4-bit carry look ahead (CLA) adder

Aryan Shrivastava
Electronics and Communication Engineering
International Institute of Information Technology
Hyderabad, India
2023102025, aryan.s@students.iiit.ac.in

***Abstract*—The project involves designing a 4-bit carry-lookahead (CLA) adder using a given 180 nm technology file for NGSPICE simulations and MAGIC layout. The project emphasizes layout and post-layout simulation results and incorporates D-flip-flops to meet timing constraints. Tasks include verifying functionality, calculating delays, creating stick diagrams, extracting SPICE netlists, and determining operational clock frequencies.**

I. Introduction

In this project, I designed a 4-bit carry look-ahead adder circuit utilizing the CMOS logic style to build the logic gates. Our objective was to reduce the power delay product (PDP) as much as possible relative to the traditional 4-bit CLA adder. My design incorporates multiple input XOR and NAND gates, since NAND gates have a lesser delay compared to AND and OR gates [1].

***Index Terms*—CLA, CMOS, Static Logic, flip flops**

II. CLA adder bit calculation

Let us have two 4-bit numbers $a_3a_2a_1a_0$ and $b_3b_2b_1b_0$, then the propagate bit (p_i) and generate bit (g_i) are defined as :

$$P_i = a_i \oplus b_i$$

$$G_i = a_i \cdot b_i$$

$$C_{(i+1)} = (P_i \cdot C_i) + G_i, \quad i = 1, 2, 3, 4$$

$$sum_i = P_i \oplus C_i$$

We have assumed C_0 as 0.

III. Proposed structure for the adder

Traditional CLA is constructed by XOR, AND, and OR gates. The proposed circuit uses NAND gates to replace the AND and NOT gates in CLA, it can decrease the cost of CLA and increase the speed of CLA. In this proposed CLA circuit, all components are implemented with NAND gates except for the outputs of P and $\overline{G}$ (as seen in the circuit diagram).
The sum bits are obtained using the XOR gates, whereas for obtaining the carry bits we will use the NAND gate implementation. As $c_i = \overline{\overline{c_i}}$.[1]
Therefore, we have $C_{(i+1)} = (P_i \cdot C_i) + G_i$, and hence for the implementation of the NAND gate we will use $C_{(i+1)} = \overline{\overline{G_i} \cdot \overline{P_i C_i}}$.

Hence the sum bits and carry bits at each stage can be obtained as :

$$S_i = a_i \oplus b_i \oplus c_i, \quad i = 1, 2, 3, 4$$
$$C_0 = \textit{input carry},$$
$$C_1 = \overline{\overline{G_0} \cdot \overline{P_0 C_0}},$$
$$C_2 = \overline{\overline{G_1} \cdot \overline{P_1 G_0} \cdot \overline{P_1 P_0 C_0}},$$
$$C_3 = \overline{\overline{G_2} \cdot \overline{P_2 G_1} \cdot \overline{P_2 P_1 G_0} \cdot \overline{P_2 P_1 P_0 C_0}},$$
$$C_4 = \overline{\overline{G_3} \cdot \overline{P_3 G_2 P_3 P_2 G_1} \cdot \overline{P_3 P_2 P_1 G_0} \cdot \overline{P_3 P_2 P_1 P_0 C_0}}.$$
$$\overline{G} = \overline{G_3} \cdot \overline{P_3 G_2} \cdot \overline{P_3 P_2 G_1} \cdot \overline{P_3 P_2 P_1 G_0}$$
$$P = P_3 P_2 P_1 P_0$$

Therefore, the C_4 of second level can be produced from G, P and C_o of the first level, which is :

$$C_4 = \overline{\overline{G} \cdot \overline{P \cdot C_0}}$$

.

