CARRY LOOK AHEAD ADDER

Nitin Rama Sai Grandhi
International Institute of Information Technology, Hyderabad
grandhi.sai@research.iiit.ac.in
2023112026

***Abstract*—The project involves the design and implementation of a 4-bit Carry Look Ahead (CLA) adder, a high-speed arithmetic circuit used for efficient binary addition. The CLA adder is composed of modular blocks for propagating and generating signal computation, carry look-ahead logic, and the sum generation block. The designs ensures that input bits are provided before the rising edge of the clock, with outputs computed and available at the subsequent rising edge, adhering to a synchronous timing scheme.**
The D flip-flops, used to store intermediate values and synchronize the carry signals, are designed with specific parameters, including a width ratio of $\frac{W_p}{W_n} = \frac{20\lambda}{10\lambda}$, and are operated with a clock signal to ensure proper synchronization.

I. Introduction

This project is used to efficiently solve the carry generation problem where the conventional case is to use Ripple Carry Adder which is disadvantageous in terms of delay as the $i-1$-th carry goes in the calculation of i-th carry

Let the numbers to be added be $a_4a_3a_2a_1$ and $b_4b_3b_2b_1$.The efficient way to optimis delay is to pre-compute a few values namely propagate (p_i) and generate (g_i). The (p_i) and (g_i) signals for each bit position can be defined as:

$$p_i = a_i \oplus b_i$$

$$g_i = a_i \cdot b_i$$

For the carry-out (c_{i+1}) of the i-th bit position, assuming $c_0 = 0$, we have:

$$c_{i+1} = (p_i \cdot c_i) + g_i, \quad i = 1, 2, 3, 4$$

Thus, the carry-out c_{i+1} can be expressed entirely in terms of the p_i and g_i functions. The sum (sumi) can be represented as:

$$\text{sum} = p_i \oplus c_i$$

The carry-out for each bit position is generated as:

$$c_1 = g_1 + p_1 \cdot c_0$$

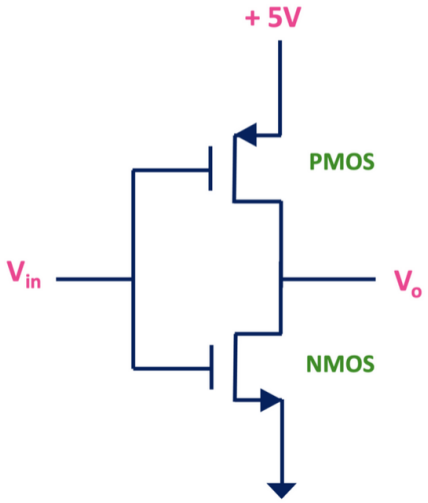
$$c_2 = g_2 + p_2 \cdot g_1 + p_2 \cdot p_1 \cdot c_0$$

$$c_3 = g_3 + p_3 \cdot g_2 + p_3 \cdot p_2 \cdot g_1 + p_3 \cdot p_2 \cdot p_1 \cdot c_0$$

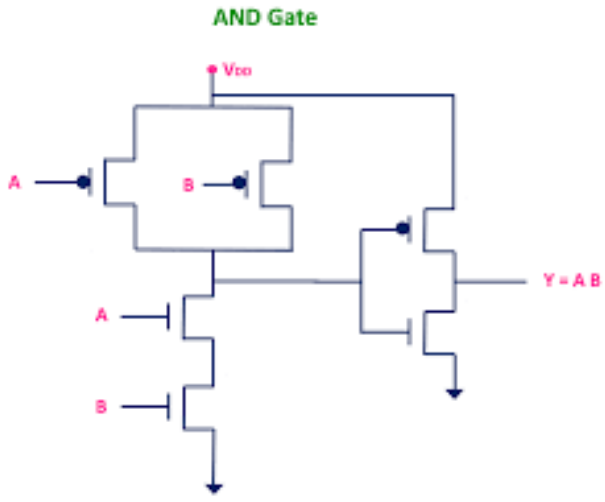
$$c_4 = g_4 + p_4 \cdot g_3 + p_4 \cdot p_3 \cdot g_2 + p_4 \cdot p_3 \cdot p_2 \cdot g_1 + p_4 \cdot p_3 \cdot p_2 \cdot p_1 \cdot c_0$$

II. Design Topology

INVERTER The inverter is implemented using cmos static logic.They act as buffers in circuits, providing isolation and strengthening weak signals to drive larger loads.CMOS inverters switch between logic levels quickly, making them suitable for high-speed digital applications.

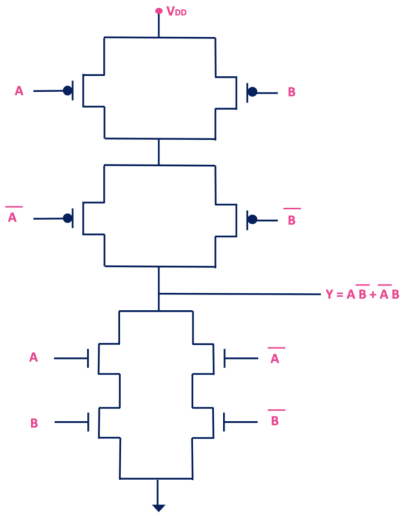


AND GATE The AND gate is implemented using CMOS static logic. CMOS static logic is preferred due to its low power consumption, high noise immunity, and reliable operation across a wide range of conditions. It ensures minimal static power dissipation as current flows only during state transitions. Additionally, CMOS circuits are highly scalable and well-suited for integration in modern VLSI designs. These features make them ideal for implementing robust and efficient digital logic gates. From using CMOS NAND gate and minimum sized inverter combination AND gate is designed.The sizes for the and gate are like the PMOS of AND gate is 2*W and the width for NMOS is 2*W where w is the width of NMOS in inverter.



