International Institute of Information Technology, Hyderabad

VLSI Design

Course Project: 5-bit Carry Look Ahead (CLA) Adder

Author:
Madhur Kankane
madhur.kankane@students.iiit.ac.in
Roll No.: 2024102061

Instructor:
Prof. Abhishek Srivastava, CVEST

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

1 INTRODUCTION

Modern digital systems demand high-speed and energy-efficient arithmetic operations to support a wide range of computational tasks. Among these, addition is a fundamental operation that significantly influences the performance of arithmetic units in processors, signal processors, and embedded systems. Traditional ripple-carry adders, though simple in design, suffer from poor scalability, as their addition delay grows linearly with the number of bits due to carry propagation. This limitation makes them unsuitable for high-performance applications where low latency is critical.

To address these challenges, **Carry Look-Ahead Adders (CLAs)** have emerged as a key solution by minimizing carry propagation delay. CLAs achieve this by generating *propagate* (P_i) and *generate* (G_i) signals for each bit position and leveraging these signals to compute carry bits in parallel. This parallel computation significantly improves addition speed, making CLAs an attractive choice for modern digital systems.

In this project, a **5-bit CLA adder** is designed and implemented using **180nm CMOS technology**. The design process involves:

- **Schematic Design:** The CLA adder is initially designed at the transistor level using various MOSFET logic to achieve robust and energy-efficient operation.
- **Functional Verification:** The functionality of the designed modules is verified through pre-layout simulations using NGSPICE.
- **Physical Design:** The design is implemented in MAGIC layout, ensuring compliance with design rules and minimizing parasitics.
- **Post-Layout Validation:** Post-layout simulations are performed to evaluate critical performance metrics, such as propagation delay, power consumption, and area utilization.

To validate the design's practicality, the CLA adder is also implemented on an FPGA platform, demonstrating its hardware applicability. A comprehensive analysis compares the schematic and post-layout simulation results, highlighting discrepancies introduced by layout parasitics. Key performance metrics, such as maximum operating frequency, worst-case delay, and energy efficiency, are thoroughly examined.

The circuit is designed to drive an inverter of size $W_n = 10 * \lambda$ and $W_p = 2 * W_n$, where $\lambda = 0.09\mu m$. The implemented circuit follows the arrangement given below.

2 DESIGN OF THE CIRCUIT

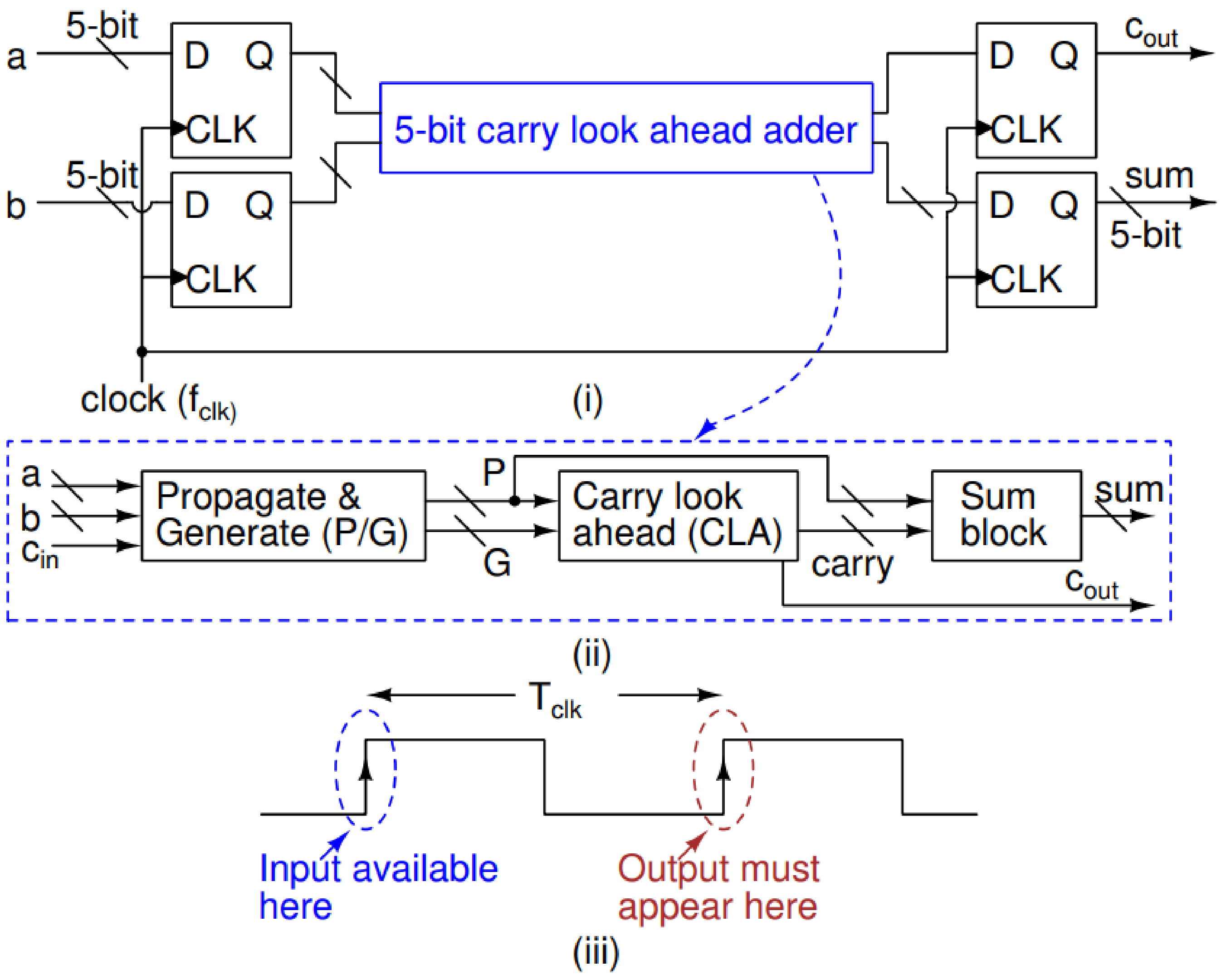


Figure 1: High level overview of circuit

To implement the 5-bit CLA adder the block for Carry propagation and generation is used. The propogate and generate equations are as follows:

$$P_i = A_i \oplus B_i \tag{1}$$

$$G_i = A_i \cdot B_i \tag{2}$$

The carry equations for a Carry-Lookahead Adder (CLA) are as follows:

$$C_{i+1} = G_i + P_i C_i \tag{3}$$

On Simplifying the Equations (substituting the C_i value in C_{i+1} Equations), we get:

$$C_1 = G_0 + P_0C_0$$

$$C_2 = G_1 + P_1G_0 + P_1P_0C_0$$

$$C_3 = G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0$$

$$C_4 = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 + P_3P_2P_1P_0C_0$$

$$C_5 = G_4 + P_4G_3 + P_4P_3G_2 + P_4P_3P_2G_1 + P_4P_3P_2P_1G_0 + P_4P_3P_2P_1P_0C_0$$

- Static CMOS implementation is used for most of the gates, and for some gates (specifically XOR) pass transistor logic styling is used.
- 2-input gates are used in place of multiple input gates in order to minimize the delay.
- Standard CMOS static sizing is used, so that the delay is minimized and the other unwanted effects don't bother.

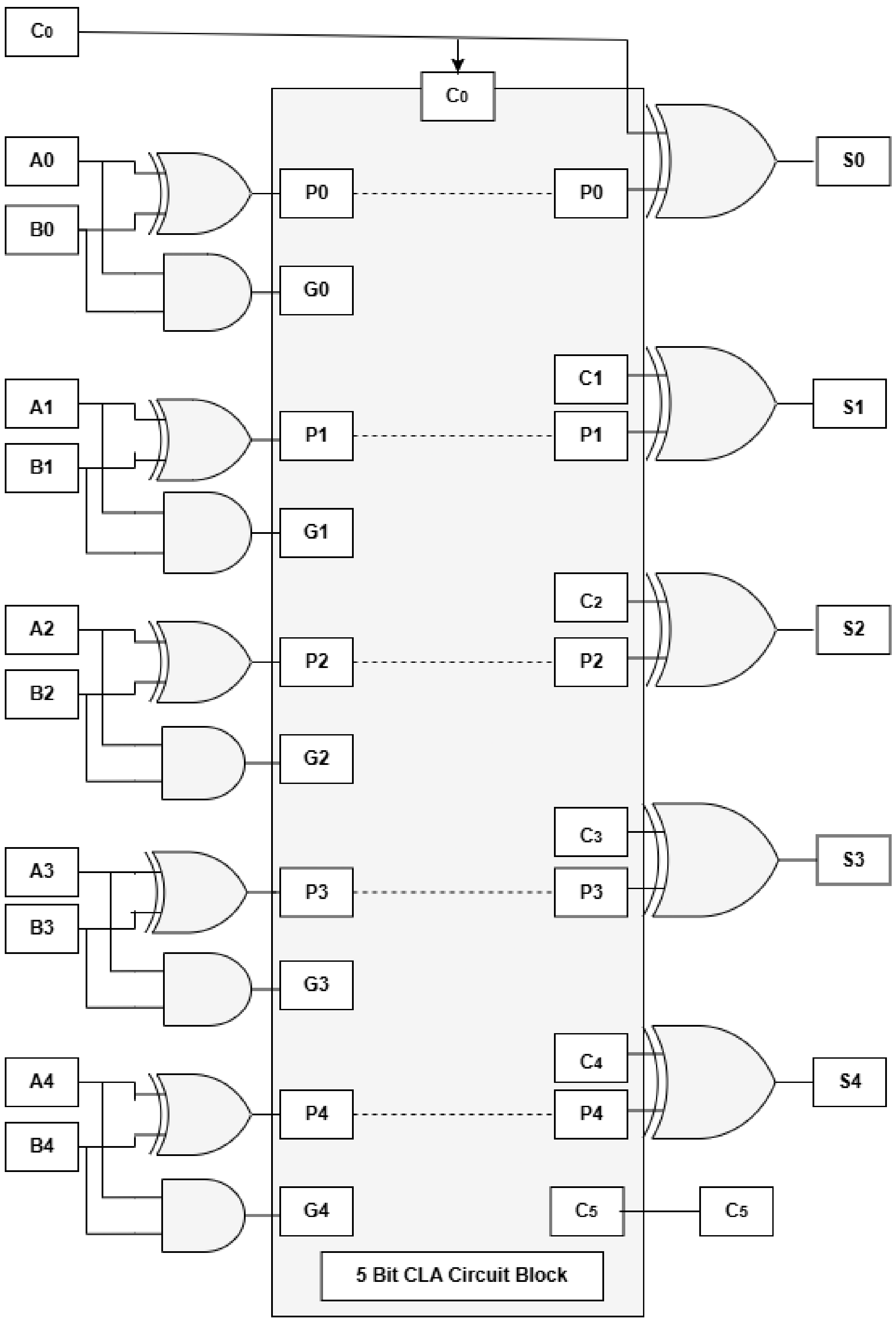


Figure 2

3 Design Details of Blocks Used

$W_n = 10 * LAMBDA$ and $W_p = 2 * W_n$, where $LAMBDA = 0.09\mu m$ are used as a standard size of static CMOS gates. The sizing of the following gates are now described in terms of W_n and W_p. The length of each MOSFET used is $2 * LAMBDA$ (min length).

3.1 Inverter

A CMOS inverter uses complementary PMOS and NMOS transistors to achieve low power dissipation, high noise margins, and efficient signal inversion. The static CMOS inverter used at all the place has the $\frac{W_p}{W_n}$ ratio=2 and W_n and W_p are the same as described above.

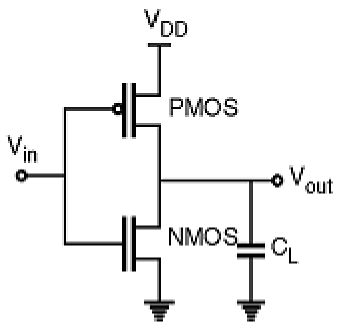


Figure 3

3.2 XOR Gate

The XOR gate is implemented using PTL style, with PMOS and NMOS sized identically to those in the CMOS inverter.

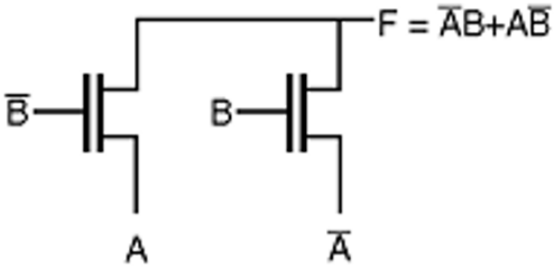


Figure 4

3.3 AND Gate

The AND gate is implemented by cascading a NAND gate and an inverter, using series NMOS and parallel PMOS transistors in the NAND stage.

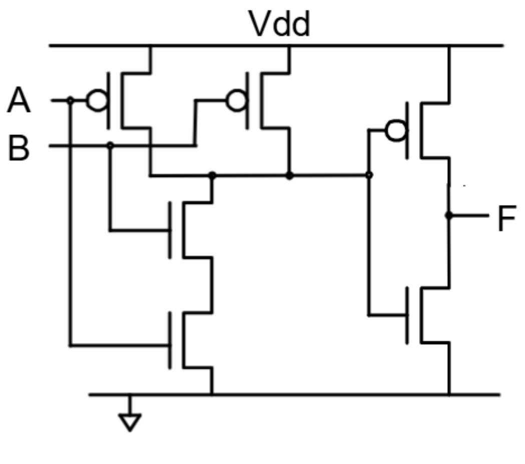


Figure 5

3.4 OR Gate

The OR gate is implemented by cascading a NOR gate and an inverter, using parallel NMOS and series PMOS transistors in the NOR stage.

