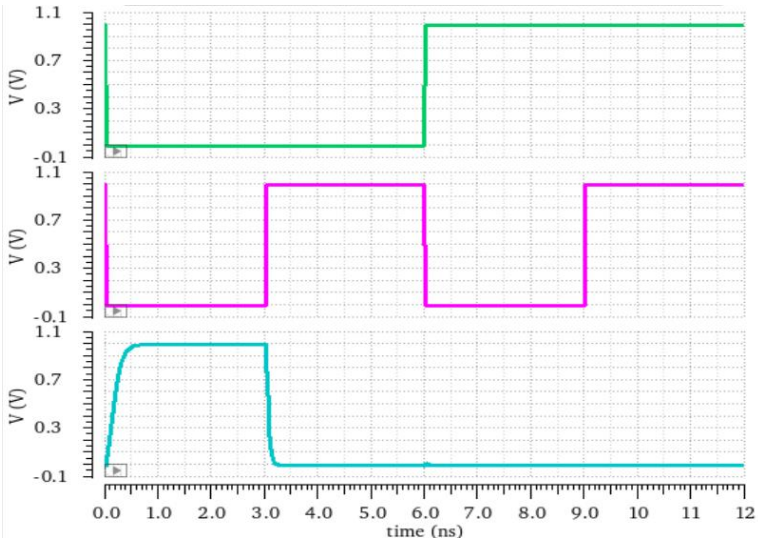


Figure 2.16 : *Transient of NOR2*

Parameter	Measure
t_{rise} – *Rising time* (10% – 90%) (s)	268.1p
t_{fall} – *Falling time* (90% – 10%) (s)	31.2p
t_{pdr} – *Rising propagation delay* (90% – 50%) (s)	11.03p
t_{pdf} – *Falling propagation delay* (10% – 50%) (s)	11.03p
t_{pd} – *Average propagation delay* (50% – 50%) (s)	6,154p
Power consumption (W)	1,373p

Table 2.8 : *Measurement results of NOR2*

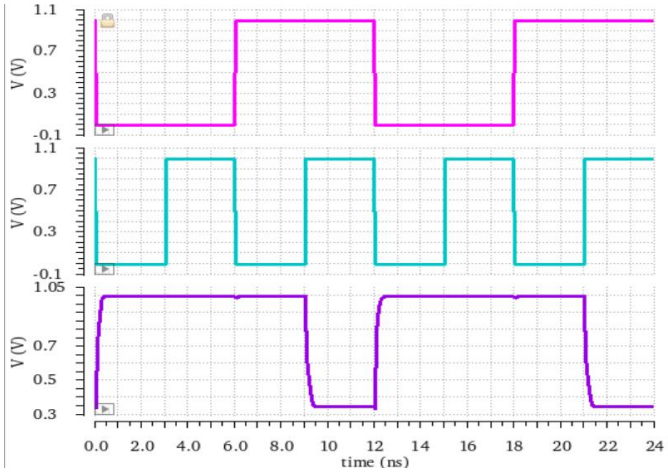


Figure 2.17 : *Transient of* $EX - OR2$

Parameter	Measure
t_{rise} – *Rising time* (10% – 90%) (*s*)	194.6*p*
t_{fall} – *Falling time* (90% – 10%) (*s*)	336.3*p*
t_{pdr} – *Rising propagation delay* (90% – 50%) (*s*)	3,074*p*
t_{pdf} – *Falling propagation delay* (10% – 50%) (*s*)	3,074*p*
t_{pd} – *Average propagation delay* (50% – 50%) (*s*)	30.1*p*
Power consumption (*W*)	5,509*p*

Table 2.9 : *Measurement results of* $EX - OR2$

2.1.4 Layout

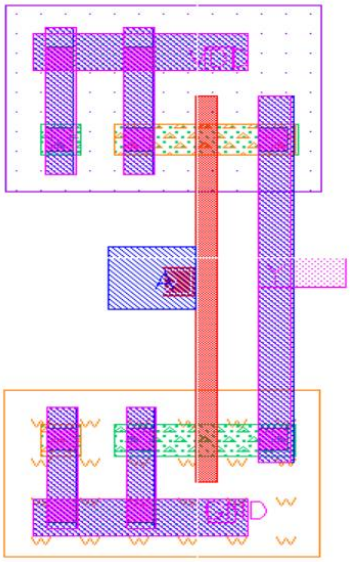


Figure 2.18 : *Layout of* INV

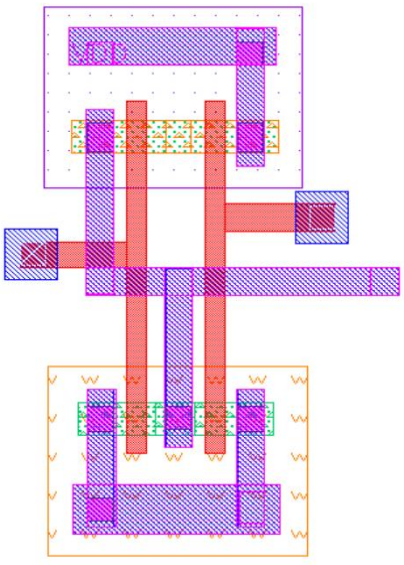


Figure 2.19 : *Layout of* $NOR2$

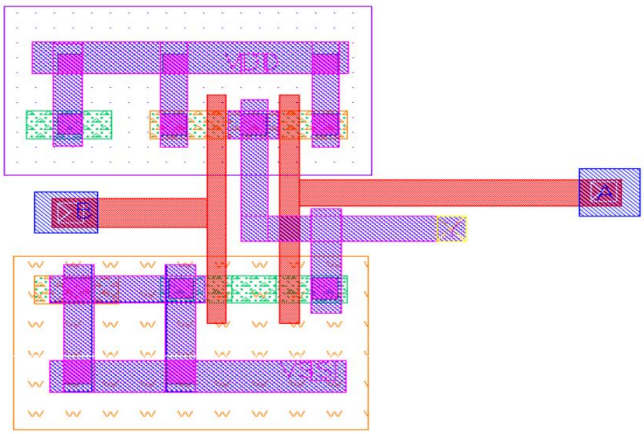


Figure 2.20 : *Layout of* $NAND2$

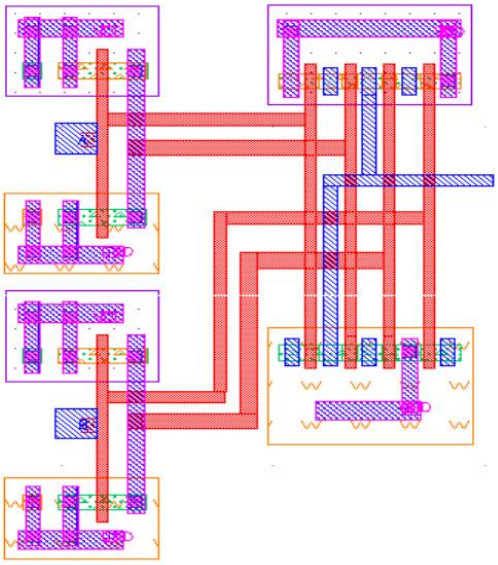


Figure 2.21 : *Layout of* $EX-OR2$

2.2 Experiment 2: Implement basic combination component

2.2.1 Truth table and Schematic of 2-to-1 MUX compound gate

D1	D0	S	Y
0	X	0	0
0	X	0	0
1	X	0	1
1	X	0	1
X	0	1	0
X	1	1	1
X	0	1	0
X	1	1	1

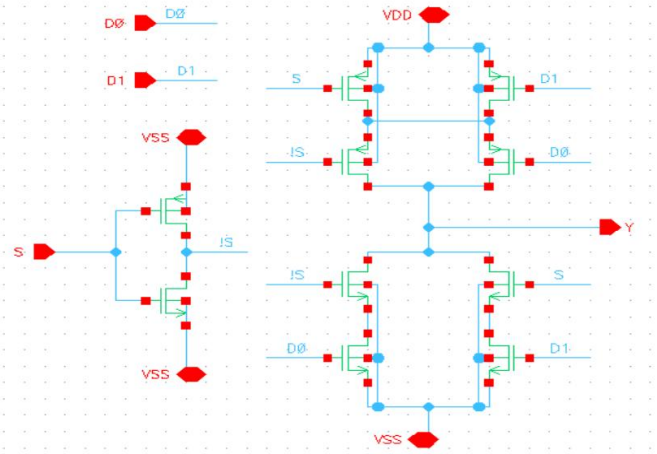


Figure 2.22 : Truth table & Schematic of 2 – to – 1 channel MUX using compound gate

The 2-to-1 MUX circuit is represented by its dependence on the Select pin to determine the Y output. More specifically, when the Select pin is 1, the output value of Y will be taken from D1 regardless of D0. and vice versa.

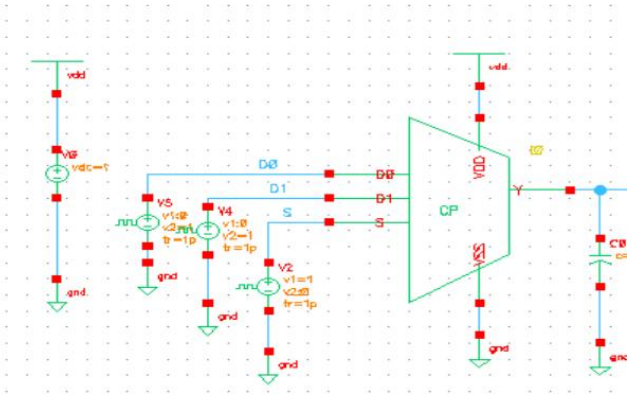


Figure 2.23 : Testbench of 2 – to – 1 channel MUX using compound gate

2.2.2 DC Analysis Simulation

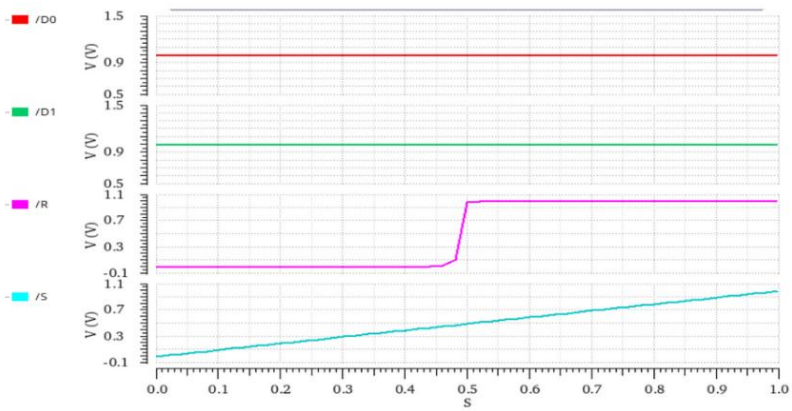


Figure 2.24 : A , B, & Y DC analysis of compound gate when sweeping S

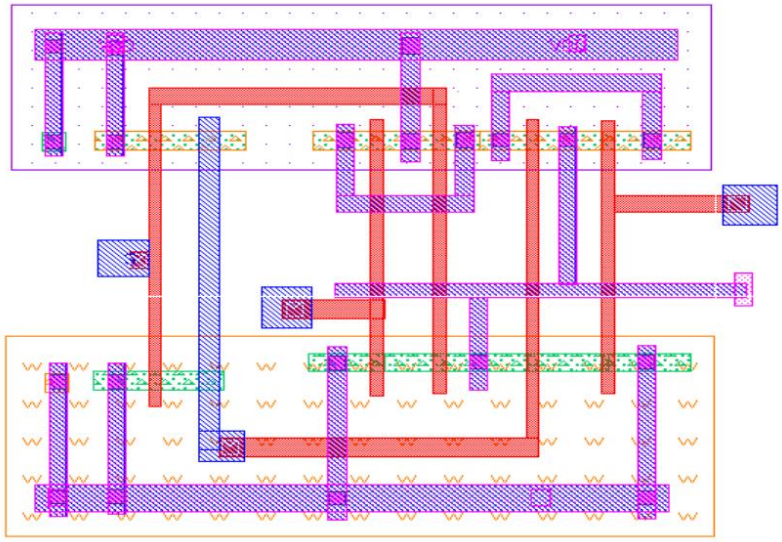


Figure 2.25 : *Layout of* $2-to-1$ *channel MUX using compound gate*

2.2.3 Transient Analysis Simulation

After running Transient, we will receive eight states corresponding to each 1ns in the 8ns time domain. In the first 4ns, we see that the Select pin is set at the negative edge, then it is known that Y owns the entire value of D1, this is shown on the truth table. In the next 4ns continuation, we can witness the opposite when Y takes the entire conductance value of D0.

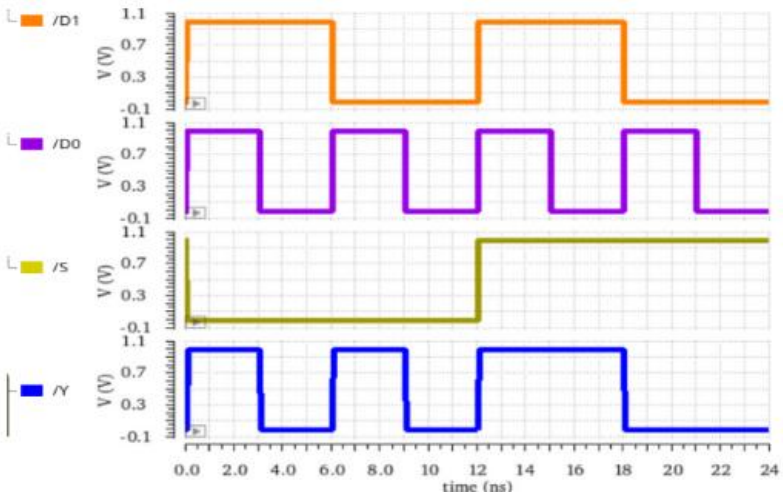


Figure 2.26 : *Transient of MUX compound gate*

2.3 Experiment 3: Implement a simple storage Modified TSPC D flip-flop

2.3.1 Truth table and Schematic of Modified TSPC D flip-flop

A sequential logic circuit that, upon the rising or falling edge of a single clock signal (CLK), records and captures a data input (D). The clock (CLK), data input (D), and

output (Q) are commonly included in the truth table of a TSPC D flip-flop. The output Q takes the value of the input D when the clock transitions (it might be a rising or falling edge, depending on the design); if not, Q stays in its prior state.

NMOS and PMOS transistors are stacked in a Modified TSPC D flip-flop circuit to provide two or more stages of dynamic logic. While later stages employ additional transistors to transport and store the data, the initial stage typically uses an NMOS transistor controlled by the clock to sample the data input. Optimizing the transistor configuration in TSPC flip-flops is frequently necessary to save power consumption, minimize propagation latency, or enhance noise margins. This guarantees reliable operation in low-power, high-speed digital circuitry.