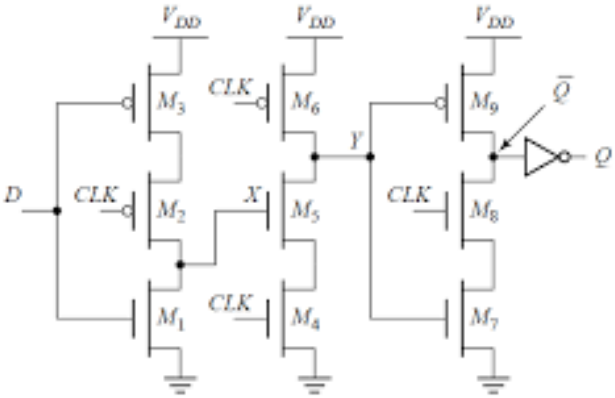
XOR GATE The XOR gate also implemented using CMOS static logic. CMOS static logic is preferred due to its low power consumption, high noise immunity, and reliable operation across a wide range of conditions. It ensures minimal static power dissipation as current flows only during state transitions.For xor gate the part where inputs and inputs bar are taken there the width of PMOS is 4*W and the width of NMOS is 2*W where W is the width of NMOS in inverter.

XOR Gate



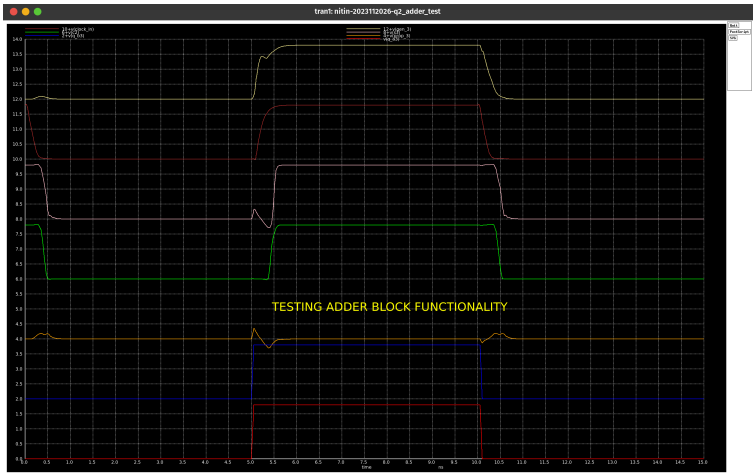
III. Flip Flop Description

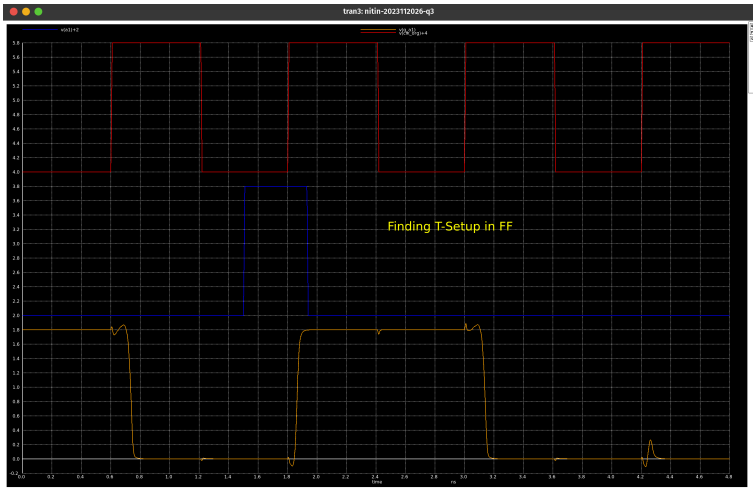
D-Flip-Flop: The flip-flop is implemented using True Single-Phase Clocked (TSPC) logic. This choice ensures high-speed operation with reduced clock skew, enabling reliable edge-triggered storage of the carry-in signal.



- The D Flip-Flop is implemented using True Single-Phase Clocking (TSPC) logic, which dynamically stores and transfers data with reduced transistor count and a single clock phase.
- This design uses 11 MOSFETs instead of the traditional 14 by optimizing the logic as follows:
  - The design combines pull-up and pull-down paths efficiently, using fewer transistors while maintaining functionality.
  - A clocked NMOS network is used for data capture, and the stored data is dynamically transferred to the output through successive stages.
- The sizing of transistors:
  - NMOS and PMOS transistor widths are chosen to ensure proper rise and fall times. Typical sizing:
    - NMOS transistors: $W_n = W$.
    - PMOS transistors: $W_p = 2W$.
- Advantages of this TSPC implementation:
  - Reduced power consumption due to fewer transistors and dynamic operation.
  - Simplified clocking scheme with a single-phase clock, reducing clock skew issues.
  - Compact design with fewer MOSFETs, saving area.

IV. Simulation Of Blocks(PRE-LAYOUT)





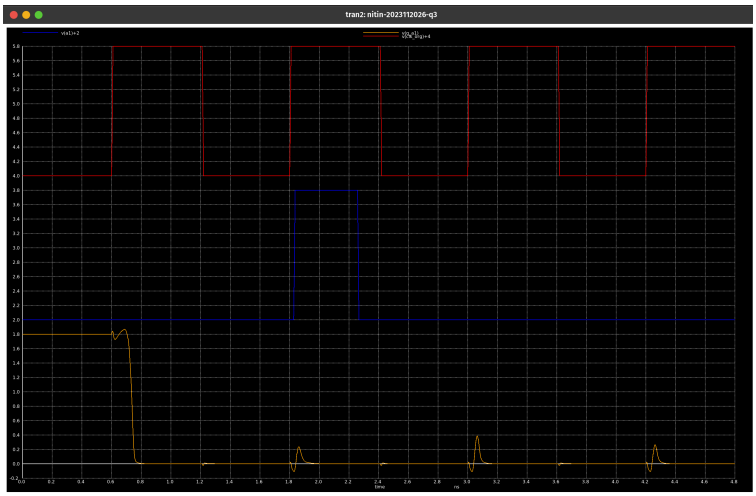


Figure: Finding T-HOLD

```
tpcq_0_to_1          =  6.009264e-11
tpcq_1_to_0          =  1.363941e-10

tsetup               =  1.000000e-10
```

