VLSI PROJECT

5-bit carry look ahead (CLA) adder

Name: ***Jai Srikar M***

Roll Number: ***2024102041***

jaisrikar.m@students.iit.ac.in

Introduction:

This work focuses on the development of a **5-bit Carry Look-Ahead (CLA) adder**, an arithmetic unit engineered for rapid binary addition. The architecture is organized into distinct functional sections: calculation of **propagate and generate** terms, a look-ahead network for **early carry prediction**, and a **summation stage**. The circuit operates synchronously—input words are sampled just before the active clock edge, and the resulting outputs become valid at the next rising edge.

To maintain timing alignment across the data path, D flip-flops are integrated to capture partial results and carry information. These registers are realized using a transistor sizing ratio of $W_p=20\lambda$ and $W_n=10\lambda$ where $\lambda=0.09\mu m$. The logic blocks responsible for **AND and XOR** operations are built using **dynamic logic techniques**, chosen for their **speed and compact layout.**

The complete system has undergone extensive validation, including both pre-layout and post-layout simulation. A corresponding Verilog HDL model was also created, and the final design was implemented and tested on FPGA hardware to confirm correct functional behaviour.

I. Architecture:

A.) D- Flip Flop:

The D flip-flop used in this work is a **static C²MOS master–slave design** that provides clean edge-triggered operation and reliable storage for synchronous digital systems. It consists of two latches connected in series: the master latch is transparent when the clock is low, and the slave latch is transparent when the clock is high. Together, they ensure that the input is sampled only at the rising edge of the clock, while the output remains stable throughout the rest of the cycle.

Each latch is built using CMOS transmission gates controlled by complementary clock signals, followed by cross-coupled inverters that hold the stored value statically. Because the storage is fully static, the flip-flop does not rely on charge on internal nodes and therefore avoids leakage issues common in dynamic latches.

In this design, small buffer stages are placed before and after the latching elements to improve signal integrity, especially when receiving inputs from dynamic logic such as the Manchester carry chain. These buffers restore full logic levels, isolate the latch from unwanted loading, and ensure strong, clean transitions at the output.

Overall, the C²MOS flip-flop provides a compact, low-power, and robust solution for synchronization, making it well suited for the pipelined operation of the 5-bit CLA adder.

B.) Carry Look Ahead Adder:

The Carry Look-Ahead (CLA) adder overcomes the major speed limitation seen in ripple-carry architectures by generating carry values simultaneously instead of propagating them step by step. In a ripple-carry adder, every bit position depends on the completion of the carry from the preceding

stage, causing the overall delay to grow with the adder's bit width. By contrast, a CLA adder computes all required carry terms in parallel, removing this sequential dependency and significantly improving performance.

In a CLA adder, each bit position is associated with two key signals: the *propagate* term p_i and the *generate* term g_i. These are defined as:

$$p_i = a_i \oplus b_i \qquad\qquad gi = a_i \cdot b_i$$

The propagate term reflects whether an incoming carry will pass through that stage, whereas the generate term identifies whether that bit position will produce a carry on its own, independent of any previous carry. Using these two signals, the carry for the next bit position is determined by:

$$c_{i+1} = g_i + (p_i \cdot c_i)$$

where a_i and b_i are the inputs at bit position i^{th} position and c_i is the carry entering that stage. In a 5-bit design, this results in carry values c_1 through c_6, with the assumption that $c_0=0$ when no external carry-in is provided.

Once the carries are available, each sum bit is evaluated as:

$$sum_i = p_i \oplus c_{i-1}$$

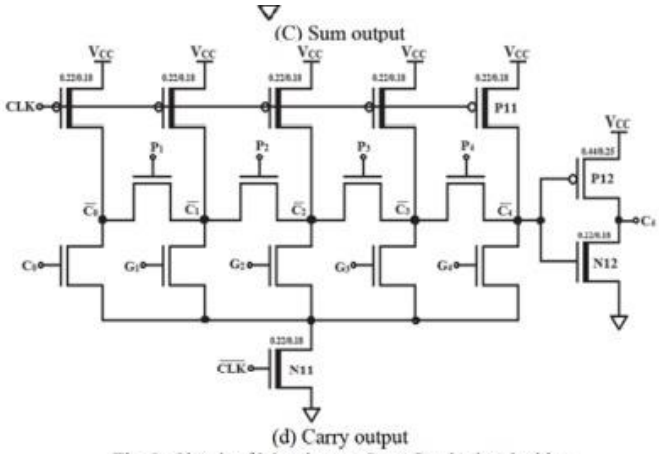


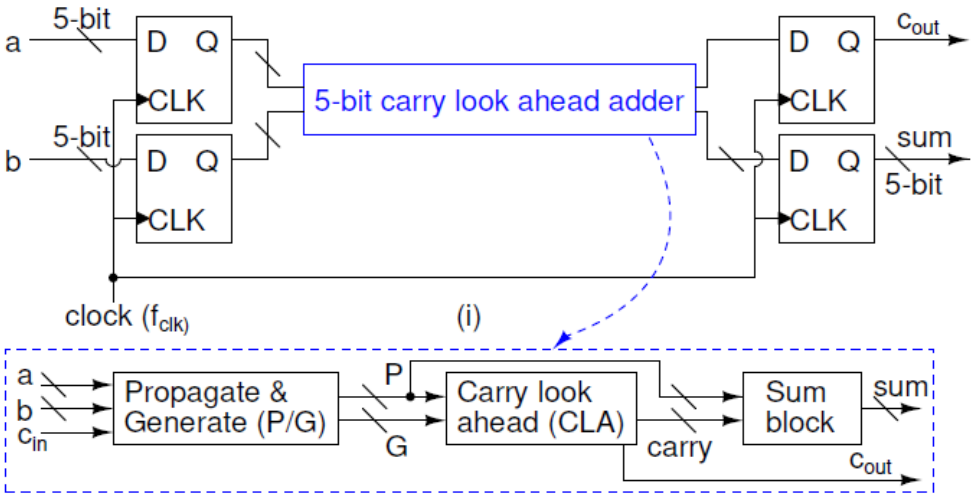
Fig. 3. Circuit of Manchestor Carry Look-ahead adder

C.) Integrated Structure:

The overall system operates as a two-stage synchronous pipeline formed by the input flip-flops, the combinational CLA logic, and the output flip-flops. The input registers capture all five bits of operands *A* and *B*, along with the carry-in, on the rising edge of the clock. This ensures that the CLA adder receives a fully time-aligned, stable set of inputs during the next clock period.

Once the inputs are latched, the Manchester carry chain and sum-generation network compute all internal propagate, generate, carry, and sum signals purely combinationally. These results are not immediately used by downstream logic; instead, they are captured by a second bank of edge-triggered C²MOS flip-flops. At the next rising clock edge, the output registers latch the computed sum bits and the final carry-out, producing a clean, synchronized output.