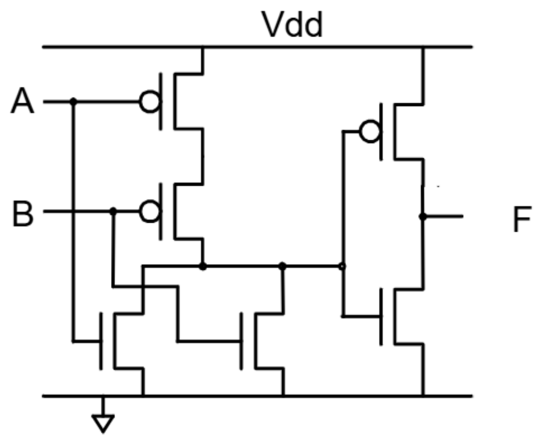


Figure 6

3.5 D Flip Flop

The D Flip-Flop is implemented using TSPC (True Single Phase Clock) logic for high-speed operation and reduced clock skew, with a buffer added at the output to prevent signal degradation.

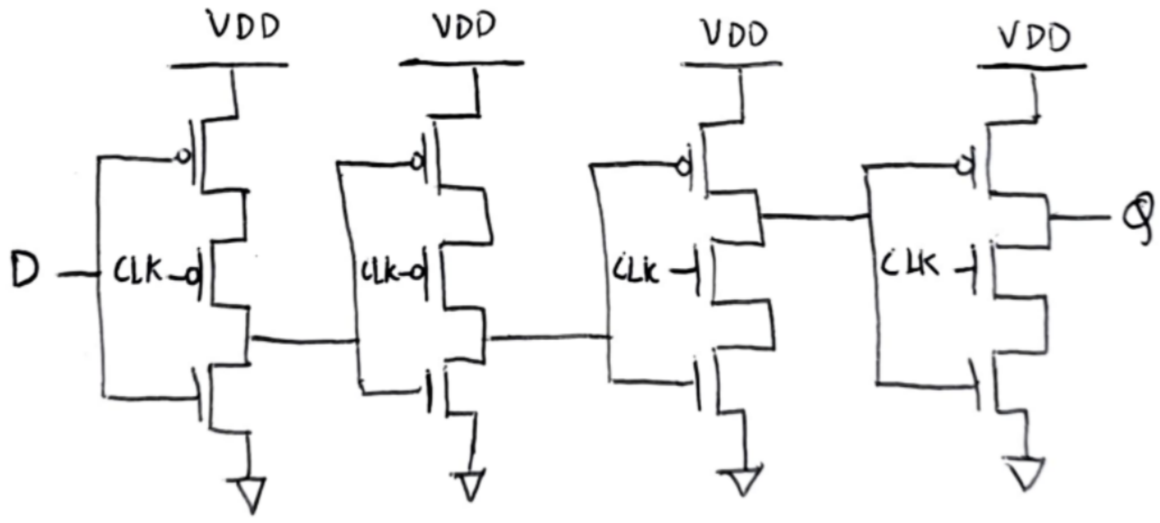


Figure 7

4 Verification and Simulation of Gates Used

4.1 Inverter

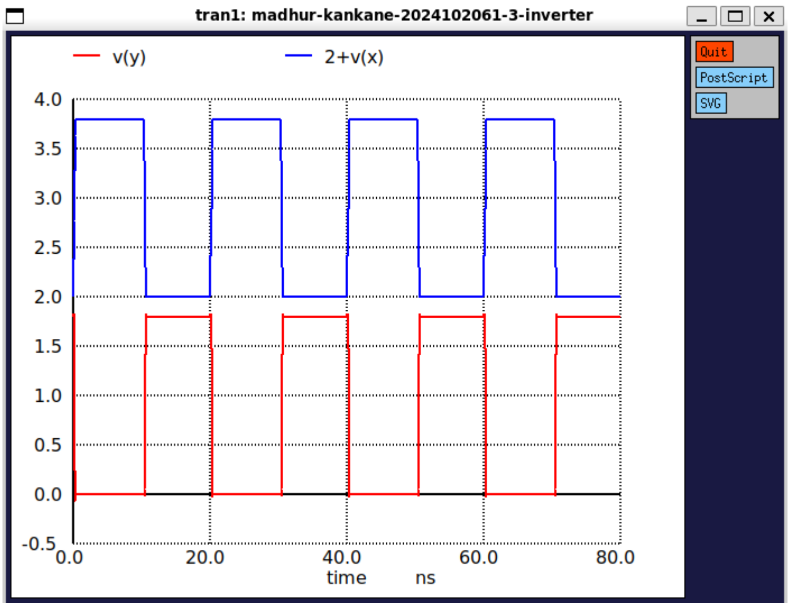


Figure 8

4.2 XOR Gate

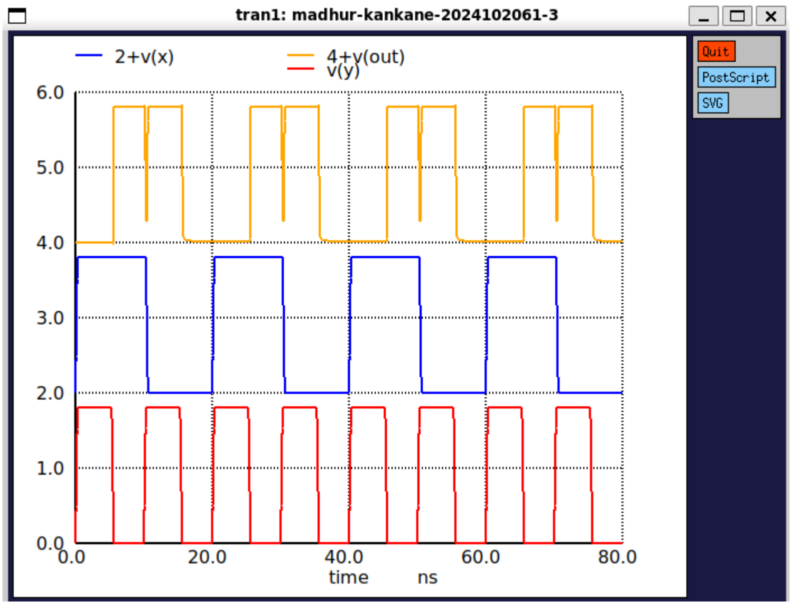


Figure 9

4.3 AND Gate

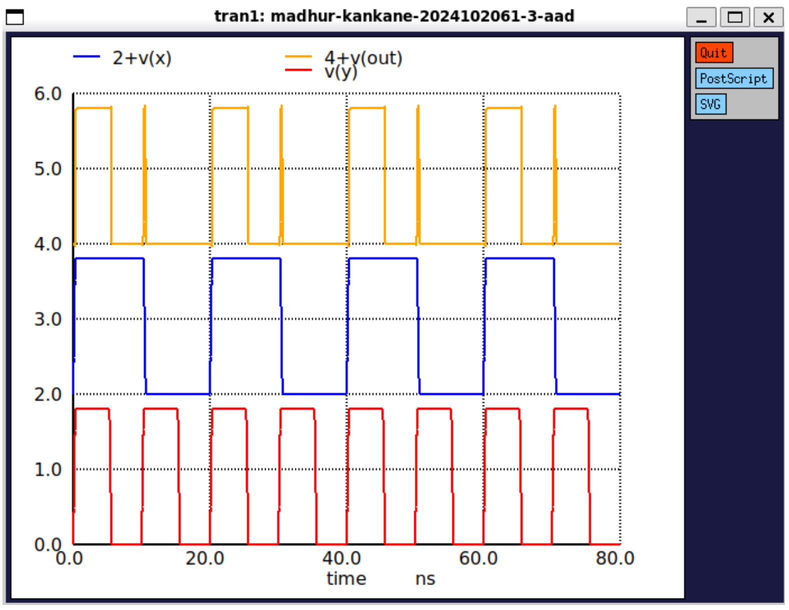


Figure 10

4.4 OR Gate

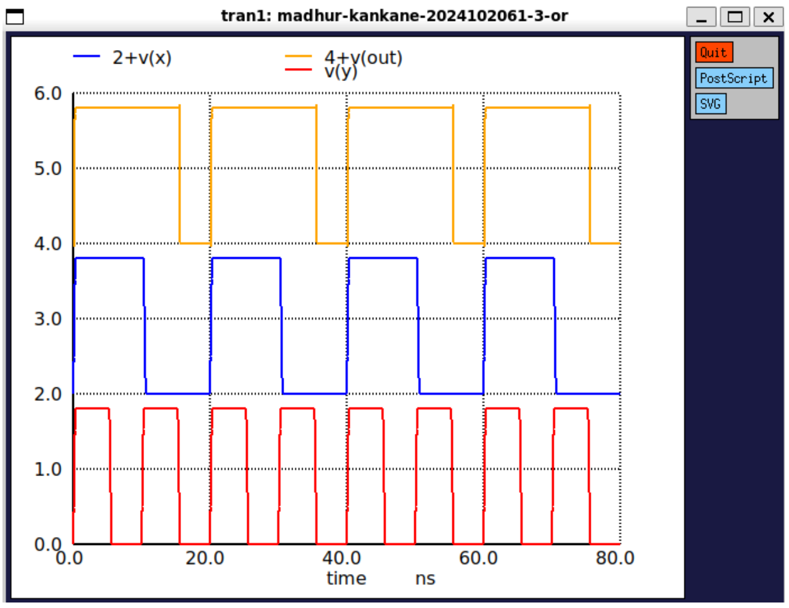


Figure 11

4.5 D Flip Flop

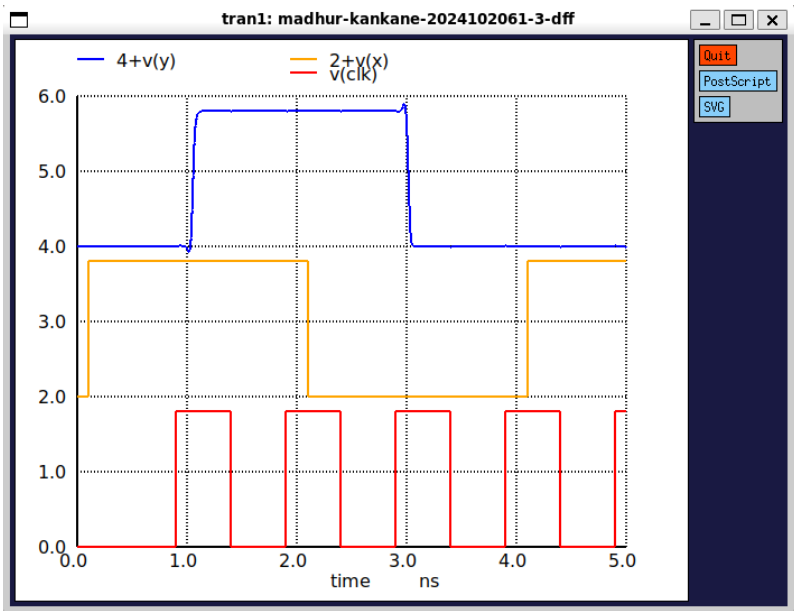
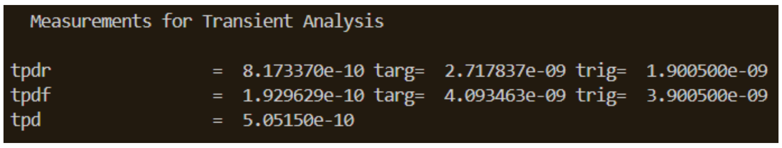


Figure 12

5 Determining Setup Time, Hold Time And Clock To Q Delay



```
  Measurements for Transient Analysis

tpdr                 =  8.173370e-10 targ=  2.717837e-09 trig=  1.900500e-09
tpdf                 =  1.929629e-10 targ=  4.093463e-09 trig=  3.900500e-09
tpd                  =  5.05150e-10
```

Figure 13

The schematic simulation plot shows that the clock to Q delay is measured to be 0.505 ns, representing the time it takes for the output to respond to a change in the clock signal.

5.1 Hold time

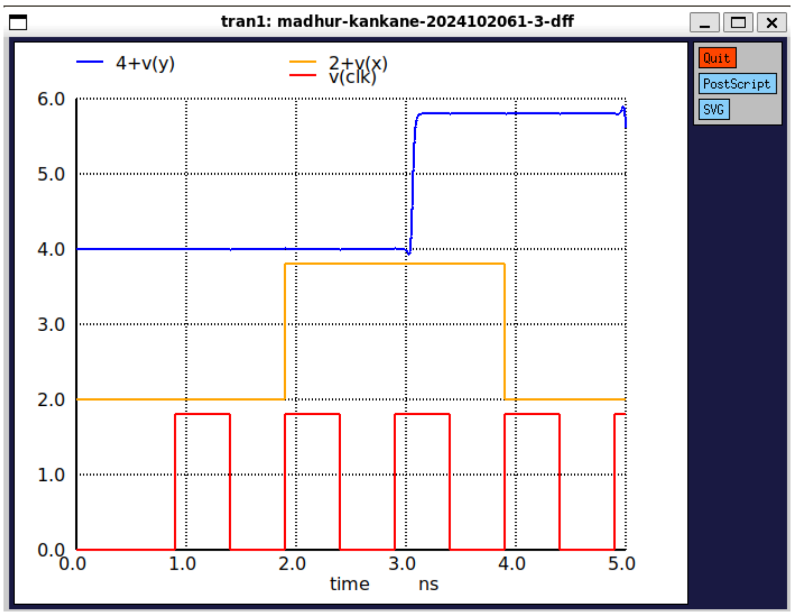


Figure 14

The plot illustrates that changing the input during the clock pulse does not affect the output, confirming that the hold time is zero, as the D flip-flop retains its output state even when the input changes at this point.