Figure: Flip Flop Characteristics

```
tpd_max              =  5.039673e-10
```

Figure: Finding Tpdmax until Carry Generation

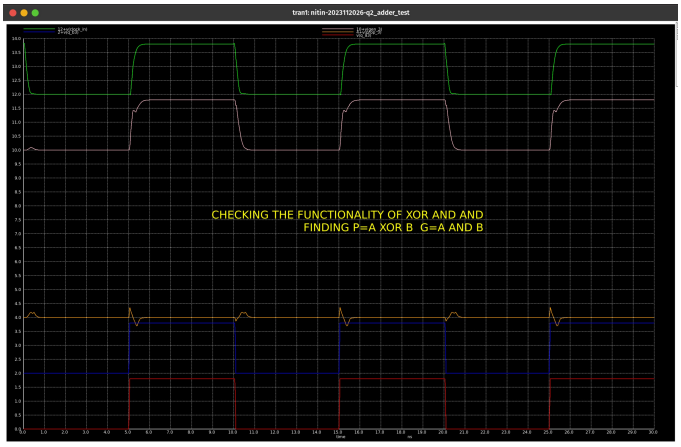
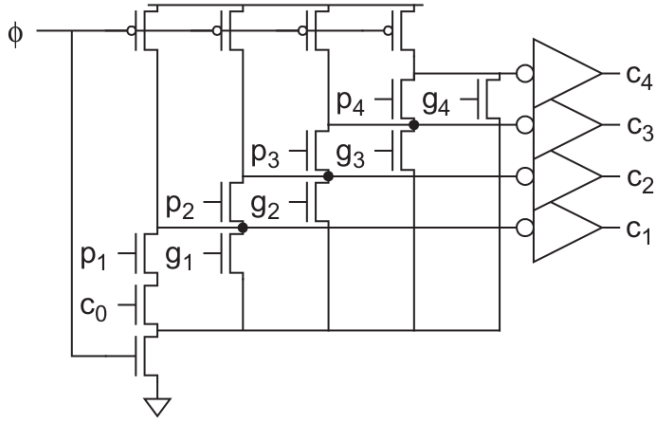


Fig. 1. XOR and AND Functionality

Carry Generation:

The chosen carry generation topology is ***Manchester carry chain***. It is often necessary to compute multiple functions where one is a sub-function of another or shares a subfunction. ***Multiple-output domino logic (MODL)*** saves area by combining all of the computations into a multiple output gate. For this adder, we must define:



Carry Generation Logic

If p_i were defined as $a_i + b_i$, a **sneak path** could exist when a_4 and b_4 are 1 and all other inputs are 0. In that case: g_4=p_4=1. c_4 would fire as desired, but c_3 would also fire incorrectly.

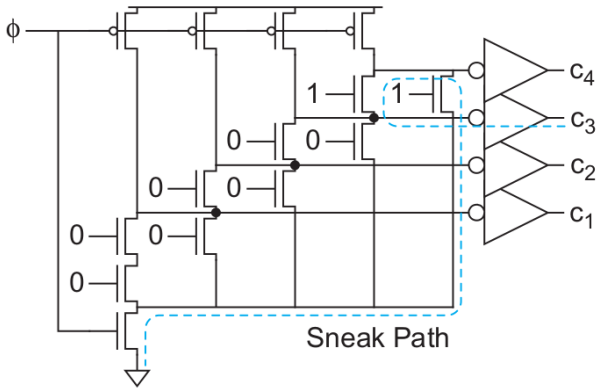
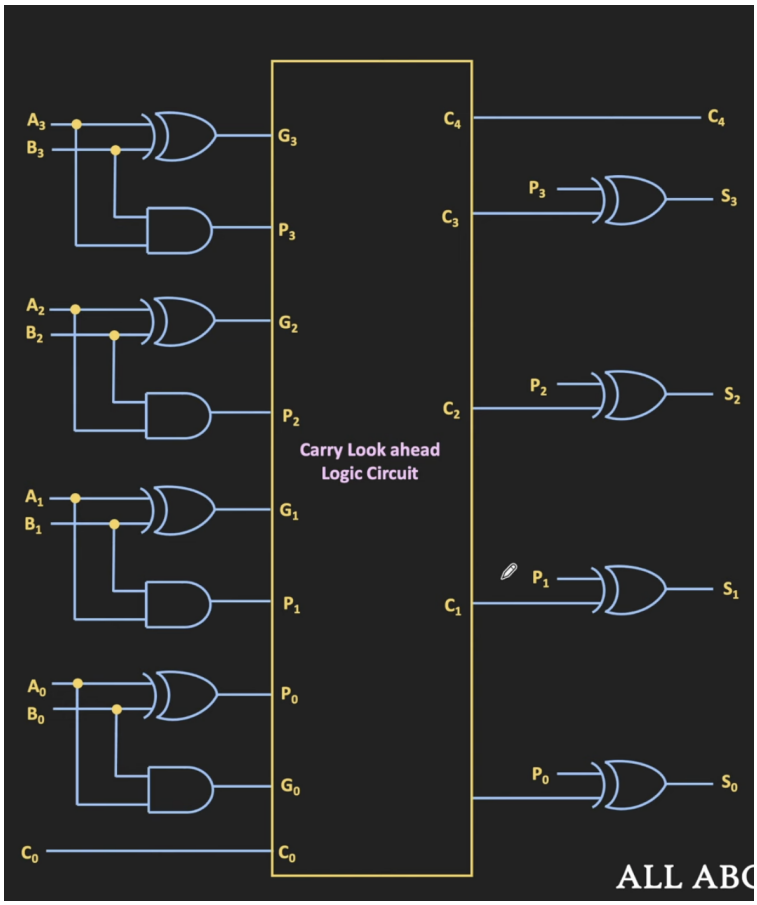


