

5-bit Carry-Lookahead Adder

Sai Poojith
2025122010

International Institute of Information Technology, Hyderabad, India
mididoddi.poojith@research.iiit.ac.in

Abstract—This paper presents the design and implementation of a synchronous 5-bit addition module using 180nm CMOS technology. The module integrates a carry look-ahead adder (CLA) with D flip-flops for synchronized, high-speed operations. The proposed design is implemented in NGSpice and Verilog, with layout done in MAGIC software. A Verilog-based structural description enables FPGA implementation for hardware validation. It achieves extremely low delay while optimizing parameters like area and power consumption.

Keywords—D flip-flop, CLA, NGSPICE, MAGIC, Delay, Verilog, FPGA

I. INTRODUCTION

Binary addition is a fundamental operation and a critical component of Arithmetic Logic Units (ALUs). While various adder architectures exist—such as Ripple Carry Adders and Carry Select Adders—conventional designs suffer from significant propagation delay due to the serial dependency of higher-order bits on the carry-out from lower-order positions. The Carry Look-ahead Adder (CLA) architecture addresses this bottleneck. By concurrently calculating the carry status for each bit position rather than waiting for serial propagation, the CLA decouples carry generation from summation. This predictive capability substantially reduces latency and optimizes the overall performance of the adder.

II. IMPLEMENTATION OF THE CLA

A. Principle of the 5-bit CLA

The function of the Carry-Lookahead Adder (CLA) is to directly compute the carry-in for each bit, rather than waiting for the carry-out of each sum. In the case of a two-input addition, the external carry-in is denoted as C_0 . Both inputs are 4-bit values, represented as $A_3A_2A_1A_0$ and $B_3B_2B_1B_0$, respectively. The sum of the addition, $S_3S_2S_1S_0$, is determined by the full adders, which yields

$$S_4S_3S_2S_1S_0 = A_4A_3A_2A_1A_0 \oplus B_4B_3B_2B_1B_0 \oplus C_4C_3C_2C_1C_0.$$

The carry-in calculations for each bit are facilitated by the introduction of two new variables: G (carry-generate) and P (carry-propagate), D (carry-delete) defined as follows:

$$\begin{aligned} P_i &= A_i \oplus B_i \\ G_i &= A_i B_i \\ D_i &= \overline{A_i + B_i} \end{aligned}$$

The output sum and carry can respectively, be expressed in the following manner:

$$\begin{aligned} S_i &= P_i \oplus C_i \\ C_{i+1} &= G_i + P_i C_i \end{aligned}$$

Here, i takes the value 0, 1, 2, 3, or 4. We thus obtain the set of equations for the carry bits as:

$$C_1 = G_0 + P_0 C_0 \quad (1)$$

$$C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0 \quad (2)$$

$$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0 \quad (3)$$

$$\begin{aligned} C_4 &= G_3 + P_3 G_2 + P_3 P_2 G_1 \\ &\quad + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0 \end{aligned} \quad (4)$$

$$\begin{aligned} C_5 &= G_4 + P_4 G_3 + P_4 P_3 G_2 + P_4 P_3 P_2 G_1 \\ &\quad + P_4 P_3 P_2 P_1 G_0 + P_4 P_3 P_2 P_1 P_0 C_0 \end{aligned} \quad (5)$$

B. Limitations of Static CMOS Implementation

As observed in Equations (4) and (5), the complexity of the carry generation logic increases linearly with the bit significance. Specifically, implementing C_4 and C_5 directly using static CMOS logic would require 5-input and 6-input AND/OR gates respectively. In standard CMOS technology, stacking more than four transistors in series (as required for high fan-in NAND/NOR gates) significantly degrades signal propagation speed due to the body effect and increased parasitic capacitance. Furthermore, these large gates occupy considerable silicon area and suffer from slow rise/fall times.

C. Manchester Carry Chain Implementation

To overcome the limitations of high fan-in static gates, the Manchester Carry Chain (MCC) architecture is adopted. The MCC utilizes a pass-transistor logic network to propagate carry signals. Instead of calculating the entire expression in a single complex gate, the MCC employs a chain of pass switches controlled by the propagate (P_i), delete (D_i) and generate (G_i) signals. This structure allows the carry to propagate through the chain via a low-resistance path when P_i is high, while G_i signals can independently discharge nodes to generate carries. This approach significantly reduces the transistor count and parasitic capacitance, thereby achieving the rapid carry propagation required for high-speed arithmetic operations.

Also, positive edge triggered flip-flops are used to provide input and obtain output. If input bits are available before the

rising edge of the clock, then output will be computed and present at the next rising edge of the clock.

III. TOPOLOGY AND SIZING

The architecture of the proposed 5-bit Carry Lookahead Adder is realized using a hybrid topology centered around the Manchester Carry Chain (MCC). The design is partitioned into three functional blocks: setup, carry propagation, and summation.

The setup stage, responsible for generating the bit-wise Generate (G_i), Propagate (G_i) and Delete (D_i) signals. G_i, D_i implemented using standard static CMOS AND and NOR logic. This ensures robust rail-to-rail switching and high noise margins for the control signals. However, for the critical carry propagation path, the design shifts from static logic to a Manchester Carry Chain implemented using Transmission Gates (TG). The transmission gate implementation is chosen over simple pass-transistors to eliminate threshold voltage drops and improve drive strength through the chain.

Finally, the XOR operations required for the Propagate (P_i) signals and the final Sum (S_i) calculation are not implemented in static CMOS. Instead, a Transmission Gate-based XOR topology is employed. This non-static approach significantly reduces the transistor count compared to a 12-transistor static XOR and minimizes parasitic capacitance, thereby optimizing the delay of the adder's critical path.

A. 2-INPUT CMOS AND GATE

1) *Functionality and Boolean Definition:* The 2-Input AND gate implements the fundamental Boolean operation of logical conjunction. The output (Y) is High (logic '1') only when both inputs (A and B) are simultaneously High. The functionality is defined by the following Truth Table:

Input A	Input B	Output $Y = A \cdot B$
0	0	0
0	1	0
1	0	0
1	1	1

2) *Topology and Static CMOS Design:* The 2-Input AND gate is realized using a cascaded **Static CMOS** topology. It is constructed by connecting a 2-Input **NAND Gate** in series with a **CMOS Inverter** (INV), as shown in the conceptual expression:

$$Y = \overline{\overline{A \cdot B}} = A \cdot B \quad (6)$$

3) *Transistor Sizing for Performance Optimization:* To achieve maximum performance and minimize the propagation delay (τ_{pd}), the transistors in the 2-Input NAND gate are meticulously sized to ensure its equivalent resistance in the worst-case scenario matches that of a minimum-sized static CMOS inverter. This technique is known as **Logical Effort** based sizing, aiming for **symmetric rise and fall delays** ($\tau_{pLH} \approx \tau_{pHL}$).

$$W_{NMOS} (\text{NAND}) = 8\lambda \quad (7)$$

$$W_{PMOS} (\text{NAND}) = 8\lambda \quad (8)$$

This sizing strategy ensures that the delay contribution of the 2-Input NAND gate is comparable to the minimal delay contributed by the reference Inverter, thereby optimizing the overall speed of the cascaded AND structure.

4) *Physical Design and Layout:* The physical implementation of the AND gate was carried out using the **Magic VLSI** layout tool. The design process began with the creation of a Stick Diagram to visualize the topology and optimize the routing of the polysilicon and diffusion layers.

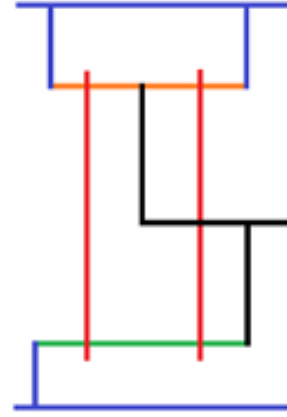


Fig. 1: Stick Diagram for CMOS NAND Gate.