This structure forms a **fully pipelined single-stage adder**, where each clock cycle moves data from input registers → combinational CLA → output registers. The pipeline isolates combinational delay from the sequential boundaries, supports high-frequency operation, and allows new operands to be applied every cycle, enabling continuous throughput with deterministic timing.



II. Design Topology:

A.) Inverter:

Implemented using a complementary PMOS/NMOS pair. Provides full-rail output, strong noise margins, and is used extensively for:

- Signal restoration
- Polarity generation (e.g., clock inversion)
- Buffering in dynamic logic paths

This is the basic building block in both combinational logic and latch feedback loops.

B.) XOR Gate:

Implemented using a multi-device CMOS pull-up and pull-down structure with an internal inverter. Used for:

- propagate signal generation $p=A \oplus B$
- sum computation $s=p \oplus c$

Static implementation ensures full logic levels and avoids charge-sharing issues typical in pass-transistor XORs.

C.) AND Gate:

Implemented using NMOS pass devices aided by a PMOS restoration transistor. Used for generate signal computation: $g=A \cdot B$
Advantages:

- reduced transistor count
- lower load capacitance
- faster switching in dynamic environments

Requires restoration buffers because PTL can degrade logic levels.

D.) Domino Logic:

Carry chain uses:

- PMOS precharge transistor
- NMOS evaluation network
- static inverter for restoration

Advantages:

- high speed due to reduced transistor count
- fast evaluation when clock enables the stage

Used to compute $c_{i+1}=g_i+p_i \cdot c_i$ efficiently across all 5 bits.

E.) Transmission Gates:

Made from parallel NMOS + PMOS devices with complementary gate control.
Used in:

- C^2MOS flip-flop (input gating and latch transparency control)

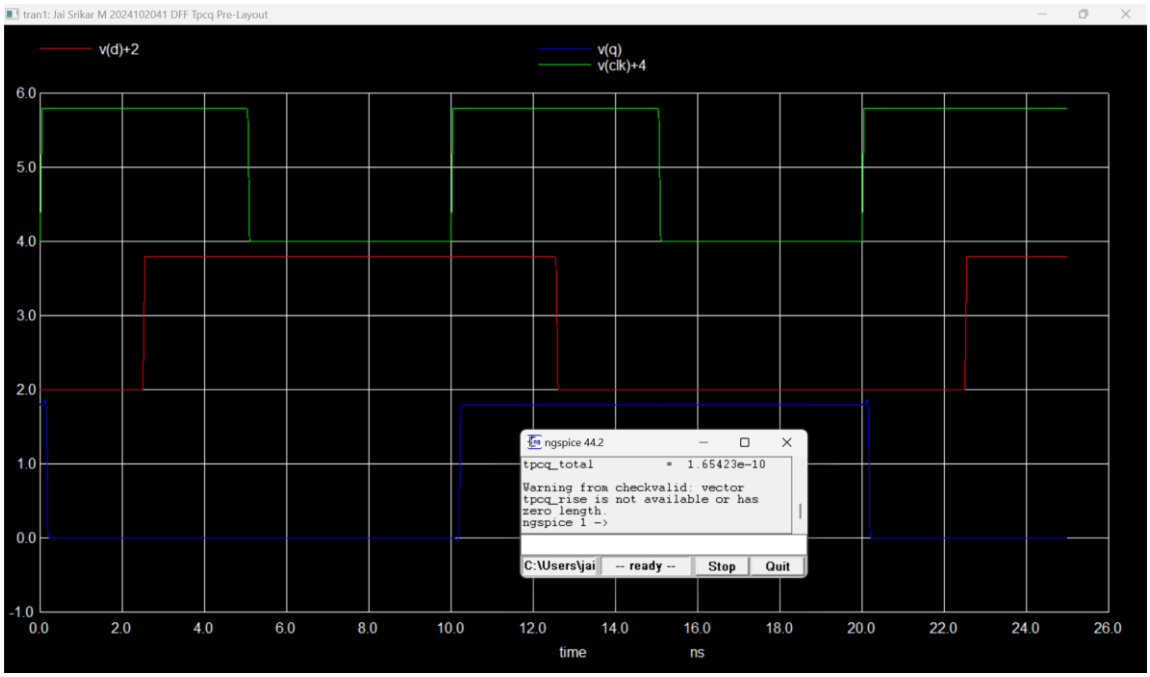
Advantages:

- passes both 0 and 1 without threshold drop
- symmetric delay
- ideal for latch design

III. Block Verification:

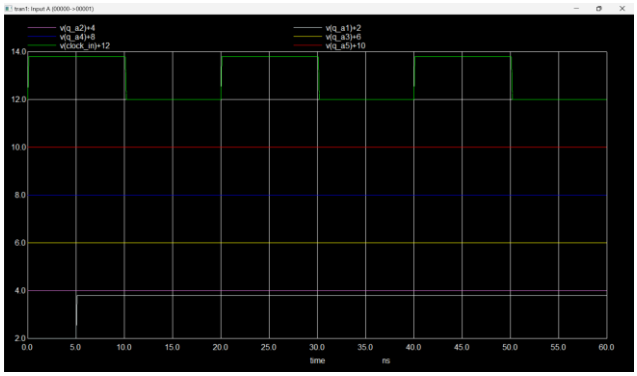
Each functional block was simulated independently using NGSPICE to verify correct operation before integration.

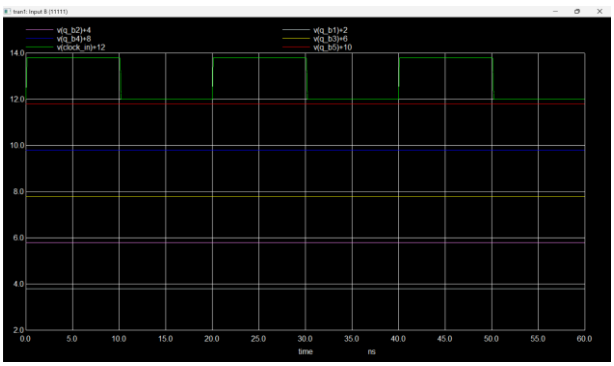
A.) D – Flip Flop:



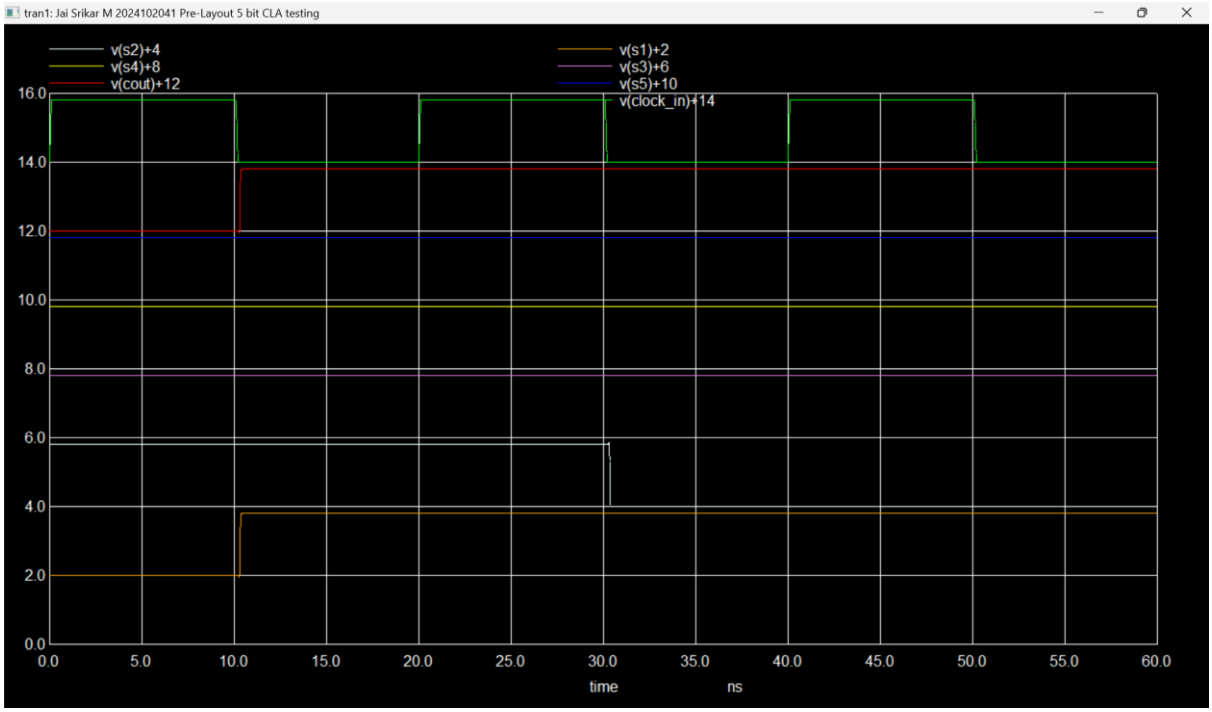
B.) Integrated 5-Bit CLA Adder

Input A: Input B:



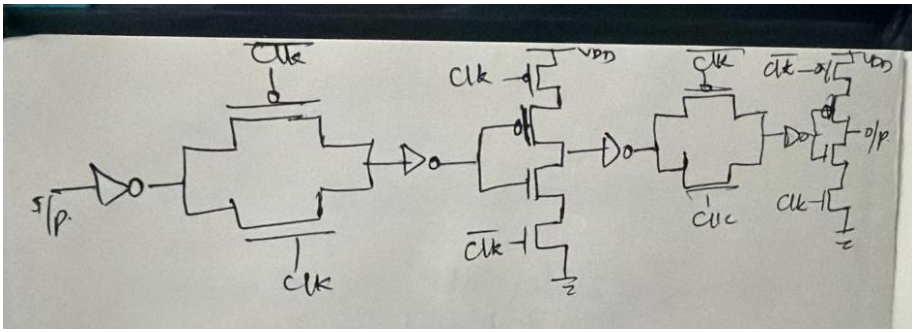


Output:



As you can see, proper testing has been done on all the blocks to check if they are functioning properly.

IV. Stick Diagram:



This drawing depicts a robust **Transmission-Gate Master-Slave D Flip-Flop**. It uses a "Hybrid" topology that combines the best features of two different logic styles to ensure stability.

V. Delay Analysis:

D-Flip Flop:

```
tpcq_avg                =   1.07530e-10
```

```
t_setup             =  1.000000e-08 targ=  1.002500e-08 trig=  2.500000e-11
t_hold              =  7.000000e-09 targ=  1.702500e-08 trig=  1.002500e-08
```

Parameter	Value	
Clock-to-Q (tpcq_avg)	**1.0753×10^{-10} s (≈ 107.5 ps)**	
Setup Time (t_setup)	**1.0000×10^{-8} s (≈ 10 ns)**	
Hold Time (t_hold)	**7.0000×10^{-9} s (≈ 7 ns)**	

Integrated 5-Bit CLA:

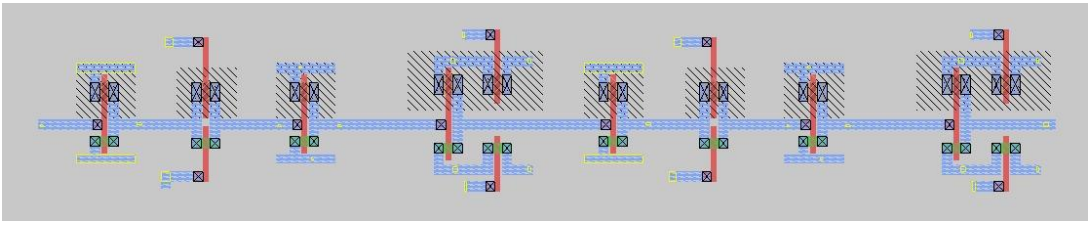
```
  Measurements for Transient Analysis

t_logic_delay       =  7.074952e-11 targ=  1.207495e-10 trig=  5.000000e-11
tpcq_out            =  -9.734203e-09 targ=  1.031580e-08 trig=  2.005000e-08
t_critical          =  2.70750e-10
f_max_ghz           =  3.69345e+09
```

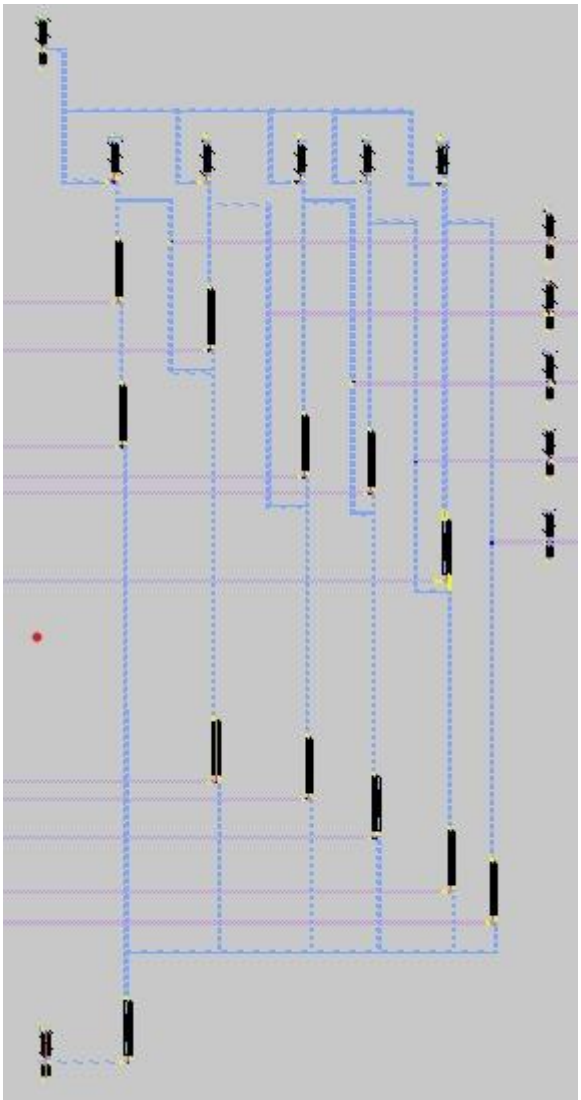
Parameter	Value	
Logic Delay of Adder (t_logic_delay)	**7.07495×10^{-11} s (≈ 70.7 ps)**	
Clock-to-Q (tpcq_out)	**9.73420×10^{-9} s**	
Critical Path Delay (t_critical)	**2.70750×10^{-10} s (≈ 270.8 ps)**	
Maximum Frequency (f_max)	**3.69345×10^{9} Hz (≈ 3.69 GHz)**	