Preset	D	CLK	Q
0	0	1	1
0	1	0	1
0	1	1	1
0	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1
1	0	0	0

Table 2.10 : *Truth table of D flip – flop*

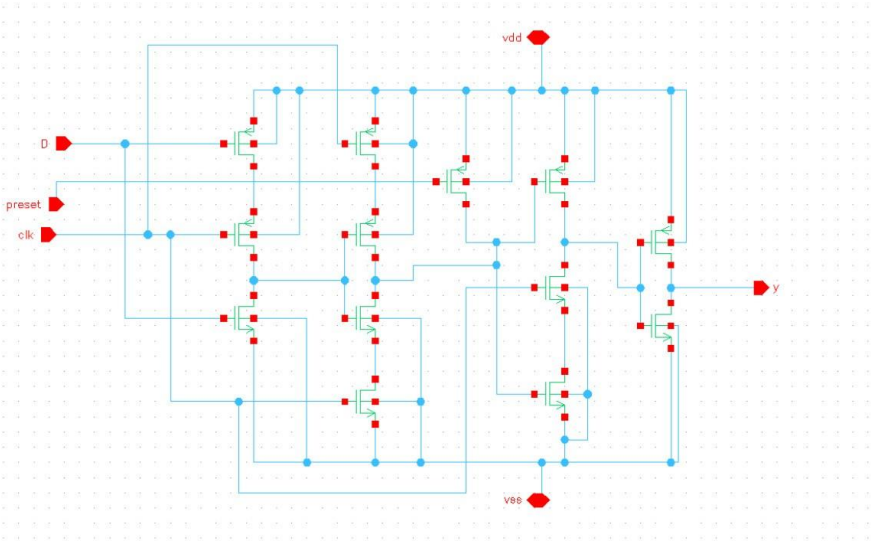


Figure 2.27 : *Schematic of TSPC D flip – flop*

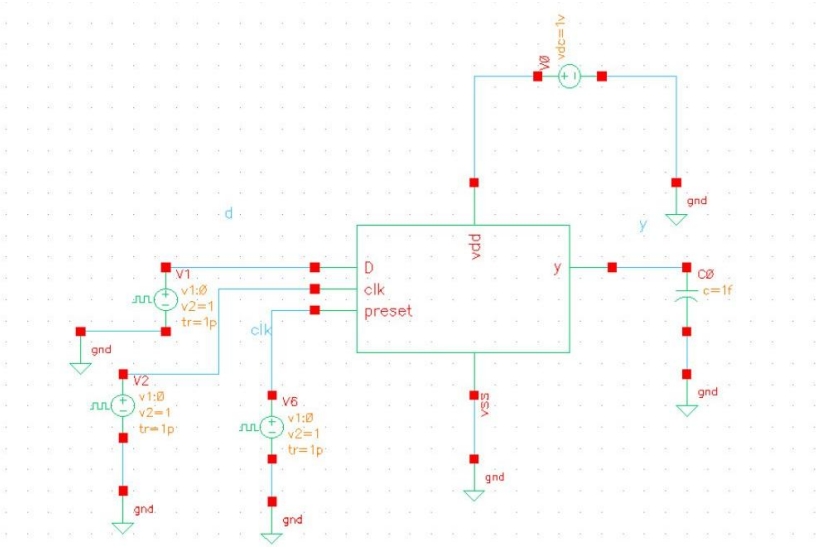


Figure 2.28 : *Testbench of TSPC D flip – flop*

Entails setting up a test environment to assess the functioning and performance of the flip-flop. In order to test various scenarios, the testbench generates the required input signals, which include a data input signal (D) that changes state at predefined times and a clock signal (CLK) with a given frequency and duty cycle. The testbench should also have output probes to watch the flip-flop's output (Q) and the proper power supply connections (VDD and GND). In order to record important parameters like setup time, hold time, propagation delay, and power consumption, the testbench usually has measuring components. The behavior of the TSPC D flip-flop can be examined under different circumstances by doing transient simulations, ensuring that it functions as intended and satisfies design requirements.

2.3.2 Transient Analysis Simulation

A True Single-Phase Clock (TSPC) D flip-flop's dynamic behavior as it reacts to input signals is assessed during the transient analysis simulation process using a tool such as Cadence. The time-dependent features of the flip-flop's operation are captured in this simulation by concentrating on how it switches between states in response to shifting clock and data inputs. A data input signal that varies at predetermined intervals is applied during the simulation, along with a clock signal with a given frequency and duty cycle. The transient response is captured and examined, along with any potential glitches or metast ability problems, setup and hold periods, and propagation delays. This aids in confirming that the TSPC D flip-flop functions

appropriately, capturing and holding data in accordance with clock transitions and operating consistently under various circumstances.

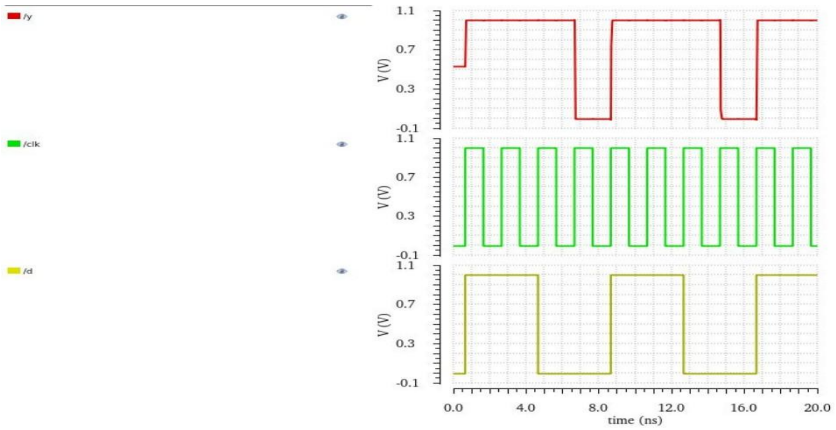


Figure 2.29 : *Transient of TSPC D flip – flop*

For a True Single-Phase Clock (TSPC) D flip-flop, the setup time (t setup) and hold time (t hold) are crucial timing factors. The setup time is the bare minimum of time before the clock edge that the data input (D) needs to be maintained steady in order for the flip-flop to sample it accurately. On the other hand, the hold time is the smallest amount of time that follows the clock edge that the data input needs to stay steady in order to be recorded accurately. Correct TSPC and thold times are crucial for the reliable operation of a TSPC D flip-flop. Incorrect tscp and thold times can result in incorrect data being latched, which could cause timing mistakes in the digital circuit.

Parameter	**Measure**
t_{pdr} – *Rising propagation delay* (90% – 50%) (s)	$51.2p$
t_{pdf} – *Falling propagation delay* (10% – 50%) (s)	$619.1p$
t_{pd} – *Average propagation delay* (50% – 50%) (s)	$335.79p$
Power consumption (W)	$19.9n$
t_{hold}	$0.2n$
t_{setup}	$0.2n$

Table 2.11 : *Delay time measurement of D flip – flop*

In order to determine the minimum stable periods for the data input with respect to the clock edge, a thorough timing study through simulation is usually required for accurate identification of these parameters. This involves testing several input situations and clock circumstances.

Laboratory 3: Combinational and Sequential Circuit

3.1 Experiment 1: Design a combinational circuit - a 1-bit Full Adder

3.1.1 Truth table, schematic and symbol for transient simulation

Our goal in the lab is to clarify the complexities of one-bit full adder circuits by examining various topologies that are intended to reduce transistor consumption in design, beginning with the traditional schematic. Our goal in exploring these options is to find a solution that balances simplicity and efficiency. Furthermore, we consider important aspects like power consumption and delay limitations in addition to just circuitry. Our goal is to offer a comprehensive approach to one-bit full adder architecture that optimizes performance without sacrificing important features. In addition, we are eager to provide methods for figuring out which clock speeds work best for these designs so that efficiency and functionality in the world of digital circuits are seamlessly integrated.

A	B	CIN	COUT	SUM
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

Table 3.1 : *Truth table of* 28*T Full Adder*

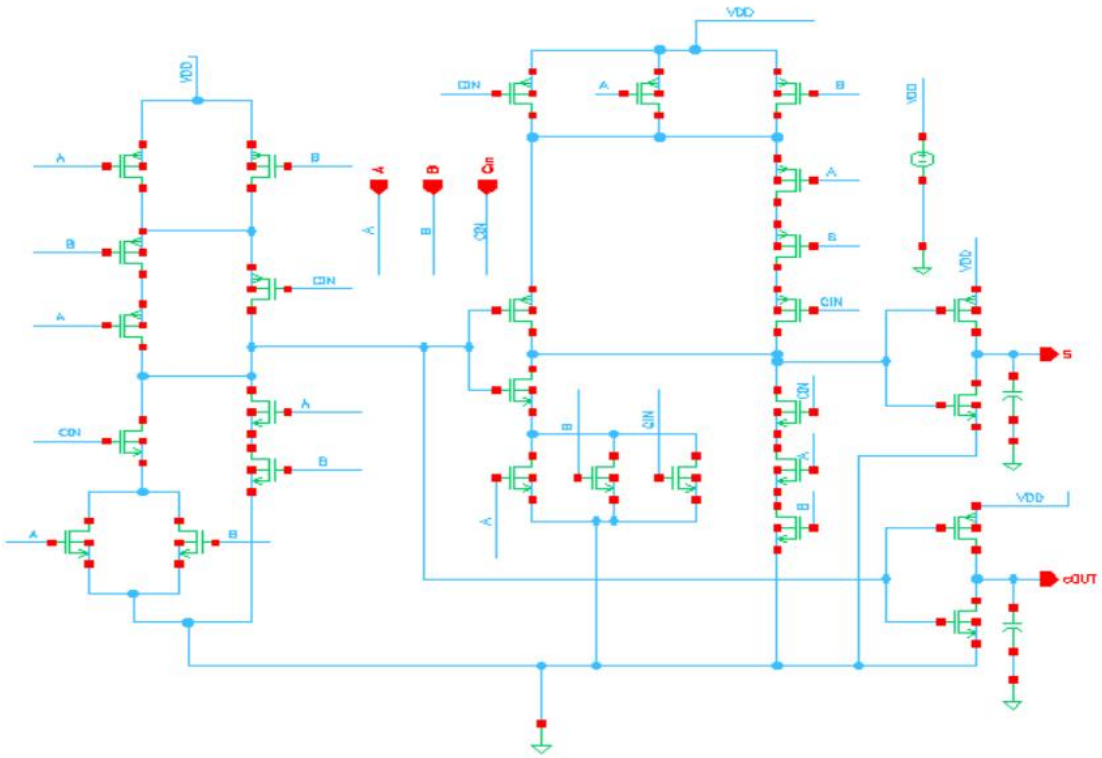


Figure 3.1 : *Schematic of 28T Full Adder*

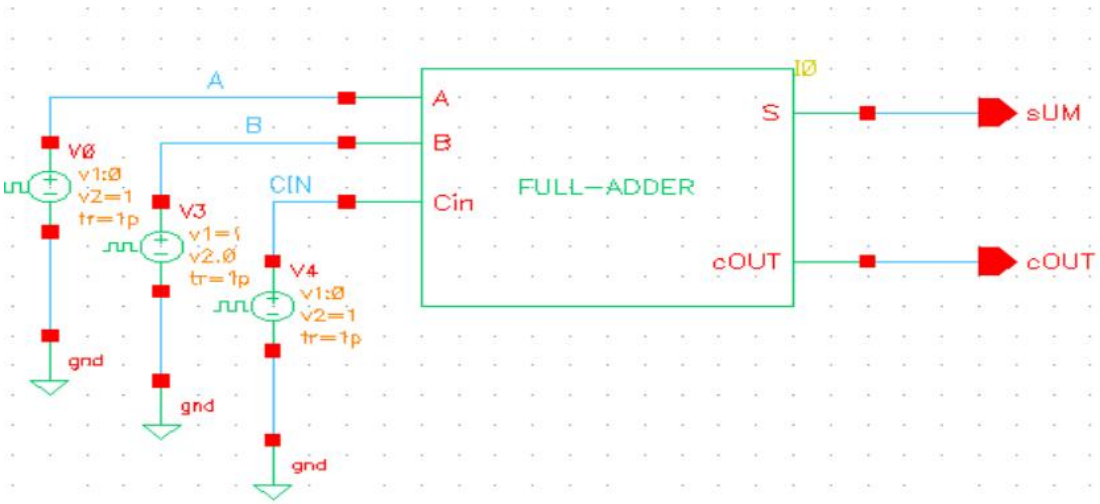


Figure 3.2 : *Testbench of 28T Full Adder*

3.1.2 Transient Analysis Simulation

A 28T (Twenty-Eight Transistor) Full Adder involves dynamic modeling and analysis of the adder's behavior during operation. This full adder architecture typically consists of multiple stages of logic gates, including XOR and AND gates, implemented using CMOS (Complementary Metal-Oxide-Semiconductor) technology. In the transient analysis simulation, various input scenarios representing different binary inputs (A, B, and carry-in) are applied to the full adder, and the transient response of the output signals (sum and carry-out) is observed over time. The simulation helps validate the

correctness and functionality of the full adder design, assessing its performance metrics such as propagation delay, power consumption, and signal integrity. Additionally, transient simulations allow designers to optimize the full adder's architecture and transistor sizing for speed, area efficiency, and power consumption, ensuring it meets the desired specifications for arithmetic and logic operations in digital circuits.

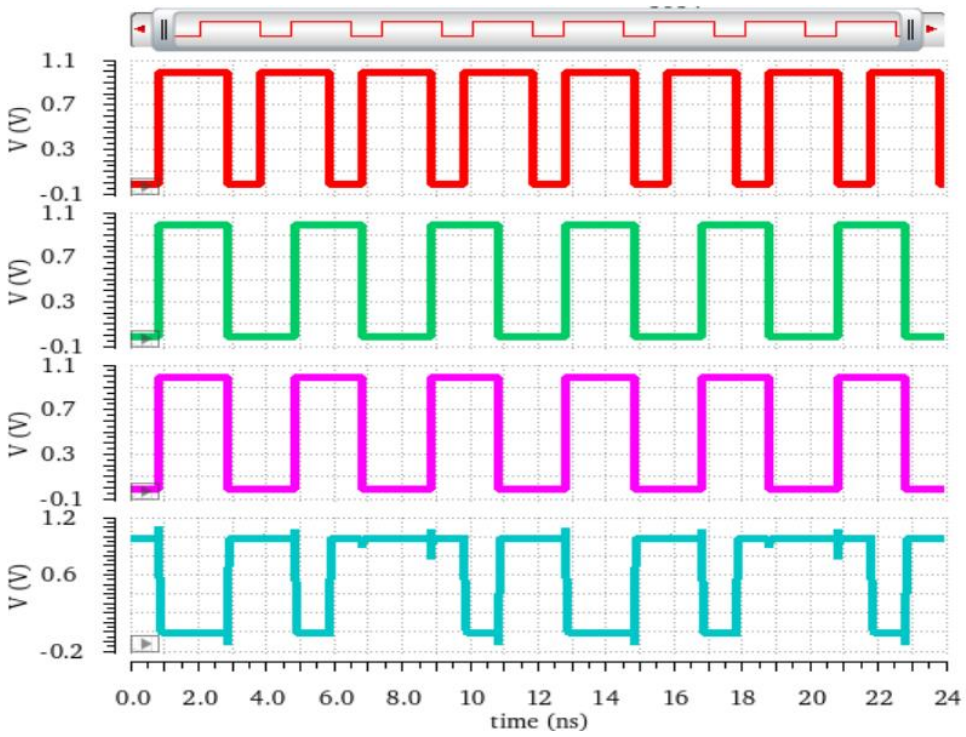


Figure 3.3 : *Transient of* 28*T Full Adder*

Question

Due to the topology shown in Figure 3, why do people implement the PDN of the first stage using equation Cout = (A + B)Cin + AB **instead of** Cout = AB + ACin + BCin?