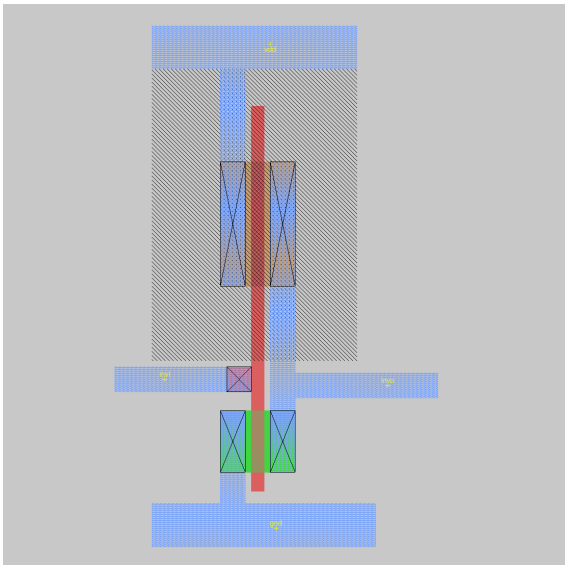
Fig. 2. *
Sneak Path in Carry Chain

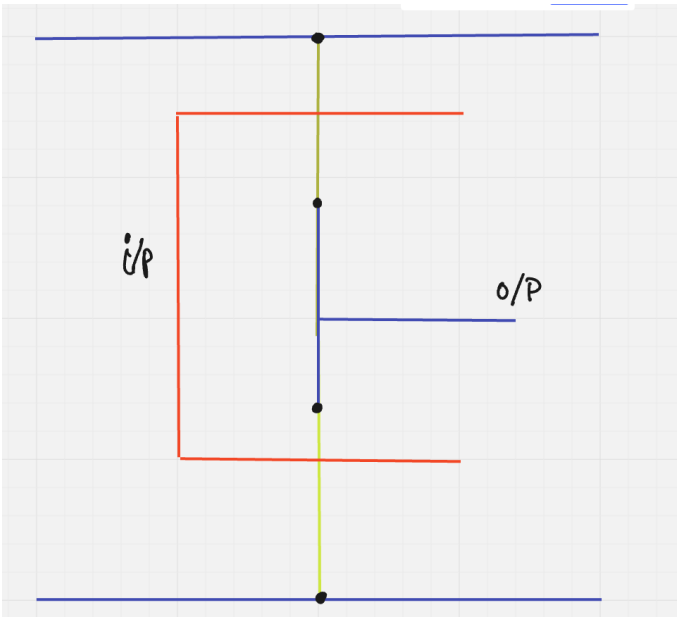
Overall Adder Circuit:



V. Stick Diagrams And Layout Extraction

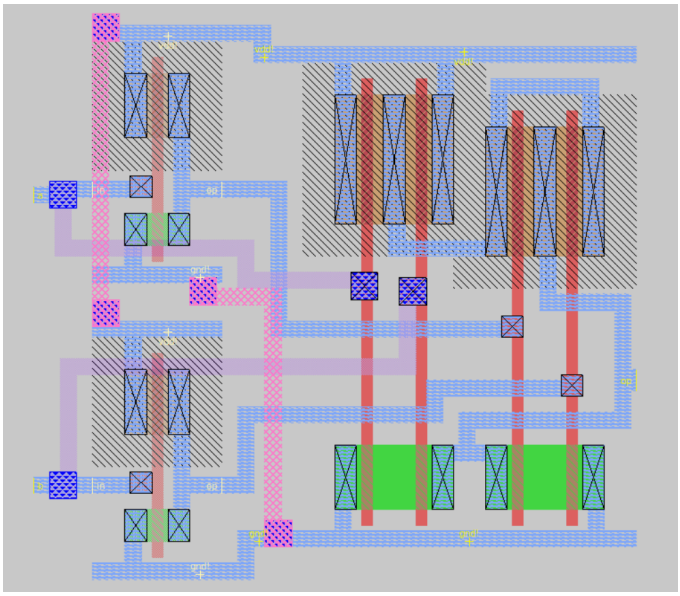
A. INVERTER

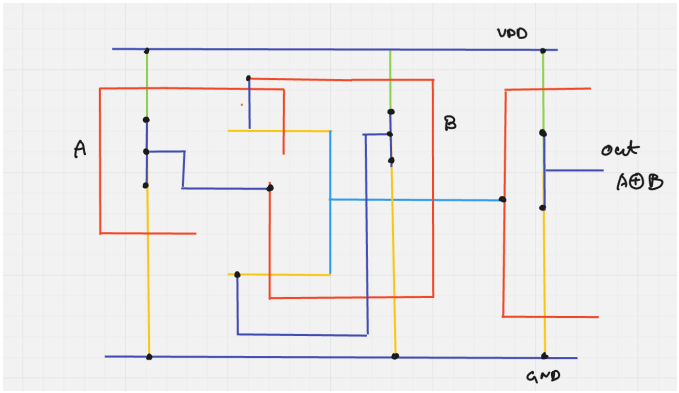




INV

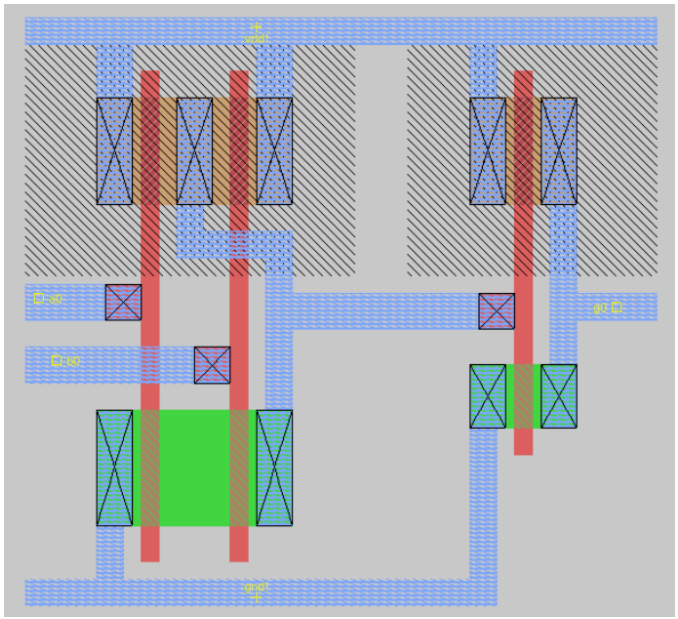
C. XOR GATE

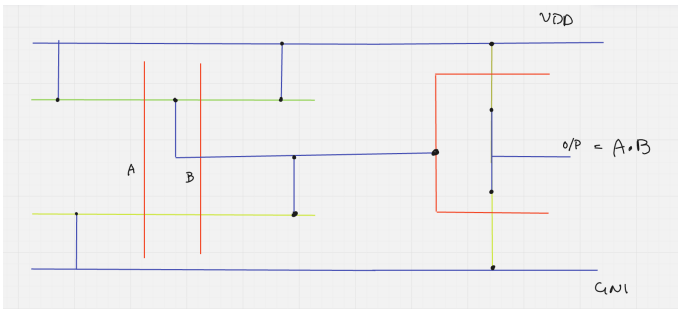




XOR

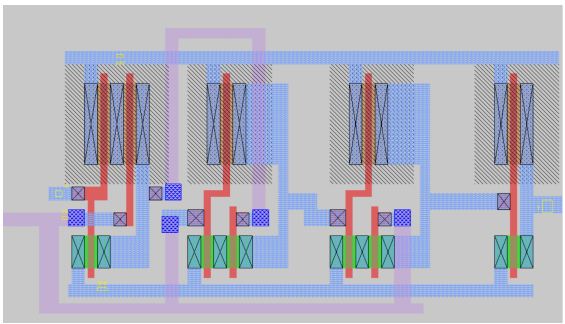
B. AND GATE

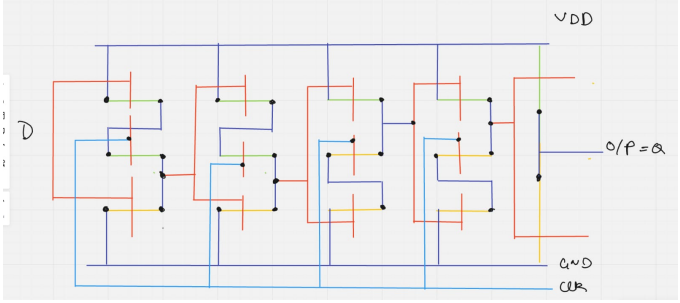




AND

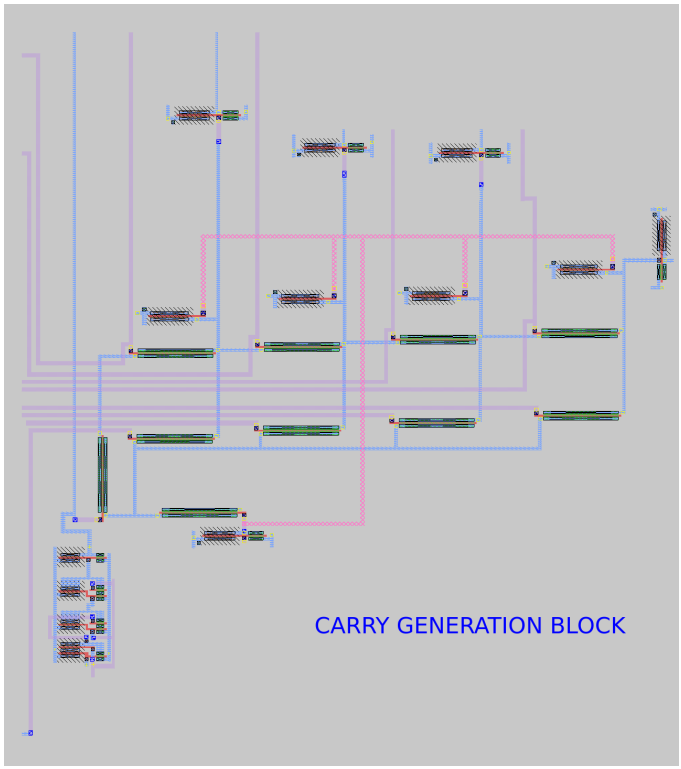
D. D Flip Flop