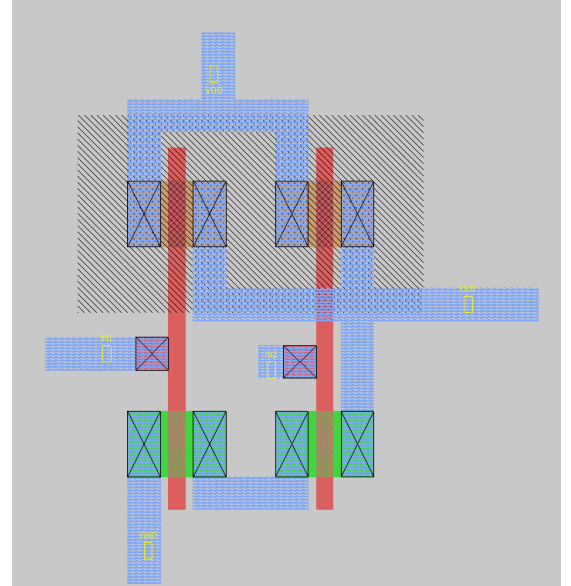


Fig. 2: Layout of the 2-Input NAND Gate.

5) *Functional Verification:* The functionality of the designed gate was verified through circuit simulation using **NGSpice**. A transient analysis was performed to observe

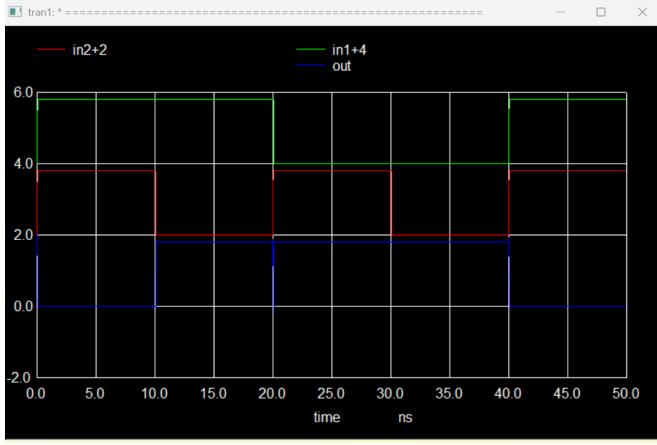


Fig. 3: NGSpice transient analysis for NAND Gate.

the output response (Y) against all possible transitions of inputs A and B .

The simulation results, shown in Figure 3, confirm that the circuit correctly follows the truth table defined in Section 2.1.1, with clear logic levels and appropriate switching characteristics.

B. 2-INPUT CMOS NOR GATE

1) *Functionality and Boolean Definition:* The 2-Input NOR gate implements the logical NOR operation, which produces a High output ('1') only when both inputs are Low. This gate is crucial for the generation of the Delete (D_i) signal in the setup stage. The functionality is defined by the following Truth Table:

Input A	Input B	Output $Y = \overline{A + B}$
0	0	1
0	1	0
1	0	0
1	1	0

2) *Topology and Static CMOS Design:* The 2-Input NOR gate is designed using a complementary **Static CMOS** topology. Unlike the NAND gate, the NOR topology places the **PMOS transistors in series** (Pull-Up Network) and the **NMOS transistors in parallel** (Pull-Down Network). The Boolean expression is:

$$Y = \overline{A + B} \quad (9)$$

3) *Transistor Sizing for Performance Optimization:* To optimize the propagation delay, the transistor sizes are adjusted based on the Logical Effort. The NOR gate is intrinsically slower than a NAND gate because the slower PMOS devices are stacked in series. To compensate for this and match the drive strength of the reference inverter:

$$W_{PMOS} \text{ (NOR)} = 4\lambda \quad (10)$$

$$W_{NMOS} \text{ (NOR)} = 16\lambda \quad (11)$$

This sizing ensures that the rise time (τ_{pLH}) is balanced with the fall time (τ_{pHL}), ensuring a symmetric output response.

4) *Physical Design and Layout:* The physical implementation of the NOR gate was carried out using the **Magic VLSI** layout tool. The Stick Diagram was first created to plan the diffusion paths and minimize the polysilicon routing.

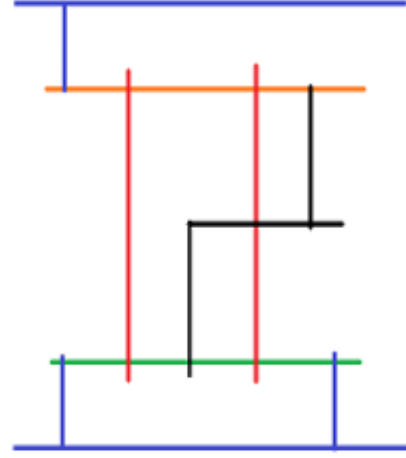


Fig. 4: Stick Diagram for CMOS NOR Gate.

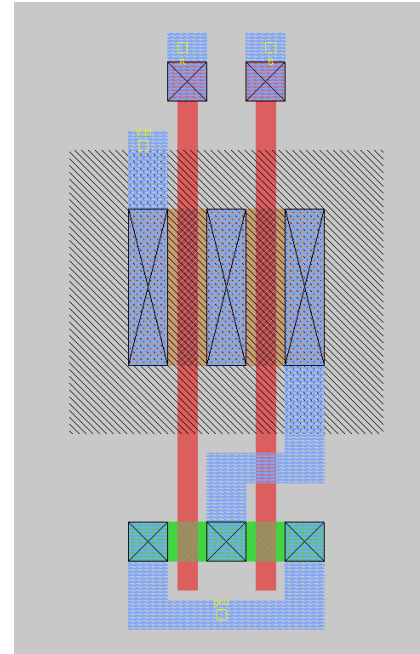


Fig. 5: Layout of the 2-Input NOR Gate.

5) *Functional Verification:* The functionality was verified using **NGSpice** transient analysis. The simulation tested all input combinations to ensure the output transitions correctly according to the NOR truth table.

As shown in Figure 6, the output remains Low for all cases except when both inputs are Low, verifying the correct operation of the gate.

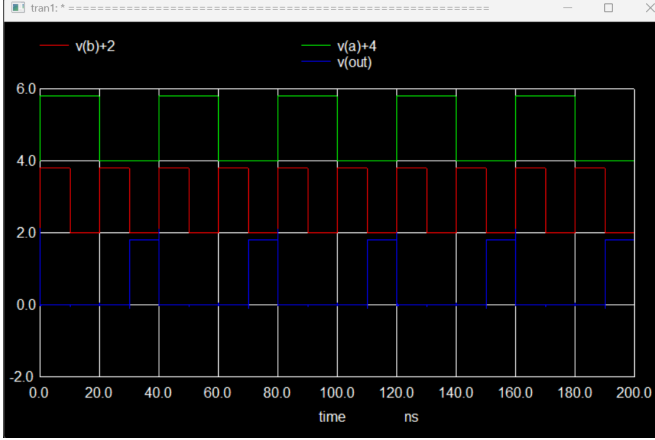


Fig. 6: NGSpice transient analysis for NOR Gate.

C. TRANSMISSION GATE (TG)

1) *Functionality and Operation:* The Transmission Gate (TG) acts as a bidirectional voltage-controlled switch. Unlike standard logic gates that drive a '0' or '1' based on a truth table, the TG either passes the input signal to the output (Low Resistance state) or isolates the output from the input (High Impedance state).

It is controlled by a control signal C and its complement \bar{C} . The operation is defined as follows:

C	\bar{C}	Switch State
0	1	OFF (Open)
1	0	ON ($V_{out} = V_{in}$)

2) *Topology and Circuit Design:* The Transmission Gate is constructed by connecting an **NMOS transistor** and a **PMOS transistor** in parallel. This specific topology is chosen to eliminate the "Threshold Voltage Drop" problem associated with single pass transistors.

$$Y = \begin{cases} A & \text{if } C = 1 \\ Z & \text{if } C = 0 \end{cases} \quad (12)$$

- **NMOS Role:** The NMOS transistor passes a strong Logic '0' (GND) but a weak Logic '1' ($V_{DD} - V_{tn}$).
- **PMOS Role:** The PMOS transistor passes a strong Logic '1' (V_{DD}) but a weak Logic '0' ($|V_{tp}|$).
- **Parallel Combination:** By using both, the TG is able to pass the full voltage swing from Rail-to-Rail (0V to V_{DD}) without signal degradation.