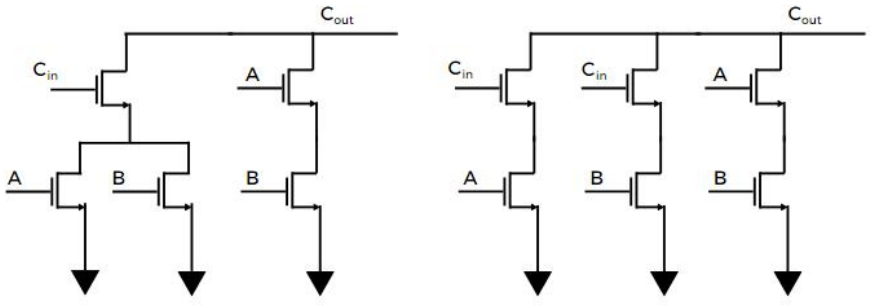


Figure 3.4 : *The PDN using* $C_{out} = (A+B) * C_{in} + AB$ *instead of* $C_{out} = AB + AC_{in} + BC_{in}$

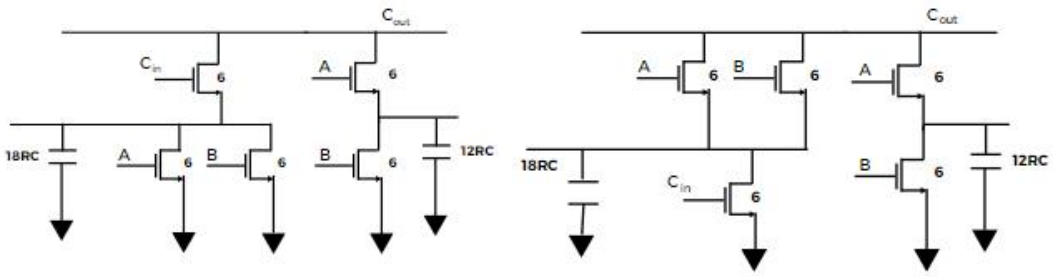


Figure 3.5 : C_{in} *connects to* C_{out} $(left)$ *and* C_{in} *connects to ground* $(right)$

The illustrated RC circuits offer a perceptive examination, highlighting the key distinctions between cases 1 and 2. Case 1 (left) highlights a particular rate of change in the circuit's response with the branch C(A+B) displaying a time constant equal to 2.5RC. However, for the identical branch, example 2 (right) shows a different time constant of 4RC, indicating a different dynamic in its behavior. Relocating the two parallel A+B branches to the output is an important decision since it introduces differences in the overall time constants of the branches that are next to it, with the AB branch being affected the most. The AB branch's total output capacitance in case 1 is 2C, whereas in case 2, it is 3C. The circuit's overall time constant is directly impacted by this change in the output total capacitance, which sheds light on the dynamics and performance differences between the two scenarios.

3.2 Experiment 2: Design a sequential circuit

3.2.1 Schematic of PRBS (pseudo-random binary sequence)

A Pseudo-Random Binary Sequence (PRBS) generator schematic comprises a feedback shift register configuration, consisting of flip-flops connected in series. These flip-flops store binary values that are shifted with each clock pulse. Feedback logic, typically implemented with XOR gates, feeds the output of certain flip-flops back to the input of others, creating a feedback loop. This arrangement generates a pseudo-random sequence at the output, driven by a clock signal controlling the shifting operation. The resulting sequence appears random but follows a predetermined pattern determined by the feedback configuration, making it useful in various applications such as testing and communication systems.

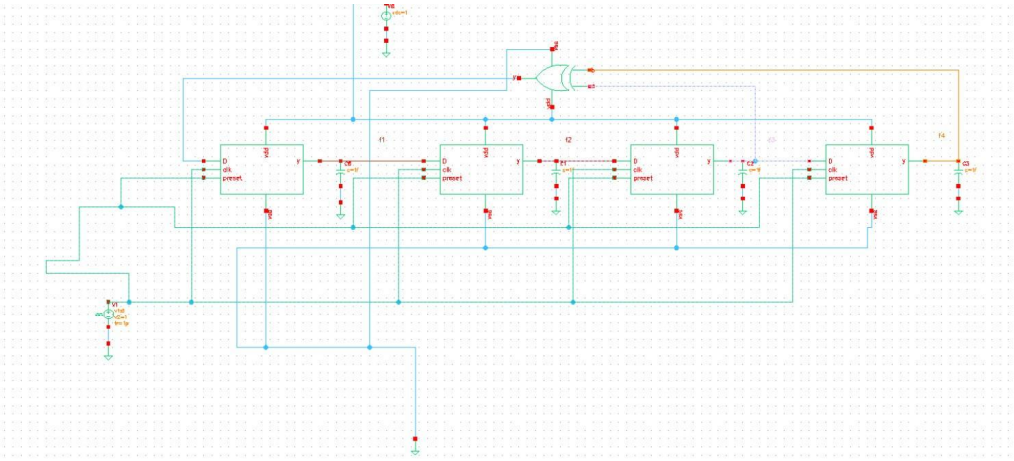


Figure 3.6 : *PRBS schematic*

3.2.2 Transient Analysis Simulation of PRBS (pseudo-random binary sequence)

Our primary focus in this test bench is the creation of the ADC module, which is specifically designed for easy and effective Verilog testing, as explained in the appendix. It skillfully converts analog signals into digital equivalents by imitating the operation of an analog-to-digital converter (ADC). Interestingly, the use of a shared pulse source simplifies period control and adds another level of ease to the implementation procedure as a whole.

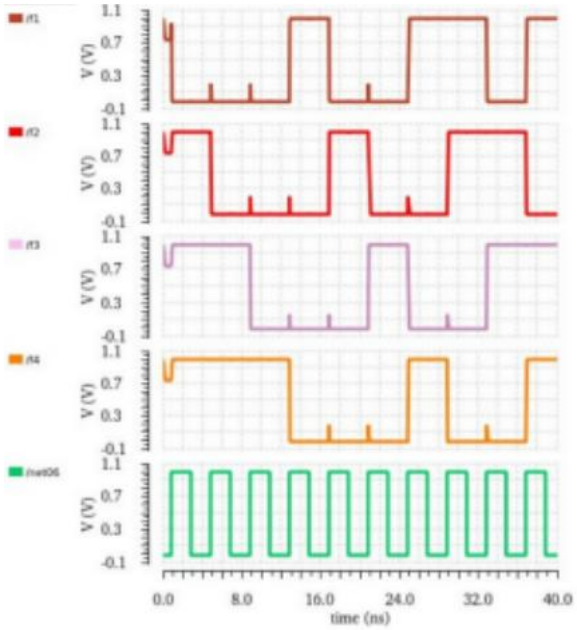


Figure 3.7 : *Transient of PRBS*

Laboratory 4: Arithmetic logic unit (ALU) & register file

4.1 Experiment 1: Design an ALU performing eight functions with 4-bits Flag.

4.1.1 Truth table, schematic and symbol for transient simulation

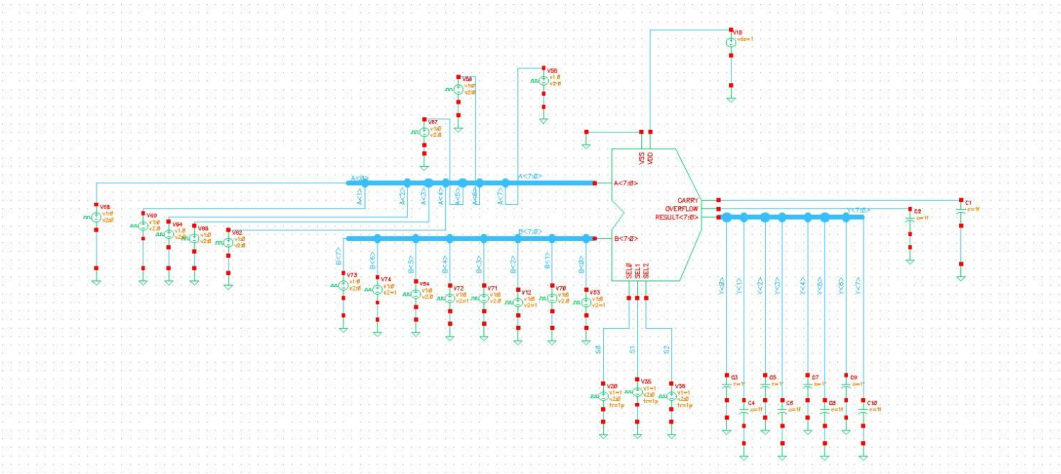


Figure 4.1 : *The schematic of ALU*

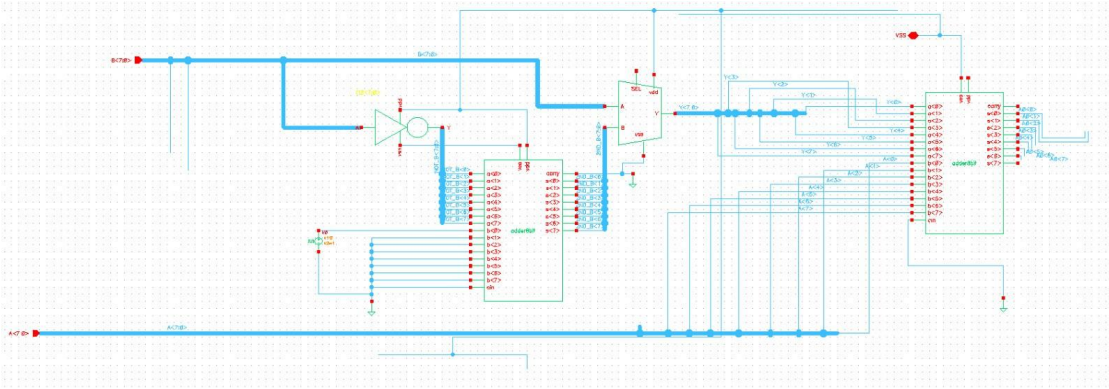


Figure 4.2 : *The FullADDER*

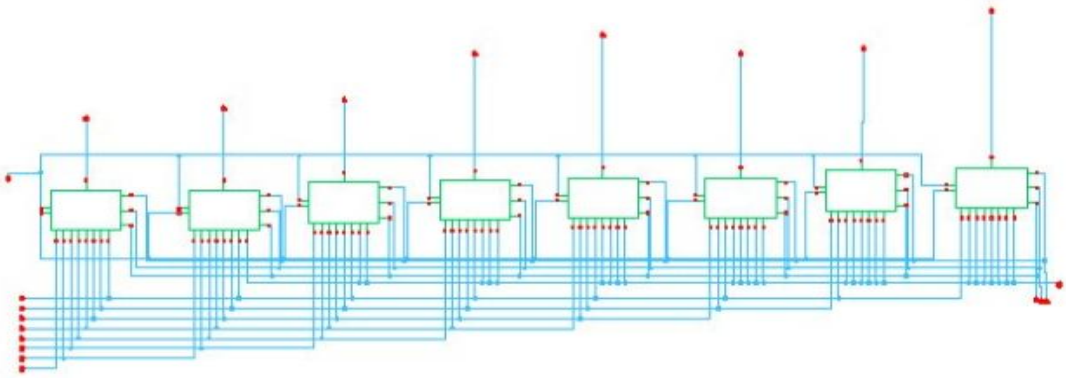


Figure 4.3 : Shift right of ALU

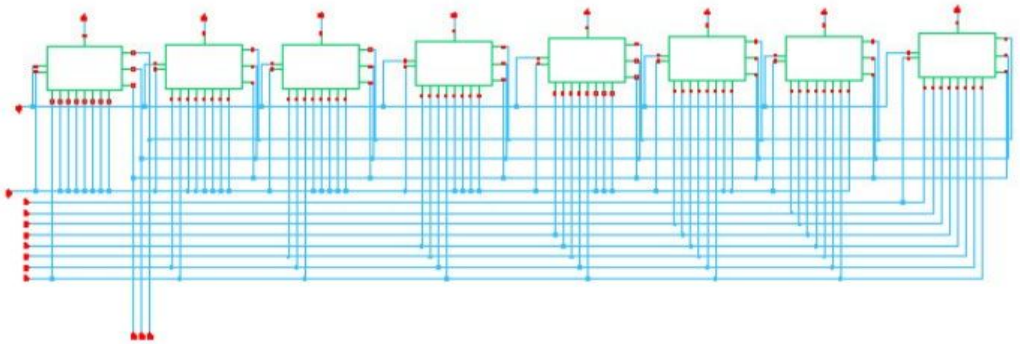


Figure 4.4 : Shift left of ALU

$A\ [7:0]$	$B\ [7:0]$	$ALU\_op\_i\ [2:0]$	$ALU\_result\_o\ [7:0]$	*Function*
0000_0000	*0101_0101*	*000*	*0101_0101*	*add*
0000_0000	*0101_0101*	*001*	*1010_1011*	*sub*
0000_0000	*0101_0101*	*010*	*0000_0000*	*and*
0000_0000	*0101_0101*	*011*	*0101_0101*	*or*
0000_0000	*0101_0101*	*100*	*0000_0000*	*right shift*
0000_0000	*0101_0101*	*101*	*1111_1111*	*not*
0000_0000	*0101_0101*	*110*	*0000_0000*	*left shift*
0000_0000	*0101_0101*	*111*	*0000_0000*	*fw A*

Table 4.1 : First Test Vector Truth Table

$A\ [7:0]$	$B\ [7:0]$	$ALU\_op\_i\ [2:0]$	$ALU\_result\_o\ [7:0]$	*Function*
1111_1111	*0110_0110*	*000*	*0110_0101*	*add*
1111_1111	*0110_0110*	*001*	*1001_1001*	*sub*
1111_1111	*0110_0110*	*010*	*0110_0110*	*and*
1111_1111	*0110_0110*	*011*	*1111_1111*	*or*
1111_1111	*0110_0110*	*100*	*1100_0000*	*right shift*
1111_1111	*0110_0110*	*101*	*0000_0000*	*not*
1111_1111	*0110_0110*	*110*	*0000_0011*	*left shift*
1111_1111	*0110_0110*	*111*	*1111_1111*	*fw A*

Table 4.2 : Second Test Vector Truth Table

$A\ [7:0]$	$B\ [7:0]$	$ALU\_op\_i\ [2:0]$	$ALU\_result\_o\ [7:0]$	*Function*
0010_1010	*1001_1111*	*000*	*1100_1001*	*add*
0010_1010	*1001_1111*	*001*	*1000_1011*	*sub*
0010_1010	*1001_1111*	*010*	*0000_1010*	*and*
0010_1010	*1001_1111*	*011*	*1011_1111*	*or*
0010_1010	*1001_1111*	*100*	*0000_0000*	*right shift*
0010_1010	*1001_1111*	*101*	*1101_0101*	*not*
0010_1010	*1001_1111*	*110*	*0000_0000*	*left shift*
0010_1010	*1001_1111*	*111*	*0010_1010*	*fw A*

Table 4.3 : Fiourth Test Vector Truth Table

Transient Analysis Simulation for an Arithmetic Logic Unit (ALU) involves modeling and analyzing the dynamic behavior of the ALU during operation. The ALU is a key component in a processor responsible for executing arithmetic and logical operations on input data. In the transient analysis simulation, various input scenarios are applied to the ALU, representing different arithmetic or logical operations, and the transient response of the ALU's output is observed over time. This includes monitoring the output signals, such as the result of addition, subtraction, bitwise AND, OR, or XOR operations, as well as carry-out and overflow flags. The simulation helps verify the ALU's correctness, performance, and timing characteristics under different operating conditions, aiding in debugging and optimization. Additionally, transient simulations can assess the ALU's speed, power consumption, and signal integrity, ensuring it meets the desired specifications for the overall processor design.