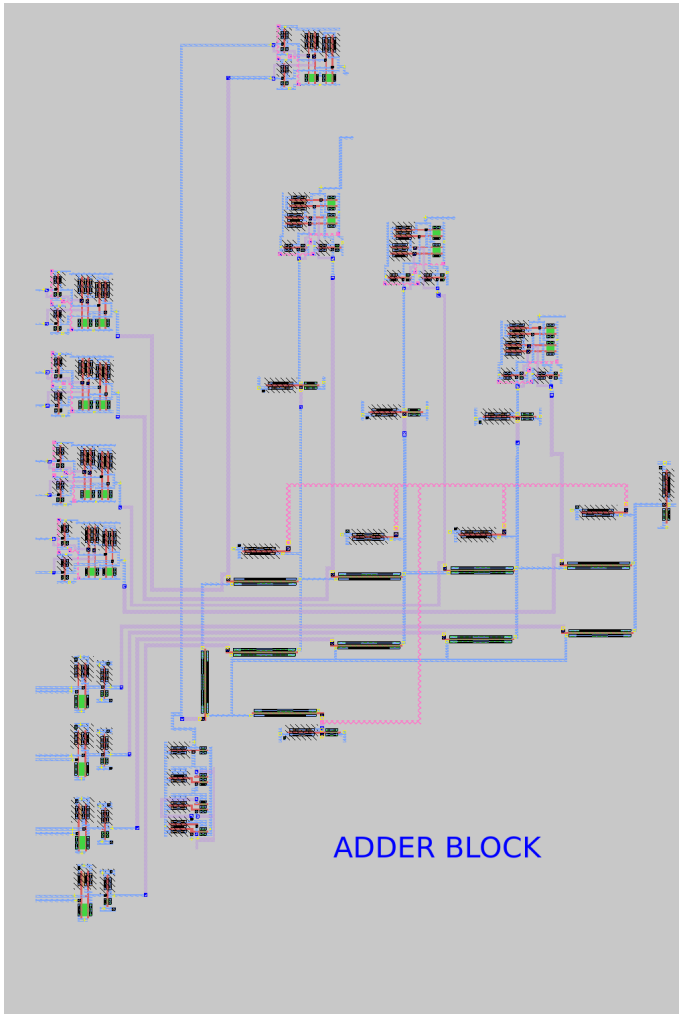


TSPC FLIP FLOP

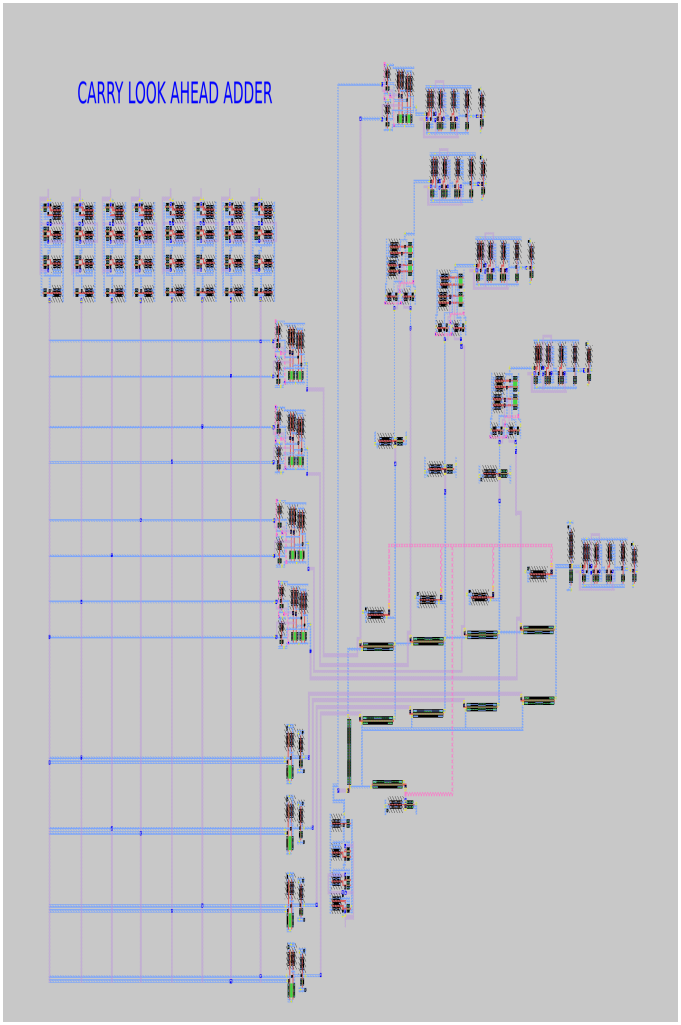
E. Carry Generator



F. ADDER



G. CARRY LOOK AHEAD ADDER

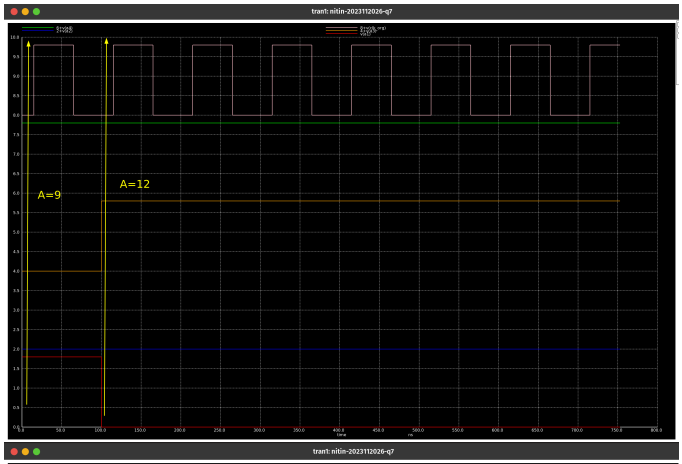


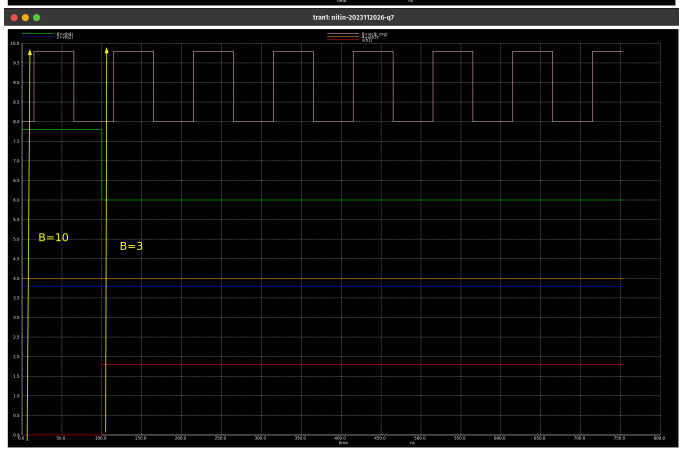

```
tpcq_0_to_1_post    =  7.663109e-11
tpcq_1_to_0_post    =  1.264172e-10

thold               =  2.700000e-11

tsetup_post         =  5.800000e-11

tpd_max_post        =  4.535991e-10
```