5.2 Set-up time

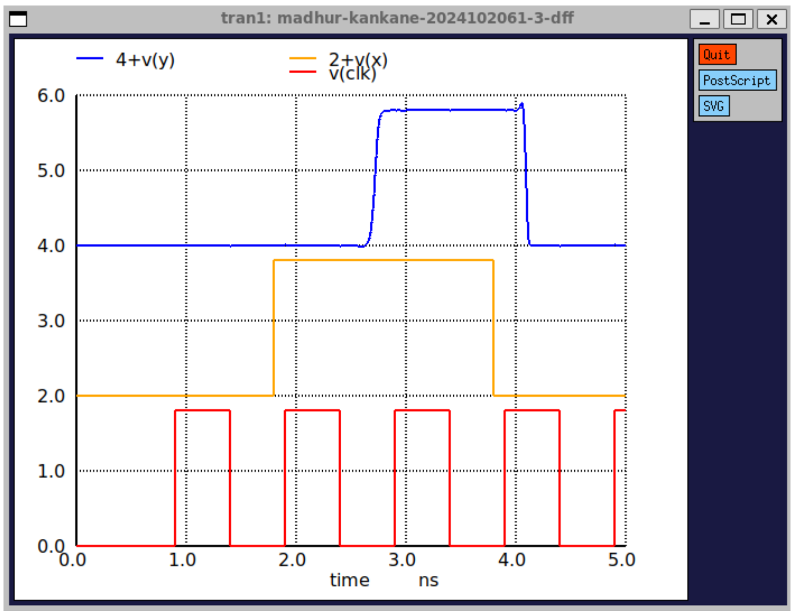


Figure 15: output changes at 1.792ns

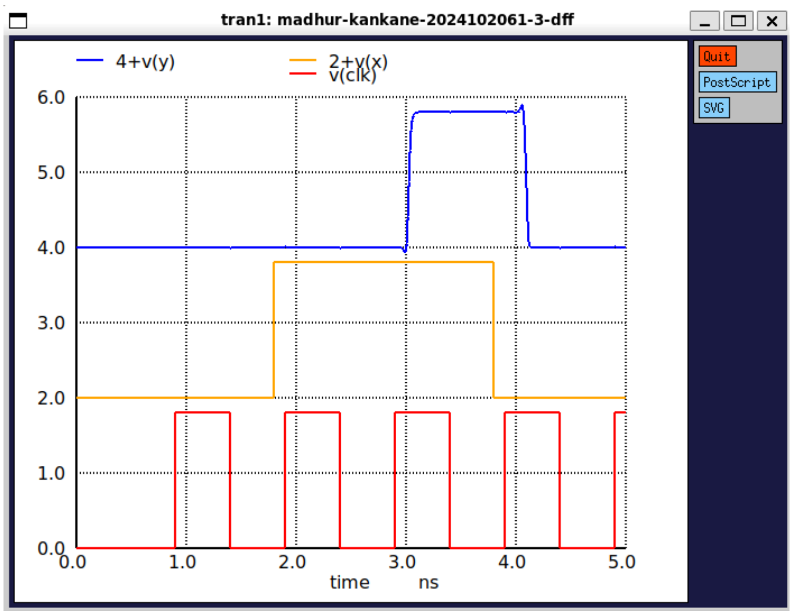


Figure 16: output does changes at 1.793ns

The plot demonstrates that the clock pulse arrives at 1.9 ns, and when the input changes at 1.792 ns, the output is correct. However, changing the input at 1.793 ns results in an incorrect output, indicating a setup time of 107 ps for the D flip-flop.
$\text{SetupTime} = 1.9\text{ns} - 1.793\text{ns} = 0.107\text{ns}$

6 Stick Diagrams of Blocks

Below is the implementation of the stick diagram for each gate used to implement this project, illustrating the physical layout of the gates and their interconnections.

