ELECTRIC-ELECTRONICS ENGINEERING DEPARTMENT
EEM/EEE 410 INTEGRATED CIRCUIT DESIGN
Spring 2023-2024

ASSIGNMENT 2

An 8-bit SYNCHRONOUS COUNTER WITH PARALLEL LOAD AND ASYNCHRONOUS RESET

Samet Bayat
22293730

Table of Contents

ABSTRACT

This report presents the design of an 8-bit synchronous counter with parallel load and asynchronous reset capabilities, implemented using Electronic Design Automation (EDA) tools. Synchronous counters are fundamental building blocks in digital systems, and this design incorporates features that enhance its functionality and versatility.

Keywords: Counter, synchronous, VLSI, Electric-VLSI-is-tough :/

1. INTRODUCTION

This report details the design of an 8-bit synchronous counter with parallel load and asynchronous reset capabilities. This counter finds application in various digital systems for tasks such as:

- **Generating timing signals:** Synchronous counters are crucial for generating precise timing signals within a system. They can be used to create clock pulses, delays, and control intervals for other digital circuits.
- **Implementing sequence control:** In many digital systems, specific sequences of operations need to be executed. Synchronous counters can be employed to define the order and timing of these operations, ensuring proper functionality.
- **Addressing memory locations:** Microprocessors and other digital processors utilize counters to address memory locations during data read and write operations. An 8-bit counter can effectively address up to 256 (2^8) memory locations.
- **Frequency dividers:** By manipulating the counting direction and enabling signals, synchronous counters can be configured to divide an input clock signal by a specific factor. This is useful for generating lower frequency signals from a higher frequency source.

The counter operates synchronously, meaning all flip-flops update their outputs on a common clock signal. This eliminates the propagation delays associated with asynchronous (ripple) counters, leading to more predictable and reliable operation at high frequencies.

Furthermore, the inclusion of parallel load functionality allows the counter to be directly loaded with a desired initial value, enhancing its versatility. Additionally, the asynchronous reset provides a mechanism to immediately force the counter's output to zero, regardless of the clock signal.

This report will delve into the design considerations, circuit schematics, logic implementation, and simulation results for the 8-bit synchronous counter with parallel load and asynchronous reset.

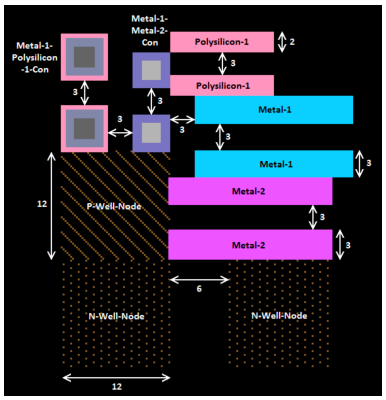


Figure 1. Fundamental rule of designing an IC layout (All values arein lambda) [1]

* Grammar check has been done by AI.

2. 8-bit COUNTER DESIGN & SIMULATIONS

This part explains what an 8-bit Synchronous Counter with Parallel Load and Asynchronous Reset includes & their simulations.

P1.i.

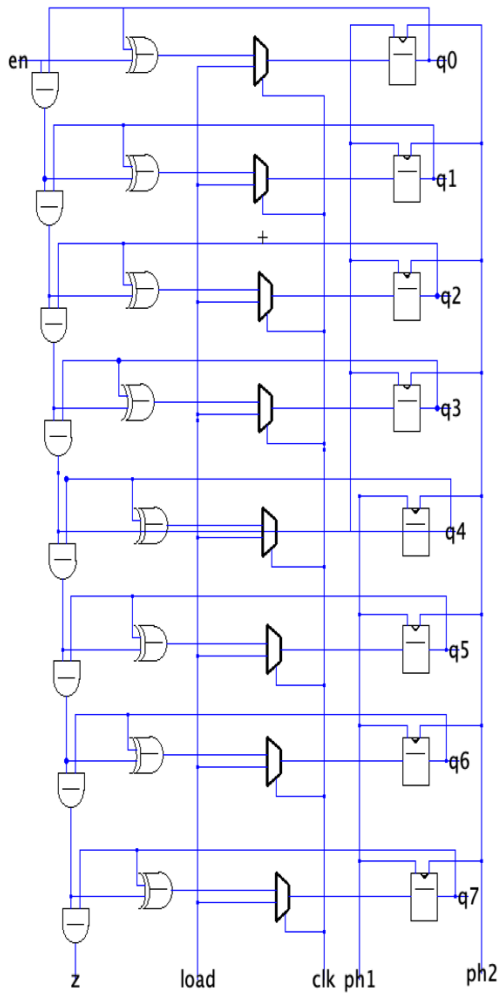


Fig 1. 8-bit Counter Schematic

This counter circuit can count up to 255 (8 bits) and offers two functionalities:

1. **Synchronous Counting:** Increments the count on every positive clock edge (CLK) when enabled.
2. **Parallel Load:** Allows setting the counter output to a specific value by applying data on parallel lines.
3. **Asynchronous Reset:** Forces the counter output to zero regardless of the clock or enable signals.