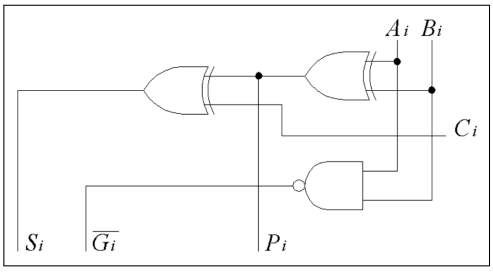

Fig. 1. **Circuit diagram for** S_i, P_i, G_i

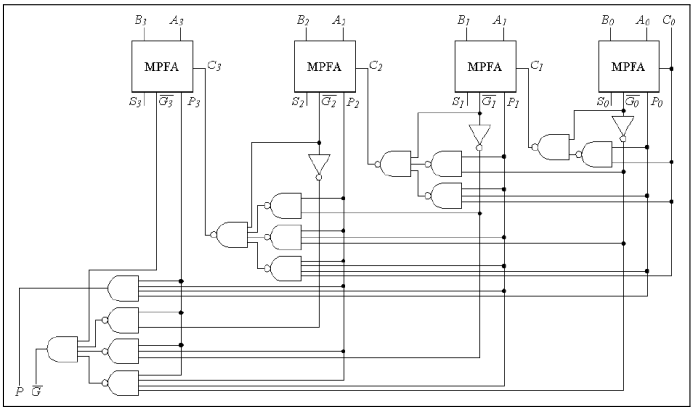

Fig. 2. **4-bit Simplified carry lookahead adder**

IV. Design Details

A. D-flip-flop

For flip-flops, I have used dynamic logic. I have implemented The True Single-Phase Clock (TSPC) flip-flop that operates using only a single clock phase. Compared to the minimum sized inverter, the sizing for the PMOS and NMOS mosfets in the TSPC is indicated below in the circuit diagram.

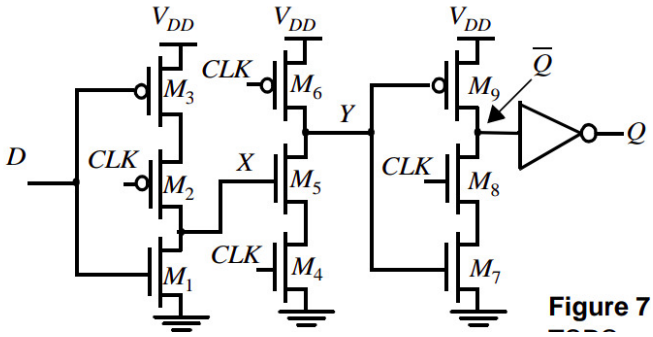

Fig. 3. Schematic for TSPC

B. Adder Module

Gates used in the circuit building are based on Static CMOS logic, in which we have Pull Up Network (PUN) connected to VDD and Pull Down Network (PDN) connected to GND. Sizing for PMOS and NMOS in each gate is mentioned below: Sizing is done in terms of LAMBDA (λ) with width of PMOS and NMOS being factor of LAMBDA

1) Inverter
   - NMOS -10λ
   - PMOS -20λ

2) 2 Input Nand Gate
   - NMOS - 20λ
   - PMOS - 20λ

3) 3 Input Nand Gate
   - NMOS - 30λ
   - PMOS - 20λ

4) 4 Input Nand Gate
   - NMOS - 40λ
   - PMOS - 20λ

5) 4 Input And Gate
   - NMOS - 40λ
   - PMOS - 20λ

6) 2 XOR Gate
   - NMOS -10λ
   - PMOS -20λ

V. NGSPICE Simulation and Functionality

A. D-Flip Flop

Flip Flop will follow the input when there comes the rising edge in the clock whereas it will remain unchanged irrespective of input changes.

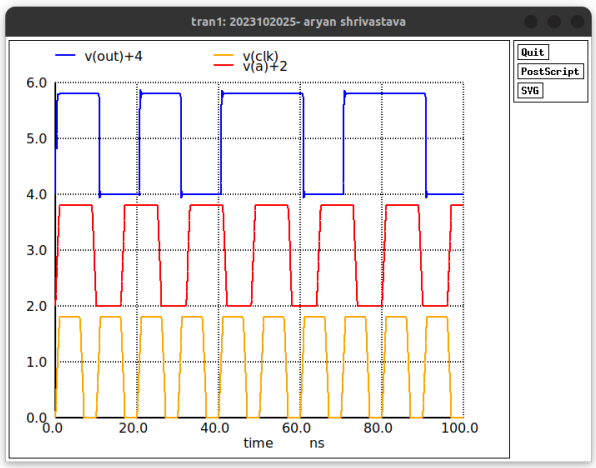

Fig. 4. Output Bits from the Flip Flop

- Orange graph represents the clock input.
- Red graph represents the input.
- Blue Graph represents the output.

B. 4-Bit CLA Adder Module

- The inputs have identical numbers, resulting in the LSB of the output consistently being 0.
- In the initial MPFA, the $C0$ carry bit, used as an input, consistently remains 0.
- The resulting waveform for both input and output is shown below.