6.1 Inverter

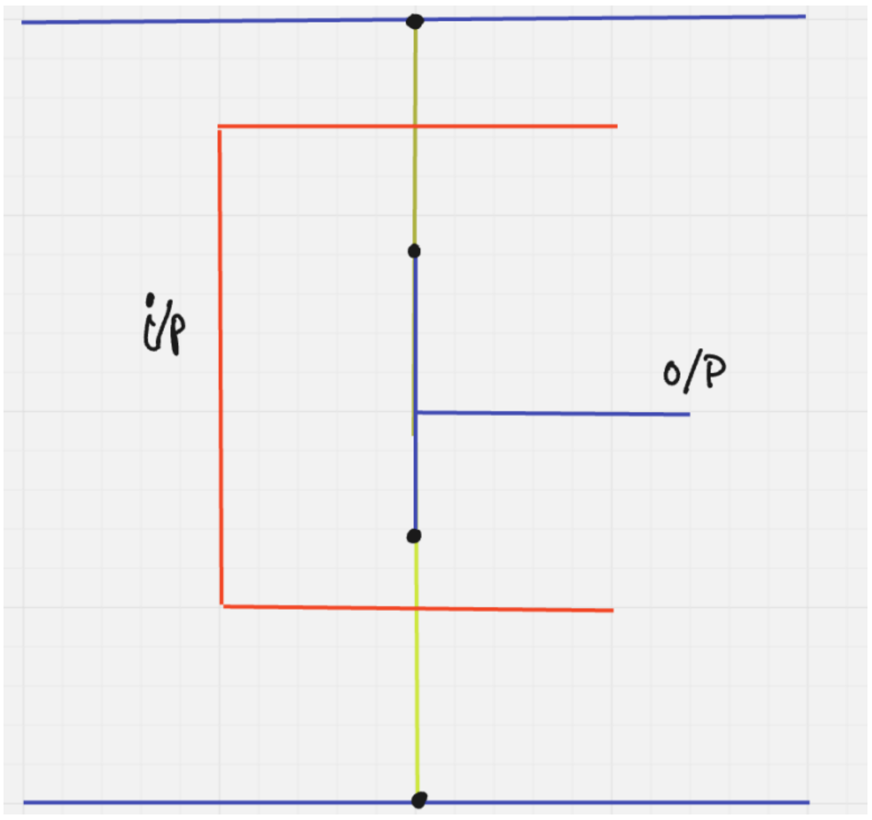


Figure 17

6.2 XOR Gate

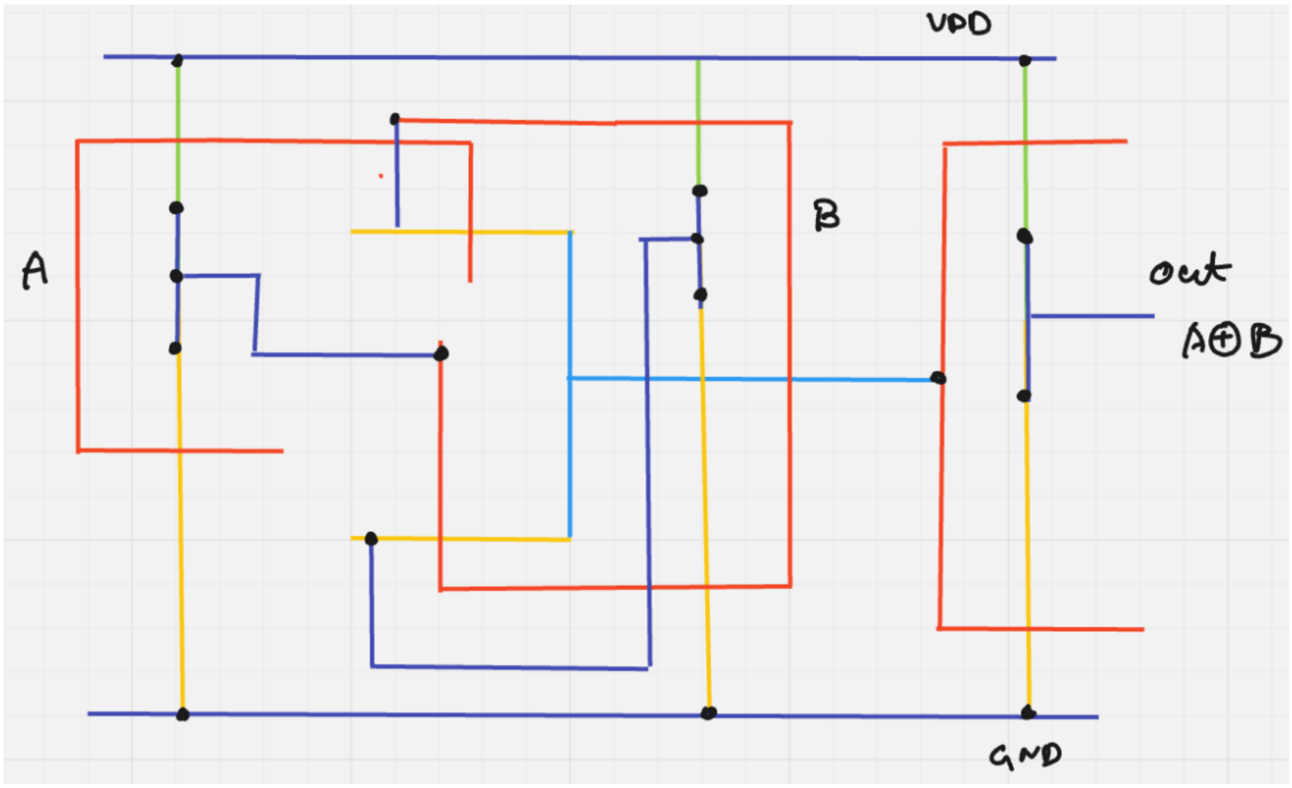


Figure 18

6.3 AND Gate

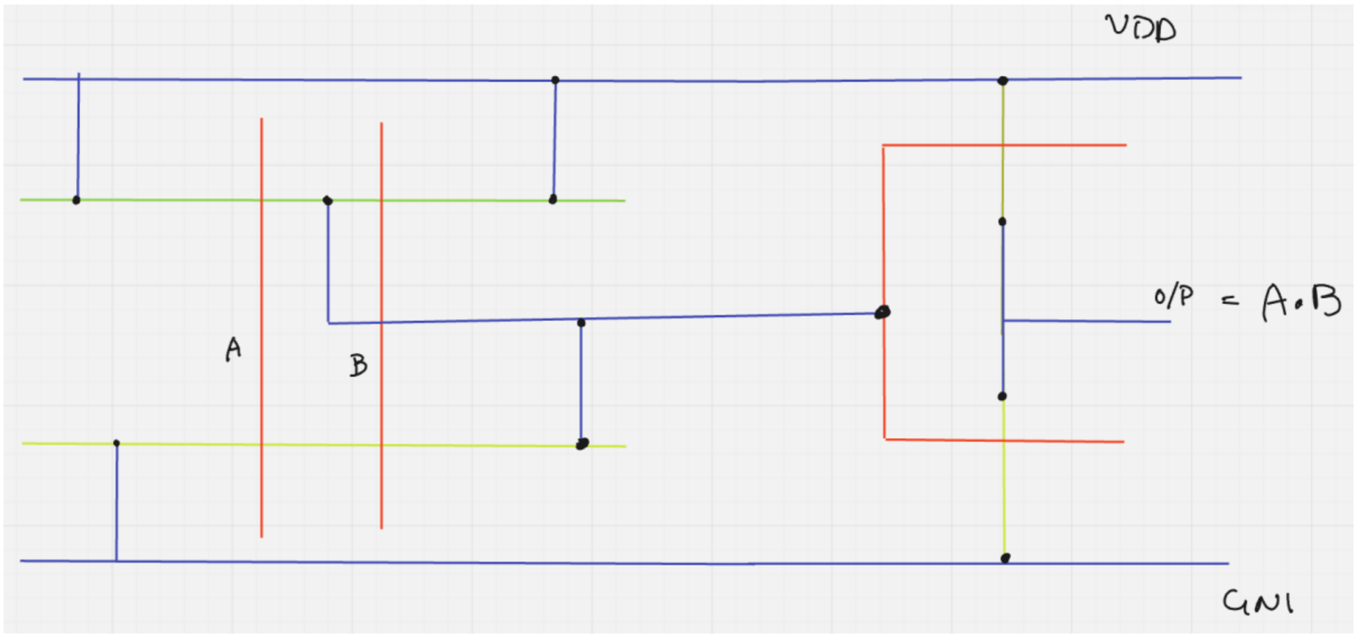


Figure 19

6.4 OR Gate

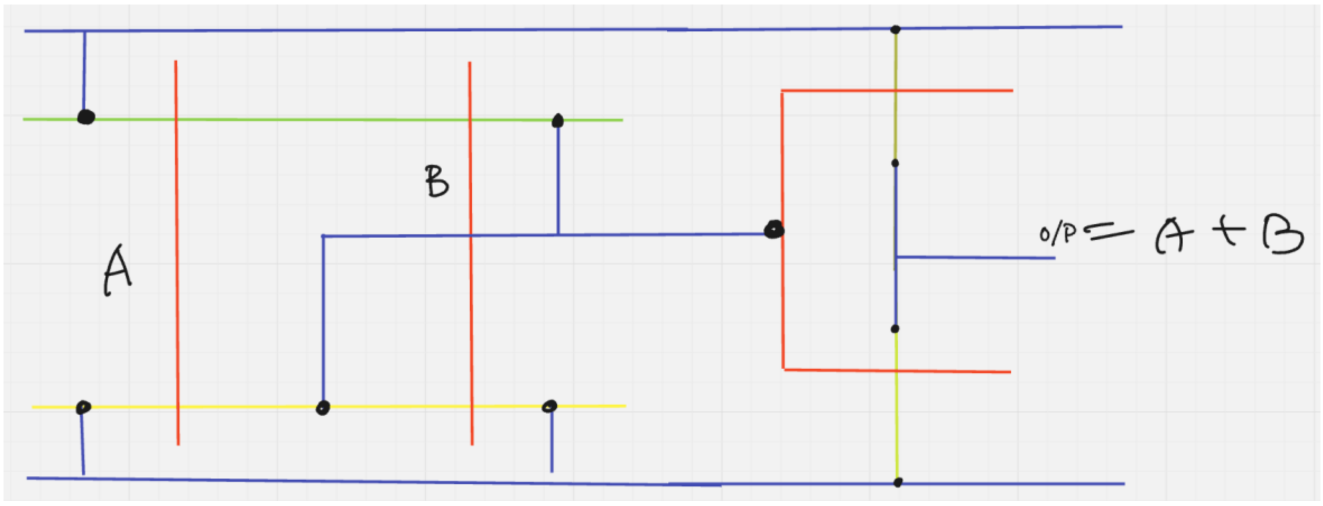


Figure 20

6.5 D Flip Flop

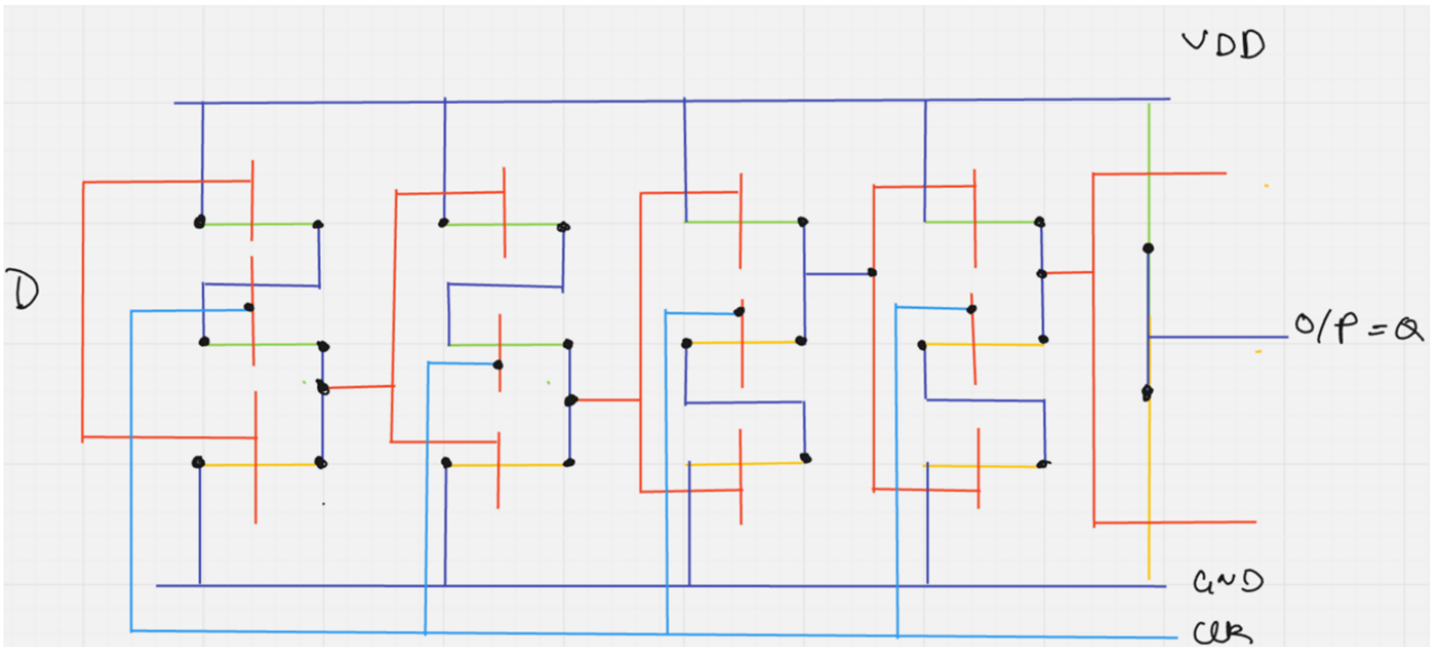


Figure 21

7 MAGIC Layouts and Post Layout Simulations of Blocks

7.1 Inverter

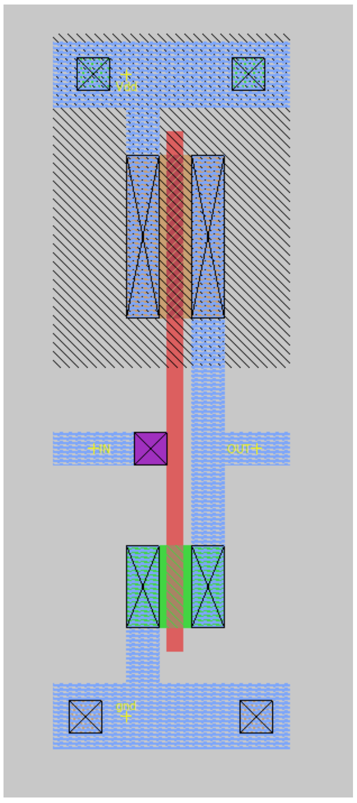


Figure 22

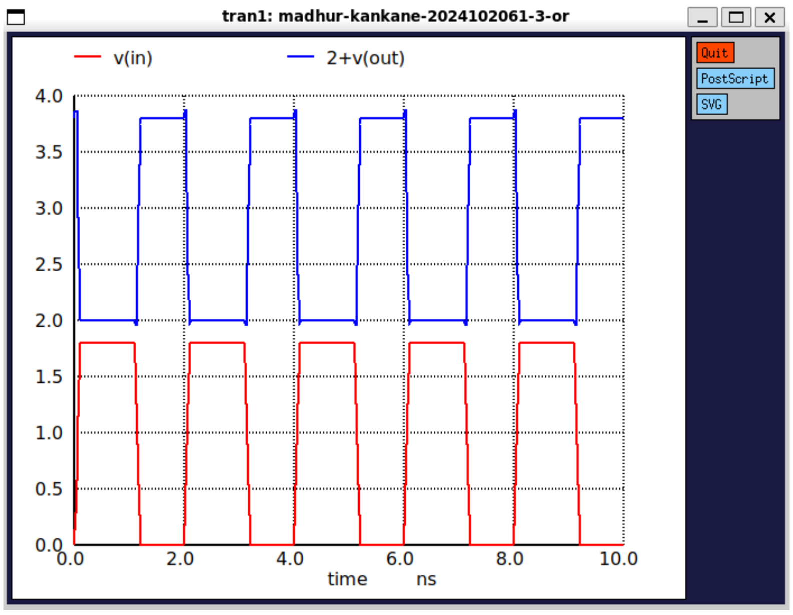


Figure 23

7.2 XOR Gate

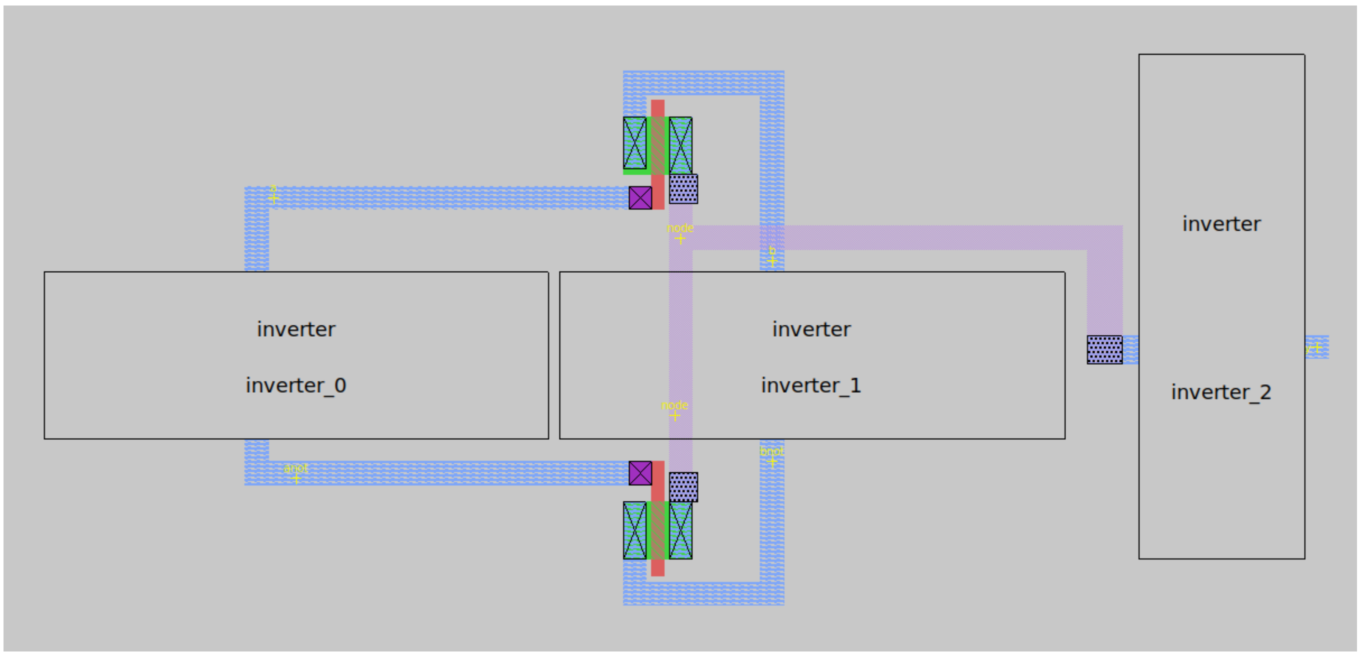


Figure 24

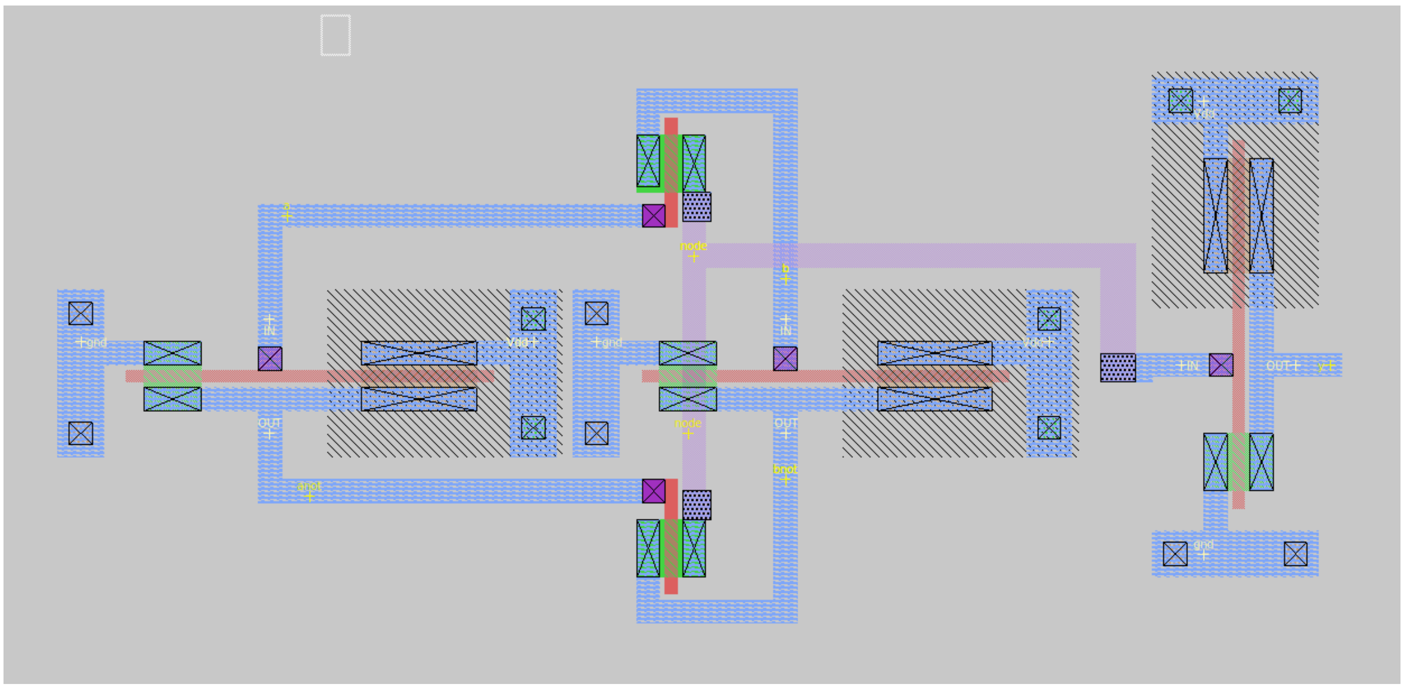


Figure 25

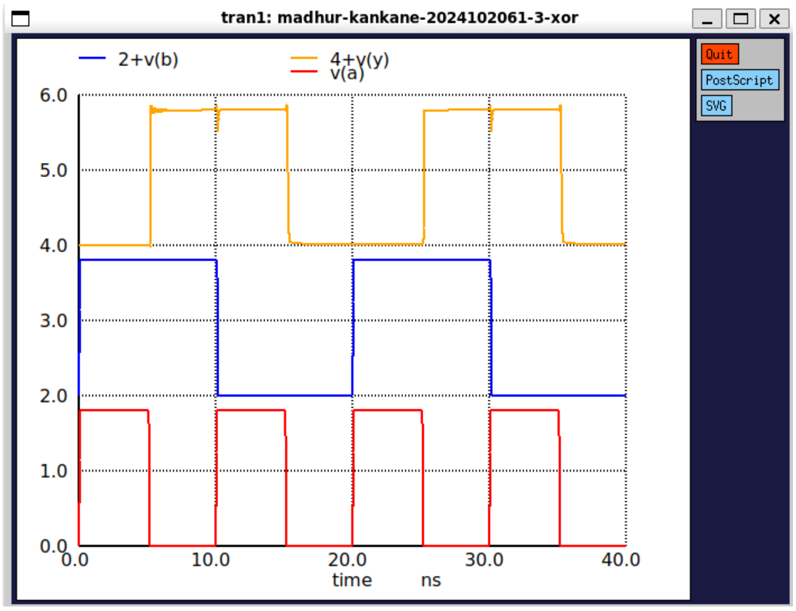


Figure 26

7.3 AND Gate

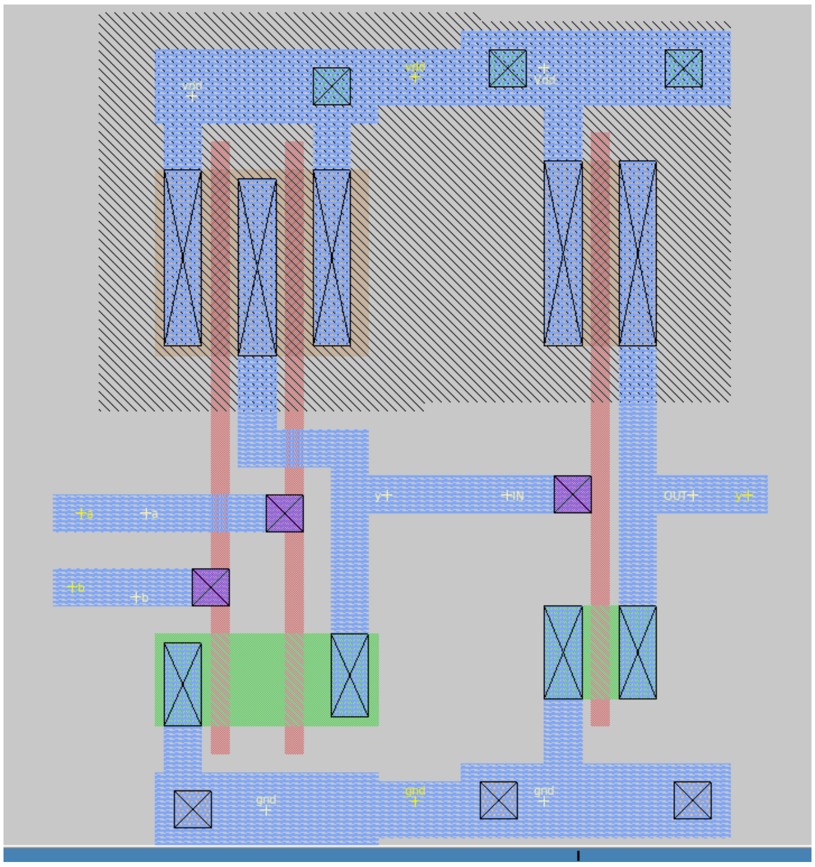


Figure 27

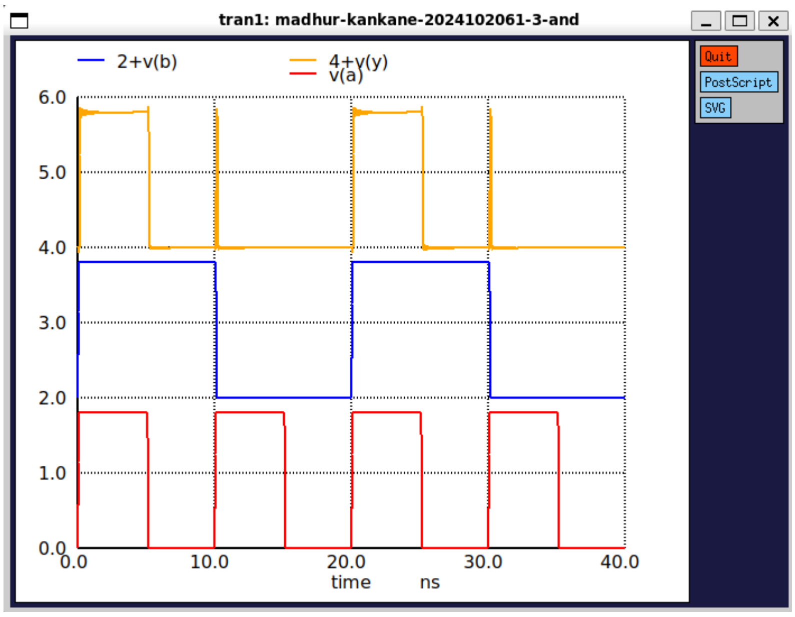


Figure 28

7.4 OR Gate

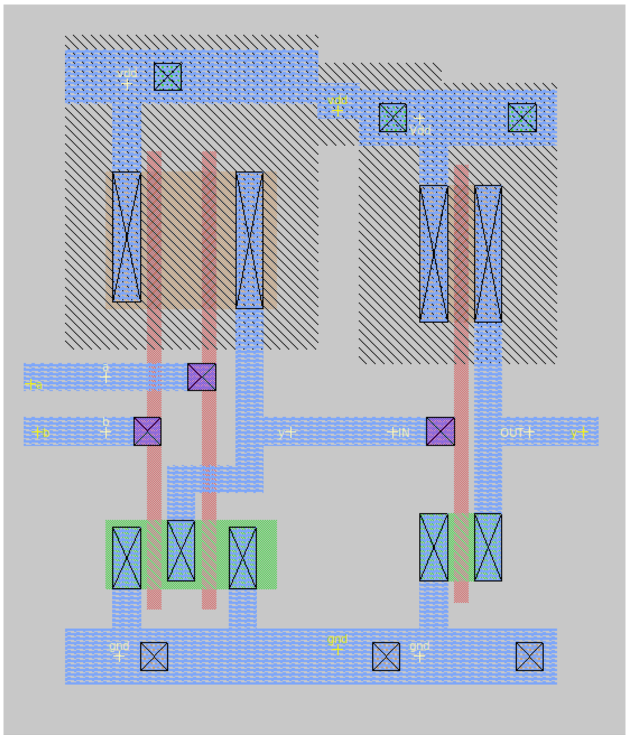


Figure 29

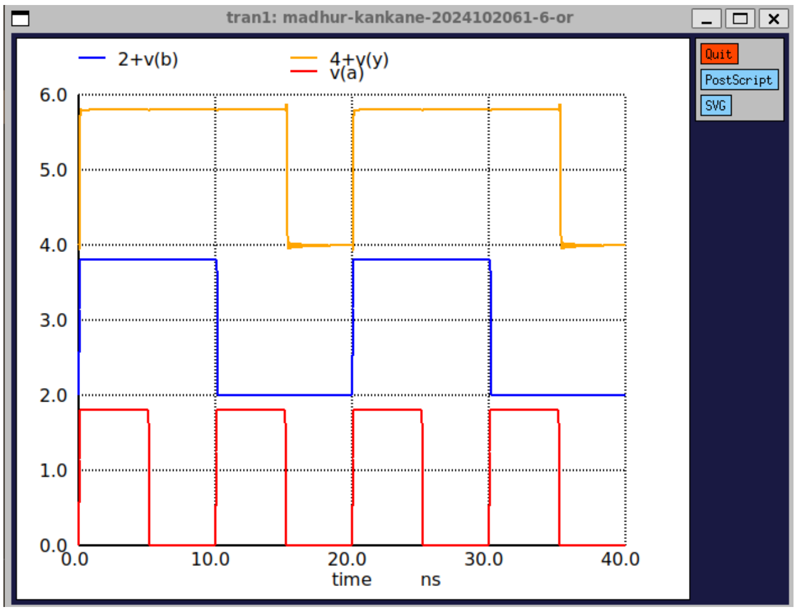