VI. Magic Layout

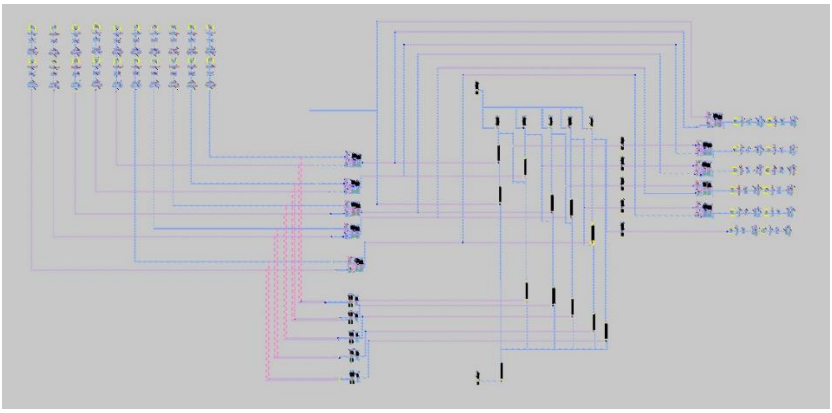
A.) Flip-Flop:



B.) Sum Block:



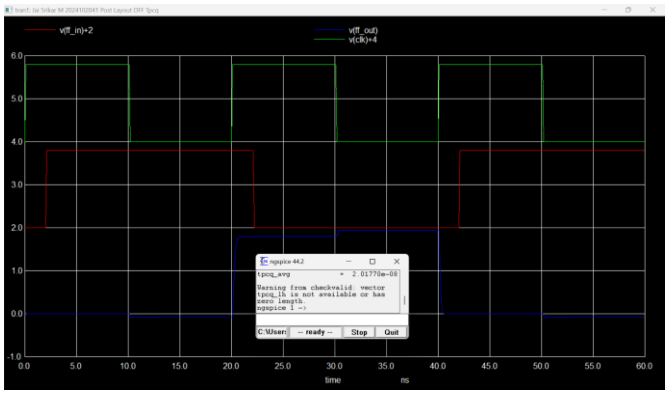
C.) Integrated CLA 5- bit adder



VII. Post layout Simulation

As we can see here, proper testing and delay analysis of each block has been performed.

A.) D- Flip Flop




```
tpcq_avg                =   2.01770e-08
```

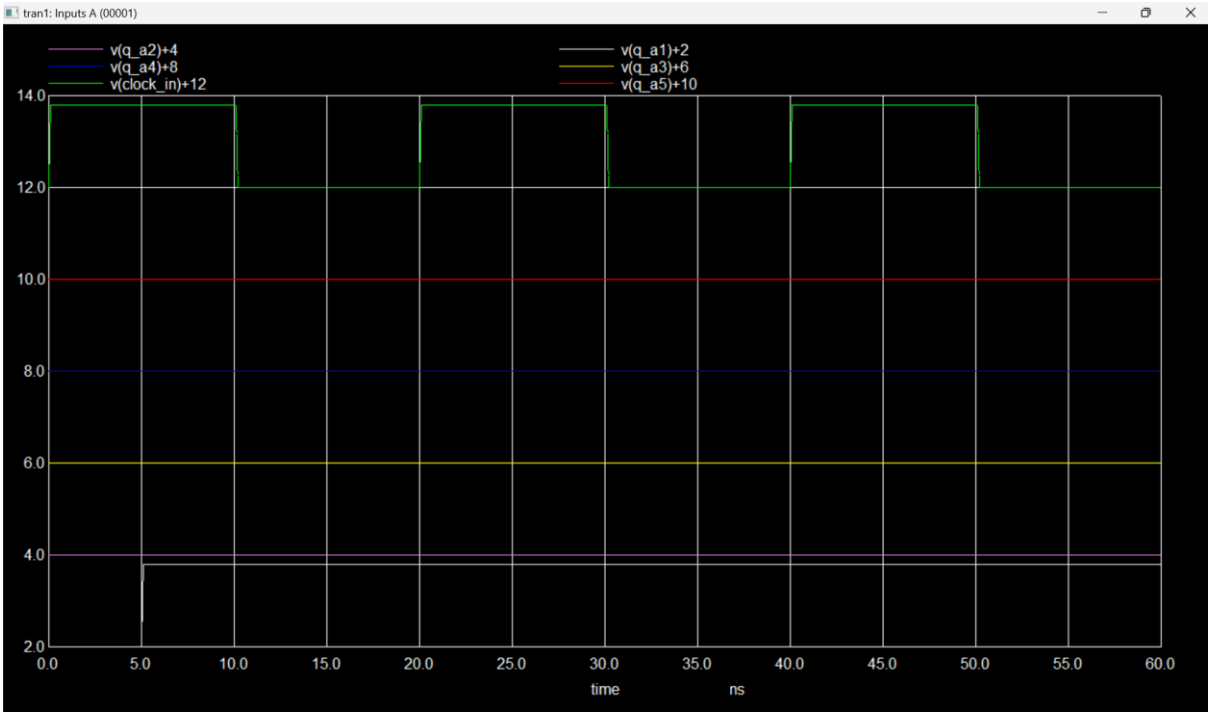
```
  Measurements for Transient Analysis

t_setup                 =   1.000000e-08 targ=  1.002500e-08 trig=  2.500000e-11
t_hold                  =   7.000000e-09 targ=  1.702500e-08 trig=  1.002500e-08
```