3) *Transistor Sizing:* To ensure symmetric performance for both rising and falling signal transitions, the internal resistance of the switch must be balanced. Since hole mobility (μ_p) is lower than electron mobility (μ_n), the PMOS transistor is sized wider than the NMOS transistor.

For this implementation, to maintain low on-resistance (R_{ON}) for the Carry Chain:

- **NMOS Width:** $W_{NMOS} = 4\lambda$
- **PMOS Width:** $W_{PMOS} = 8\lambda$

4) *Physical Design and Layout:* The physical layout was designed in **Magic VLSI**. The layout places the NMOS and PMOS devices close together to share the source/drain diffusion regions where possible, minimizing parasitic capacitance.

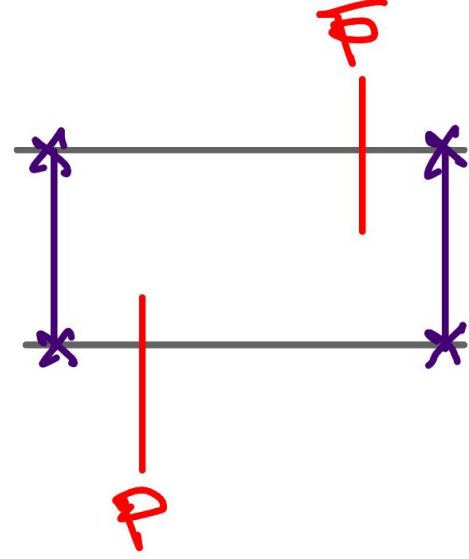


Fig. 7: Stick Diagram of the Transmission Gate.

5) *Functional Verification:* The switching characteristics were verified using **NGSpice**.

As seen in Figure 9, when C is High, the Output tracks the Input perfectly. When C is Low, the output effectively floats (or holds value depending on load), confirming the switch behavior.

D. 2-INPUT XOR GATE

1) *Functionality and Boolean Definition:* The Exclusive-OR (XOR) gate is a critical component in the Carry Lookahead Adder, used for generating the Propagate signal ($P_i = A_i \oplus B_i$) and calculating the final Sum ($S_i = P_i \oplus C_i$). The output is High only when the inputs are different.

Input A	Input B	Output $Y = A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

2) *Topology and Circuit Design:* While a conventional static CMOS XOR gate typically necessitates 12 transistors, the proposed 5-bit CLA utilizes a compact **pass-transistor XOR topology**. This implementation, depicted in Figure 10, achieves the logic function using only **4 transistors**. This architectural choice significantly mitigates silicon area overhead and minimizes parasitic capacitance at the input nodes compared to the standard CMOS approach.

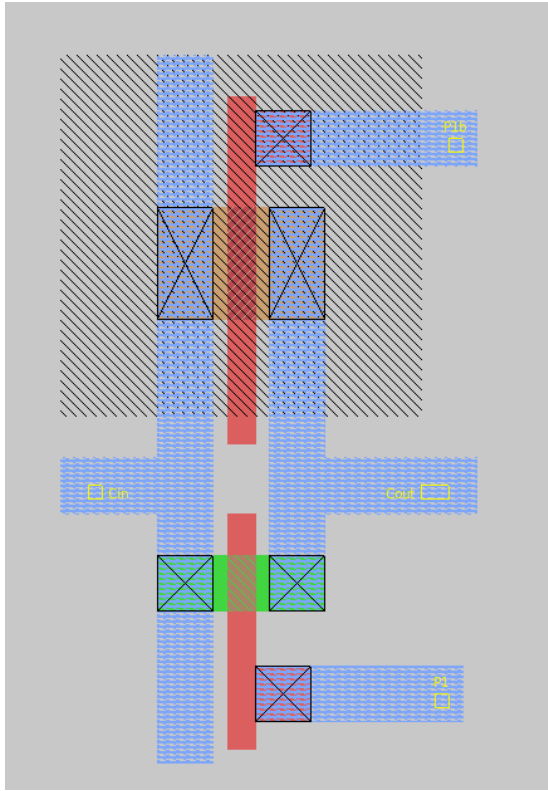


Fig. 8: Layout of the Transmission Gate.

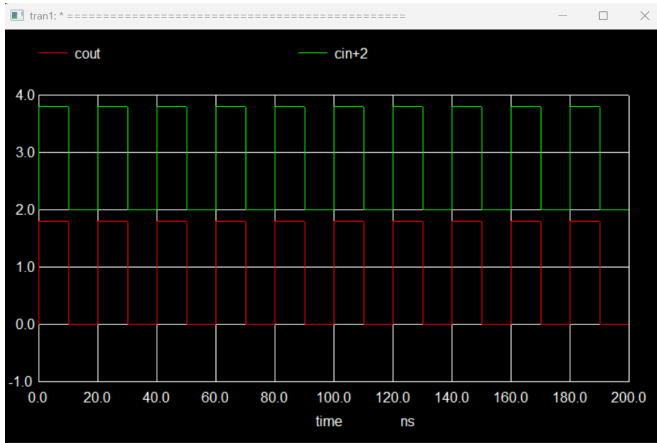


Fig. 9: NGSpice transient analysis for Transmission Gate.

3) *Transistor Sizing*: To ensure the drive strength is comparable to the standard gates and to balance the rise and fall times, the transistors are sized to match the standard inverter delay:

- **PMOS Width:** 8λ
- **NMOS Width:** 4λ

4) *Physical Design and Layout*: The physical implementation was carried out in **Magic VLSI**. A Stick Diagram was created to visualize the signal flow and minimize diffusion breaks.

The final layout achieves a high density by utilizing the

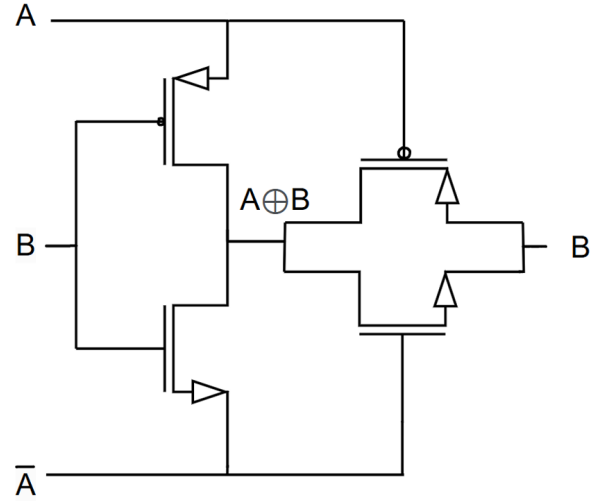


Fig. 10: Circuit Diagram of the XOR Gate.

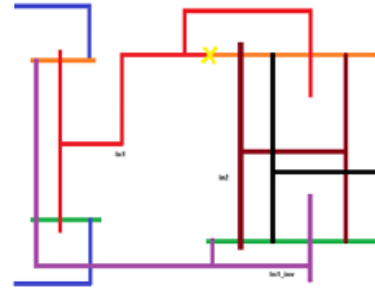


Fig. 11: Stick Diagram of the XOR Gate.

pass-transistor structure, which requires fewer contacts and metal interconnects than the static CMOS counterpart.

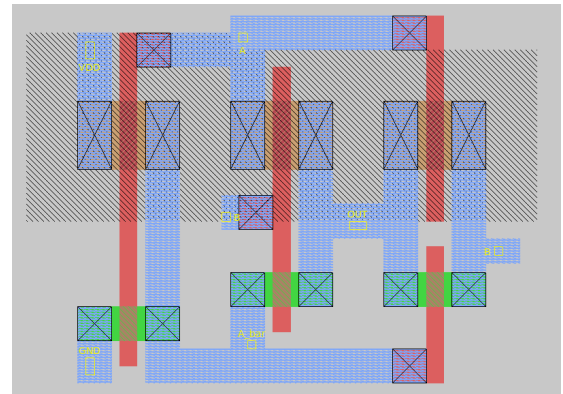


Fig. 12: Layout of the XOR Gate.