Figure 30

7.5 D Flip Flop

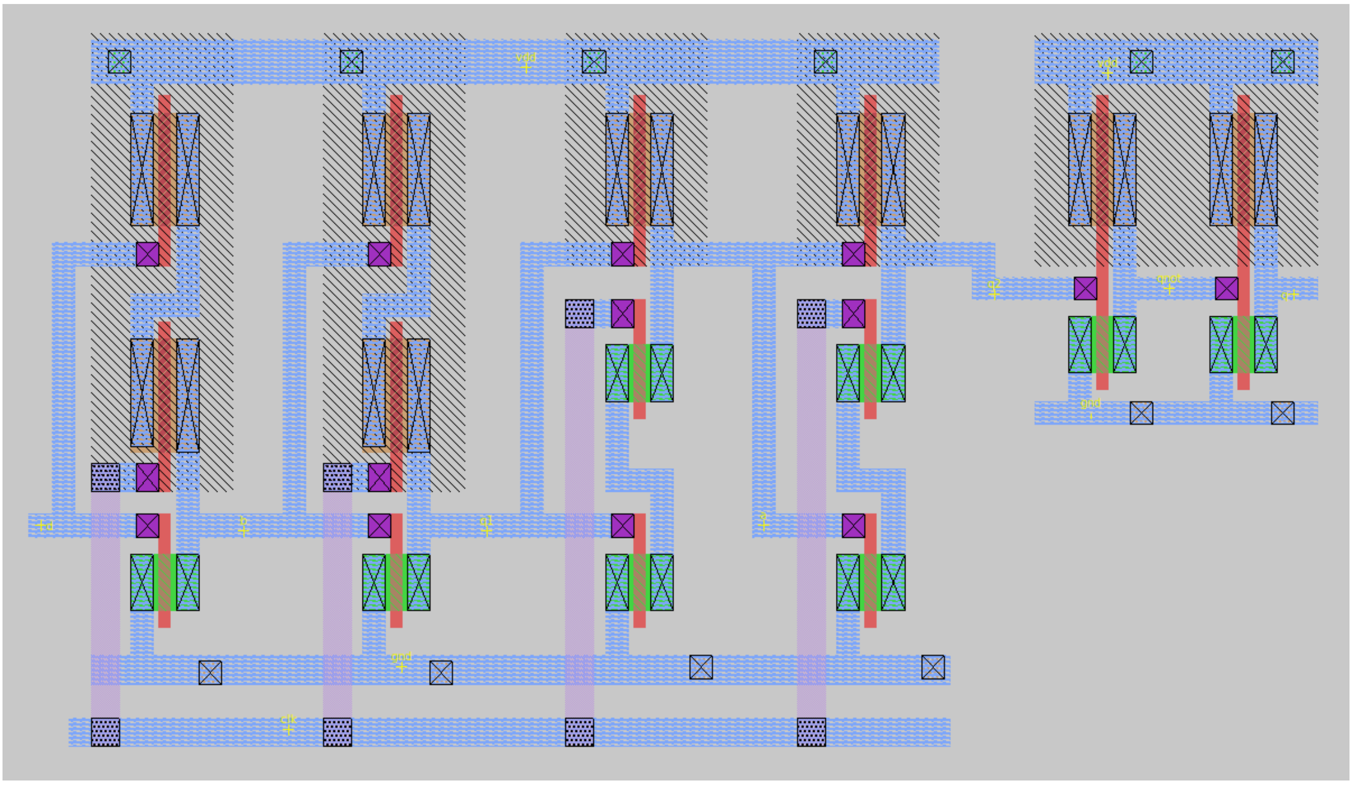


Figure 31

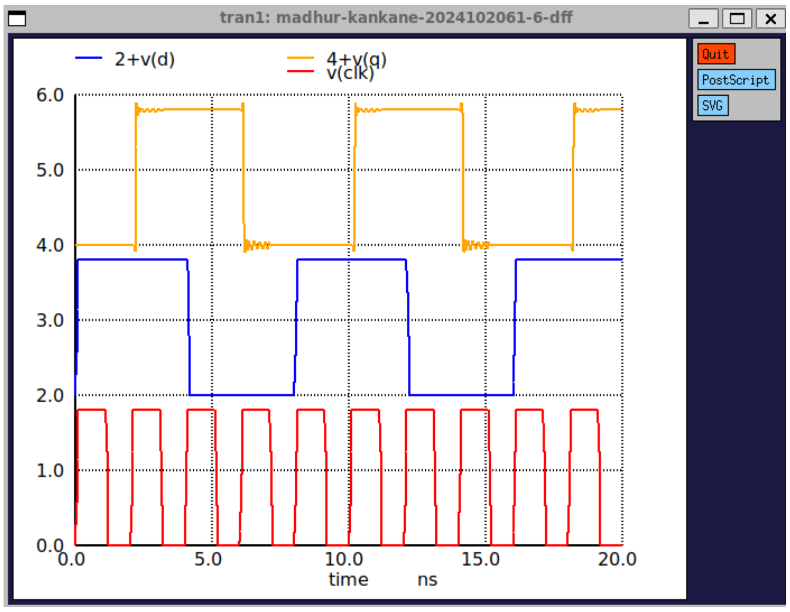
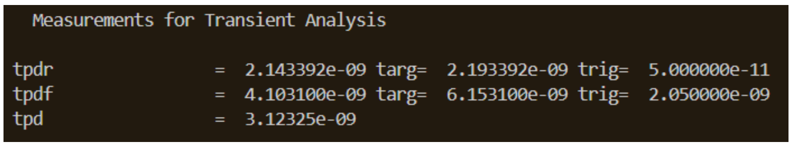


Figure 32

The Post layout simulation plot shows that the clock to Q delay is measured to be 3.12 ns, representing the time it takes for the output to respond to a change in the clock signal.



```
   Measurements for Transient Analysis

tpdr                 =  2.143392e-09 targ=  2.193392e-09 trig=  5.000000e-11
tpdf                 =  4.103100e-09 targ=  6.153100e-09 trig=  2.050000e-09
tpd                  =  3.12325e-09
```

Figure 33

8 Full Circuit Schematic Simulation

The inputs A4, A3, A2, A1, A0 and B4, B3, B2, B1, B0, CIN are fed to the D Flipflop. After one clock cycle, the updated values A3D, A2D, A1D, A0D, B3D, B2D, B1D, B0D, CIND are generated and processed within the D Flipflop and fed to CLA Block. The final outputs are S3, S2, S1, S0, COUT.