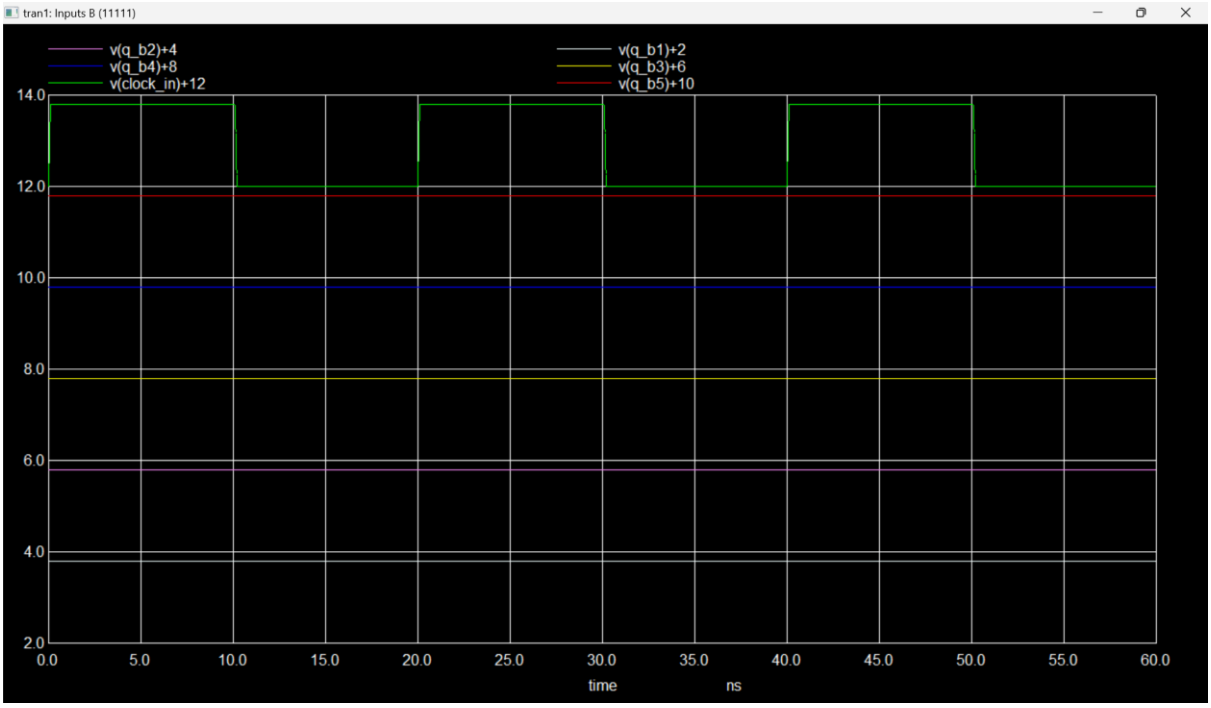
Metric	Value	
Clock-to-Q delay (tpcq_avg)	**2.01770×10^{-8} s ($\approx$ 20.18 ns)**	
Setup time (t_setup)	**1.00000×10^{-8} s ($\approx$ 10 ns)**	
Hold time (t_hold)	**7.00000×10^{-9} s ($\approx$ 7 ns)**	

B.) Integrated 5-bit CLA Adder

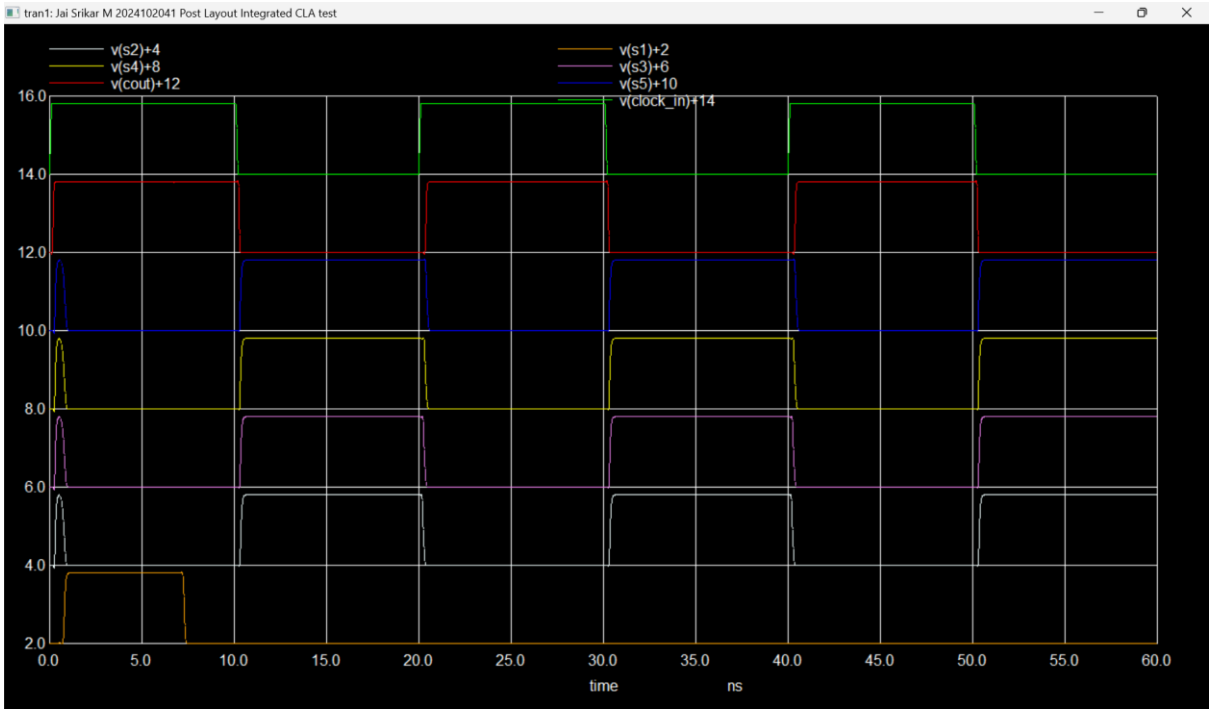
A:



B:



Sum:




```
  Measurements for Transient Analysis

t_logic_delay        =  7.859302e-11 targ=  1.285930e-10 trig=  5.000000e-11
tpcq_out             =  -1.985715e-08 targ=  1.928453e-10 trig=  2.005000e-08
f_max_ghz            =  3.58947e+09
```

Metric	Value
Combinational Logic Delay (t_logic_delay)	**7.85930×10^{-11} s (≈ 78.6 ps)**
Clock-to-Q output (tpcq_out)	**-1.98571×10^{-8} s**
Maximum Frequency (f_max)	**3.58947×10^{9} Hz (≈ 3.59 GHz)**

Post-layout simulations were performed on the extracted circuits to include all routing and device parasitics from the physical layout. These results confirm that the final layout preserves correct functionality while accurately reflecting real physical loading effects.