5) *Functional Verification*: The proposed XOR topology was verified using **NGSpice**. The transient analysis confirms that the gate provides a full logic swing for all inputs.

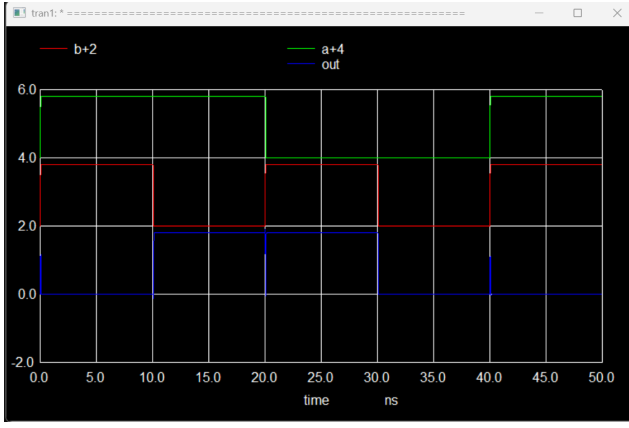


Fig. 13: NGS Spice transient analysis for the XOR Gate.

E. MODIFIED TSPC (MTSPC) D FLIP-FLOP

1) *Rationale and Functionality*: While the standard True Single Phase Clock (TSPC) D flip-flop offers the advantage of single-phase operation, it often exhibits *numerous glitches and noise* at the output. This phenomenon is primarily caused by *unnecessary toggling at the intermediate nodes* during non-critical clock phases, which results in dynamic power waste and signal integrity issues.

To alleviate this problem, a *Preset-able Modified TSPC (MTSPC)* topology is implemented. The MTSPC architecture introduces an *extra PMOS transistor* into the pull-up network. This additional device acts as a control switch to *suspend the toggling of intermediate nodes* when they are not driving the output, thereby significantly reducing redundant switching activity and power consumption.

The flip-flop operates as a positive edge-triggered device, capturing the input D on the rising edge of the clock (CLK).

2) *Performance Comparison*: The proposed MTSPC design was benchmarked against the standard TSPC architecture to validate its efficiency. Table I presents a detailed performance comparison at an input frequency of 1 GHz.

The results highlight that the MTSPC design achieves a *71% reduction in average power consumption* (from $75.43\mu W$ to $21.83\mu W$) and a significant improvement in the Clock-to-Q delay ($91.99ps$ average).

TABLE I: Performance Comparison: TSPC vs. MTSPC D Flip-Flop

Performance Parameters	TSPC DFF	MTSPC DFF
Input Clock Frequency	1 GHz	1 GHz
Clock-to-Q Delay ($L \rightarrow H$)	92.95 ps	61.08 ps
Clock-to-Q Delay ($H \rightarrow L$)	143.6 ps	122.9 ps
Average Clock-to-Q Delay	118.27 ps	91.99 ps
Setup Time (t_{setup})	70.13 ps	64.14 ps
Hold Time (t_{hold})	≈ 0	≈ 0
Average Power Consumption	$75.43 \mu W$	$21.83 \mu W$

3) *Transistor Sizing*: Proper transistor sizing is critical in dynamic logic to ensure charge retention and correct evalua-

tion. The transistors in the MTSPC circuit are sized to balance the rise/fall times and ensure the "extra PMOS" effectively controls the intermediate nodes without introducing excessive parasitic capacitance.

Using the standard λ -based design rules: To achieve maximum performance and minimize the propagation delay (τ_{pd}), the transistors are meticulously sized to ensure its equivalent resistance in the worst-case scenario matches that of a minimum-sized static CMOS inverter. This technique is known as *Logical Effort* based sizing, aiming for *symmetric rise and fall delays* ($\tau_{pLH} \approx \tau_{pHL}$).

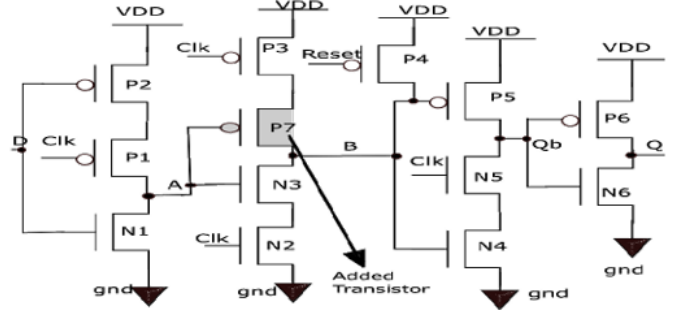


Fig. 14: Circuit Schematic of the Modified TSPC (MTSPC) D Flip-Flop.

4) *Physical Design and Layout*: The physical layout was designed in *Magic VLSI*. Despite the addition of the extra PMOS for glitch suppression, the overall area remains compact due to the elimination of other redundant transistors found in the standard TSPC intermediate stages.

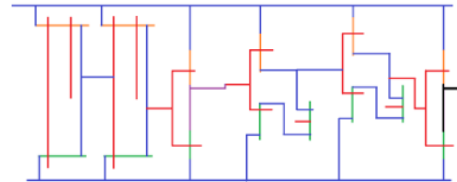


Fig. 15: Stick Diagram representing the MTSPC topology.

5) *Functional Verification*: The functionality and glitch-free operation were verified using *NGSpice*. The transient analysis confirms that the output follows the input on the rising edge of the clock, with stable logic levels during the hold phase.

6) *Simulation Results and Timing Analysis*: The transient response and timing characteristics of the designed MTSPC D Flip-Flop were analyzed using NGSpice with a supply voltage of 1.8V. The Setup Time (t_{setup}) and Propagation Delay (t_{pcq}) were extracted by measuring the delay between the 50% voltage crossing points of the input and output signals.

The simulation results, as captured in Figure 18, yield the following precise timing parameters:

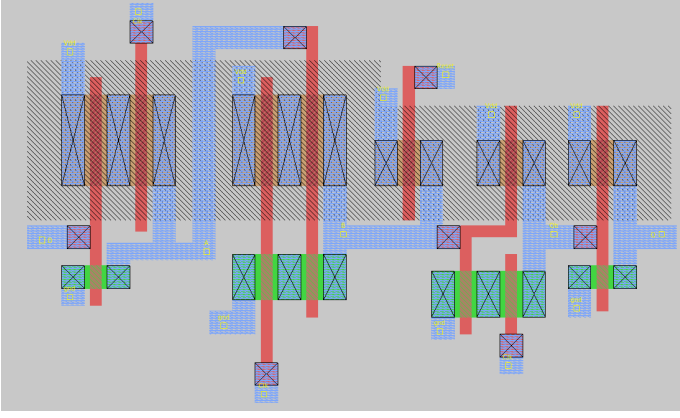


Fig. 16: Layout of the MTSPC D Flip-Flop designed in Magic VLSI.

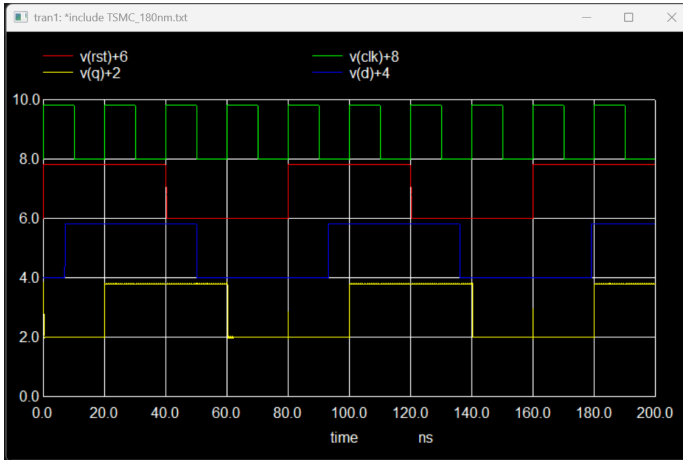


Fig. 17: NGSpice transient analysis showing the edge-triggered behavior.

```
=====
TIMING ANALYSIS RESULTS
=====
t_setup = 5.628354e-11
t_pcq = 4.054320e-11
=====
```

Fig. 18: NGSpice simulation Setup Time and Clock-to-Q delay.