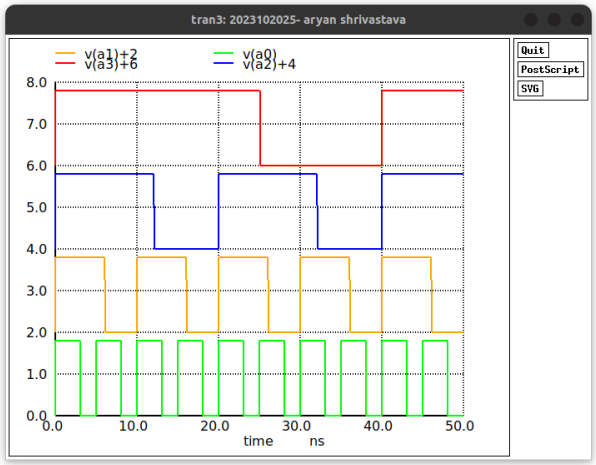

Fig. 5. Input Bits for the 1st Number

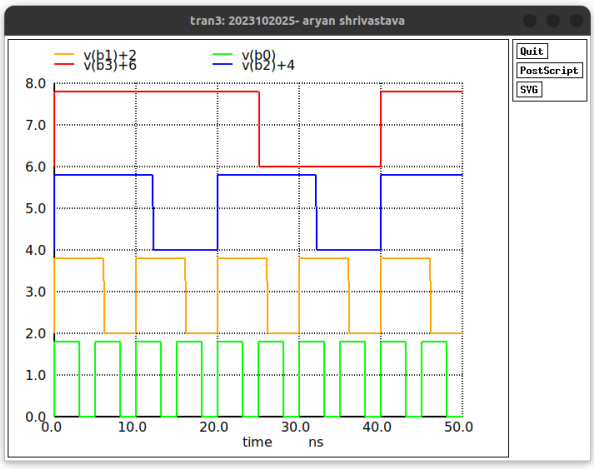

Fig. 6. Input bits for the 2nd number

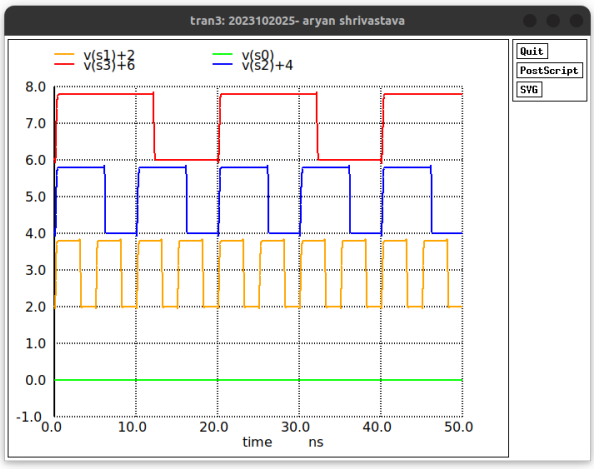

Fig. 7. Waveform for the Output bits

VI. Setup, Hold Time and TPCQ Delay

- **Setup Time** : Setup time is the minimum time before the clock edge during which the data input of a flip-flop must remain stable to ensure correct data is captured.
- **Hold Time** : Hold time is the minimum time after the clock edge during which the data input of a flip-flop must remain stable to ensure correct data is captured.
- **TPCQ Delay** : TPCQ delay (Time Propagation Clock to Q delay) is the time it takes for the output Q of a flip-flop to respond after the clock edge triggers the flip-flop. It measures the delay from the active clock edge (rising or falling, depending on the design) to the stabilization of the Q output. This delay is crucial for determining the flip-flop's contribution to the overall timing in sequential circuits.