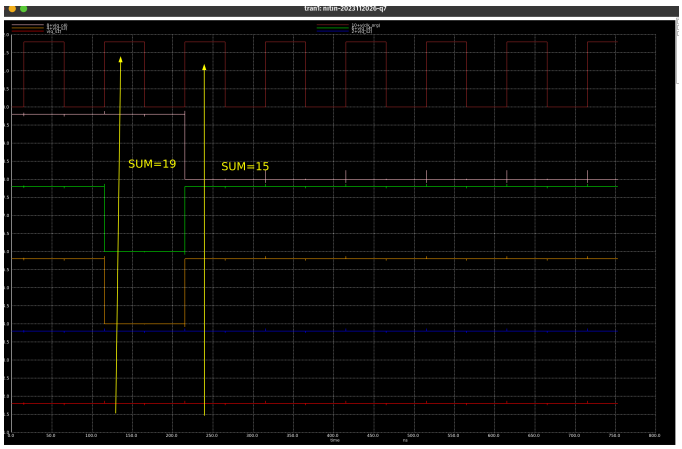
VI. Pre-Layout Integration

Category	Parameter	Pre-Layout
And Gate	$t_{\text{pd max}}$	0.057 ns
Xor Gate	$t_{\text{pd max}}$	0.06 ns
D Flip-Flop	t_{setup}	0.1 ns
	t_{hold}	27.8 ps
	t_{pcq} $(0 \rightarrow 1)$	68.09 ps
	t_{pcq} $(1 \rightarrow 0)$	130 ps
Sum Block without Xor	$t_{\text{pd max}}$	0.4399 ns
Minimum Clock Period	T_{clk}	1.1998 ns
Clock Frequency	T_{freq}	833 MHz

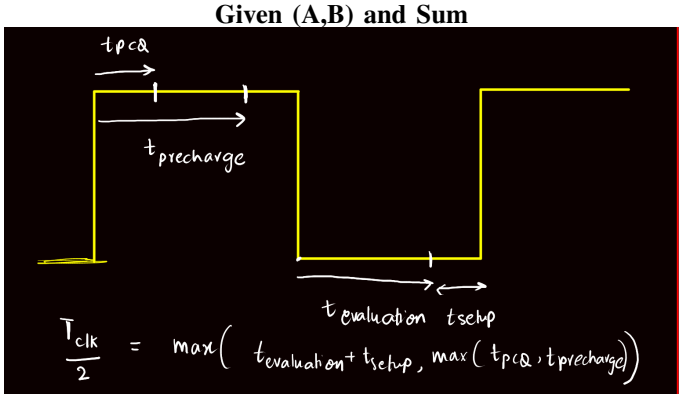
Pre-Layout Delays







Given (A,B) and Sum



Clock calculation

$$T_{clk} = 2 * (t_{pdmax} + t_{setup})$$

Advantages

- High-speed operation due to parallel carry computation and domino logic.
- Reduced gate count compared to static CMOS designs.
- Efficient area utilization in high-performance applications.

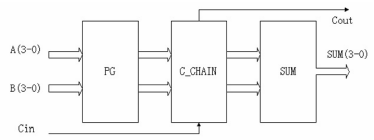
```
V_a1 a1 0 PWL(0ns 1.8 {Tperiod} 1.8 {Tperiod+0.1n} 0 {2*Tperiod} 0)
V_a2 a2 0 PWL(0ns 0 {Tperiod} 0 {Tperiod+0.1n} 0 {2*Tperiod} 0)
V_a3 a3 0 PWL(0ns 0 {Tperiod} 0 {Tperiod+0.1n} 1.8 {2*Tperiod} 1.8)
V_a4 a4 0 PWL(0ns 1.8 {Tperiod} 1.8 {Tperiod+0.1n} 1.8 {2*Tperiod} 1.8)
V_b1 b1 0 PWL(0ns 0 {Tperiod} 0 {Tperiod+0.1n} 1.8 {2*Tperiod} 1.8)
V_b2 b2 0 PWL(0ns 1.8 {Tperiod} 1.8 {Tperiod+0.1n} 1.8 {2*Tperiod} 1.8)
V_b3 b3 0 PWL(0ns 0 {Tperiod} 0 {Tperiod+0.1n} 0 {2*Tperiod} 0)
V_b4 b4 0 PWL(0ns 1.8 {Tperiod} 1.8 {Tperiod+0.1n} 0 {2*Tperiod} 0)
```

- Lower power consumption in short or less frequently used paths.
- Scalable design for larger bit-widths.

Disadvantages

- High power consumption due to precharge and evaluation phases.
- Complex design requiring careful clocking and timing considerations.
- Susceptibility to noise due to dynamic nodes.
- Clock synchronization issues, prone to clock skew.