- **Setup Time (t_{setup}):** Measured as the delay from the data input rising edge to the internal node evaluation.

$$t_{setup} = 56.28 \text{ ps} \quad (13)$$

- **Clock-to-Q Delay (t_{pcq}):** Measured from the rising edge of the clock to the valid output Q .

$$t_{pcq} = 40.54 \text{ ps} \quad (14)$$

These measured values indicate that the designed MTSPC flip-flop is operating with high-speed performance, surpassing the standard theoretical delays for 180nm technology. Table II summarizes the final achieved performance metrics.

TABLE II: Measured Performance Metrics of the MTSPC D Flip-Flop

Parameter	Pre-Simulation	Post-Simulation
Technology Node	180 nm	180 nm
Supply Voltage	1.8 V	1.8 V
Setup Time (t_{setup})	56.28 ps	58 ps
Propagation Delay (t_{pcq})	40.54 ps	41.62 ps

F. STATIC MANCHESTER CARRY CHAIN (MCC)

1) *Theoretical Framework and Operation:* The core acceleration mechanism of the proposed 5-bit adder is the **Static Manchester Carry Chain (MCC)**. Unlike ripple carry adders that rely on complex gate logic for every bit, the MCC utilizes a high-speed switch-based topology to propagate the carry signal (C_i) directly to the next stage (C_{i+1}).

The circuit operates using three mutually exclusive control signals derived from the Setup Stage. To align with the transistor requirements shown in the schematic, the inverted versions of Generate and Propagate are utilized:

- **Generate ($\overline{G_i}$ is Low):** When active, the PMOS transistor turns ON, pulling the carry output (C_{out}) to V_{DD} (Logic '1').
- **Delete (D_i is High):** When active, the NMOS transistor turns ON, pulling the carry output to Ground (Logic '0').
- **Propagate (P_i is High):** When active, the Transmission Gate turns ON, creating a low-resistance path that passes the input carry (C_{in}) directly to C_{out} .

2) *1-Bit Circuit Topology:* The schematic for a single bit of the Static MCC is shown in Figure 19. It employs a hybrid pass-transistor/transmission-gate structure:

- A **PMOS pull-up** driven by $\overline{G_i}$.
- An **NMOS pull-down** driven by D_i .
- A **Transmission Gate (TG)** driven by complementary signals P_i and $\overline{P_i}$.

3) *5-Bit Implementation and Layout:* The final implementation consists of **five such instances cascaded in series**. The output C_o of bit i drives the input C_i of bit $i+1$. The physical layout, shown in Figure 20, minimizes diffusion capacitance to ensure rapid signal propagation.

The functional verification in Figure 21 confirms the correct carry propagation across all 5 stages

4) *Functional Verification (5-Bit Chain):* The functionality of the complete 5-bit chain was verified using NGSpice. The transient analysis in Figure 21 demonstrates the carry propagation across the full 5-bit width, confirming that the static logic levels are maintained without degradation.

IV. STATIC MCC CLA DELAY ANALYSIS

A. Critical Path Definition

The performance of the 5-bit Carry Lookahead Adder is characterized by its **Critical Path Delay** (t_p), defined as the maximum time required for a signal transition at the input (A_0, B_0) to propagate to the most significant output (S_4 or C_{out}).

For the Static Manchester Carry Chain topology, this path consists of:

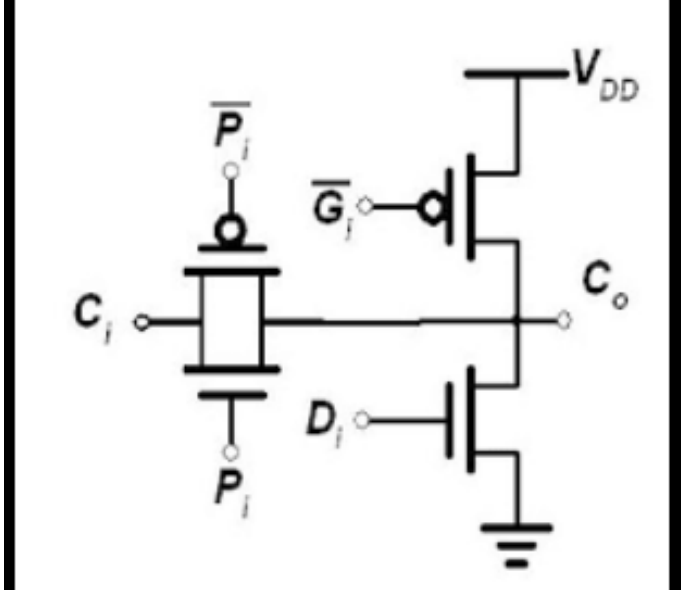


Fig. 19: Circuit Schematic of the 1-Bit Static Manchester Carry Chain Cell.

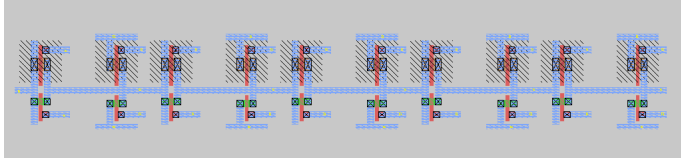


Fig. 20: Complete Layout of the 5-Bit Manchester Carry Chain.

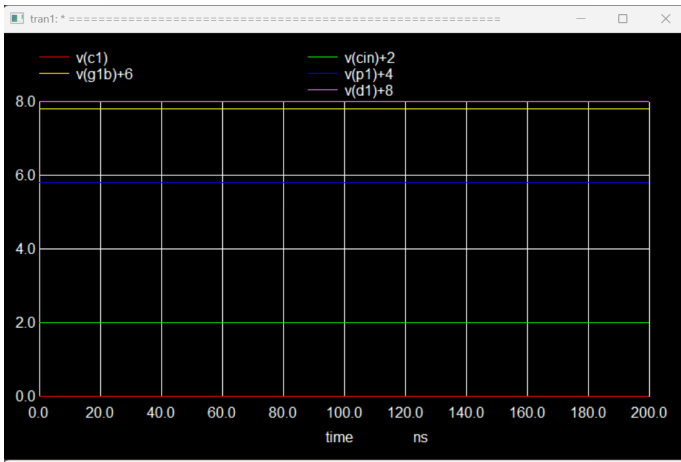


Fig. 21: Transient Analysis of the 5-Bit Manchester Carry Chain.

- 1) **Setup Stage:** Generation of P_0, G_0 signals.
- 2) **Carry Propagation:** Propagation through the 5-stage transmission gate chain ($C_0 \rightarrow C_1 \rightarrow \dots \rightarrow C_4$).
- 3) **Summation Stage:** The final XOR operation ($S_4 = P_4 \oplus C_4$).

B. Pre-Layout vs. Post-Layout Performance

To accurately assess the physical design impact, the delay was measured in two simulation environments:

- **Pre-Layout Simulation:** Ideal schematic simulation considering only intrinsic transistor capacitances.
- **Post-Layout Simulation:** Simulation performed on the extracted netlist from Magic VLSI, which includes parasitic wire capacitances (C_{par}).

Metric	Value
Pre-Layout Delay ($t_{pd,pre}$)	78 ps
Post-Layout Delay ($t_{pd,post}$)	100 ps
Percentage Increase	28%

C. Analysis of Parasitic Effects

The simulation results indicate a delay degradation of approximately **22 ps (28%**) in the post-layout phase. This increase is consistent with expected physical design overheads in 180nm technology.

The extra delay can be theoretically modeled using the **Elmore Delay** approximation. Since the Manchester Carry Chain essentially functions as an RC ladder network, the delay increases quadratically with the number of series stages if unbuffered, but linearly in this short 5-bit chain. The post-layout extraction reveals two primary contributors:

- **Junction Diffusion Capacitance (C_{diff}):** The shared source/drain regions between the cascaded Transmission Gates add significant parasitic capacitance. In the schematic, this is idealized, but in the layout, the area of the diffusion regions (AD, PD) creates a real capacitive load that must be charged by the signal.
- **Interconnect Resistance (R_{metal}):** The metal routing connecting the P, G, D setup logic to the carry chain introduces series resistance. This additional resistance interacts with the gate capacitance of the TGs, increasing the RC time constant of the critical path.