4.1.2 Transient Analysis

A truth table for a "select" operation typically illustrates the behavior of a multiplexer (MUX) or a similar circuit element. In the context of a MUX, the select input determines which of multiple data inputs is routed to the output. For instance, in a 2-to-1 MUX, with two data inputs (A and B) and one select input (S), the truth table would showcase how the output (Y) changes based on the select input. When S is 0, Y will match input A; when S is 1, Y will match input B. This concise representation allows for easy comprehension of how the MUX operates and enables designers to understand its functionality within a larger circuit.

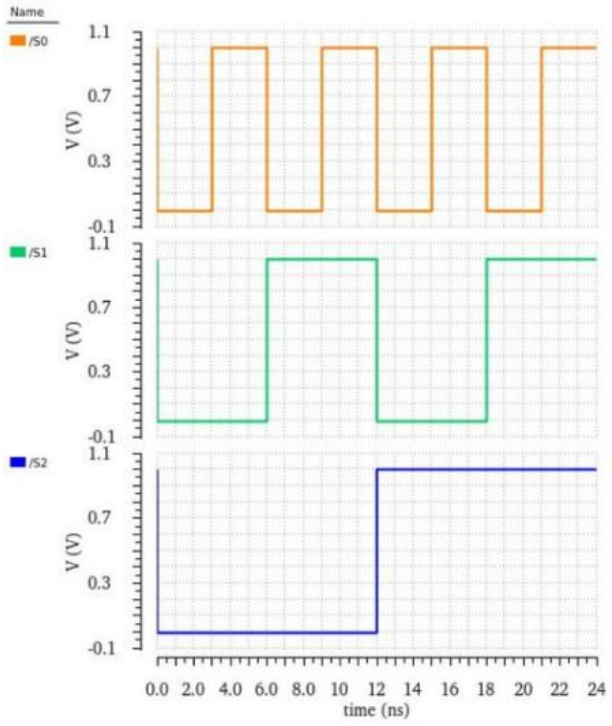


Figure 4.5 : *Signal of Selectin ALU*

We can see that vpulse setup is extremely important when D0, 1, 2 are increased by one according to the pre-set wavelength. With this design, it will be easy to see that we measure 8 statements of Sel one by one to get the result Y reaching ALU_op_i.

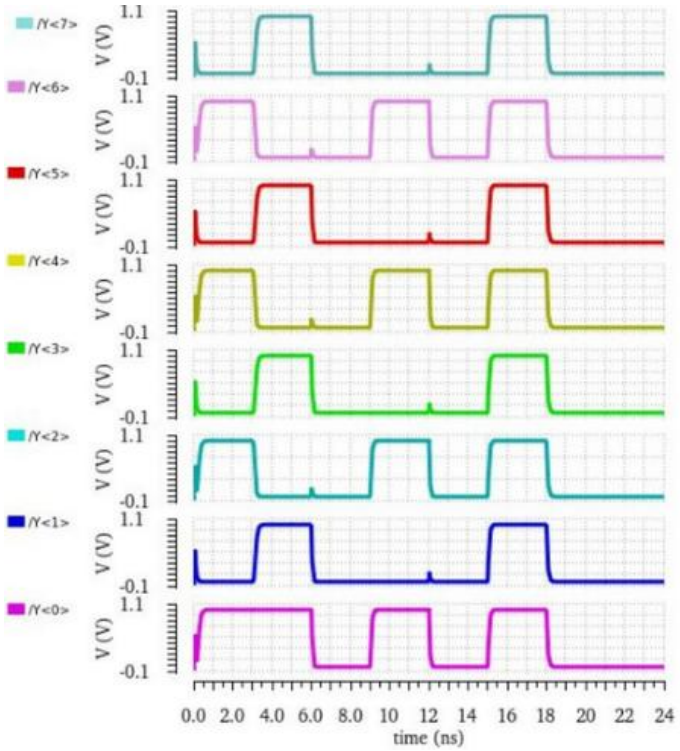


Figure 4.6 : *Transient of based on Table* 4.1

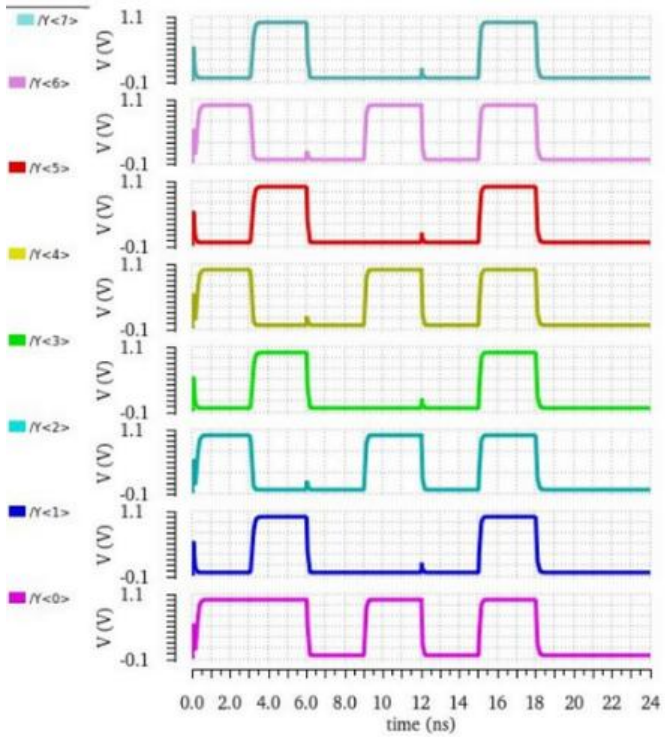


Figure 4.7 : *Transient of based on Table* 4.2

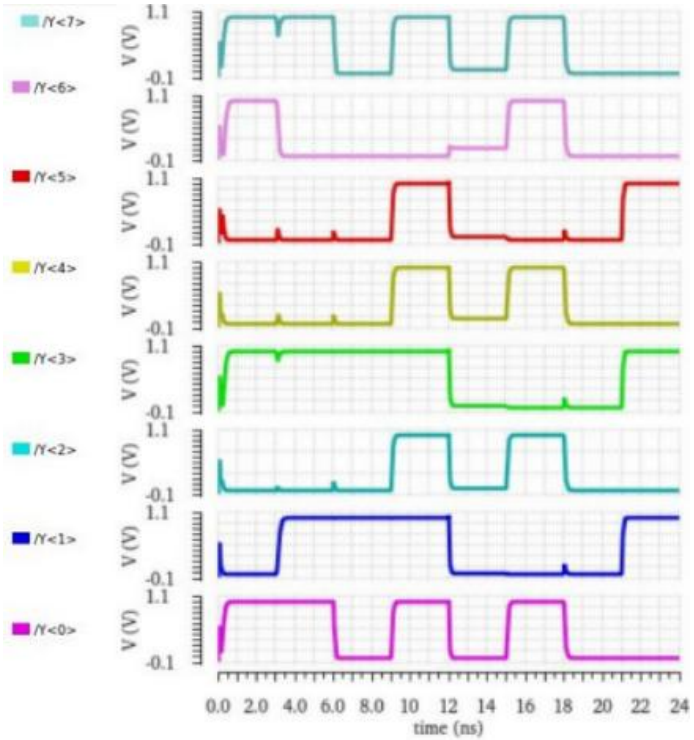


Figure 4.8 : *Transient of based on Table* 4.3

4.1.3 HDL Implementation Waveform

```
module tb_alu;

  // Testbench variables
  logic [7:0] numA;
```

```
  logic [7:0] numB;
  logic [2:0] sel;
  logic [8:0] result;

  // Instantiate the ALU
  alu uut (
    .numA(numA),
    .numB(numB),
    .sel(sel),
    .result(result)
  );

  // Test procedure
initial begin
#10;
    // Test case 1: Addition
    numA = 8'b00000000; numB = 8'b01010101; sel = 3'b000;
#10;
    // Test case 2: Subtraction
    numA = 8'b00000000; numB = 8'b01010101; sel = 3'b001;
#10;
    // Test case 3: AND

    numA = 8'b00000000; numB = 8'b01010101; sel = 3'b010;
#10;
    // Test case 4: OR

    numA = 8'b00000000; numB = 8'b01010101; sel = 3'b011;
#10;
    // Test case 5: Right Shift

    numA = 8'b00000000; numB = 8'b01010101; sel = 3'b100;
#10;
    // Test case 6: NOT

   numA = 8'b00000000; numB = 8'b01010101; sel = 3'b101;
#10;
    // Test case 7: Left Shift

   numA = 8'b00000000; numB = 8'b01010101; sel = 3'b110;
#10;
    // Test case 8: Pass Through

    numA = 8'b00000000; numB = 8'b01010101; sel = 3'b111;
#20;


end
endmodule
```

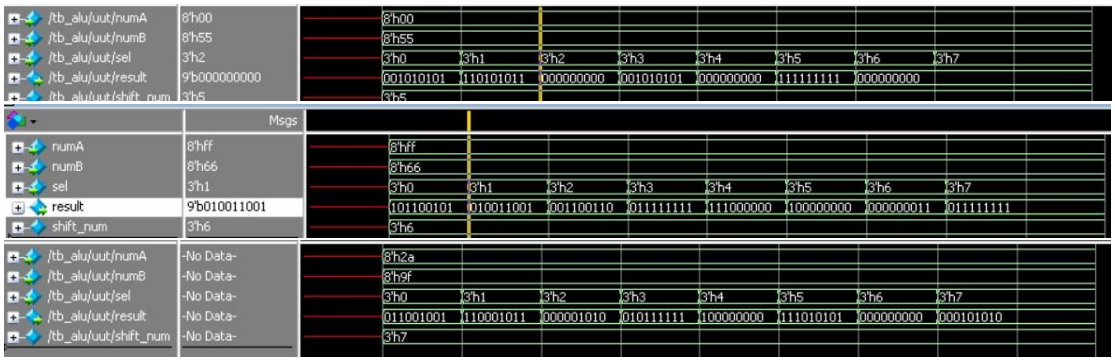


Figure 4.9 : *Output signal when check by Verilog – A*

Test cases are performed through hardware RTL code to check whether the Vpulse signals and output voltage are correct or not.

4.2 Experiment 2: Register File

4.2.1 Schematic and symbol for transient simulation of RF

For transient simulation of a Register File, the schematic would depict a collection of registers organized in rows and columns, with input and output buses connected to select and data lines. Each register typically consists of a group of flip-flops to store data bits and associated control logic for read and write operations.

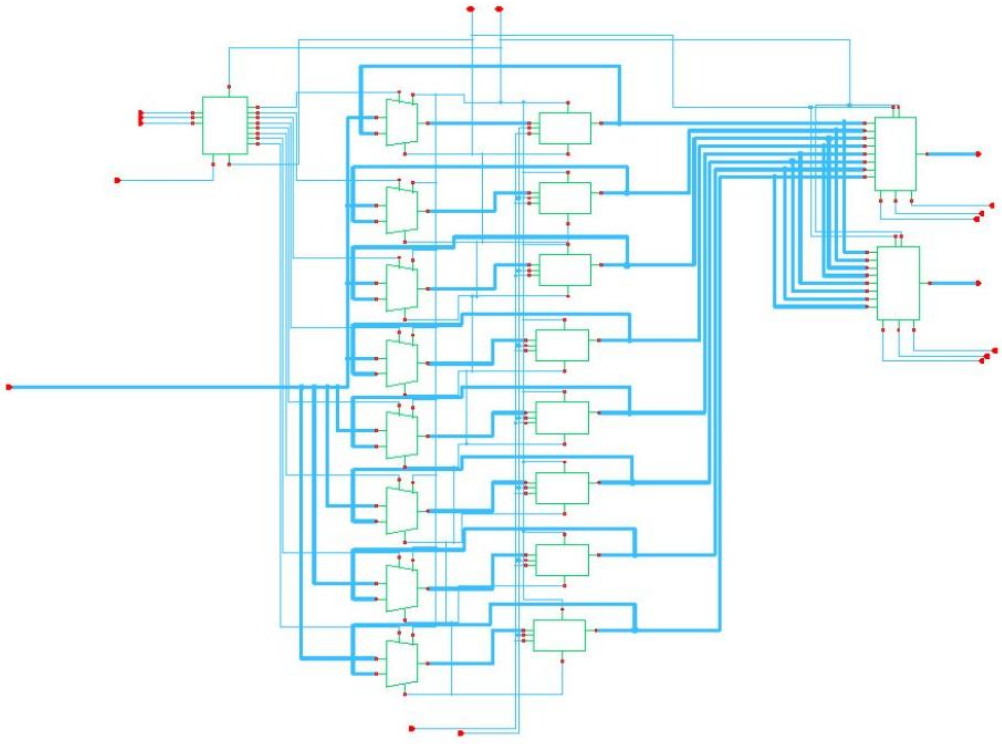


Figure 4.10 : *Register File*

The schematic would show connections for address inputs to select individual registers, data inputs for writing data, and data outputs for reading data. Additionally, control signals such as write enable (WE) and read enable (RE) would be included to govern the operation of the registers during write and read cycles. The symbol for transient simulation of a Register File would represent these components in a compact and standardized format, illustrating the inputs, outputs, and control signals required for simulation. Together, the schematic and symbol facilitate the transient simulation of the Register File, enabling analysis of its behavior over time under various operating conditions.

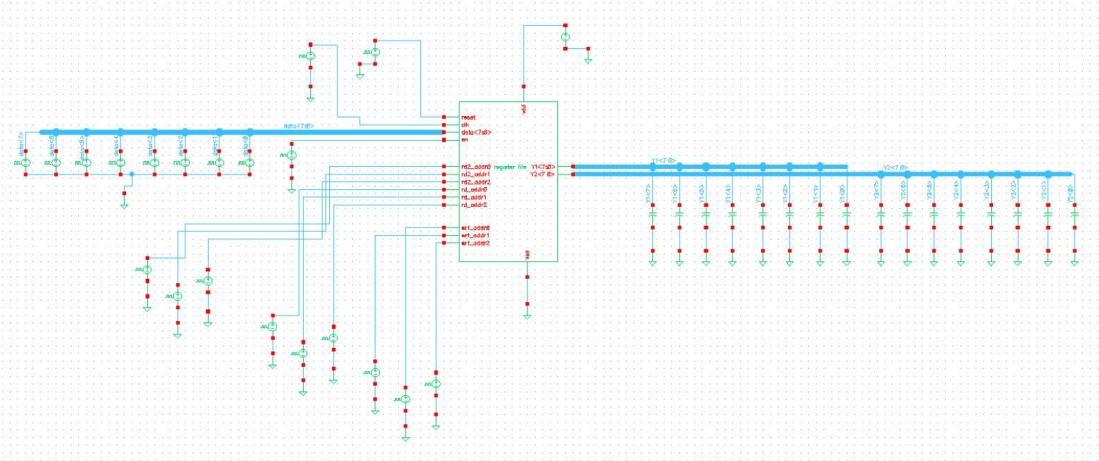


Figure 4.11 : *Register* 8 – *bits Symbol*

In a Register File (RF), a 3-to-8 decoder serves a critical function by interpreting a 3-bit address input and activating one of eight output lines, each corresponding to a specific register within the file. This decoder enables read and write operations to be directed to the desired register based on the provided address. During reads, it selects the register from which data is retrieved, while during writes, it determines the register to which incoming data will be written. Implemented using a combination of logic gates, the efficiency of the 3-to-8 decoder directly impacts the RF's performance, ensuring fast and accurate access to stored data.