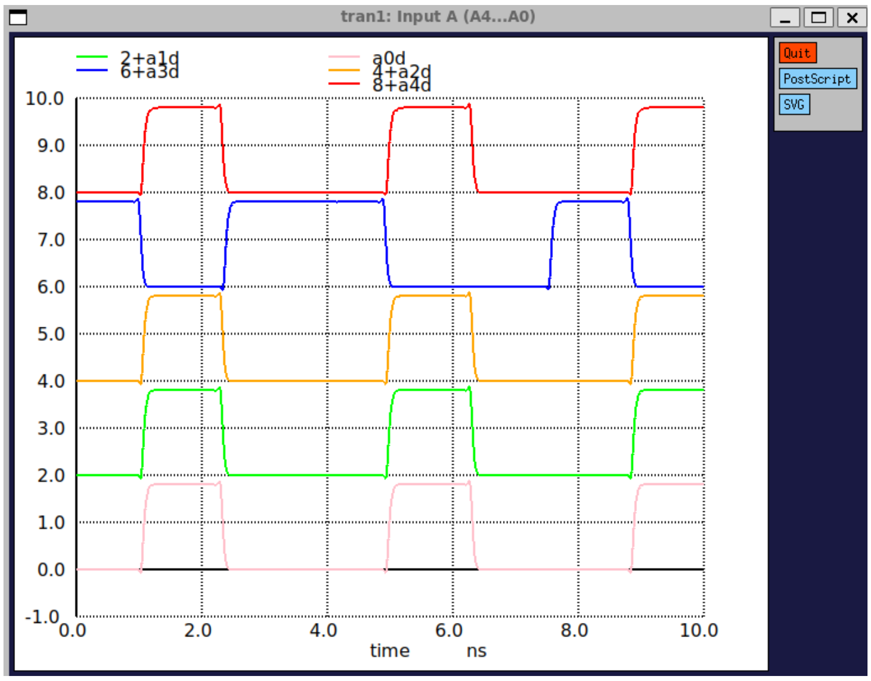


Figure 34

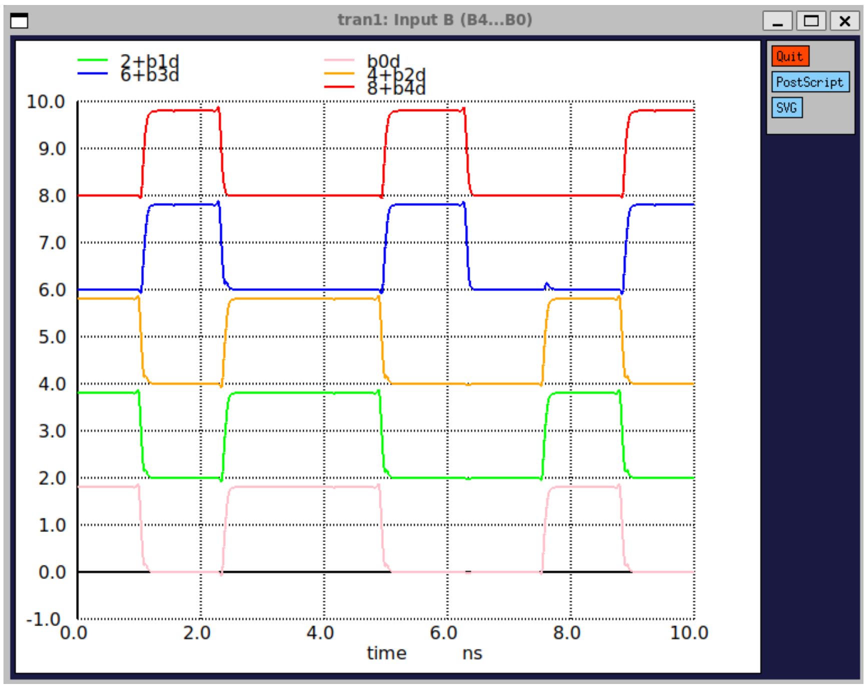


Figure 35

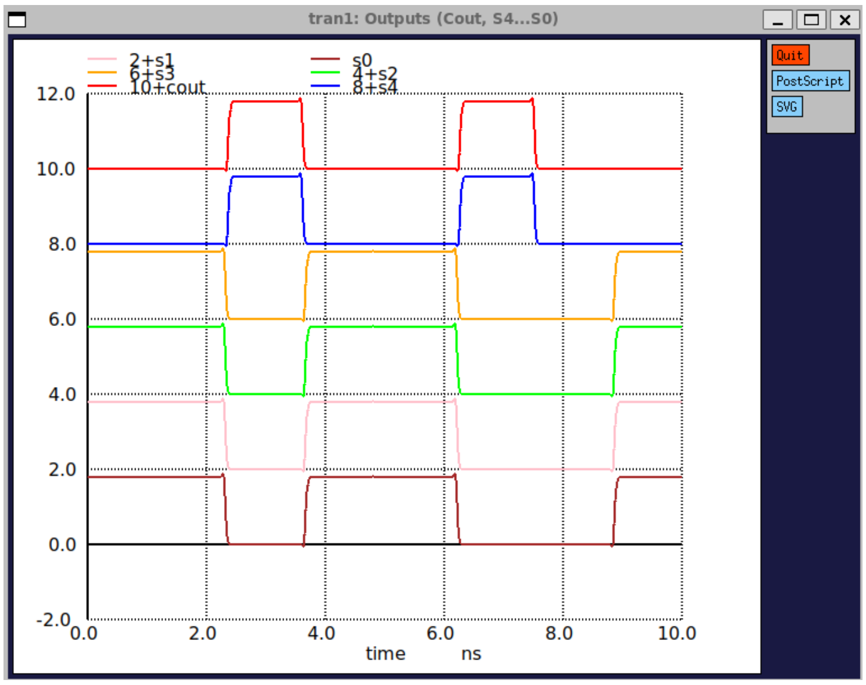


Figure 36

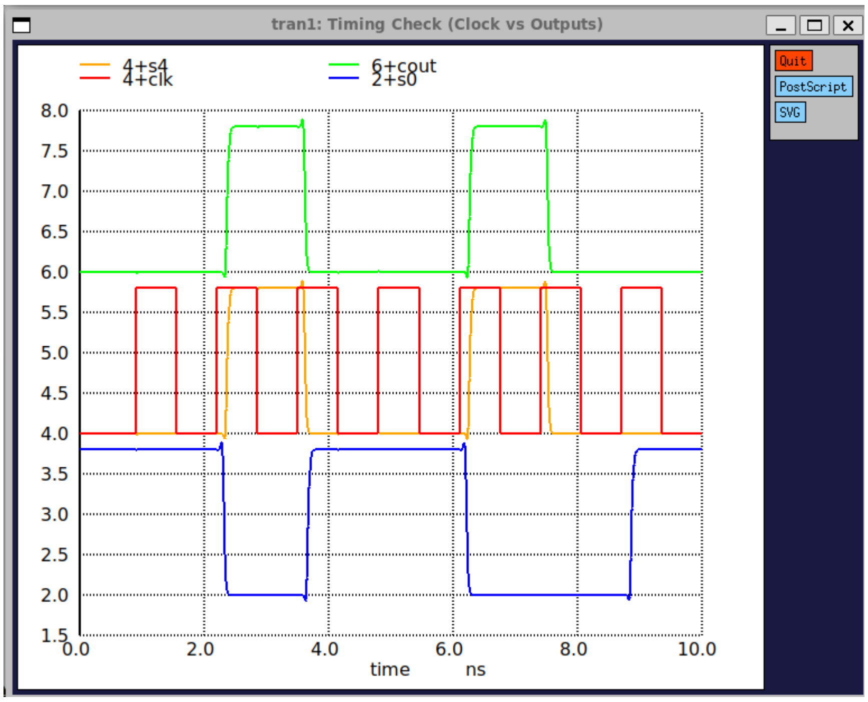


Figure 37

9 Floor Plan of The Layout

Below is the floor planning of the circuit, depicting the spatial arrangement of the components and gates used in the design. This layout optimizes the placement of each block to minimize routing delays and enhance performance.