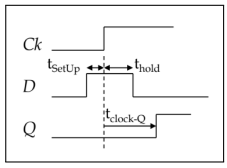

Fig. 8. T_{setup} and T_{hold}

A. Setup Time Violation

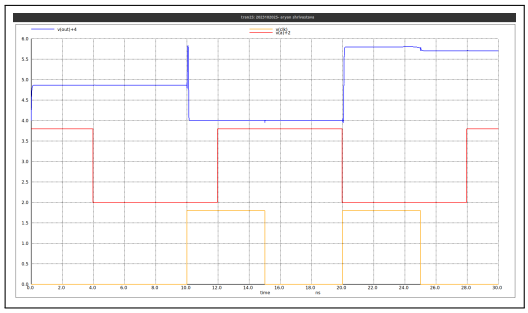

Fig. 9. Setup time

B. Hold Time Violation

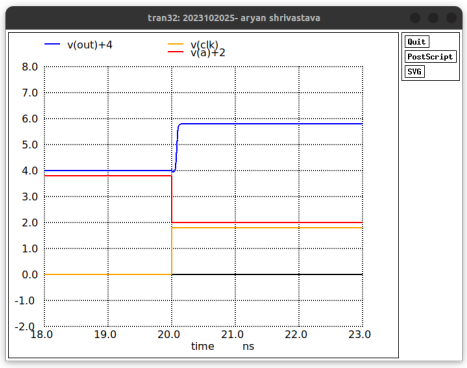

Fig. 10. Hold time

C. TPCQ Delay

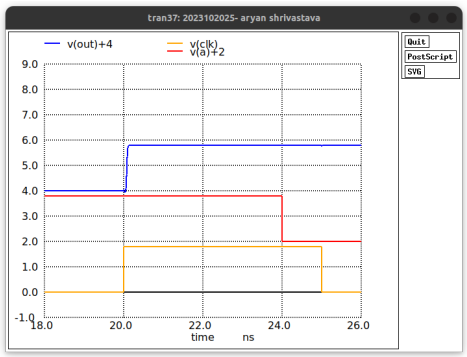

Fig. 11. TPCQ Delay

S.No.	Time	Value
1	Setup Time	30 ps
2	Hold Time	0 ps
3	TPCQ Delay	7.939050e-11

TABLE I
TIMING DIAGRAM

VII. STICK DIAGRAM

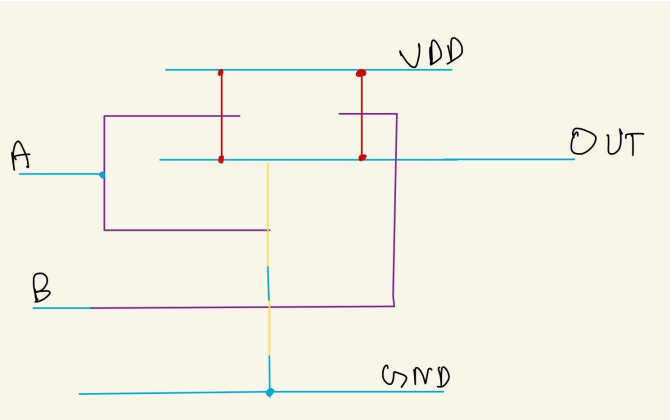

Fig. 12. Nand-2

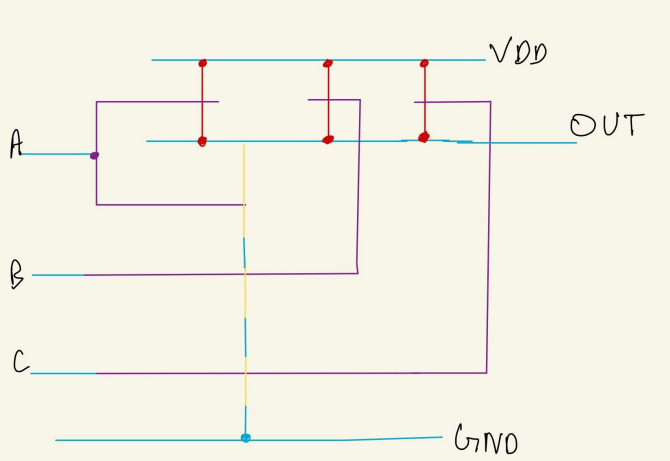

Fig. 13. Nand-3

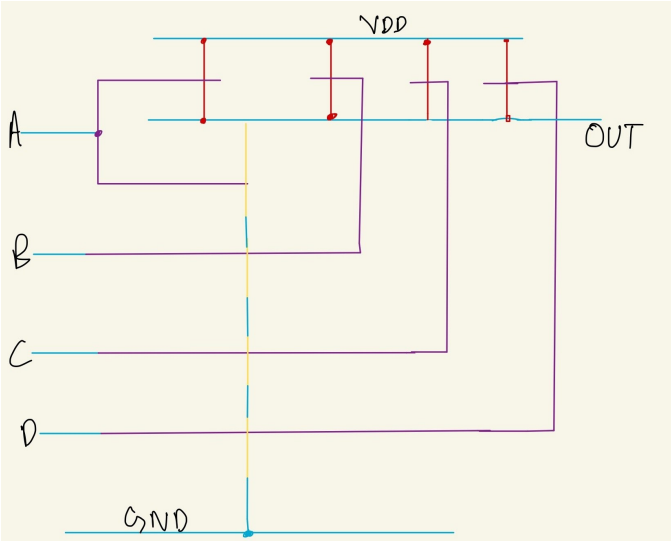

Fig. 14. Nand-4