4.2.2 Transient Analysis Simulation of RF

The accuracy of the register files has been thoroughly confirmed using the waveform measurements made after the simulation. The selected register files multiplied by the matching order of the register array are displayed in the result. Additionally, notable

about the observed waveform is the regular resetting of all registers to zero after every eighth write enable operation. This behavior is consistent with what is expected. functionality, showing how the register files' implementation of reset procedures and suitable register addressing is done.

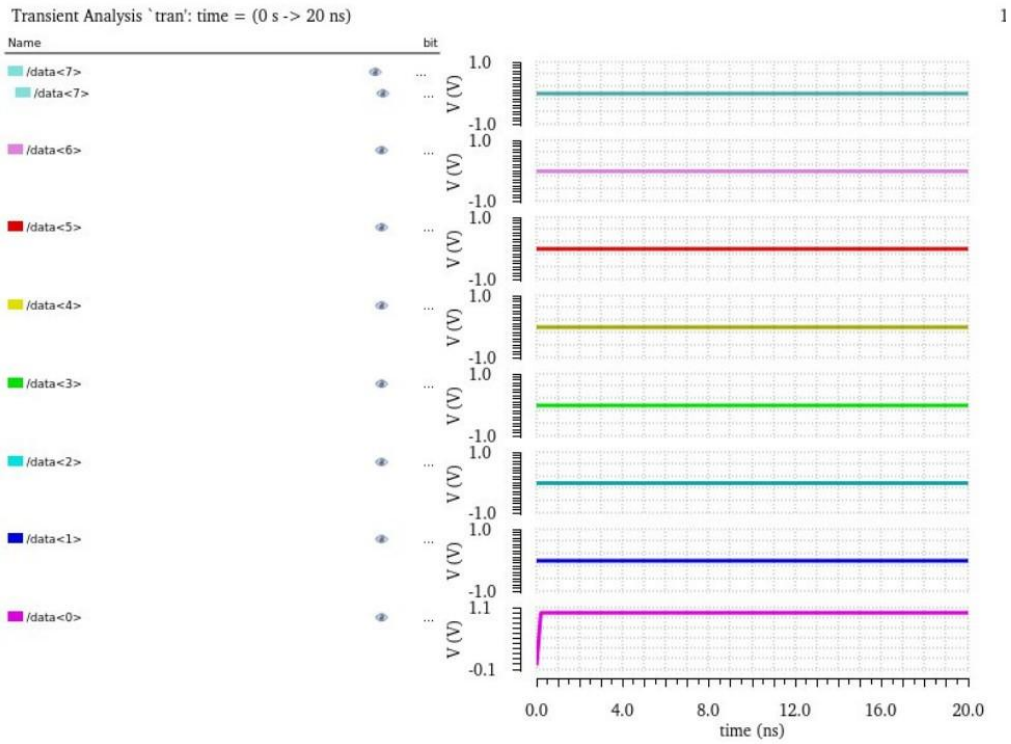


Figure 4.12 : *Transient signal of data_in*1

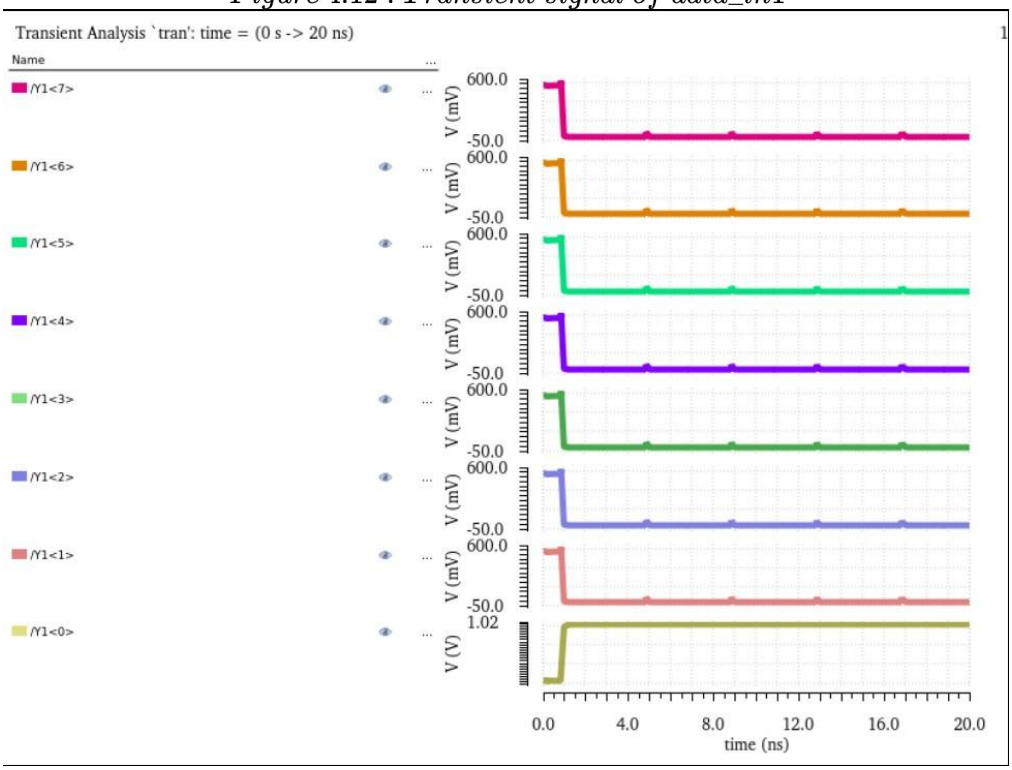


Figure 4.13 : *Transient signal of data_out*1

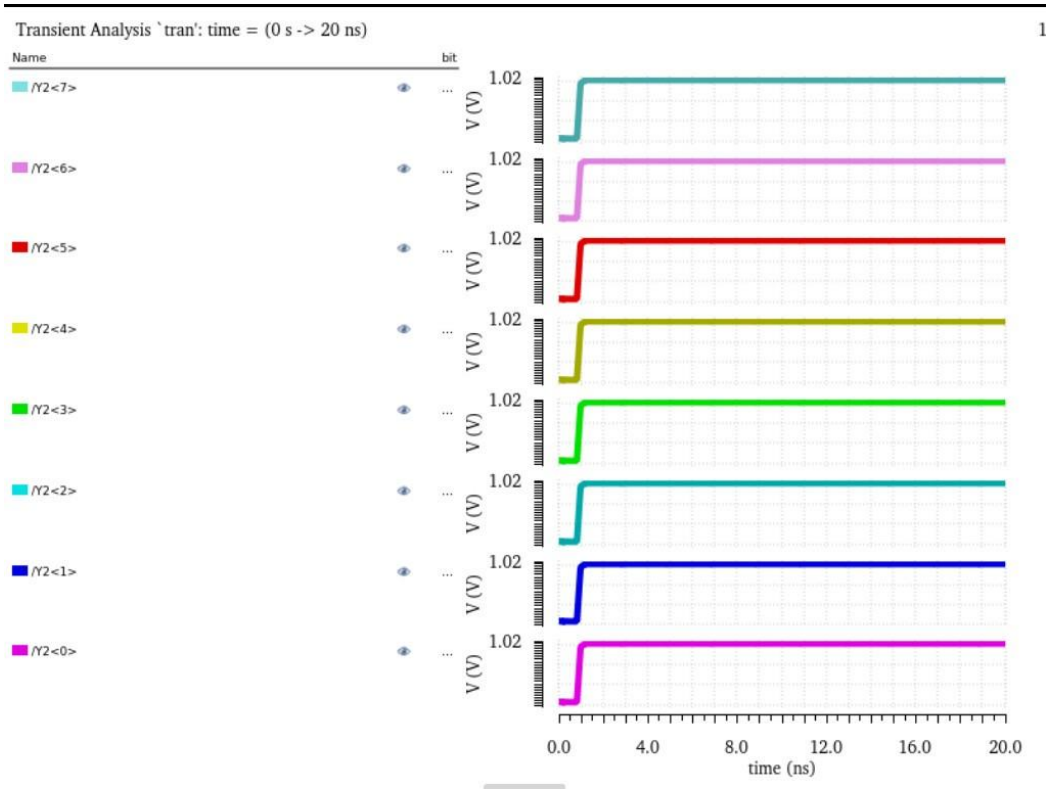


Figure 4.14 : *Transient signal of data_out2*

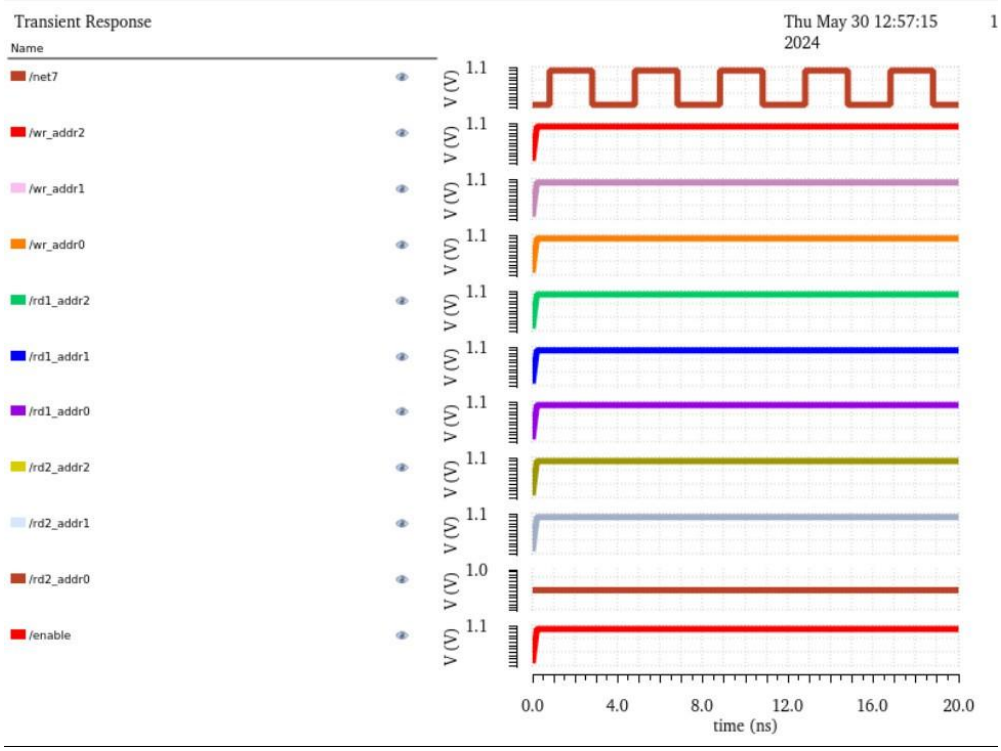


Figure 4.15 : *Transient signal of read , write address*

4.1.3 HDL Implementation Waveform

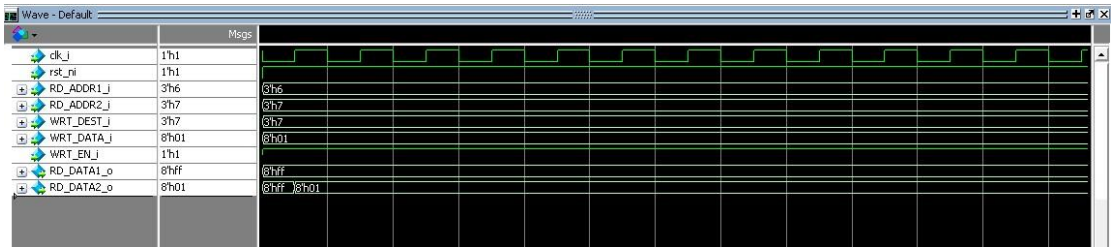


Figure 4.15 : *HDL Waveform of Regfile*

```
module REGFILE_8bit (
input logic clk_i, rst_ni,
input logic [2 : 0] RD_ADDR1_i, RD_ADDR2_i,
input logic [2 : 0] WRT_DEST_i,
input logic [7 : 0] WRT_DATA_i,
input logic WRT_EN_i,
output logic [7 : 0] RD_DATA1_o, RD_DATA2_o
);
reg [7 : 0]register [0 : 7];

assign RD_DATA1_o = register [RD_ADDR1_i];
assign RD_DATA2_o = register [RD_ADDR2_i];

always_ff @(posedge clk_i or negedge rst_ni) begin
if (!rst_ni) begin
for (integer i = 0; i < 8; i = i + 1)
register[i] <= 8'b11111111;
end
else
        begin
        register [WRT_DEST_i] <= WRT_DATA_i;
        end
end
endmodule
```

Laboratory 5: Introduction to Memory Circuit Design

5.1 SRAM (Static Random Access Memory)

5.1.1 Introduction of SRAM

The efficiency of cutting-edge technology including CPUs, register files, and networking hardware has increased thanks to integrated circuits (IC). There is a growing need for memory solutions to store and handle massive volumes of data as computing power and complexity rise. Rapidly developing memory technologies offer dependable, effective, and fast storage options. The main goals of research and development are to improve current technologies and investigate new ideas. Digital

electronics require Static Random-Access Memory (SRAM) because of its efficiency, dependability, and speed.

Figure 5.1 : *A sample of RAM*

5.1.2 Schematic and truth tabke of SRAM

The most common type of SRAM is called a 6T SRAM, which uses six transistors, as seen in Fig. 5.2, to carry out all RAM functions, such as writing, reading, and storing data. Because SRAM requires many transistors to store a single bit of data, it is more expensive than DRAM but offers a wider range of options.

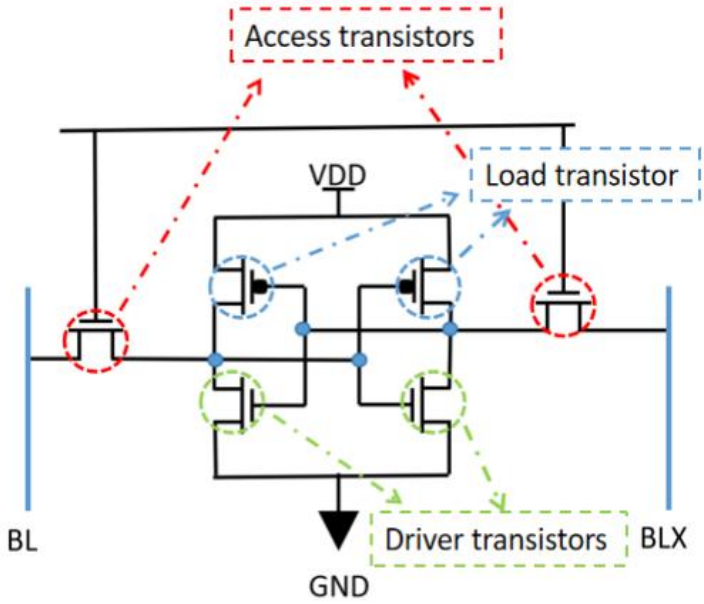


Figure 5.2 : *Schematic of 6T SRAM*

Also is a fundamental building block of memory arrays in modern integrated circuits. It consists of six MOSFETs arranged in a cross-coupled latch configuration. Two access transistors control the read and write operations, enabling data to be stored and retrieved. The stability of the stored data is maintained by the feedback loop formed by the cross-coupled inverters. The 6T SRAM cell offers fast access times, low power consumption, and high density, making it a versatile choice for on-chip memory in digital systems. Its robustness and simplicity make it suitable for various applications, ranging from cache memories in microprocessors to data buffers in communication systems. However, the 6T SRAM cell occupies a relatively large area compared to alternative memory cell designs, limiting its scalability in highly integrated circuits.