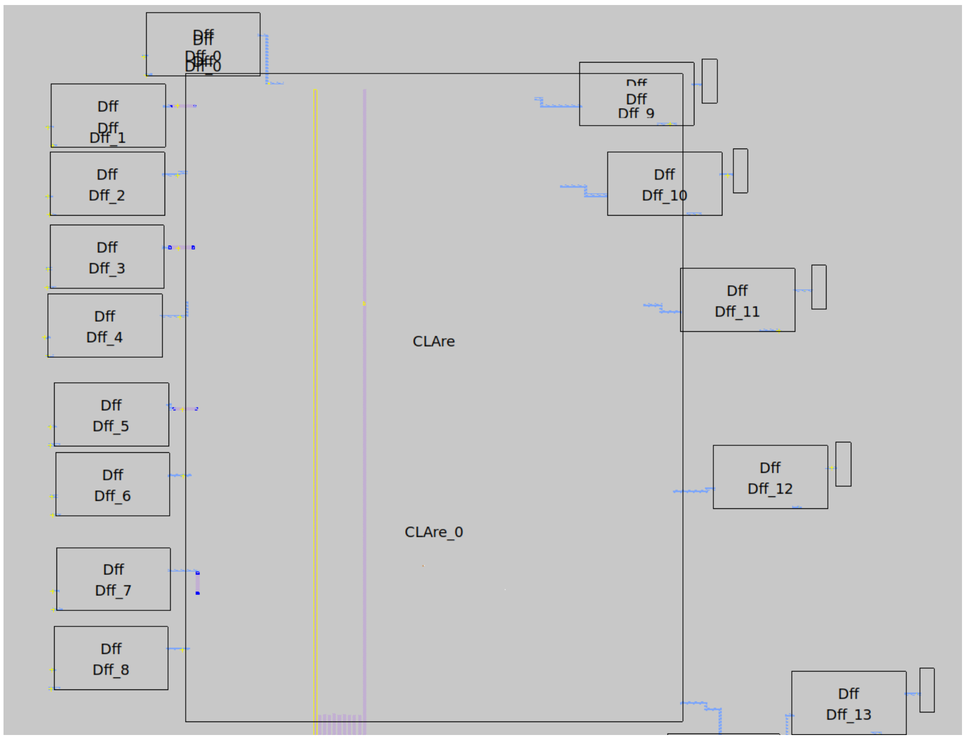


Figure 38

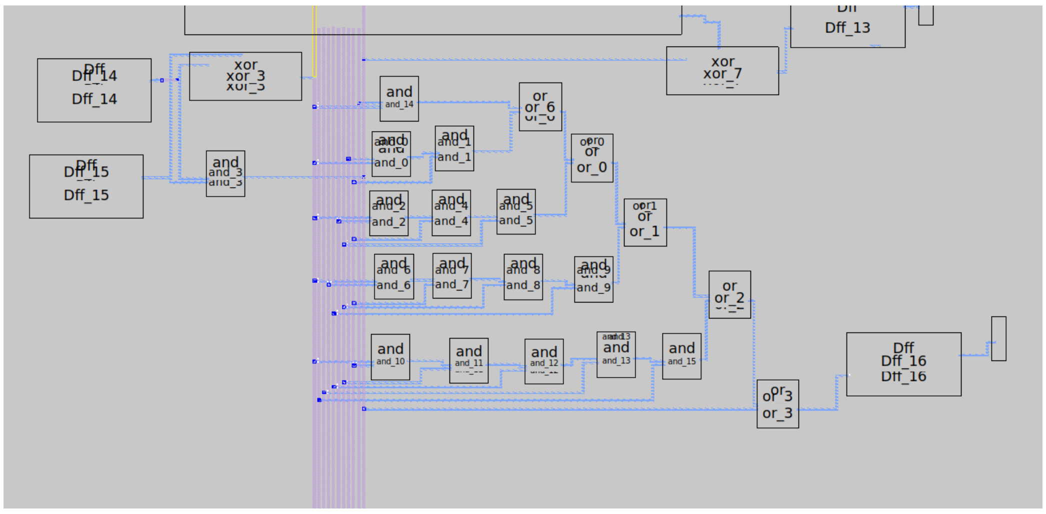


Figure 39

10 MAGIC Layout and Post layout Simulation

10.1 MAGIC Layout

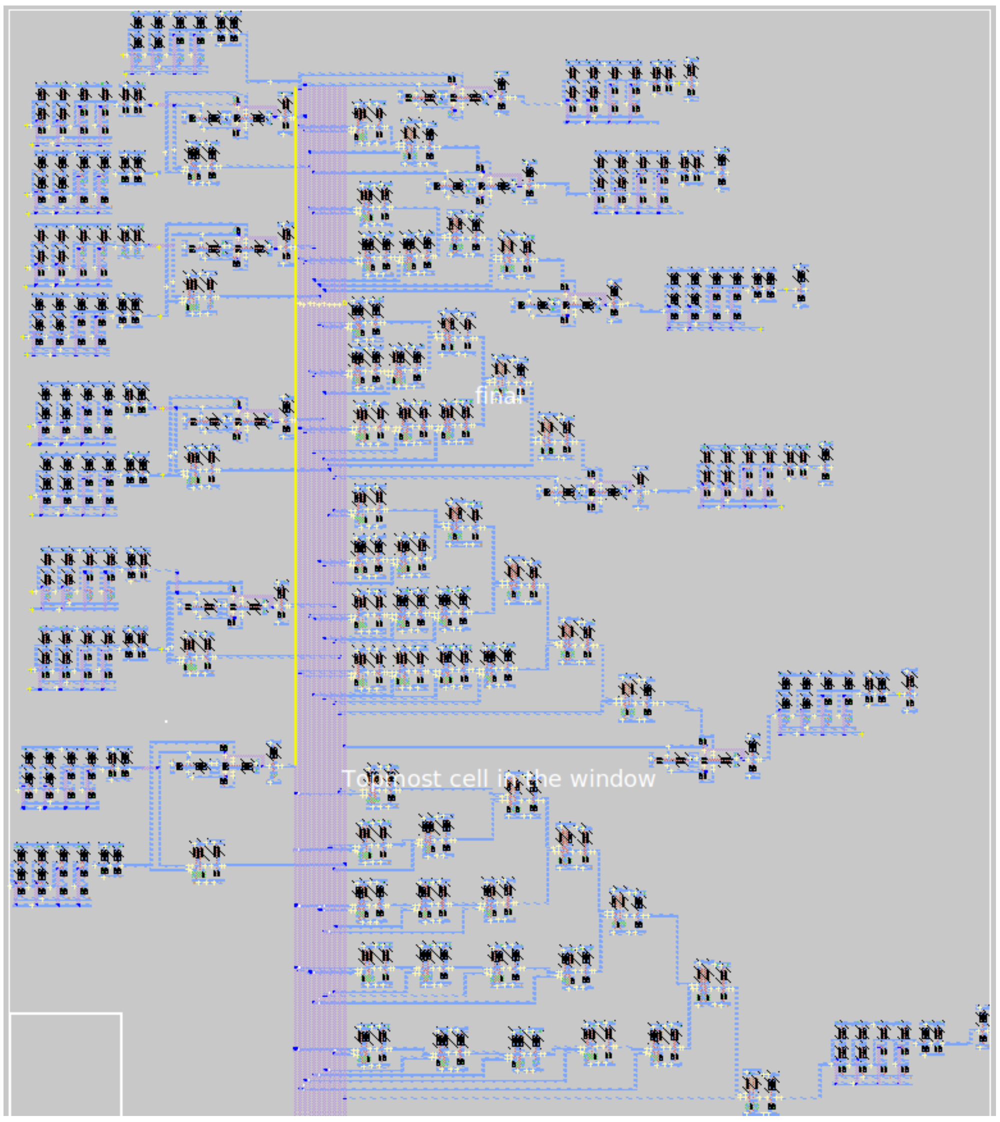


Figure 40

10.2 Post Layout SPICE simulation

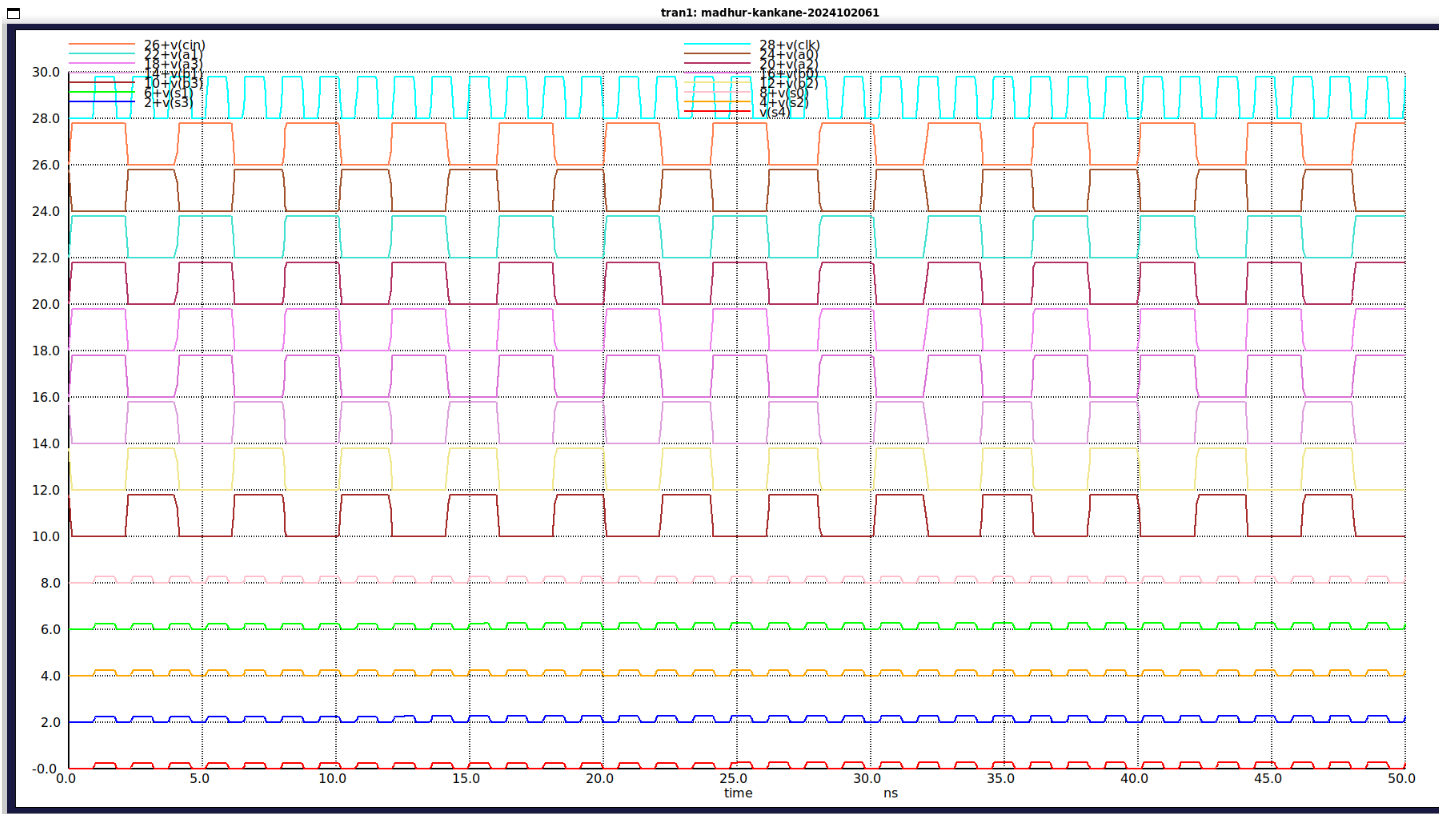
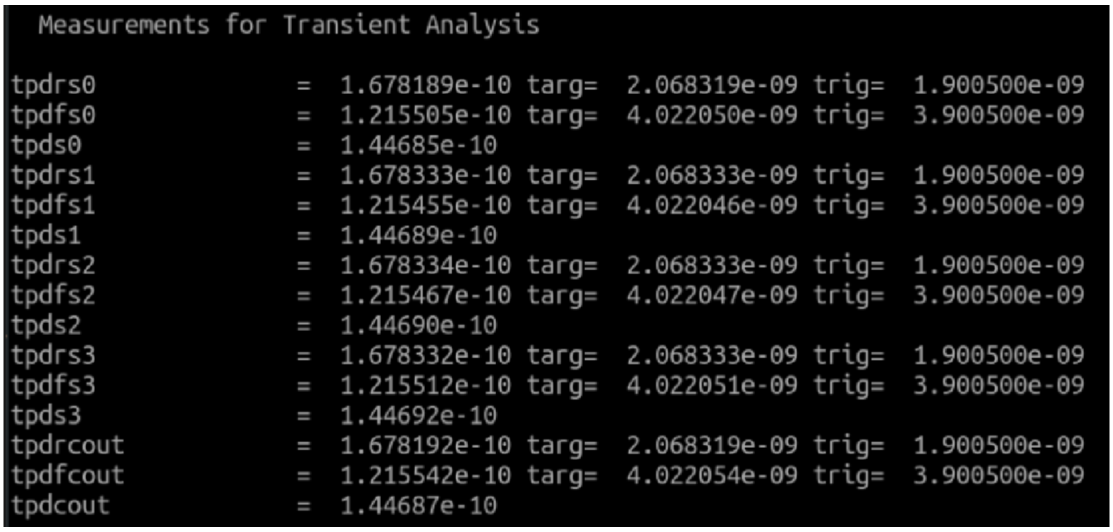


Figure 41

11 Pre and Post Layout Delay comparisions

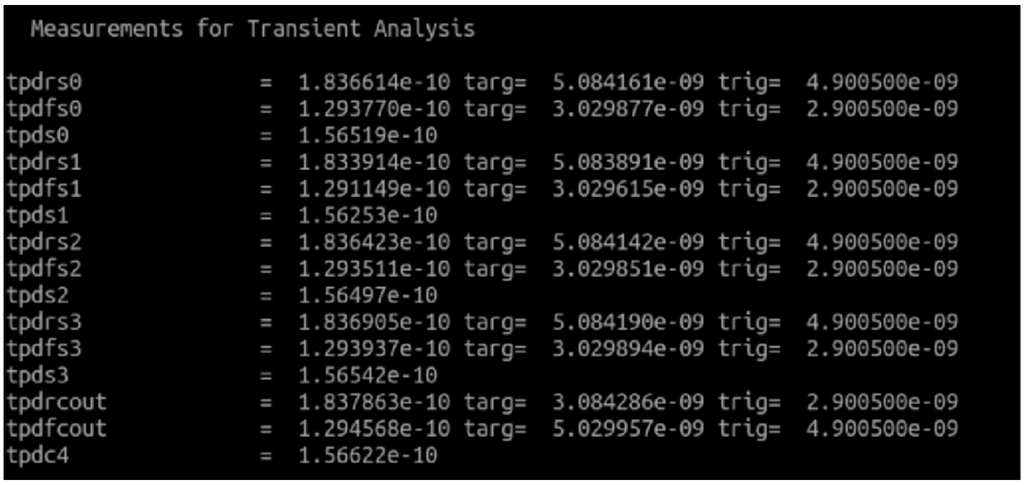
The post layout delay of the circuit has increase as compared to the prelayout delays. The increase in delay is mostly accounted due to the increase in parasitics.



```
  Measurements for Transient Analysis

tpdrs0                  =  1.678189e-10 targ=  2.068319e-09 trig=  1.900500e-09
tpdfs0                  =  1.215505e-10 targ=  4.022050e-09 trig=  3.900500e-09
tpds0                   =  1.44685e-10
tpdrs1                  =  1.678333e-10 targ=  2.068333e-09 trig=  1.900500e-09
tpdfs1                  =  1.215455e-10 targ=  4.022046e-09 trig=  3.900500e-09
tpds1                   =  1.44689e-10
tpdrs2                  =  1.678334e-10 targ=  2.068333e-09 trig=  1.900500e-09
tpdfs2                  =  1.215467e-10 targ=  4.022047e-09 trig=  3.900500e-09
tpds2                   =  1.44690e-10
tpdrs3                  =  1.678332e-10 targ=  2.068333e-09 trig=  1.900500e-09
tpdfs3                  =  1.215512e-10 targ=  4.022051e-09 trig=  3.900500e-09
tpds3                   =  1.44692e-10
tpdrcout                =  1.678192e-10 targ=  2.068319e-09 trig=  1.900500e-09
tpdfcout                =  1.215542e-10 targ=  4.022054e-09 trig=  3.900500e-09
tpdcout                 =  1.44687e-10
```

Figure 42: pre delay of the circuit



```
 Measurements for Transient Analysis

tpdrs0              =  1.836614e-10 targ=  5.084161e-09 trig=  4.900500e-09
tpdfs0              =  1.293770e-10 targ=  3.029877e-09 trig=  2.900500e-09
tpds0               =  1.56519e-10
tpdrs1              =  1.833914e-10 targ=  5.083891e-09 trig=  4.900500e-09
tpdfs1              =  1.291149e-10 targ=  3.029615e-09 trig=  2.900500e-09
tpds1               =  1.56253e-10
tpdrs2              =  1.836423e-10 targ=  5.084142e-09 trig=  4.900500e-09
tpdfs2              =  1.293511e-10 targ=  3.029851e-09 trig=  2.900500e-09
tpds2               =  1.56497e-10
tpdrs3              =  1.836905e-10 targ=  5.084190e-09 trig=  4.900500e-09
tpdfs3              =  1.293937e-10 targ=  3.029894e-09 trig=  2.900500e-09
tpds3               =  1.56542e-10
tpdrcout            =  1.837863e-10 targ=  3.084286e-09 trig=  2.900500e-09
tpdfcout            =  1.294568e-10 targ=  5.029957e-09 trig=  4.900500e-09
tpdc4               =  1.56622e-10
```

Figure 43: Post layout delay of the circuit

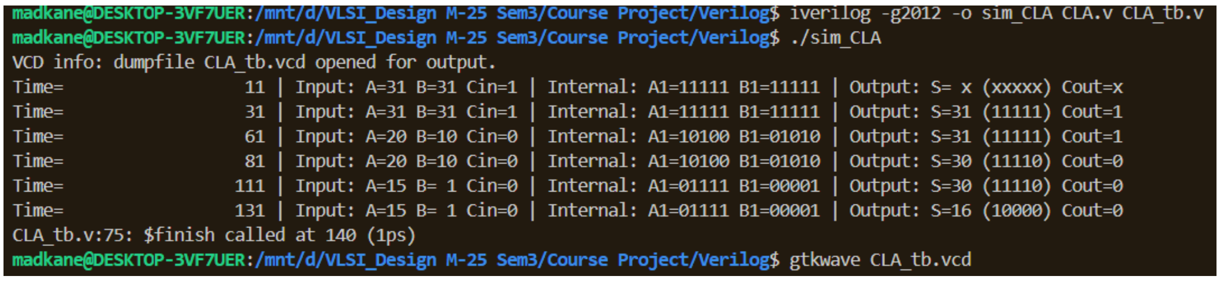
Table 1: COMPARISON OF PRE-LAYOUT AND POST-LAYOUT RESULTS

Quantity	**Pre-Layout**	**Post-Layout**
Tpcq	0.144ms	0.156ns
Tcla(max)	0.44ns	0.62ns
Tclk(min)	0.9ns	1.2ns
freq(max)	1.1GHz	833MHz

The minimum time period of the clock must be greater than or equal to $tpcq + tcla + tsetup$. Therefore, the minimum time period of our circuit prelayout was found to be around 0.8ns and for post layout it was something around 0.9ns. The minimum time period in which our circuit works correctly comes out to be 0.9ns for schematic, and for post layout it was 1.2ns.