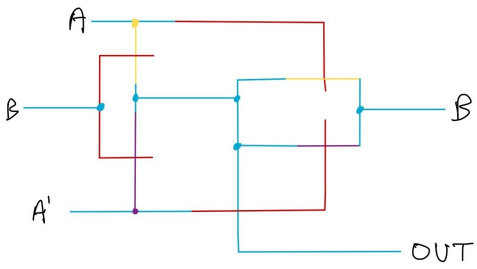

Fig. 15. XOR

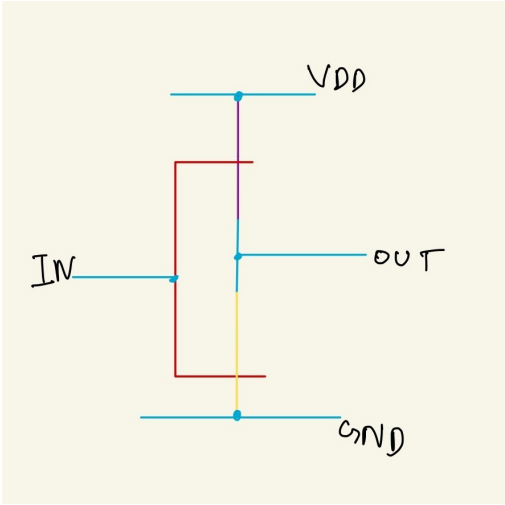

Fig. 16. inverter

VIII. Magic Simulation and Netlist

A. D Flip Flop

Following section contains the Magic layout as well simulation results for the NGSpice file created after extracting the Magic layout.

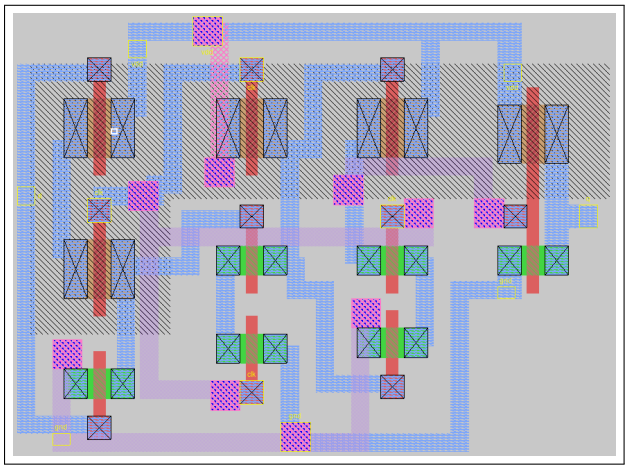

Fig. 17. Magic Layout for TSPC flip flop

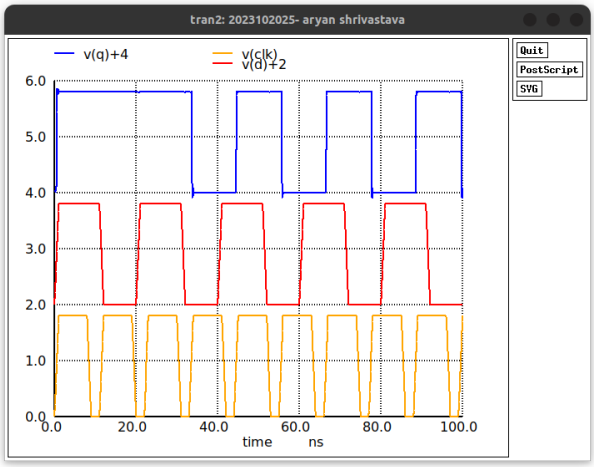

Fig. 18. Simulation results from the extracted SPICE file

B. 4-Bit CLA Adder module

Following section contains the following :

- Magic Layout for the CLA circuit.
- Input and output waveforms.