Wordline	Bit	Bít'	Output (Q
0	0	0	0
0	1	0	0
1	0	0	0
1	1	0	1

Table 5.1 : *Truth table of 6T SRAM cell write operation*

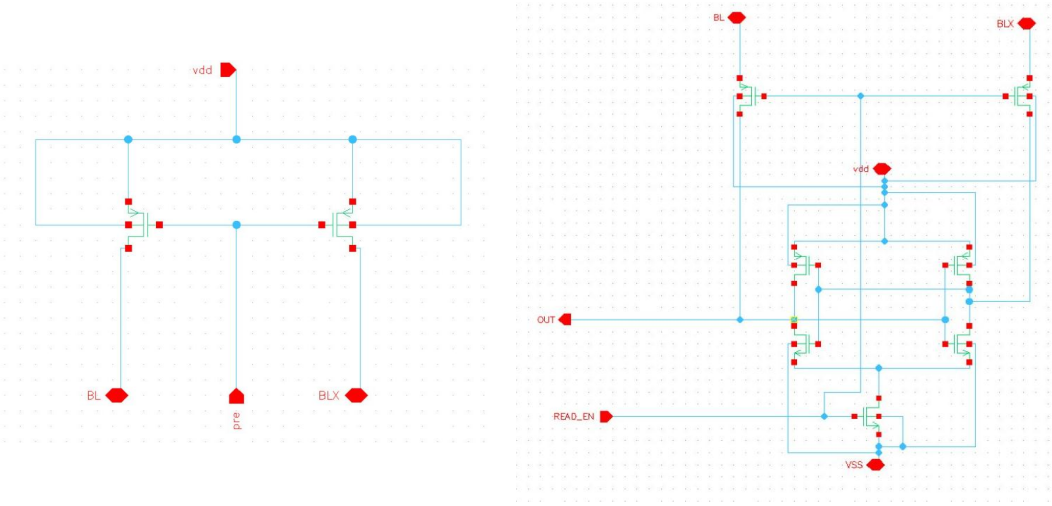


Figure 5.3 : *Precharge and sense shown in SRAM*

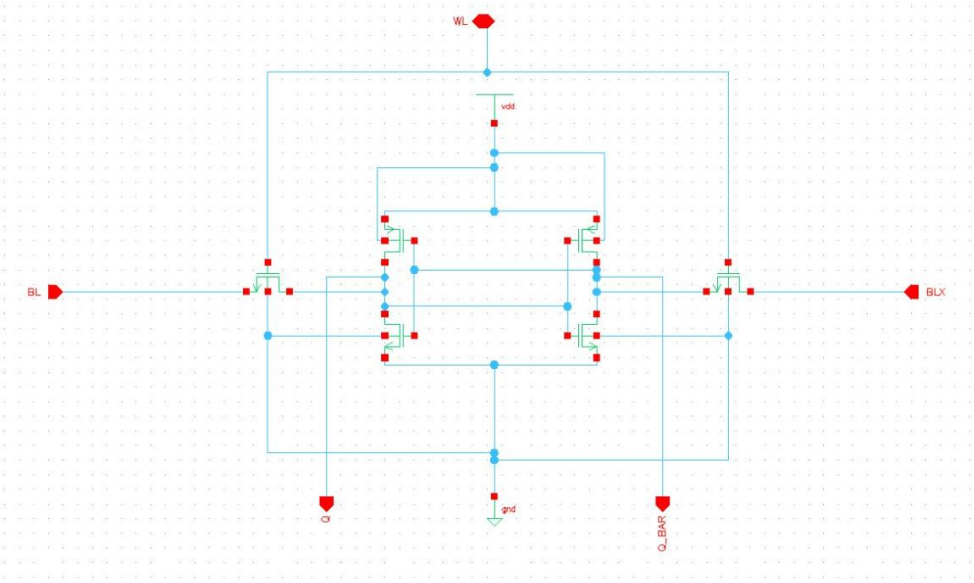


Figure 5.4 : *Core system of SRAM*

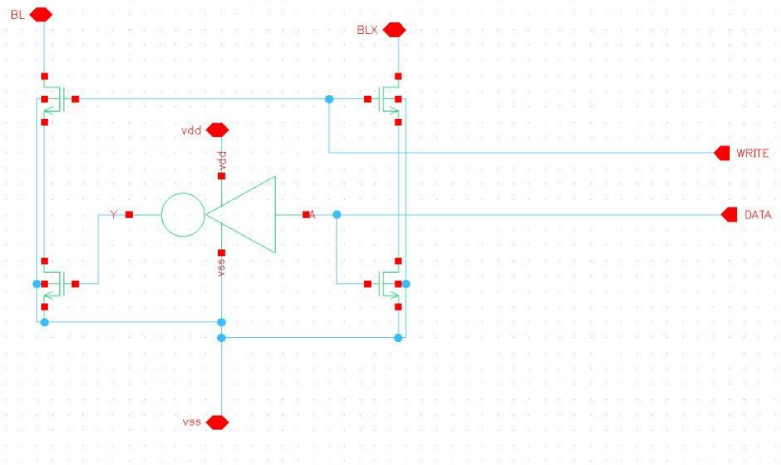


Figure 5.5 : *Write drive SRAM*

5.1.3 SRAM operation

5.1.3.1 Reading operation

During the read operation of an SRAM cell, the stored data is accessed and retrieved. The process involves the following steps:

Word Line Activation: The word line associated with the SRAM cell is activated by applying a voltage to it. This enables the access transistor to turn on.

Bit Line Precharge: The bit lines (usually labeled as BL and BL', representing complementary signals) are precharged to a reference voltage level to prepare them for the read operation.

Data Sensing: The precharged bit lines are connected to the respective storage nodes of the SRAM cell. If the cell contains a logic high (1), the storage node voltage will be high, and the complementary bit line (BL') will be discharged while the main bit line (BL) remains precharged. Conversely, if the cell contains a logic low (0), the storage node voltage will be low, and the main bit line (BL) will be discharged while the complementary bit line (BL') remains precharged.

5.1.3.2 Writing operation

An SRAM cell's write operation requires transitioning the cell from one stable state to another. The following steps are usually included in the write operation:

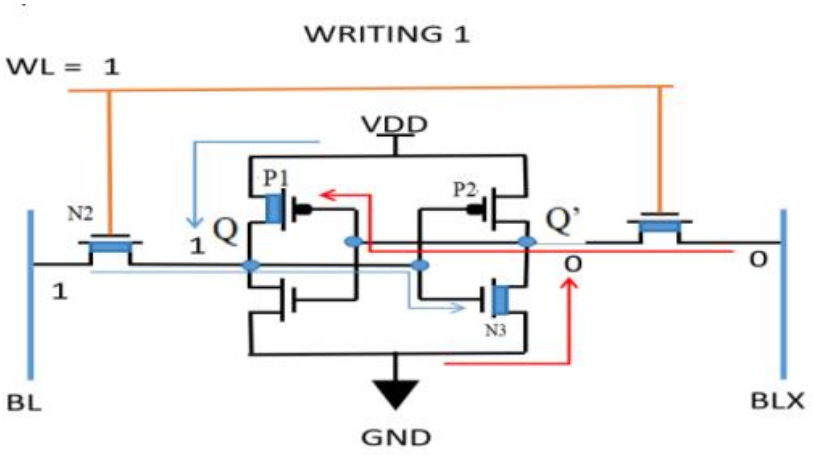


Figure 5.7 : '1' is simulated to be writing '1' for 6T SRAM

Precharging: A high voltage is applied to the bitlines, which are the SRAM cell's input/output lines.

Activation of the wordline: The wordline is brought into contact with the access transistor gates. This makes it possible to access the cell and perform a write operation.

Data input: The bitlines receive the data that has to be written. A "1" in the data causes the bitline voltage to rise, and a "0" in the data causes it to fall.

Cell operation: The data is saved in the cell when the latch changes states because the voltage on one bitline is higher than the voltage on the other.

Wordline deactivation: To stop more modifications to the cell, the wordline is disabled after the data has been stored there.

Refresh: The SRAM cells must be routinely refreshed in order to stop data loss via leakage current.

5.1.4 Transient of SRAM

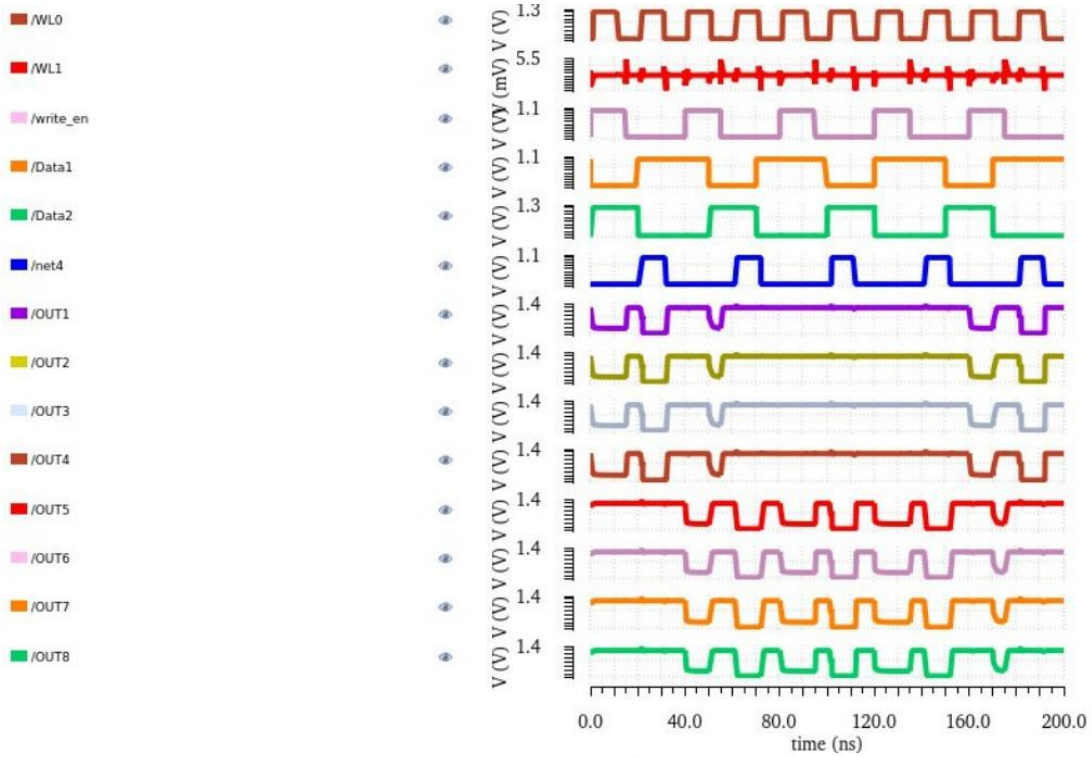


Figure 5.7 : *Transient of The* 6*T SRAM*

The simulation report shows that:

- At the start of the simulation, the cell receives data “1”, at the moment the signal output also read out “1”.
- At 11ns, the read operation occurs even when WL is active LOW, and the output remains “1” as the last read.
- At 20ns, the cell receives data “0”, and again receives data “1” at 40ns. In this period, output remain “0” until the 44ns the output changes to “1” since data stored in the cell changed to “1” at 40ns.

PART II: IC DESIGN FLOW

Chapter 1: Introduction of Integrated Circuit Design

1.1 Definition of IC Design

Integrated Circuit (IC) design refers to the process of creating intricate electronic circuits and systems that are miniaturized and integrated onto semiconductor substrates. It involves meticulously designing individual components, such as transistors, capacitors, and resistors, and interconnecting them to form complex functional units.

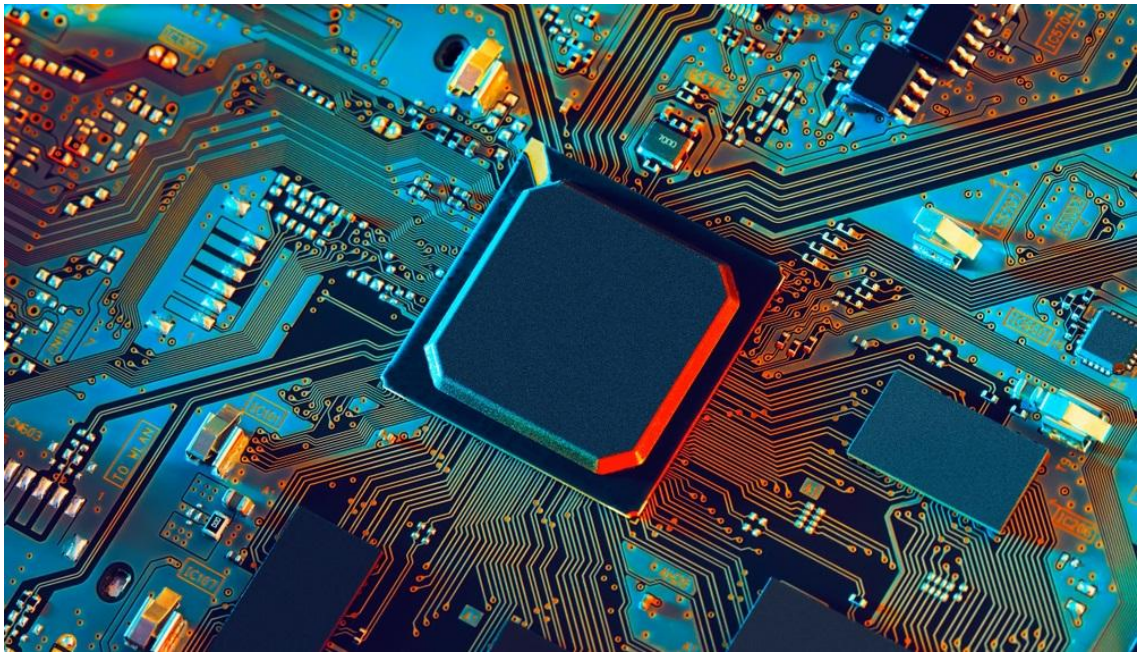


Figure 1.1 : *Integrated Circuit*

IC design encompasses various stages, starting from conceptualization and architectural planning, followed by transistor-level design, layout implementation, and verification using specialized software tools and methodologies. Designers employ techniques such as simulation, emulation, and prototyping to validate the functionality, performance, and reliability of the IC before fabrication. The objectives of IC design include optimizing factors such as speed, power consumption, area utilization, and manufacturing yield while meeting stringent specifications and constraints.