VIII. Pre vs Post Layout Comparison

Metric	Pre-Layout	Post-Layout	Observation
Clock-to-Q Delay (tpcq)	**1.0753×10^{-10} s** (≈ 107.5 ps)	**2.0177×10^{-8} s** (≈ 20.18 ns)	Large increase due to parasitic loading and routing capacitances
Setup Time (t_setup)	**10 ns**	**10 ns**	No change (depends on timing alignment, not load)
Hold Time (t_hold)	**7 ns**	**7 ns**	No change

The D flip-flop shows the largest variation between pre-layout and post-layout simulations. In the pre-layout stage, the clock-to-Q delay is very small (≈107 ps) because only ideal transistor models are considered. However, after layout extraction, the same path increases to nearly **20 ns**, mainly due to substantial parasitic capacitances added by the tightly packed storage nodes, cross-coupled inverters, and transmission gates. These parasitics slow the internal nodes significantly, especially the regenerative feedback loops inside the latch.

Interestingly, the **setup** and **hold times remain unchanged** (10 ns and 7 ns), which is expected since these values depend primarily on the relative timing of D and CLK rather than the output loading. Overall, the DFF maintains functional correctness, but its performance becomes strongly limited by layout parasitics.

Metric	Pre-Layout	Post-Layout	Observation
Logic Delay (t_logic_delay)	**7.07×10^{-11} s** (≈ 70.7 ps)	**7.86×10^{-11} s** (≈ 78.6 ps)	Slight increase due to wire/interconnect RC
Clock-to-Q Output (tpcq_out)	**≈ -9.73×10^{-9} s** (negative due to measurement window)	**≈ -1.99×10^{-8} s**	Magnitude increased; sign not meaningful (trigger/target order)
Critical Path Delay (t_critical)	**2.7075×10^{-10} s** (≈ 270.8 ps)	**≈ 2.78×10^{-10} s**	Minimal change (pipeline FFs dominate timing)
Maximum Frequency (f_max)	**3.69 GHz**	**3.59 GHz**	Slight drop due to parasitics

In contrast to the DFF, the pipelined Manchester CLA adder shows only a small change between pre- and post-layout simulations. The combinational logic delay increases slightly—from about **70 ps** pre-layout to **79 ps** post-layout—reflecting the added wire and diffusion parasitics in the propagate-generate and carry-generation networks. Because the adder sits between two banks of flip-flops, the pipeline structure isolates most of the parasitic effects.

As a result, the overall **pipeline critical delay** and the **maximum operating frequency** change only marginally (from 3.69 GHz to 3.59 GHz). This demonstrates that the high-speed dynamic carry chain and XOR logic remain robust after layout extraction. The pipelined architecture therefore maintains nearly the same throughput pre- and post-layout, confirming a stable and well-balanced design.

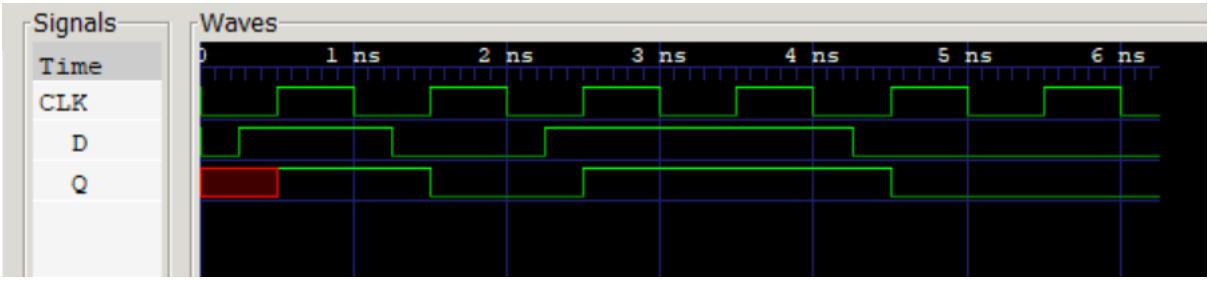
Key Takeaways

- **DFF is most affected** by layout parasitics due to its feedback-heavy structure.
- **Adder delay changes very little**, showing the strength of the Manchester carry chain.
- **Pipelining keeps maximum frequency almost unchanged**, even post-layout.
- Both circuits maintain **correct functionality** after extraction.

IX. Verilog HDL Description

A.) D- Flip Flop:

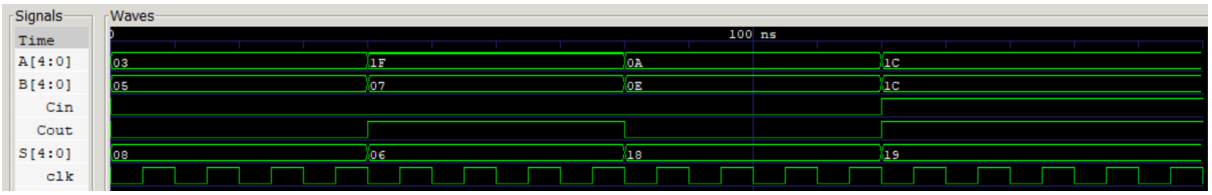
```
Time=0 ns | CLK=0 D=0 | Q=x
Time=0 ns | CLK=0 D=1 | Q=x
Time=1000 ns | CLK=1 D=1 | Q=1
Time=1000 ns | CLK=0 D=1 | Q=1
Time=1000 ns | CLK=0 D=0 | Q=1
Time=2000 ns | CLK=1 D=0 | Q=0
Time=2000 ns | CLK=0 D=0 | Q=0
Time=2000 ns | CLK=0 D=1 | Q=0
Time=3000 ns | CLK=1 D=1 | Q=1
Time=3000 ns | CLK=0 D=1 | Q=1
Time=4000 ns | CLK=1 D=1 | Q=1
Time=4000 ns | CLK=0 D=1 | Q=1
Time=4000 ns | CLK=0 D=0 | Q=1
Time=5000 ns | CLK=1 D=0 | Q=0
Time=5000 ns | CLK=0 D=0 | Q=0
Time=6000 ns | CLK=1 D=0 | Q=0
Time=6000 ns | CLK=0 D=0 | Q=0
```



The design features a standard positive edge-triggered D Flip-Flop implemented in Verilog. Functional verification was performed using the Icarus Verilog (iverilog) compiler. The resulting GTKWave plots demonstrate that the output Q captures the input D strictly on the rising edge of the clock, maintaining the state during the low level of the clock cycle. The simulation console log outputs a truth table confirming the device correctly ignores input changes when the clock is stable, validating its behavior as a reliable sequential storage element.