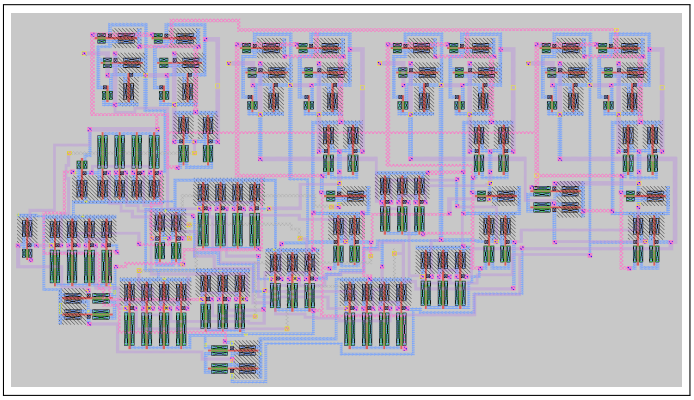

Fig. 19. Magic Layout for CLA

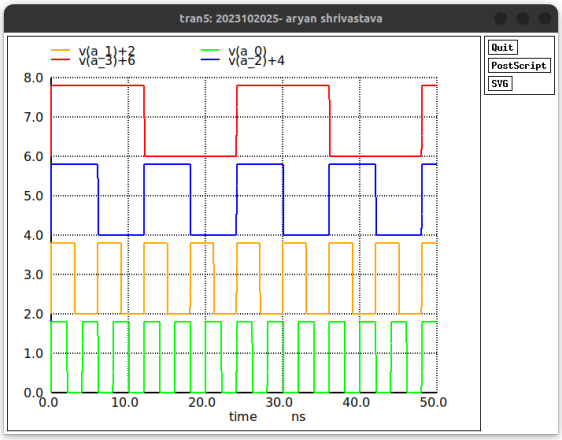

Fig. 20. First input bits

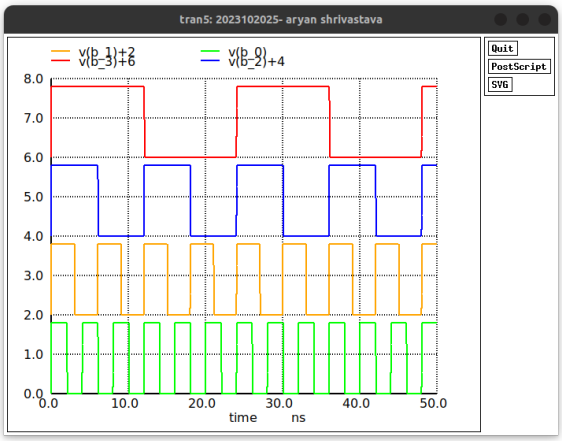

Fig. 21. Second Input bits

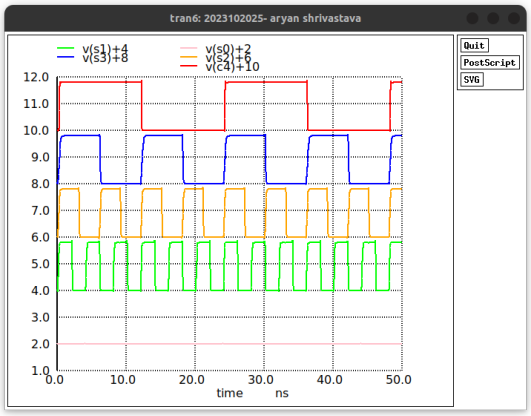

Fig. 22. Waveform for the output bits

C. Comparison

- While comparing the result for the pre-layout and post-layout of D-Flip Flop, we notice that both the outputs follow the rising edge triggering mechanism.
- Therefore, the output for both the layout is same.
- Consequently, comparing pre-layout and post-layout of the 4-bit CLA adder
- For the input $a = 1111$ and $b = 1111$ the output in pre-layout as well as post-layout is same which is $S = 1110$ and $c_4 = 1$.
- Similarly, if we try out with the next set of input i.e. $a = 1101$ and $b = 1101$, the output will be 1010 with 4^{th} carry bit equal to 1.
- Comparing the setup time, hold time and TPCQ delay of the schematic and post layout simulation :

S.No.	Time 2	Schematic	Post-layout
1	Setup time	30 ps	60 ps
2	Hold time	0 ps	0 ps
3	TPCQ Delay	7.939050e-11	9.388252e-11