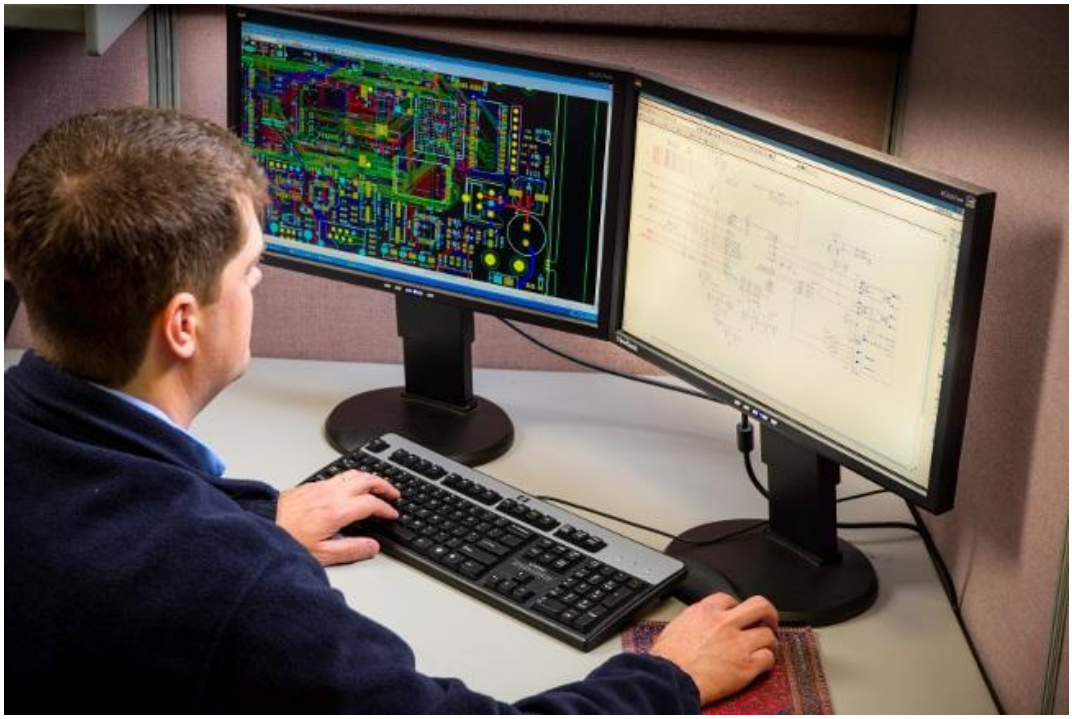


Figure 1.2 : *Integrated Circuit Engineering*

Ultimately, IC design plays a pivotal role in developing cutting-edge semiconductor solutions that power a wide array of electronic devices and systems, driving technological advancements in numerous industries.

1.1.1 Overview of IC Design

The overview of IC design entails a structured process involving several key stages. It begins with architectural exploration, where designers define the overall functionality and specifications of the integrated circuit. This is followed by high-level modeling and simulation to validate the architecture's feasibility. Next, designers move to transistor-level design, where they specify the individual components and their interconnections, optimizing for performance, power consumption, and area efficiency. The layout implementation phase involves physically arranging the components on the semiconductor substrate, considering factors such as parasitic effects, routing constraints, and manufacturability. Verification is a crucial step, encompassing simulation, formal verification, and physical verification to ensure the design meets functional, timing, and reliability requirements. Finally, the design is ready for fabrication, where the IC is manufactured using advanced semiconductor processes. Throughout the process, IC designers leverage a wide range of tools and

methodologies to address various design challenges and optimize the final product for its intended application.

Related to career opportunities, Vietnam's growing electronics industry offers promising career opportunities for professionals in Integrated Circuit (IC) design. With a focus on manufacturing and export-oriented production, the country has seen increasing demand for skilled IC designers to support the development of cutting-edge semiconductor solutions. Career opportunities in IC design in Vietnam include positions such as IC design engineer, design verification engineer, physical design engineer, and layout designer. These roles are often found in semiconductor companies, research institutions, and multinational corporations with a presence in Vietnam's burgeoning technology sector. As Vietnam continues to invest in technology and innovation, the demand for IC design talent is expected to rise, providing ample opportunities for career growth and development in this dynamic field. Additionally, with the government's support for the semiconductor industry through initiatives such as the National Semiconductor Strategy, there are prospects for local talent to contribute to the country's technological advancement and competitiveness on the global stage.

Figure 1.3 : *Statistics of semiconductor businesses in Vietnam*

1.1.2 Importance of IC Design Flow

The importance of the Integrated Circuit (IC) design flow lies in its role as a structured framework that guides the development of complex semiconductor devices,

ensuring efficiency, reliability, and innovation. A well-defined IC design flow provides a systematic approach to tackling the intricate challenges of designing integrated circuits, from conceptualization to fabrication. By following a standardized flow, designers can effectively manage the complexity of modern ICs, optimize performance parameters such as speed, power consumption, and area utilization, and meet stringent design specifications and constraints. Additionally, the IC design flow facilitates collaboration among multidisciplinary teams of engineers, enabling seamless integration of digital, analog, and mixed-signal components in advanced system-on-chip (SoC) designs. Moreover, a robust design flow enhances productivity by streamlining design iterations, accelerating time-to-market, and reducing development costs.

1.1.3 Types of ICs

Integrated Circuits (ICs) encompass various types tailored to different functions and applications. **Digital ICs** process discrete signals and perform logical operations, including microprocessors and memory chips. **Analog ICs** handle continuous signals for tasks such as amplification and filtering, featuring operational amplifiers and voltage regulators. **Mixed-signal ICs** integrate both digital and analog circuitry, facilitating seamless interaction between domains. **Application-Specific Integrated Circuits (ASICs)** are customized for specific functions, while **Field-Programmable Gate Arrays (FPGAs)** offer reconfigurability. **System-on-Chip (SoCs)** pack multiple components onto a single chip, and **memory ICs** store data temporarily or permanently. These ICs power diverse technologies, from consumer electronics to industrial applications, shaping the modern world.

1.2 Specifications

In the IC design flow, the specifications step involves defining the functional requirements, performance targets, and constraints for the integrated circuit (IC) being developed. This critical phase lays the foundation for the entire design process, guiding subsequent stages to ensure the final product meets customer expectations and market demands. During the specifications step, designers collaborate with stakeholders to gather and analyze requirements, considering factors such as functionality, speed, power consumption, area, and cost. Clear and comprehensive

specifications help establish design goals, identify design trade-offs, and inform architectural decisions. Additionally, specifications serve as a reference throughout the design flow, enabling designers to validate design choices and track progress against predefined criteria. By carefully defining specifications upfront, IC design teams can streamline the development process, minimize rework, and deliver successful IC designs that meet or exceed customer expectations.

For example, the design requirements for a digital signal processing (DSP) IC intended for wireless communication applications encompass functional, performance, and non-functional criteria. Functional requirements specify the desired features and capabilities of the IC, such as signal processing, filtering, modulation/demodulation, and error correction coding. Performance requirements define key metrics such as throughput, latency, power consumption, area utilization, and reliability targets. Non-functional requirements address factors like regulatory compliance, environmental conditions, and manufacturing constraints. These design requirements serve as a comprehensive guideline for the IC design process, guiding designers in making informed decisions and ensuring that the final IC meets the needs and expectations of its intended users.

Chapter 2: Frontend Design

1.3 System Design

In the IC design flow, the step of system design involves the conceptualization and high-level planning of the integrated circuit (IC) based on the identified requirements and specifications. This stage focuses on defining the overall architecture, functionality, and key features of the IC to meet the desired application needs. System designers collaborate closely with stakeholders, including customers, marketing teams, and domain experts, to gather requirements and understand the application domain thoroughly. Using this information, designers develop a high-level system architecture, outlining the major components, interfaces, and data flow within the IC. During system design, trade-offs and design decisions are made to balance competing requirements such as performance, power consumption, area efficiency, and cost-effectiveness. Prototyping and simulation techniques may be employed to validate the feasibility and functionality of the proposed system architecture. Overall, system

design sets the direction for subsequent stages in the IC design flow, providing a solid foundation for detailed design, implementation, and verification.

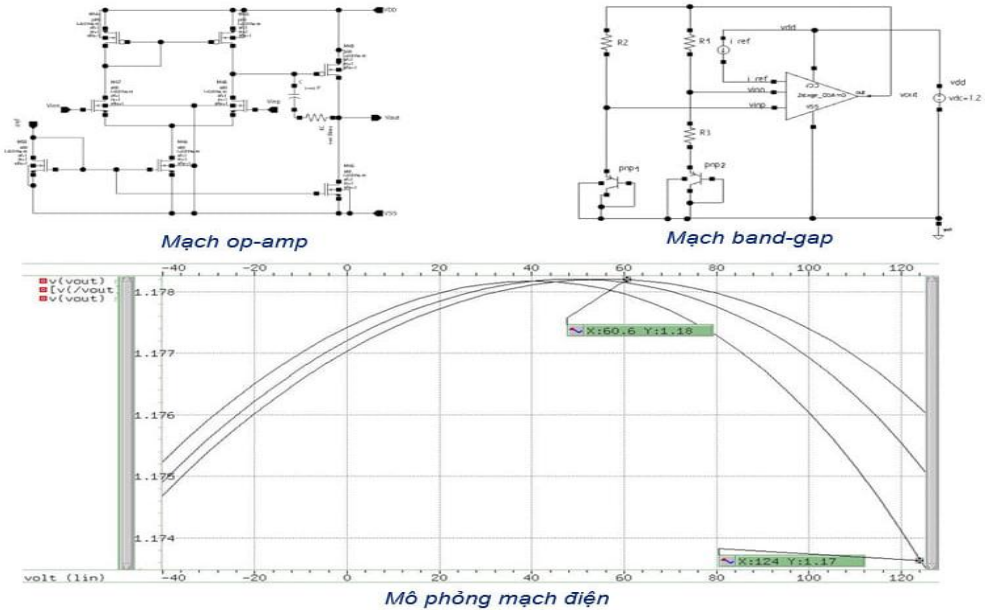


Figure 1.3 : *Analog IC Design Working Scope*

The System Architecture step in the IC design flow entails the following key aspects:

- Functional Partitioning:

System architects identify the major functional blocks or modules required to fulfill the IC's intended purpose. These functional blocks represent the core functionalities of the system, such as processing units, memory components, input/output interfaces, and communication modules.

- Interconnect Design:

Architects define the communication pathways and interfaces between the functional blocks to facilitate data exchange and control flow within the IC. This includes determining the types of interconnects (e.g., buses, point-to-point connections) and their specifications (e.g., data width, protocol).

- Interface Definition:

System architects specify the external interfaces of the IC, including input/output ports, communication protocols, and control signals. These interfaces enable the IC to interact with external components, systems, or users, and they play a crucial role in defining the IC's compatibility and interoperability requirements.

- Performance Analysis:

Architects evaluate the expected performance of the system architecture in terms of speed, throughput, latency, and power consumption. Performance analysis helps identify potential bottlenecks, optimize resource allocation, and ensure that the architecture meets the desired performance goals.

- Trade-off Analysis:

System architects assess various design trade-offs, such as performance versus area, power consumption versus speed, and cost versus functionality. By considering these trade-offs, architects can make informed decisions to balance competing requirements and optimize the overall system architecture.

- Prototyping and Validation:

Architects may create prototypes or conduct simulations to validate the feasibility and functionality of the proposed system architecture. Prototyping helps identify potential design issues early in the design process and enables iterative refinement of the architecture before proceeding to detailed design and implementation.

1.4 RTL Design (Register Transfer Level)

Is a key step in the IC design flow that focuses on describing the behavior and functionality of the integrated circuit (IC) at the register transfer level. In RTL design, designers specify the functionality of the IC using hardware description languages (HDLs) such as Verilog or VHDL, capturing the data flow and control logic of the design at the register level. The RTL description typically defines the registers, data paths, and control signals within the IC, as well as the operations performed on the data as it moves between registers. RTL descriptions are abstract representations of the hardware behavior and are independent of specific implementation details such as transistor-level circuits or physical layout. RTL descriptions are used for simulation, synthesis, and verification of the IC design. RTL simulation allows designers to verify the functionality of the design and identify potential issues before moving to synthesis. RTL synthesis translates the RTL description into a gate-level netlist, which represents the logical connectivity of the design using standard cells from a technology library. Overall, RTL Design serves as a crucial bridge between the high-level system architecture and the low-level implementation of the IC, enabling efficient and accurate development of complex digital designs.

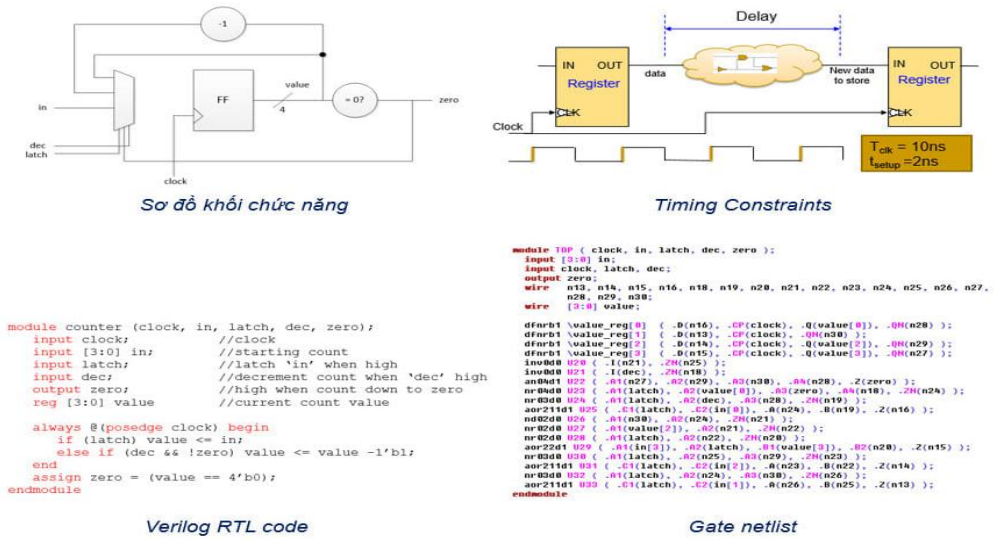


Figure 1.4 : *RTL Design*

The primary components of RTL design include:

- Registers: RTL descriptions define registers to store data temporarily during computation. Registers represent the state elements in the IC and capture the data at different points in time.

- Data Paths: Data paths represent the paths through which data flows between registers and functional units within the IC. RTL descriptions specify the logic and arithmetic operations performed on data as it traverses the data paths.

- Control Logic: Control logic determines the sequence and timing of operations performed within the IC. RTL descriptions define control signals that govern the behavior of the IC and coordinate the execution of operations.

- Sequential and Combinational Logic: RTL descriptions include both sequential logic elements (e.g., flip-flops) and combinational logic elements (e.g., logic gates) to implement the desired functionality of the IC.

- RTL design serves as a bridge between the high-level system architecture and the low-level implementation of the IC, providing a detailed description of the IC's behavior while abstracting away implementation details such as specific transistor-level circuits or physical layout. RTL descriptions are used for simulation, synthesis, and verification of the IC design. RTL simulation allows

designers to verify the functionality of the design and identify potential issues before moving to synthesis. RTL synthesis translates the RTL description into a gate-level netlist, which represents the logical connectivity of the design using standard cells from a technology library.

1.5 Design Verification

The Design Verification step in the IC Design Flow is a critical phase focused on ensuring that the integrated circuit (IC) design meets the specified requirements and functions correctly under various operating conditions. This phase involves thoroughly testing and validating the design against the defined specifications, design constraints, and performance targets. Design Verification encompasses several key activities:

- Functional Verification: Functional Verification involves verifying that the IC behaves according to its intended functionality. This includes testing the functionality of individual blocks or modules within the IC, as well as verifying the interactions between different blocks to ensure proper system behavior. Techniques such as simulation, emulation, and formal verification are used to validate the design's functional correctness.
- Timing Verification: Timing Verification ensures that the timing requirements of the design are met under different operating conditions. This includes verifying timing constraints such as setup and hold times, clock frequencies, and signal propagation delays. Static Timing Analysis (STA) and dynamic timing simulations are commonly used to analyze and verify the timing behavior of the design.

- Power Verification: Power Verification involves assessing the power consumption of the IC and ensuring that it meets specified power targets. This includes analyzing power distribution networks, estimating power consumption during different operating modes, and verifying that power constraints are satisfied. Power simulations and analysis tools are used to validate the power characteristics of the design.