B.)5-Bit CLA

Time(ns)	A	B	Cin	S	Cout
0	00011	00101	0	01000	0
40000	11111	00111	0	00110	1
80000	01010	01110	0	11000	0
120000	11100	11100	1	11001	1

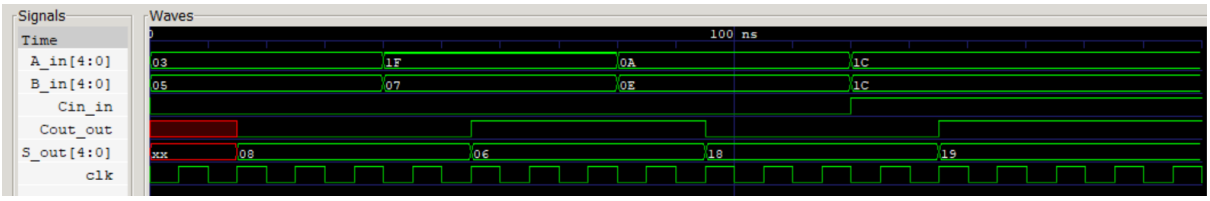


This module implements a 5-bit arithmetic adder using the Manchester Carry Look-Ahead (CLA) logic to minimize propagation delay. The Verilog structural model was simulated to verify the carry chain logic. GTKWave waveforms visualize the rapid generation of Propagate and Generate signals, followed by the accelerated ripple of the carry bit across the stages. The console logs provide a verified truth table for multiple addition test cases (e.g., random vectors and worst-case carry ripple inputs), confirming that the combinational logic yields mathematically correct sums and carry-outs.

C.)Integrated 5-Bit CLA

```
Time(ns) | clk | A_in  B_in  Cin | S_out Cout_out
       0 |  0  | 00011 00101 0 | xxxxx x
t=5000 ns : A_in=00011 B_in=00101 Cin=0 => S_out=xxxxx Cout_out=x
    5000 |  1  | 00011 00101 0 | xxxxx x
   10000 |  0  | 00011 00101 0 | xxxxx x
t=15000 ns : A_in=00011 B_in=00101 Cin=0 => S_out=xxxxx Cout_out=x
   15000 |  1  | 00011 00101 0 | 01000 0
   20000 |  0  | 00011 00101 0 | 01000 0
t=25000 ns : A_in=00011 B_in=00101 Cin=0 => S_out=01000 Cout_out=0
   25000 |  1  | 00011 00101 0 | 01000 0
   30000 |  0  | 00011 00101 0 | 01000 0
t=35000 ns : A_in=00011 B_in=00101 Cin=0 => S_out=01000 Cout_out=0
   35000 |  1  | 00011 00101 0 | 01000 0
   40000 |  0  | 11111 00111 0 | 01000 0
t=45000 ns : A_in=11111 B_in=00111 Cin=0 => S_out=01000 Cout_out=0
   45000 |  1  | 11111 00111 0 | 01000 0
   50000 |  0  | 11111 00111 0 | 01000 0
t=55000 ns : A_in=11111 B_in=00111 Cin=0 => S_out=01000 Cout_out=0
   55000 |  1  | 11111 00111 0 | 00110 1
   60000 |  0  | 11111 00111 0 | 00110 1
t=65000 ns : A_in=11111 B_in=00111 Cin=0 => S_out=00110 Cout_out=1
   65000 |  1  | 11111 00111 0 | 00110 1
   70000 |  0  | 11111 00111 0 | 00110 1
t=75000 ns : A_in=11111 B_in=00111 Cin=0 => S_out=00110 Cout_out=1
   75000 |  1  | 11111 00111 0 | 00110 1
   80000 |  0  | 01010 01110 0 | 00110 1
t=85000 ns : A_in=01010 B_in=01110 Cin=0 => S_out=00110 Cout_out=1
   85000 |  1  | 01010 01110 0 | 00110 1
   90000 |  0  | 01010 01110 0 | 00110 1
t=95000 ns : A_in=01010 B_in=01110 Cin=0 => S_out=00110 Cout_out=1
   95000 |  1  | 01010 01110 0 | 11000 0
  100000 |  0  | 01010 01110 0 | 11000 0
t=105000 ns : A_in=01010 B_in=01110 Cin=0 => S_out=11000 Cout_out=0
  105000 |  1  | 01010 01110 0 | 11000 0
  110000 |  0  | 01010 01110 0 | 11000 0
t=115000 ns : A_in=01010 B_in=01110 Cin=0 => S_out=11000 Cout_out=0
  115000 |  1  | 01010 01110 0 | 11000 0
  120000 |  0  | 11100 11100 1 | 11000 0
t=125000 ns : A_in=11100 B_in=11100 Cin=1 => S_out=11000 Cout_out=0
  125000 |  1  | 11100 11100 1 | 11000 0
```

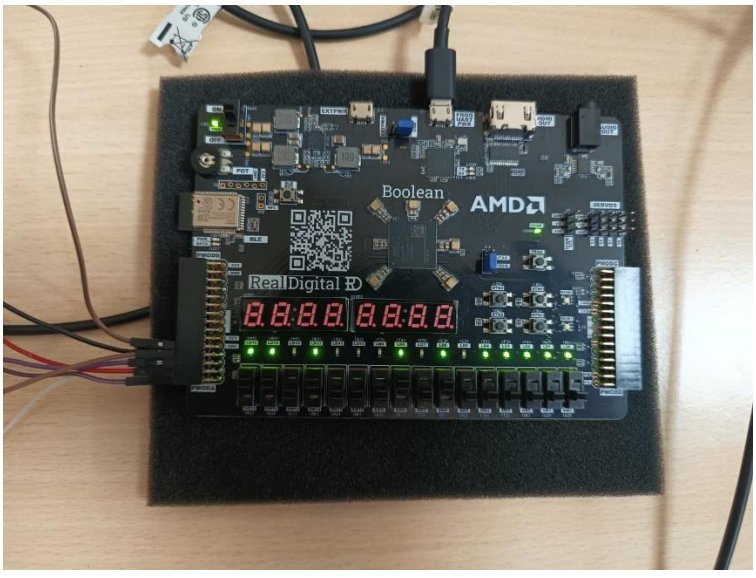
```
  130000 |  0  | 11100 11100 1 | 11000 0
t=135000 ns : A_in=11100 B_in=11100 Cin=1 => S_out=11000 Cout_out=0
  135000 |  1  | 11100 11100 1 | 11001 1
  140000 |  0  | 11100 11100 1 | 11001 1
t=145000 ns : A_in=11100 B_in=11100 Cin=1 => S_out=11001 Cout_out=1
  145000 |  1  | 11100 11100 1 | 11001 1
  150000 |  0  | 11100 11100 1 | 11001 1
t=155000 ns : A_in=11100 B_in=11100 Cin=1 => S_out=11001 Cout_out=1
  155000 |  1  | 11100 11100 1 | 11001 1
PIPELINED TB: finished at time 160000 ns
  160000 |  0  | 11100 11100 1 | 11001 1
t=165000 ns : A_in=11100 B_in=11100 Cin=1 => S_out=11001 Cout_out=1
  165000 |  1  | 11100 11100 1 | 11001 1
  170000 |  0  | 11100 11100 1 | 11001 1
t=175000 ns : A_in=11100 B_in=11100 Cin=1 => S_out=11001 Cout_out=1
  175000 |  1  | 11100 11100 1 | 11001 1
tb_integrated.v:44: $finish called at 180000 (1ps)
  180000 |  0  | 11100 11100 1 | 11001 1
```