12 Verilog Simulation and Waveforms

he following are the outputs from the structural implementation of the Verilog code for the



```
madkane@DESKTOP-3VF7UER:/mnt/d/VLSI_Design M-25 Sem3/Course Project/Verilog$ iverilog -g2012 -o sim_CLA CLA.v CLA_tb.v
madkane@DESKTOP-3VF7UER:/mnt/d/VLSI_Design M-25 Sem3/Course Project/Verilog$ ./sim_CLA
VCD info: dumpfile CLA_tb.vcd opened for output.
Time=                  11 | Input: A=31 B=31 Cin=1 | Internal: A1=11111 B1=11111 | Output: S= x (xxxxx) Cout=x
Time=                  31 | Input: A=31 B=31 Cin=1 | Internal: A1=11111 B1=11111 | Output: S=31 (11111) Cout=1
Time=                  61 | Input: A=20 B=10 Cin=0 | Internal: A1=10100 B1=01010 | Output: S=31 (11111) Cout=1
Time=                  81 | Input: A=20 B=10 Cin=0 | Internal: A1=10100 B1=01010 | Output: S=30 (11110) Cout=0
Time=                 111 | Input: A=15 B= 1 Cin=0 | Internal: A1=01111 B1=00001 | Output: S=30 (11110) Cout=0
Time=                 131 | Input: A=15 B= 1 Cin=0 | Internal: A1=01111 B1=00001 | Output: S=16 (10000) Cout=0
CLA_tb.v:75: $finish called at 140 (1ps)
madkane@DESKTOP-3VF7UER:/mnt/d/VLSI_Design M-25 Sem3/Course Project/Verilog$ gtkwave CLA_tb.vcd
```

Figure 44: Terminal Outputs

The gtkwave plots for the same set of inputs.

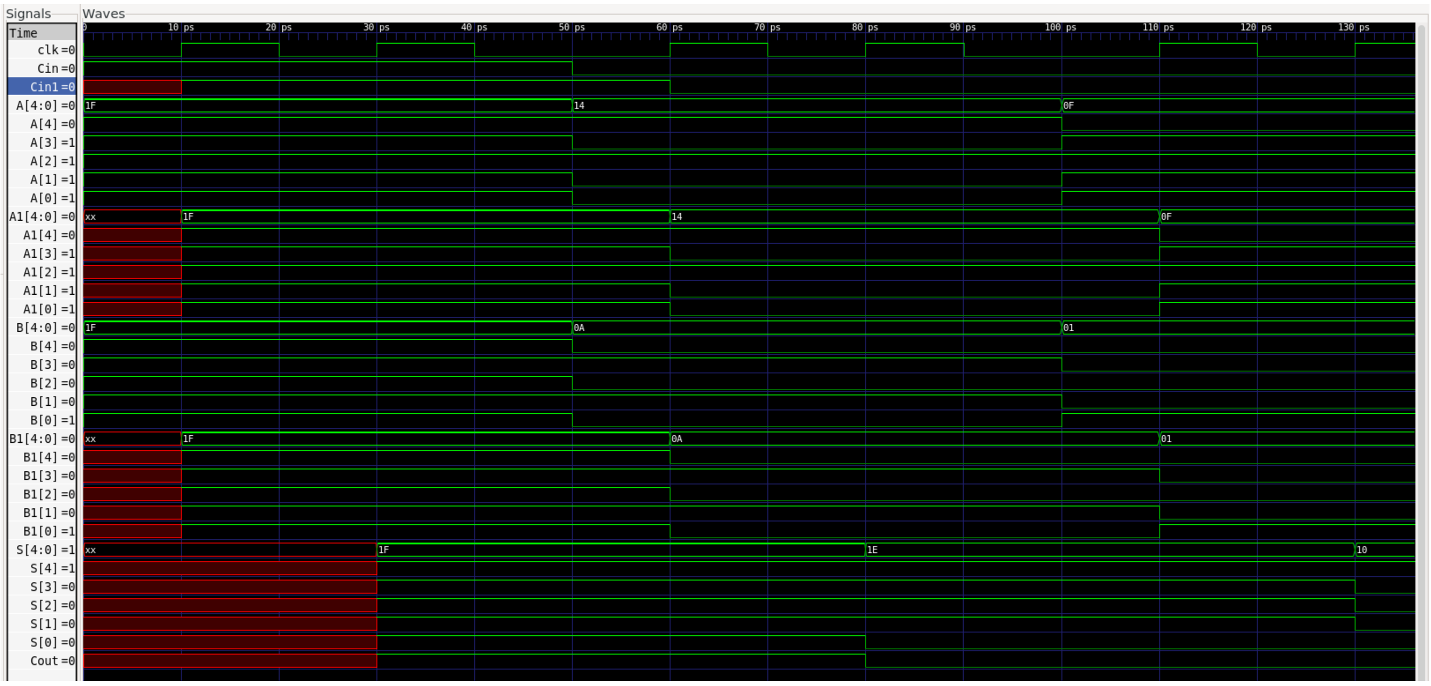


Figure 45: GTK Waveofrms for 5 bit CLA

A (5-bit)	B (5-bit)	CIN	COUT	S (5-bit)
11111	11111	1	1	11111
10100	01010	0	0	11110
01111	00001	0	0	10000

It can be observed that the inputs are made available to the CLA after the first clock edge, and the output is obtained at the second clock edge

13 FPGA Implementation

The same verilog code was synthesized and run on the FPGA. The following are the outputs observed for the inputs.

13.1 Example Testcases

Given Inputs:

$A = 10101$ (Decimal 21)
$B = 01010$ (Decimal 10)
$C_{\text{in}} = 1$
Output:
$S = 00000$ (Sum)
$C_{\text{out}} = 1$

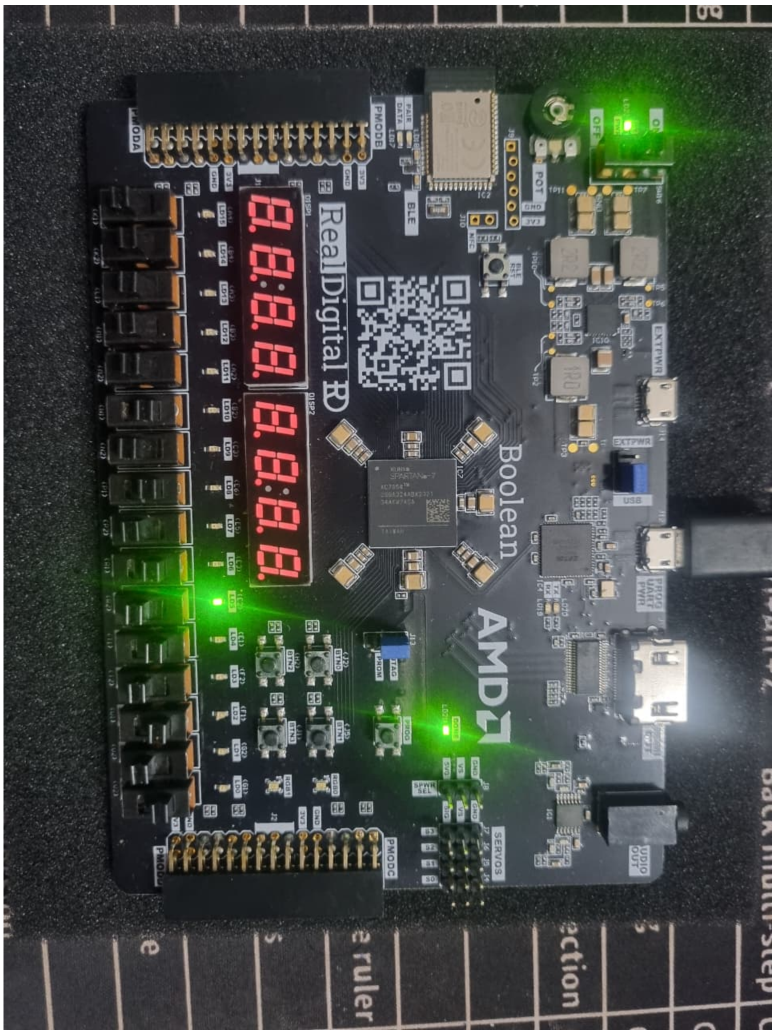


Figure 46: Terminal Outputs

Given Inputs:
$A = 11011$ (Decimal 27)
$B = 00110$ (Decimal 5)
$C_{\text{in}} = 0$
Output:
$S = 00001$ (Sum)
$C_{\text{out}} = 1$

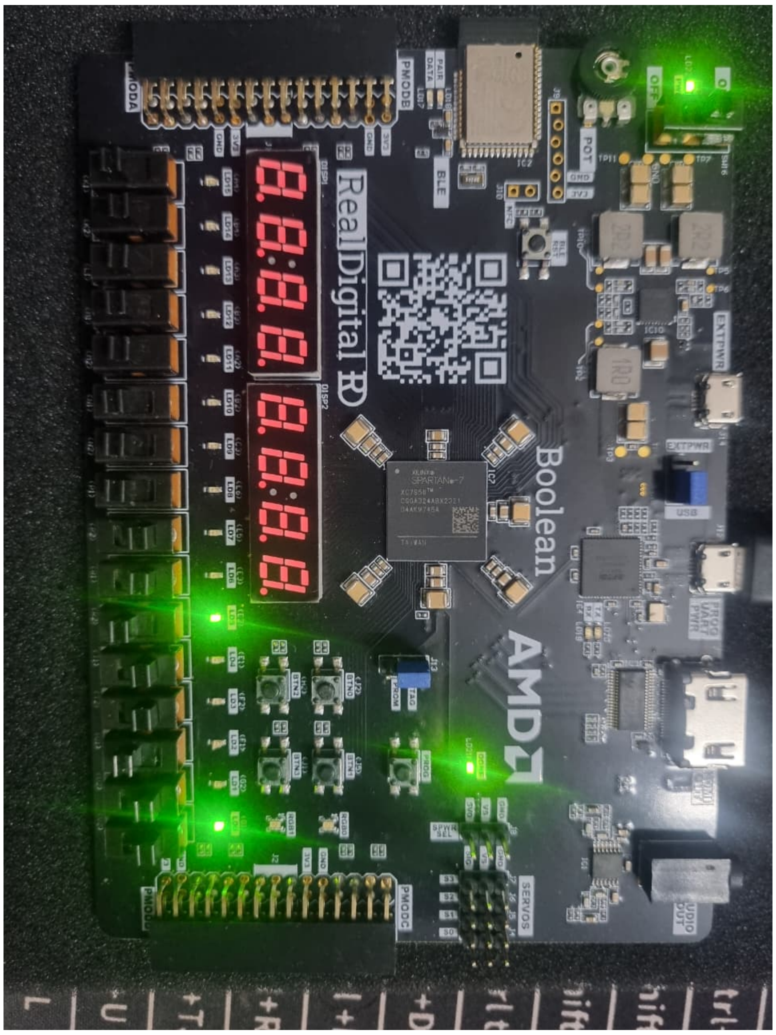


Figure 47: Terminal Outputs

Given Inputs:
$A = 11010$ (Decimal 26)
$B = 01101$ (Decimal 13)
$C_{\text{in}} = 0$
Output:
$S = 00111$ (Sum)
$C_{\text{out}} = 1$

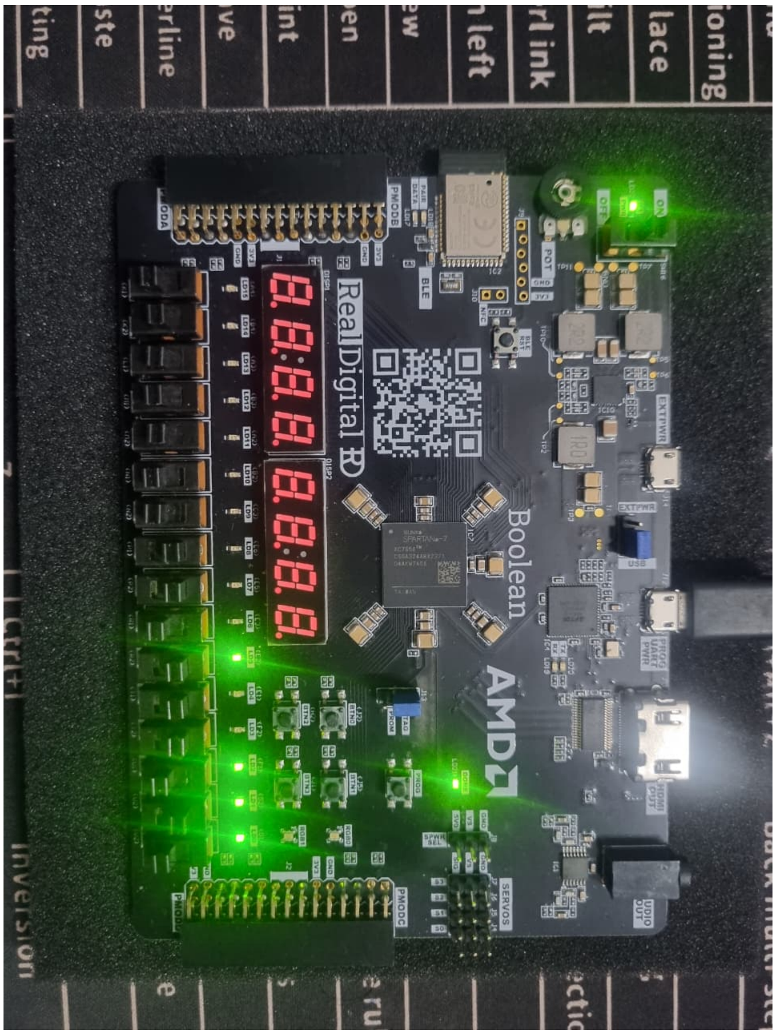


Figure 48: Terminal Outputs