- Functional Coverage Analysis: Functional Coverage Analysis assesses the completeness of the testbench and verifies that the design has been thoroughly exercised by the test cases. This includes tracking the coverage of functional features, corner cases, and error conditions to ensure that all aspects of the design have been adequately tested. Coverage metrics are used to measure the effectiveness of the verification process and identify areas that require additional testing.

- Code and Design Rule Checking: Code and Design Rule Checking involves analyzing the design code and layout to ensure compliance with coding standards, design guidelines, and manufacturing rules. This includes checking for syntax errors, coding style violations, and design rule violations that could impact the functionality or manufacturability of the IC.

- Integration Testing: Integration Testing verifies the correct integration of individual blocks or modules into the overall system. This includes testing the interfaces between different blocks, verifying data flow and control signals, and ensuring proper communication and synchronization between components.

Chapter 3: Backend Design

1.8 Physical Verification

Physical Verification is a critical step in the IC design flow aimed at ensuring the correctness, reliability, and manufacturability of the integrated circuit (IC) layout. It involves verifying that the physical layout of the IC complies with design rules, manufacturing constraints, and industry standards. Physical Verification encompasses several key activities:

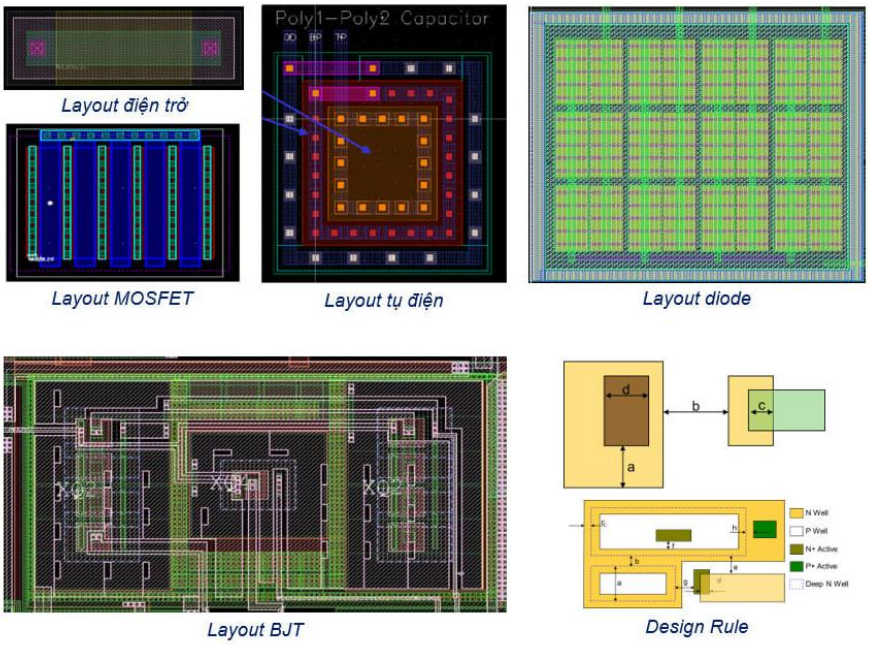


Figure 1.5 : *Layout Check*

1.8.1 Design Rule Check (DRC)

DRC verifies that the layout adheres to the design rules specified by the semiconductor foundry or manufacturing process. These rules define constraints such as minimum feature sizes, spacing requirements, layer alignments, and metal density limits. DRC tools analyze the layout data to flag violations of these rules, helping designers identify and correct potential manufacturing issues that could affect the IC's performance or yield.

1.8.2 Layout Versus Schematic (LVS)

LVS compares the layout of the IC to its corresponding schematic or netlist to ensure consistency between the design intent and physical implementation. LVS tools verify that the connectivity, transistor sizes, and device placements in the layout match the intended design specifications. Any discrepancies between the layout and schematic are flagged for further investigation and resolution.

1.8.3 Electrical Rule Check (ERC)

ERC checks for electrical connectivity and integrity issues in the layout, such as open circuits, short circuits, and floating nodes. ERC tools analyze the layout data to ensure that all electrical connections are properly established and that there are no unintended connections or signal integrity issues that could impact the IC's performance.

1.9 Timing Analysis

Timing Analysis is a critical step that focuses on assessing and optimizing the timing characteristics of the integrated circuit (IC) to ensure proper functionality and performance. This step involves evaluating the timing behavior of the IC under various operating conditions and scenarios, including worst-case conditions, to identify potential timing violations and ensure that the design meets timing requirements. Timing Analysis encompasses several key tasks:

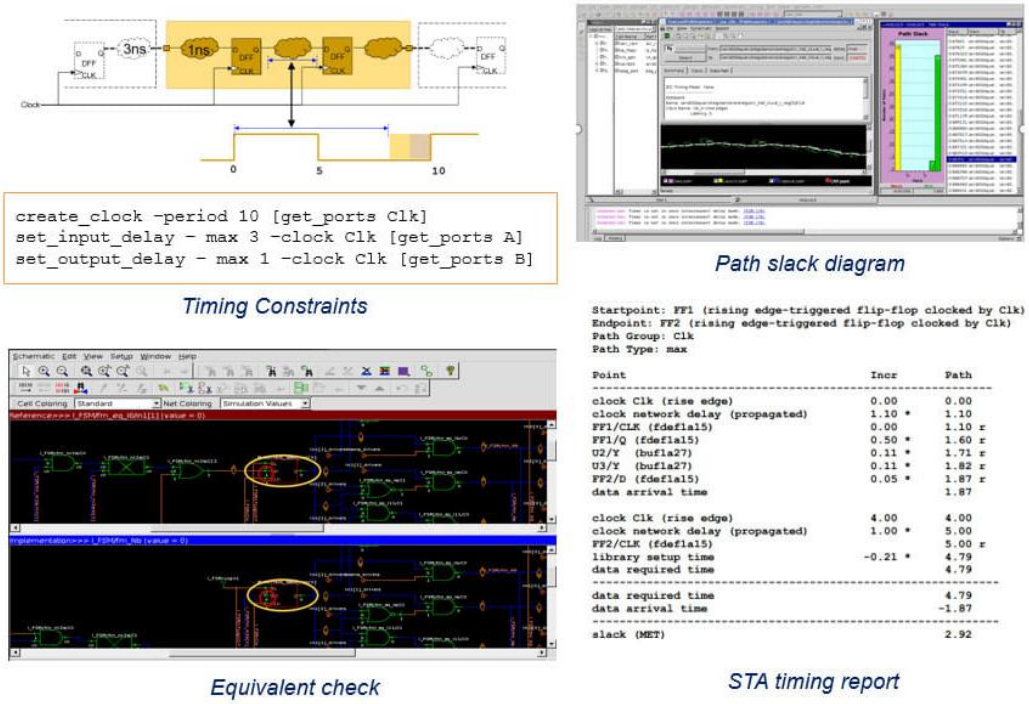


Figure 1.6 : *Timing Analysis*

1.9.1 Static Timing Analysis (STA)

STA involves analyzing the timing paths in the IC to determine the propagation delay from inputs to outputs, setup and hold times for flip-flops, and clock-to-output delays. STA tools simulate the timing behavior of the design based on the logic gates' delay models, interconnect delays, and clock signals. Designers use STA to identify critical paths, detect timing violations, and optimize the design to meet timing constraints.

1.9.2 Timing Closure

Timing Closure refers to the iterative process of modifying the design, adjusting constraints, and optimizing the timing paths to meet timing requirements. Designers use techniques such as logic restructuring, buffer insertion, and clock tree

optimization to achieve timing closure and ensure that the design operates reliably within specified timing limits.

1.9.3 Clock Domain Crossing (CDC) Analysis

CDC Analysis focuses on ensuring proper synchronization and handling of signals that cross between different clock domains in the design. CDC issues such as metastability, data loss, and skew are analyzed to prevent timing errors and ensure reliable operation of the IC.

1.10 Design for Test (DFT)

DFT is a crucial step in the IC design flow aimed at enhancing the testability of the integrated circuit (IC) and ensuring efficient testing during production. DFT techniques are integrated into the IC design to enable easy access to internal nodes, facilitate fault detection, and improve fault coverage during testing. The DFT process typically involves several key activities:

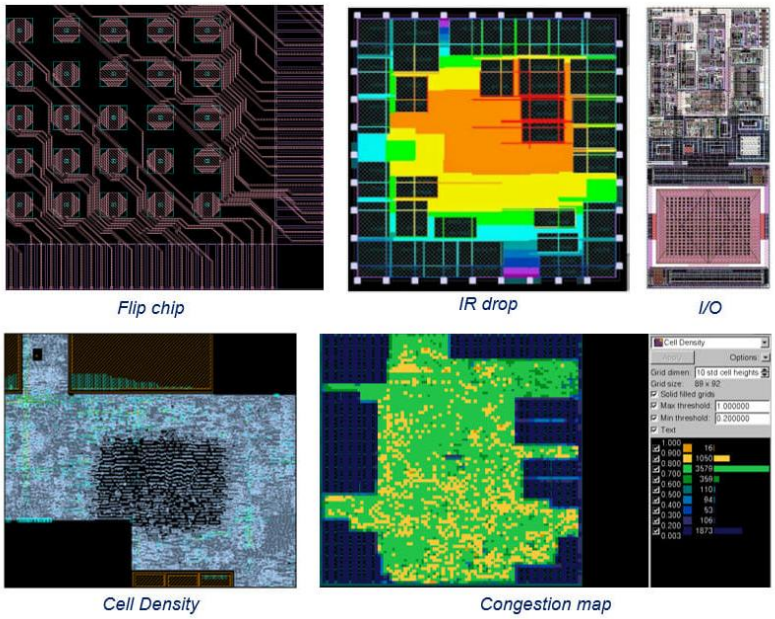


Figure 1.7 : DFT Works

- Scan Chain Insertion: Scan chains are serial chains of flip-flops inserted into the design to facilitate scan-based testing. These chains allow for the sequential shifting of test patterns into and out of the IC, enabling efficient capture and observation of internal node values during testing.

- Built-in Self-Test (BIST): BIST circuits are embedded within the IC to perform self-testing without the need for external test equipment. BIST modules generate test patterns, apply them to the IC, and analyze the responses to detect faults. BIST helps improve test coverage and reduces the need for external test resources.
- Test Access Mechanisms (TAMs): TAMs provide access to internal nodes and circuits within the IC for testing purposes. These mechanisms include test access ports (TAPs), boundary scan cells, and other interfaces that allow external test equipment to control and observe the IC during testing.
- On-Chip Instrumentation: On-chip instrumentation circuits are integrated into the design to monitor and measure various parameters such as voltage, current, and temperature during testing. These circuits provide valuable diagnostic information and help identify potential faults or defects in the IC.
- Design Verification: DFT techniques are verified and validated through simulation, emulation, and formal verification methods to ensure that they meet testability requirements and do not impact the functionality or performance of the IC.
- Post-Silicon Test: DFT features are utilized during post-silicon testing to verify the functionality and reliability of the fabricated ICs. Test patterns generated by DFT circuits are applied to the IC, and the responses are analyzed to identify and diagnose any manufacturing defects or faults.

1.11 Production

Production step in the IC design flow involves the fabrication of the integrated circuit (IC) based on the finalized design. This phase encompasses several key activities:
Mask Generation: The design data is converted into a set of photomasks, which are used to define the circuit layout on the semiconductor wafer during the fabrication process. This step involves generating masks for different layers of the IC, including the active regions, metal interconnects, and contact vias.

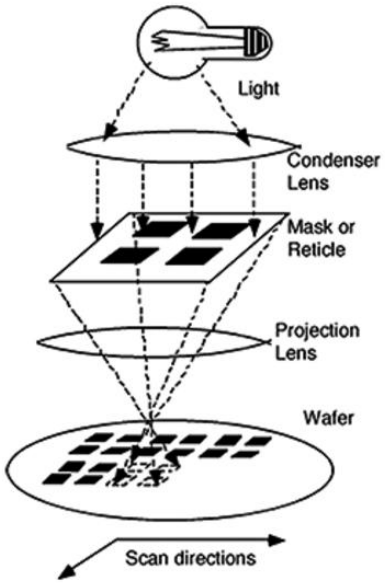


Figure 1.8 : *DFT Works*

- Wafer Fabrication: The photomasks are used to pattern the semiconductor wafer through a series of lithography, etching, and deposition steps. These processes create the physical structures and interconnections that form the IC's components and circuits on the wafer.
- Wafer Testing: After fabrication, the wafers undergo testing to ensure that the fabricated ICs meet quality and functionality specifications. Wafer testing involves electrical tests to verify the performance and functionality of the ICs, as well as physical inspections to identify defects or anomalies.
- Die Separation: Once testing is complete, the wafers are diced into individual IC chips or dies using a process called die separation. Each die contains one or more ICs, depending on the design specifications and wafer size.

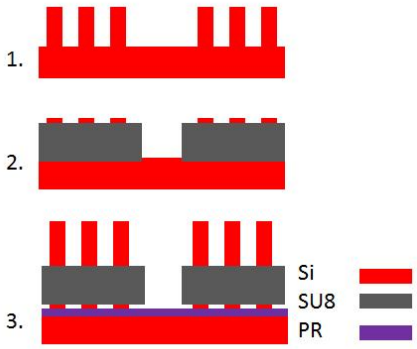


Figure 1.9 : *Die Separation*

- Packaging: The individual IC dies are packaged into protective enclosures or packages, which provide mechanical support, electrical connections, and environmental protection for the ICs. Packaging options include plastic packages, ceramic packages, and chip-scale packages, among others.
- Final Testing: Packaged ICs undergo final electrical testing to ensure proper functionality and performance. This testing may include functional tests, parametric tests, and reliability tests to verify that the ICs meet specifications under various operating conditions and environments.
- Quality Assurance: Throughout the production process, quality assurance measures are implemented to monitor and control the quality of the ICs. This includes process control, defect detection and prevention, and adherence to industry standards and regulations.

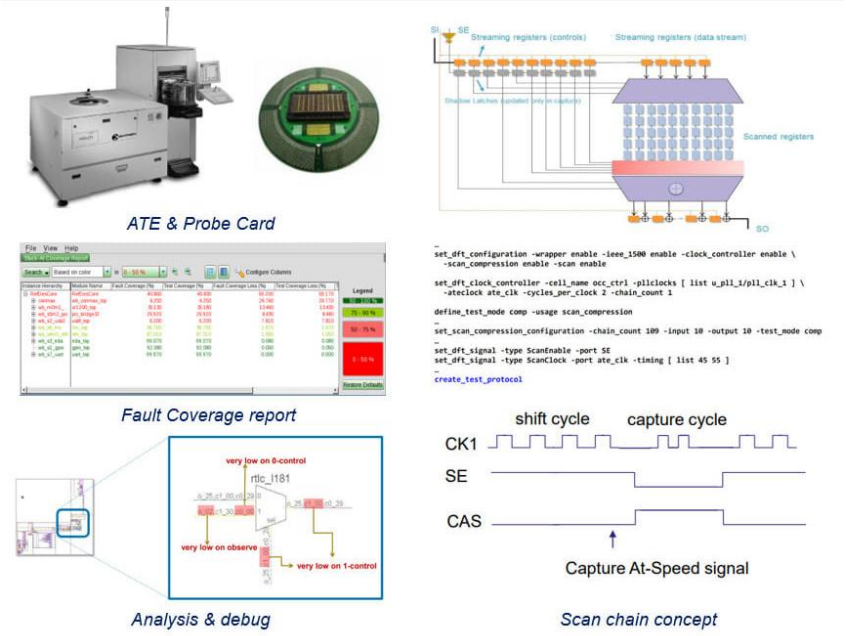


Figure 1.10