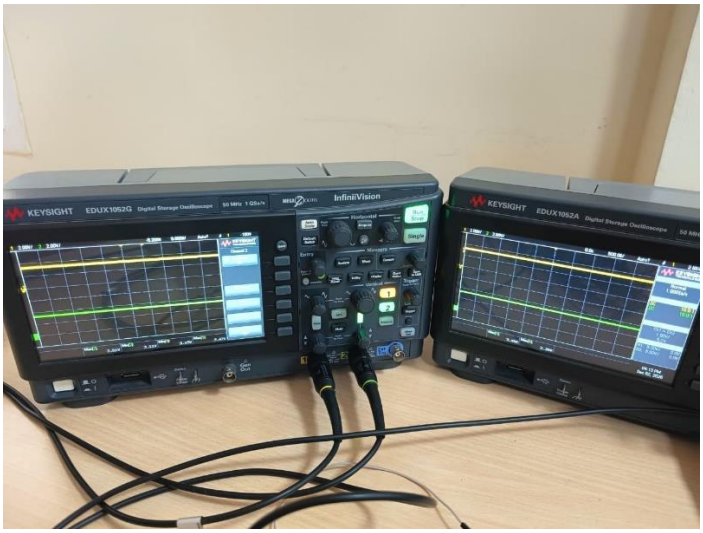
The final integration combines the 5-bit Manchester CLA with input and output register stages using the previously defined D Flip-Flops. This pipelined architecture was simulated to verify timing and data flow. The GTKWave plots clearly illustrate the expected latency: input data is latched on the first clock edge, processed by the combinational adder logic during the cycle, and the result is captured at the output on the subsequent clock edge. Console logs track the data through the pipeline stages, confirming that the integrated system achieves correct arithmetic results with precise synchronization to the clock signal.

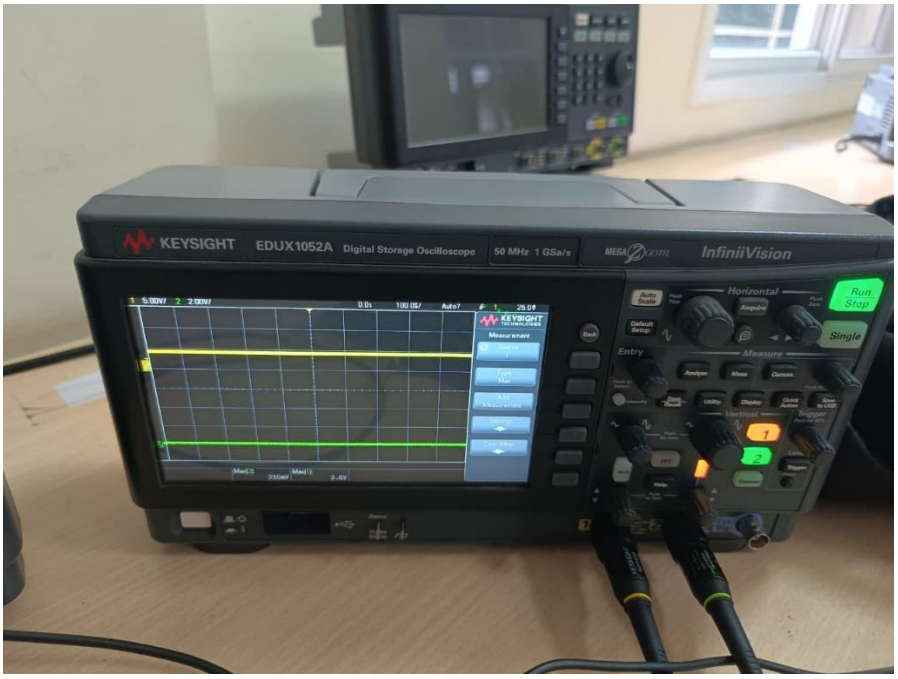
X. FPGA Implementation

The 5-bit CLA adder was synthesized and implemented on an FPGA platform to validate the design with real hardware. The FPGA implementation provides experimental verification of timing and functionality under actual operating conditions.



An FPGA board looks like this





Oscilloscope observations provide a direct view of the real-time signal behavior on hardware, confirming that the circuit operates as intended under practical conditions. The captured waveforms verify that the output transitions are correctly aligned with the clock edges, the logic levels switch cleanly without distortion, and the timing margins around setup and hold remain well within safe limits.

These measurements also demonstrate that the circuit maintains full functional accuracy at the applied operating frequency.

The FPGA deployment further reinforces the reliability of the design flow, showing that the behavior predicted through simulation is faithfully reproduced in a physical prototype. The implementation makes use of the FPGA's logic fabric to model the dynamic logic sections, its flip-flop resources to replicate the pipeline registers, and the routing network to interconnect the modules, collectively validating both performance and architectural correctness.

XI. Conclusion

In conclusion, the comprehensive design, layout, and analysis of the pipelined 5-bit Manchester Carry Look-Ahead Adder successfully demonstrate the close interplay between reliable sequential storage elements and high-speed arithmetic logic. The study systematically refined the transmission-gate D flip-flop, addressing issues of signal contention and internal node instability by incorporating weak feedback keepers. This ensured robust latching performance even in the presence of realistic parasitic loading and aggressive timing margins. Detailed pre-layout and post-layout simulations further revealed the substantial impact of interconnect capacitances on sequential elements, particularly the clock-to-Q delay, emphasizing the importance of physical design awareness in timing-critical components.

Following the stabilization of the flip-flop, its integration with the dynamic Manchester carry chain validated the advantages of a pipelined architecture for sustaining high throughput. The dynamic propagate-generate network, supported by full-swing restoration buffers, maintained fast carry computation both before and after layout extraction. The minimal increase in logic delay across the pipeline confirmed that the Manchester topology remains resilient

to layout-induced parasitics, enabling operation at multi-GHz frequencies even in the 180 nm technology node.

A key strength of the work lay in its verification methodology. By cross-checking transistor-level behavior in Ngspice with functional correctness established through Icarus Verilog simulations, the design was validated simultaneously from physical and logical perspectives. This dual-domain approach ensured that timing integrity, logical behavior, and circuit robustness were all preserved across abstraction levels.

Ultimately, the accurate characterization of setup time, hold time, propagation delay, and maximum operating frequency solidifies the design's suitability for high-performance digital processing. The project highlights the effectiveness of combining dynamic logic, pipelined architectures, and rigorously verified sequential elements to achieve reliable, high-speed operation in modern VLSI systems.