Despite this increase, the final delay of **100 ps** confirms that the layout is compact and efficient, avoiding the large interconnect delays often seen in standard cell-based designs.

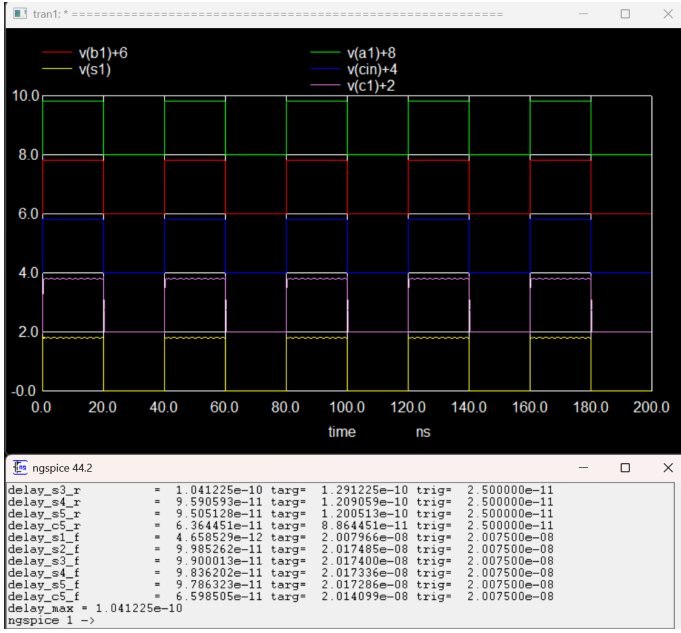


Fig. 22: NGS Spice waveform highlighting the critical path propagation delay (100 ps).

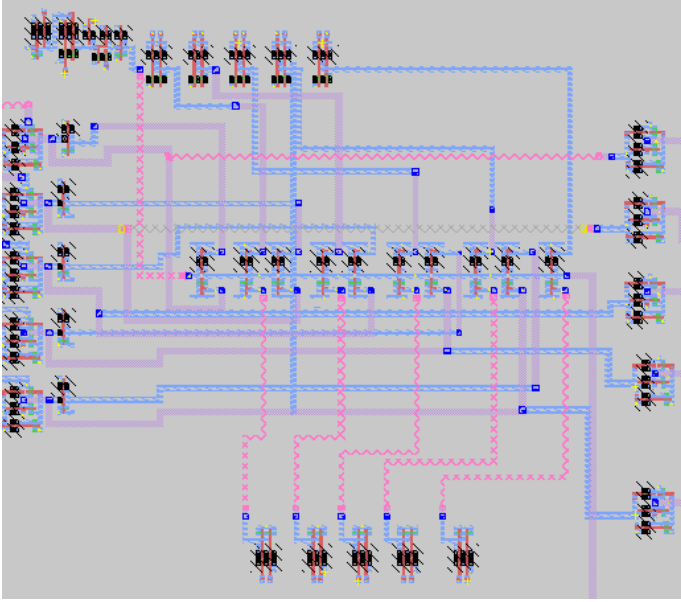


Fig. 23: Extracted Layout of the complete 5-bit CLA used for parasitic simulation.

V. REGISTERED 5-BIT CLA INTEGRATION

A. Input and Output Regulation Strategy

To transition from a strictly combinational logic block to a fully synchronous processing unit, the 5-bit Carry Lookahead Adder is integrated with **Input and Output Register Banks**. This regulation strategy is essential for modern VLSI systems to ensure signal integrity, filter out glitch propagation from previous processing stages, and synchronize the computation with the global system clock.

The design utilizes the high-performance **MTSPC D Flip-Flops** (characterized in Section IV) to form two critical regulation stages:

- 1) **Input Register Bank:** A bank of 10 D Flip-Flops captures the asynchronous input vectors $A[4 : 0]$ and $B[4 : 0]$ on the rising edge of the clock. This ensures that the Static MCC evaluates only stable inputs, preventing race conditions and false carry generation.
- 2) **Output Register Bank:** A bank of 6 D Flip-Flops captures the final results ($Sum[4 : 0]$ and C_{out}) to isolate the output load from the internal critical path.

B. Circuit Topology and Pipelining

The schematic configuration connects the Q outputs of the Input Register bank directly to the PGD Setup stage of the adder. The adder's outputs are then fed into the D inputs of the Output Register bank.

This topology introduces a **Pipeline Latency** of 1 Clock Cycle. Data sampled on the rising edge T_{clk} propagates through the adder logic during the high/low phases of the clock, and the result is latched at the output on the subsequent rising edge ($T_{clk} + T_{period}$).

C. Timing Analysis and Frequency Constraints

For the registered CLA to operate correctly without setup violations, the clock period (T_{clk}) must satisfy the critical path timing constraint. The minimum clock period is derived from the intrinsic delays of the MTSPC Flip-Flop and the propagation delay of the CLA logic:

$$T_{min} \geq t_{pcq} + t_{pd,logic} + t_{setup} \quad (15)$$

Where:

- t_{pcq} : Clock-to-Q delay of the Input Register (≈ 41.62 ps).
- $t_{pd,logic}$: Critical path delay of the Static MCC Adder (≈ 100 ps).
- t_{setup} : Setup time of the Output Register (≈ 58 ps).

D. Physical Design and Layout

The physical implementation in Magic VLSI integrates the MTSPC DFF cells with the core MCC Adder. The layout is floor-planned to minimize the distance between the register outputs and the adder inputs ($Q_{reg} \rightarrow \text{Adder}_{in}$), thereby reducing wire parasitic capacitance.

- **Area Optimization:** The DFFs are aligned in a pitch-matched row to share power rails (V_{DD}, GND) and Clock lines.
- **Clock Distribution:** A balanced clock tree structure is used to deliver the CLK signal to all 16 Flip-Flops simultaneously. This minimizes clock skew, ensuring that input and output latching events occur synchronously.

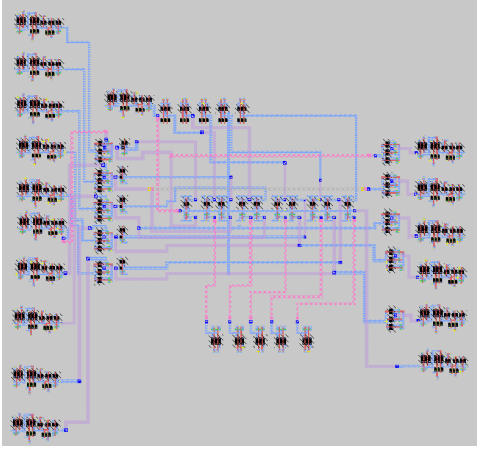


Fig. 24: Complete Layout of the 5-Bit CLA with Input/Output Registers.

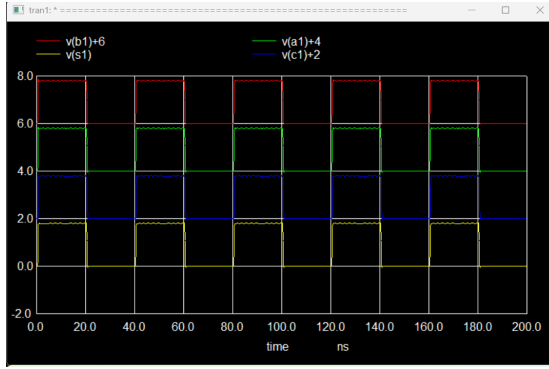


Fig. 25: Pre-Layout Transient Simulation showing synchronous input capture.