VII. Floor Plan

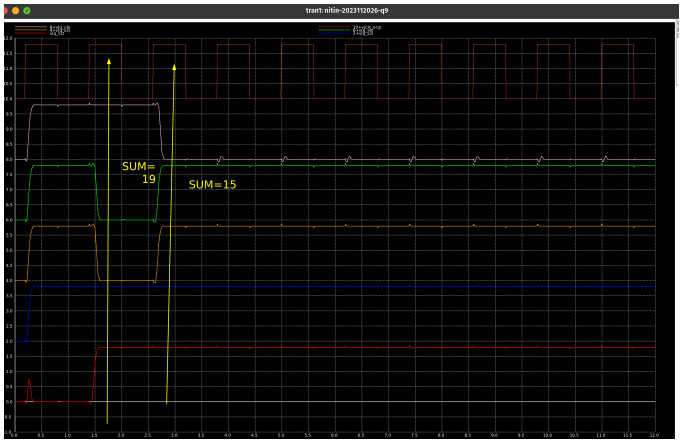


VIII. Post-Layout Integration

Category	Parameter	Pre	Post
And Gate	$t_{\text{pd max}}$	$0.057ns$	$0.061ns$
Xor Gate	$t_{\text{pd max}}$	$0.06ns$	$0.0415ns$
D Flip-Flop	t_{setup}	$0.1ns$	$0.058ns$
	t_{hold}	$27.8ps$	$24ps$
	t_{pcq} $(0 \rightarrow 1)$	$68.09ps$	$76.6ps$
	t_{pcq} $(1 \rightarrow 0)$	$0.13ns$	$0.126ns$
Sum Block without Xor	$t_{\text{pd max}}$	$0.4399ns$	$0.4535ns$
Minimum Clock Period	T_{clk}	$1.1998ns$	$1.09ns$
Clock Frequency	T_{freq}	$833MHz$	$917MHz$

TABLE I
PRE AND POST TIMING PARAMETERS

A. INPUTS check



IX. Verilog Description

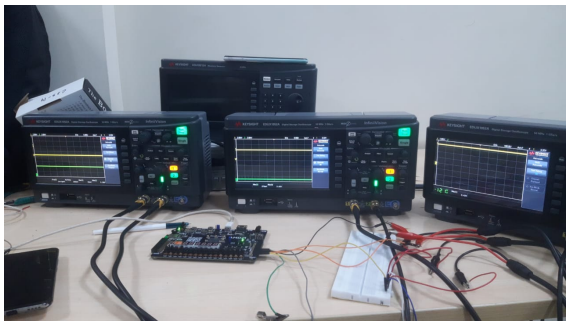
The Verilog code structurally describes a 4-bit pipelined carry-lookahead adder (CLA) circuit. It uses flip-flops (dff and dff4) for input/output synchronization and a CLA module to compute the sum and carry-out efficiently.

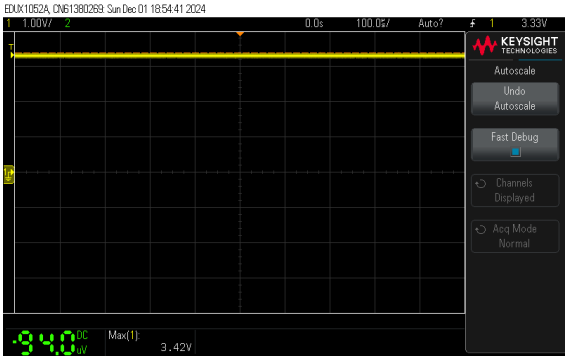
Structural Description:

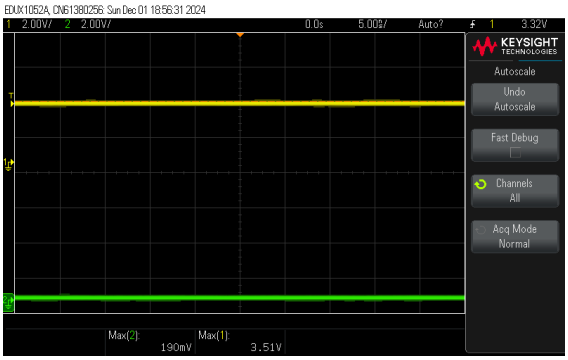
Flip-Flops: dff and dff4 are used to register inputs (A, B, cin), intermediate outputs (S, cout), and final outputs (S, cout). CLA Logic: Implements generate (G), propagate (P), and carry computation (C) for 4 bits to perform fast addition.

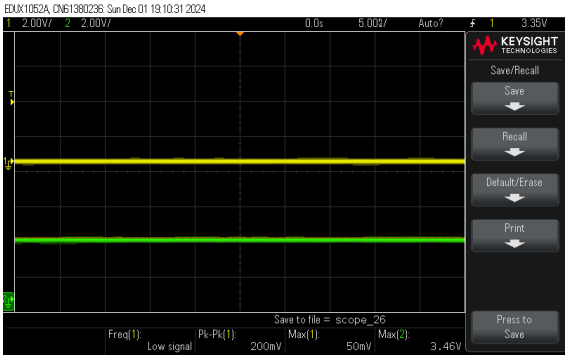

```
VCD info: dumpfile waveform.vcd opened for output.
Time = 0, a =  0, b =  0, cin = 0, sum =  x, carry_out = x
Time = 15000, a =  0, b =  0, cin = 0, sum =  0, carry_out = 0
Time = 20000, a =  5, b =  3, cin = 0, sum =  0, carry_out = 0
Time = 30000, a = 15, b = 15, cin = 1, sum =  0, carry_out = 0
Time = 35000, a = 15, b = 15, cin = 1, sum =  8, carry_out = 0
Time = 40000, a = 12, b = 10, cin = 0, sum =  8, carry_out = 0
Time = 45000, a = 12, b = 10, cin = 0, sum = 15, carry_out = 1
Time = 50000, a = 14, b =  3, cin = 1, sum = 15, carry_out = 1
Time = 55000, a = 14, b =  3, cin = 1, sum =  6, carry_out = 1
```

X. FPGA Implementation









XI. Conclusion

This project demonstrated the effectiveness of the Carry Look-Ahead Adder design in achieving high-speed addition in a 4-bit architecture. The use of CMOS technology in MAGIC layout and FPGA implementation confirmed the adder's performance. Future work could include extending the design for larger bit-widths or optimizing power consumption.

References

[1] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*, 2nd ed. Prentice Hall, 2002.

[2] N. Weste and D. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 4th ed. Addison-Wesley, 2010.

[3] M. Mano, *Digital Logic and Computer Design*. Prentice Hall, 1979.