TABLE II
COMPARISON TABLE

IX. Complete Circuit using NGSpice

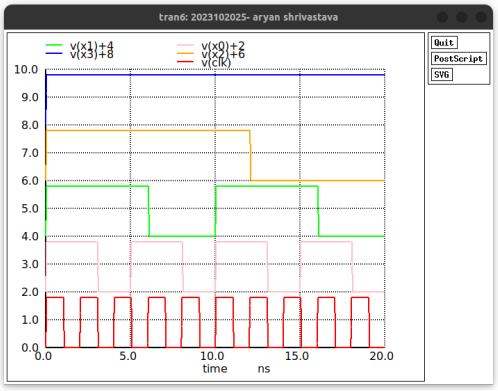

Fig. 23. Waveform for the first input

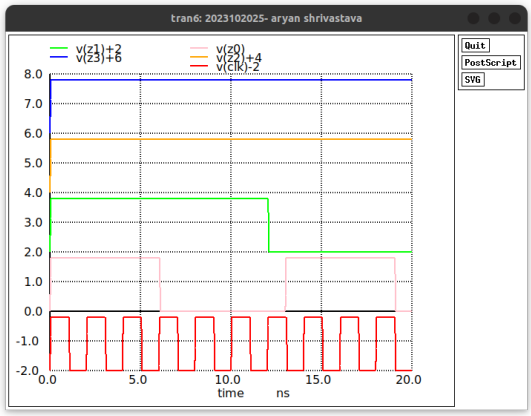

Fig. 24. Waveform for the second input

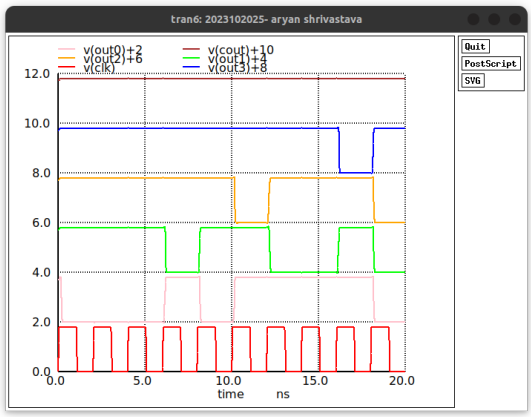

Fig. 25. Waveform for the output bits

A. Verification of functionality

- $a = 1110$ and $b = 1111$, output = 1101 with carry = 1.
- $a = 1101$ and $b = 1110$, output = 1011 with carry = 1.
- $a = 1001$ and $b = 1101$, output = 0110 with carry = 1.

Similarly, we can verify for various other example cases.

B. Delay and Maximum Clock Speed

- Worst-case delay is the maximum signal propagation time under the slowest conditions.
- Critical path is the longest delay path in a circuit, determining its maximum clock speed. Reducing the critical path minimizes worst-case delay, enhancing performance and ensuring reliable operation at higher frequencies.

Delay

- Worst case delay for the combinational circuit is $710ps$.
- Total worst case delay is almost $868ps$ after including the delay from flip flops and combinational circuits.

Maximum Clock Speed

$$t_{\text{clk (min)}} = t_{\text{su}} + t_{\text{pd}} + t_{\text{pcq}}$$

Therefore, the minimum clock time we get is $= 898ps$. This converts to $1.113GHz$ Therefore, the maximum clock speed reported at which my circuit is working fine is near $800Mhz$.

X. Floor Plan

A. Complete Layout for my Circuit

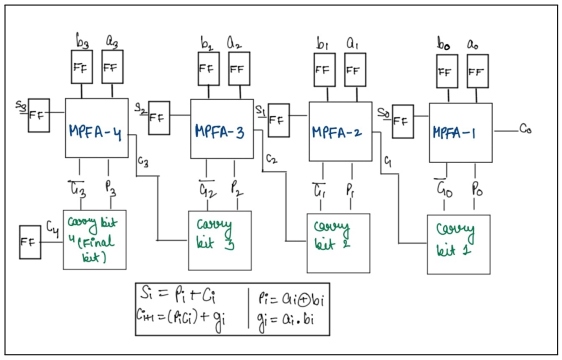

Fig. 26. Floor Plan

- Horizontal Pitch : 948 blocks or 85.32 um
- Vertical Pitch : 539 blocks or 48.51 um
- Total area covered is : 4138.8732 um