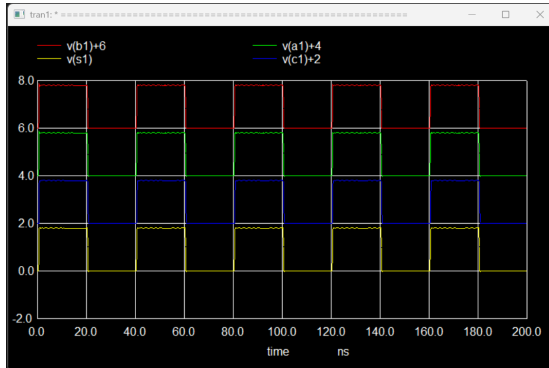


Fig. 26: Post-Layout Transient Simulation verifying operation with parasitics.

VI. MAXIMUM CLOCK FREQUENCY ANALYSIS

A. Theoretical Derivation

The maximum operating frequency (f_{max}) of the synchronous system is the inverse of the minimum allowable clock period (T_{min}). As established in the timing constraints, the clock period is limited by the sum of the sequential overheads

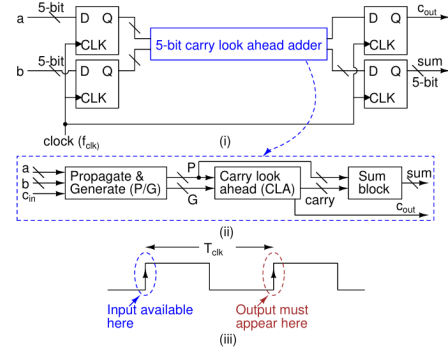


Fig. 27: Circuit Diagram.

imposed by the MTSPC registers and the combinational propagation delay of the adder logic.

The frequency is calculated as:

$$f_{max} = \frac{1}{T_{min}} = \frac{1}{t_{pcq} + t_{pd,logic} + t_{setup}} \quad (16)$$

Where the denominator represents the total time required for data to launch from the source register, propagate through the critical path of the adder, and successfully capture at the destination register before the next clock edge.

B. Performance Calculation (Pre vs. Post Layout)

To quantify the impact of physical design parasitics on system speed, the maximum frequency was calculated separately for both simulation environments.

- **Pre-Layout (Ideal):** Using intrinsic delays, the total critical path duration is approximately **174.82 ps**, yielding a theoretical maximum frequency of **5.72 GHz**.
- **Post-Layout (Extracted):** Including parasitic capacitance and resistance, the critical path extends to **199.62 ps**. This reduces the maximum safe operating frequency to **5.01 GHz**.

Table III details the component-wise breakdown and the final frequency results.

TABLE III: Maximum Frequency Calculation: Pre-Layout vs. Post-Layout

Timing Parameter	Pre-Layout	Post-Layout
Clk-to-Q Delay (t_{pcq})	40.54 ps	41.62 ps
Adder Critical Path ($t_{pd,logic}$)	78.00 ps	100.00 ps
Setup Time (t_{setup})	56.28 ps	58.00 ps
Total Period (T_{min})	174.82 ps	199.62 ps
Max Frequency (f_{max})	5.72 GHz	5.01 GHz

Despite the degradation caused by parasitic effects, the post-layout frequency of **5.01 GHz** indicates that the proposed 5-bit CLA design is suitable for high-speed digital processing applications in 180nm technology.

VII. FLOOR PLANNING AND LAYOUT METRICS

A. Floor Plan Strategy

The physical design follows a **Centralized Core Architecture**, where the critical Manchester Carry Chain (MCC) is placed in the center of the layout to minimize signal propagation delays from the surrounding control logic. The floor plan is partitioned into five distinct functional regions:

- 1) **Central Region (MCC Core):** The 5-stage Static Manchester Carry Chain is located in the exact middle of the layout. This central placement ensures balanced routing distances for the control signals coming from all directions.
- 2) **Left Region (Input Stage):** This region contains the **Input D-Flip Flop Bank** (for latching inputs A, B) and the **Propagate Generation (P_i)** logic. Placing the P_i logic here allows the propagate signals to drive the MCC chain from left to right.
- 3) **Right Region (Output Stage):** This region contains the **Summation Logic (S_i)** and the **Output D-Flip Flop Bank**. The carry signals emerging from the central MCC are immediately processed here to generate the final Sum bits and latched.
- 4) **Top Region (Generate Logic):** The logic for generating the **Generate (G_i)** signals is placed in the top block, routing vertically down into the MCC core.
- 5) **Bottom Region (Delete Logic):** The logic for generating the **Delete (D_i)** signals is placed in the bottom block, routing vertically up into the MCC core.

This concentric arrangement minimizes the "Critical Path" wire length by surrounding the carry chain with its necessary setup and summation logic.

B. Pitch Analysis of Regular Structures

To ensure layout compactness and seamless abutment, the design utilizes a **Standard Cell Approach**.

1) **Vertical Pitch (Cell Height):** The **Vertical Pitch** is defined as the fixed distance between the V_{DD} and GND power rails. All unit cells (DFF, Setup Logic, MCC Bit) were designed with a unified vertical pitch to allow them to be aligned in rows sharing a common power bus.

2) **Horizontal Pitch (Cell Width):** The **Horizontal Pitch** defines the periodicity of the repeating structures. Since the design is a 5-bit system, the layout relies on the regular repetition of 1-bit slices across the horizontal axis.

Table ?? summarizes the measured physical dimensions of the regular structures in the design.

VIII. VERILOG HDL IMPLEMENTATION

A. RTL Modeling Strategy

To validate the logical functionality of the architecture in a digital design flow, the Carry Lookahead Adder was modeled in Verilog HDL. The implementation adopts a **Register Transfer Level (RTL)** abstraction, which clearly delineates the synchronous storage elements from the combinational lookahead logic.

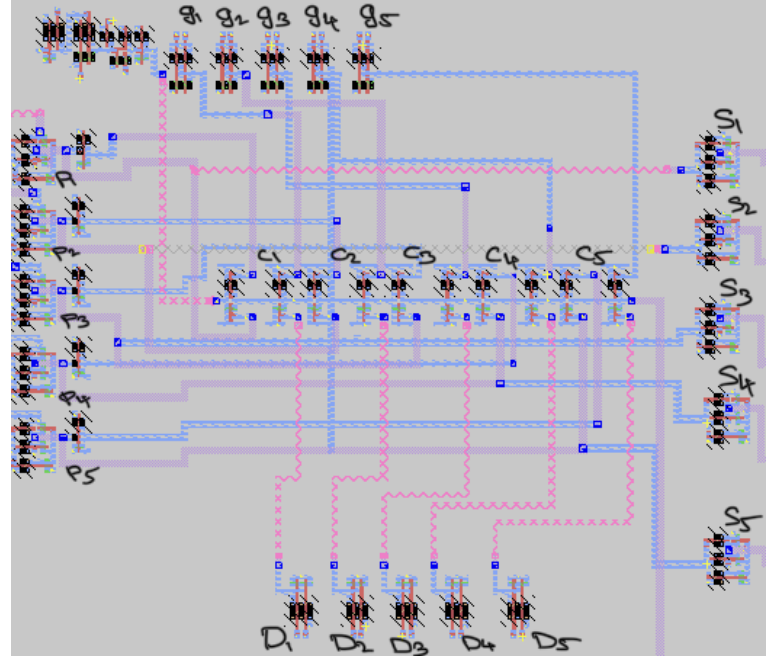


Fig. 28: Annotated Floor Plan showing the Central MCC Core surrounded by Setup Logic (Top/Bottom/Left) and Output Logic (Right).

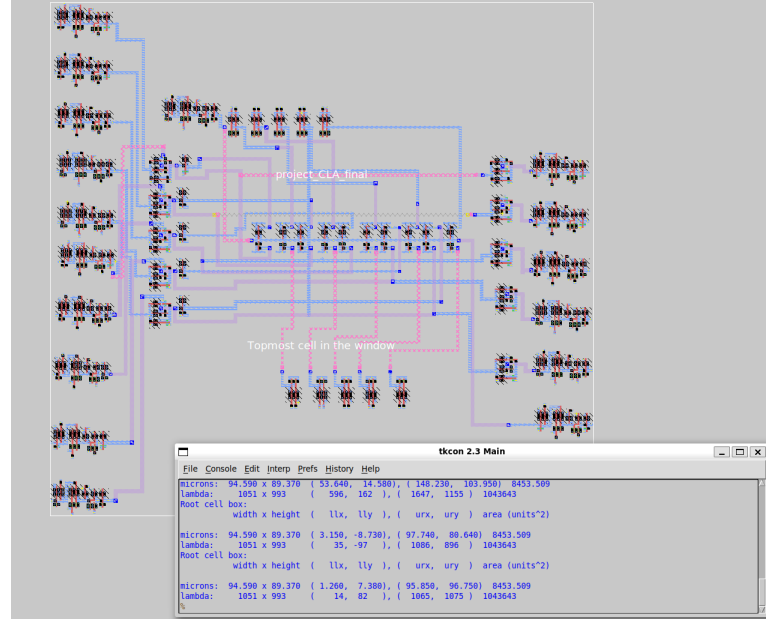


Fig. 29: Complete Layout of the 5-Bit CLA with Input/Output Registers.

The code structure, as presented in the design, consists of three distinct processing stages:

- 1) **Input Registration:** An edge-triggered always block captures the inputs (a , b , c_0) into internal registers (a_reg , b_reg , c_0_reg) to synchronize the data arrival.

2) **Lookahead Logic (Dataflow):** The core CLA equations are implemented using continuous assignment (assign) statements. This explicitly defines the Propagate (P) and Generate (G) signals, followed by the parallel calculation of the carry signals (C_1 to C_4) using the standard expansion formulas:

$$C_{i+1} = G_i + (P_i \cdot C_i) \quad (17)$$

3) **Output Registration:** A second sequential block captures the computed Sum (`s_wire`) and Carry-out (`c4_wire`) into output registers (`s_reg`, `c4_reg`), effectively creating a pipelined stage.

B. Simulation and Verification

The Verilog model was verified using a testbench that stimulates the clock and applies random input vectors. The simulation waveforms confirm the synchronous behavior: the output updates exactly one clock cycle after the input is applied, verifying the correctness of the registered pipeline topology.

```

1 timescale 1ns / 1ps
2
3 module cla_adder(
4     input wire [3:0] a,
5     input wire [3:0] b,
6     input wire c0,
7     input wire clk,
8     output wire [3:0] s,
9     output wire c4
10 );
11     wire [3:0] p, g, c1, s_wire;
12     wire c4_wire;
13     reg [3:0] a_reg, b_reg;
14     reg c0_reg;
15     reg [3:0] s_reg;
16     reg c4_reg;
17
18     always @(posedge clk) begin
19         a_reg <= a;
20         b_reg <= b;
21         c0_reg <= c0;
22     end
23
24     assign p = a_reg ^ b_reg;
25     assign g = a_reg & b_reg;
26
27     assign c1 = c0_reg;
28     assign c2 = g[0] | (p[0] & c0);
29     assign c3 = g[1] | (p[1] & g[0]) | (p[1] & p[0] & c0);
30     assign c4_wire = g[2] | (p[2] & g[1]) | (p[2] & p[1] & g[0]) | (p[2] & p[1] & p[0] & c0);
31     assign c4_wire = g[3] | (p[3] & g[2]) | (p[3] & p[2] & g[1]) | (p[3] & p[2] & p[1] & g[0]) | (p[3] & p[2] & p[1] & p[0] & c0);
32     [0];
33     assign s_wire = p ^ c4_wire;
34
35     always @(posedge clk) begin
36         s_reg <= s_wire;
37         c4_reg <= c4_wire;
38     end
39
40     assign s = s_reg;
41     assign c4 = c4_reg;
42
43 endmodule

```

Fig. 30: Verilog RTL description of the Registered Carry Lookahead Adder.

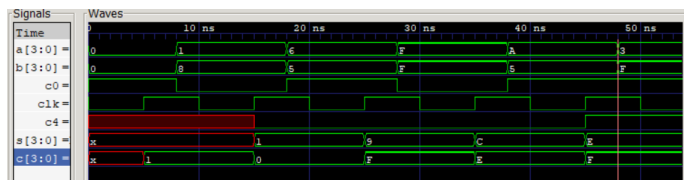


Fig. 31: Simulation waveforms showing the clock-triggered input and output updates.

IX. FPGA IMPLEMENTATION AND SYNTHESIS

A. Implementation Flow

To validate the design in a physical hardware environment, the Registered Carry Lookahead Adder was implemented on

an FPGA platform (e.g., Xilinx Artix-7). The Verilog RTL description described in the previous section served as the input for the synthesis tool. The implementation flow follows a standard three-stage process:

B. Hardware Validation Results

Two specific test vectors were applied to validate the carry propagation and overflow logic:

• Example 1 :

- **Inputs:** $A = 9$, $B = 4$, $C_0 = 0$.
- **Expected Result:** $9 + 4 = 13$.
- **Observed Output:** Green LEDs displayed 01101 (13). Correctness confirmed.

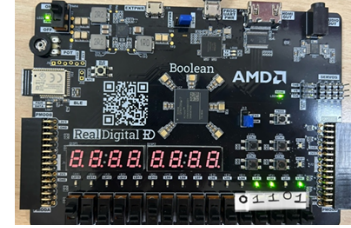


Fig. 32: FPGA Board Setup showing the Green LED output results for the test cases.

• Example 2:

- **Inputs:** $A = 13$, $B = 12$, $C_0 = 0$.
- **Expected Result:** $13 + 12 = 25$.
- **Observed Output:** Green LEDs displayed 11001 (25). Correctness confirmed.

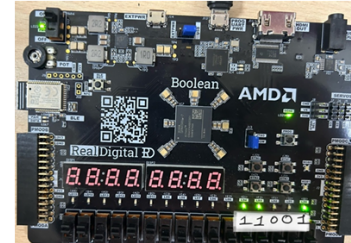


Fig. 33: FPGA Board Setup showing the Green LED output results for the test cases.

X. CONCLUSION

This project successfully presented the design, physical implementation, and hardware validation of a high-speed 5-bit Carry Lookahead Adder (CLA) accelerated by a Static Manchester Carry Chain (MCC).

The design leveraged 180nm CMOS technology to address the critical path latency often found in conventional ripple carry adders. By utilizing a transmission-gate-based carry chain, the propagation delay was significantly minimized. The integration of MTSPC D Flip-Flops facilitated a robust registered architecture, ensuring reliable synchronous operation.

Key achievements of this work include:

- **High-Speed Performance:** The critical path delay was measured at 78 ps (pre-layout) and 100 ps (post-layout), validating the efficiency of the static MCC topology.
- **Physical Design Verification:** The complete layout in Magic VLSI demonstrated a compact floor plan with an estimated maximum operating frequency of 5.01 GHz.
- **Hardware Validation:** The logic functionality was successfully prototyped on an FPGA platform, confirming correct carry propagation and overflow detection under real-world test vectors.

Future work may involve scaling the design to 32-bit or 64-bit architectures and exploring dynamic logic styles to further reduce power consumption.

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to the teaching assistants for their invaluable guidance and continuous encouragement throughout the course of this project. Their insights into VLSI design and physical layout were instrumental in the successful completion of this work.

I am also deeply thankful to the professor for delivering an engaging and challenging course that significantly enhanced my understanding of VLSI design and its intricacies. Undertaking this project was a truly enriching experience, effectively bridging the gap between theoretical concepts and their practical application.

REFERENCES

- [1] N. H. Weste and D. M. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 4th ed., Boston, MA: Addison-Wesley, 2011.
- [2] J. M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits: A Design Perspective*, 2nd ed., Upper Saddle River, NJ: Prentice Hall, 2003.
- [3] J. Yuan and C. Svensson, "High-speed CMOS circuit technique," *IEEE Journal of Solid-State Circuits*, vol. 24, no. 1, pp. 62–70, Feb. 1989. (Reference for TSPC/MTSPC logic).
- [4] T. Lynch and E. E. Swartzlander, "A Spanning Tree Carry Lookahead Adder," *IEEE Transactions on Computers*, vol. 41, no. 8, pp. 931–939, Aug. 1992.
- [5] P. Nenzi and H. Vogt, *Ngspice Users Manual*, Version 26, 2014. [Online]. Available: <http://ngspice.sourceforge.net>.