XI. Complete Layout using magic

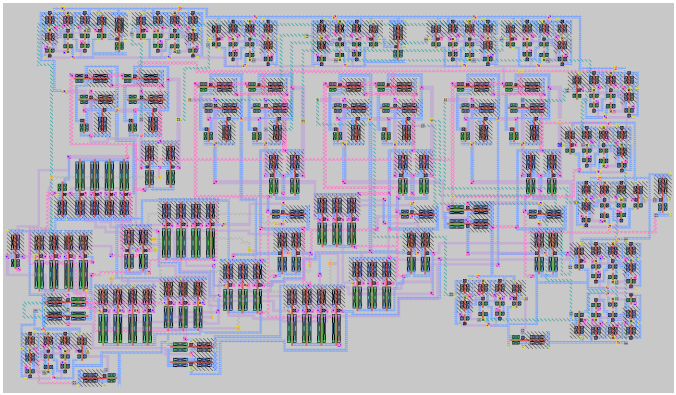

Fig. 27. Combined layout

A. Simulation results

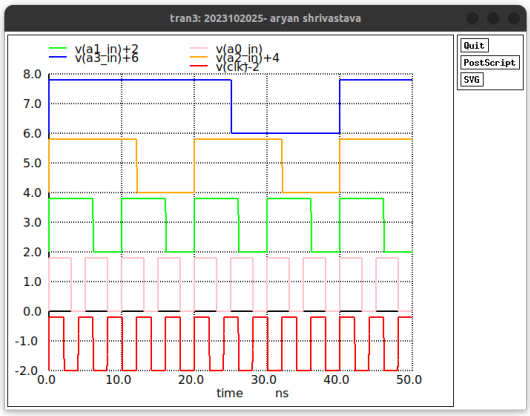

Fig. 28. 1st Input

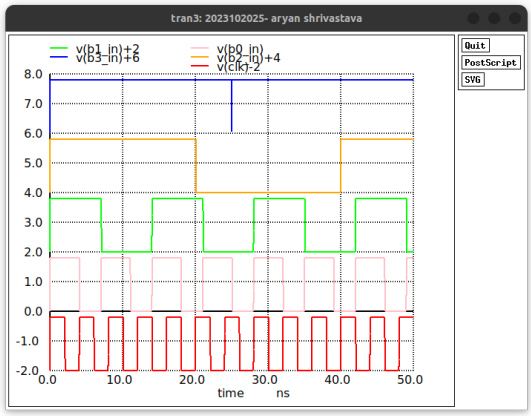

Fig. 29. 2nd input

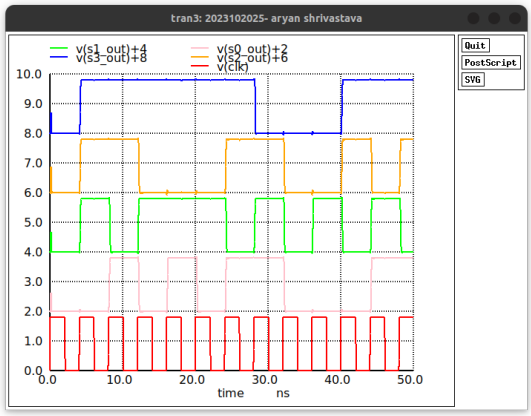

Fig. 30. Output Waveform

Here also, if we take the same input as taken earlier we get the same output.

Although, the calculated clock speed for both the simulations

S.No.	Value	Schematic	Post layout
1	Delay	614 ps	647 ps
2	clock Speed	880 Mhz	800 Mhz

TABLE III
YOUR TABLE CAPTION

is higher than 800 Mhz but for the Flip Flop to work reliably for all the inputs, we have put it down a bit.

XII. HDL Verilog and FPGA results

A. Verilog

Fig. 31. Verilog Simulation

- The inputs are appearing at the first rising edge.

- The outputs are calculated and then displayed the second rising edge.

B. FPGA simulation

The inputs taken were 0011 and 1100, therefore the outputs should be 1111 with carry= 0.

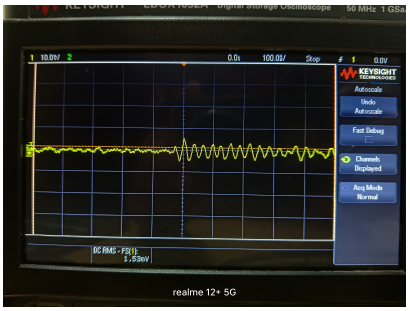

Fig. 32. Carry bit

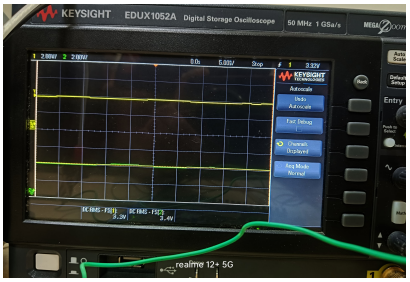

Fig. 33. S3 and S2

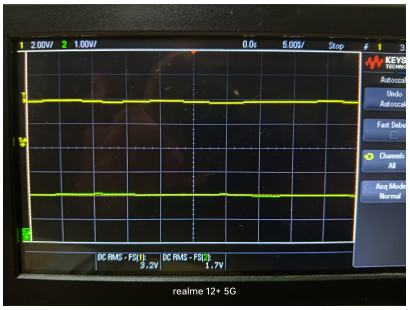

Fig. 34. S1 and S0

XIII. PDP

- Power consumption is 4.399×10^{-5}
- Delay obtained is $990ps$.
- The PDP found is 4.286×10^{-14}.

REFERENCES

[1] The Fastest Carry Lookahead Adder, Yu-Ting Pai and Yu-Kumg Chen, IEEE XPLORE.