Here's a breakdown of the schematic:

- **en (Enable):** This is a control signal. When LOW, the counter increments on every positive clock edge. When HIGH, the counter holds its current value.
- **clk (Clock):** This provides the timing signal for counting. The counter increments its value on the positive edge (rising edge) of the clock pulse.
- **load (Load):** This is a control signal for parallel loading. When LOW, the data on the parallel lines (ph1-ph8) is loaded onto the counter outputs (q0-q7) on the next positive clock edge. When HIGH, the parallel load function is disabled.
- **ph1-ph2:** Allows the counter to show any specific value. Also it can be used for reset.
- **q0-q7 (Outputs):** These are the 8 output bits of the counter, representing the current count value in binary format.

Internal Circuit (likely using D Flip-Flops): The counter likely uses eight D flip-flops to store the count value. Each flip-flop captures the data present at its D input on the positive clock edge, provided the enable signal (en) is LOW.

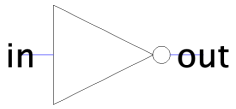
Parallel Load Operation:
When the load signal (load) is LOW, the data on the parallel lines (ph1-ph2) is directly loaded onto the D inputs of the flip-flops on the next positive clock edge. This allows setting the counter to any specific value between 0 and 255.

Synchronous Counting Operation:
When the load signal (load) is HIGH and the enable signal (en) is LOW, the counter increments its value on every positive clock edge. The logic for incrementing likely involves a ripple counter or a carry look-ahead adder circuit using the outputs of the flip-flops (q0-q7).

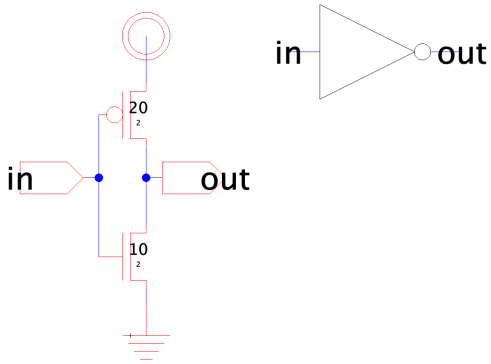
Asynchronous Reset Operation:
When the asynchronous reset signal (z) is LOW, it overrides the clock and enable signals and forces the outputs of all flip-flops (q0-q7) to LOW on the next clock edge, effectively resetting the counter to zero.

INVERTER

ICON



SCHEMATIC



SIMULATION

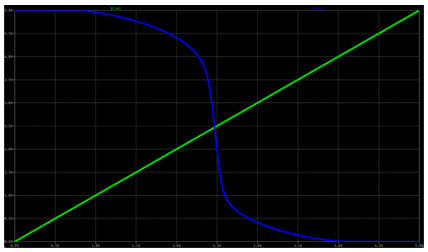
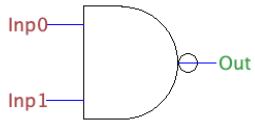


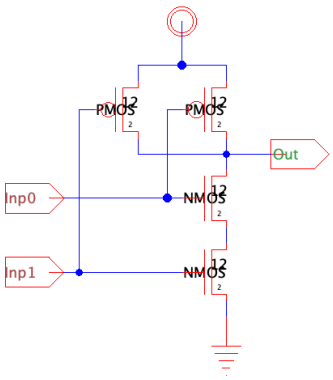
Fig 2. Inverter DC Sweep Analysis

NAND (2 INPUT)

ICON



SCHEMATIC



SIMULATION

```
vdd vdd 0 DC 5
va A 0 DC pwl 10n 0 20n 5 50n 5 60n 0 90n 0 100n 5 130n 5 140n 0 170n 0 180n 5
vb B 0 DC pwl 10n 0 20n 5 100n 5 110n 0
.measure tran tf trig v(AB) val=4.5 fall=1 td=4ns targ v(AB) val=0.5 fall=1
.measure tran tf trig v(AB) val=0.5 rise=1 td=4ns targ v(AB) val=4.5 rise=1
.tran 200n
.include tsmc180.txt
```

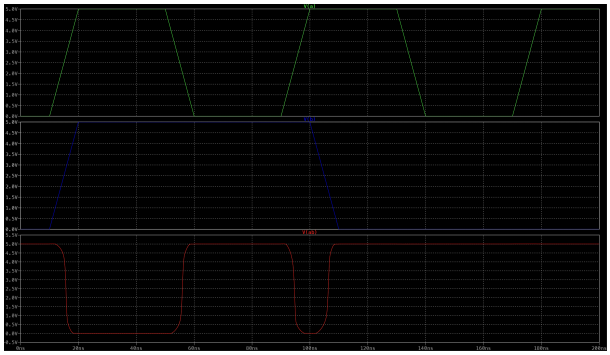
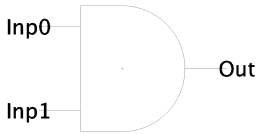


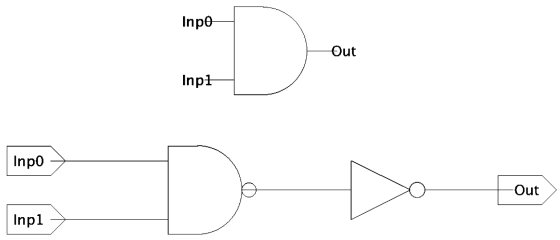
Fig 3. NAND Behavior for Various Input Conditions

AND

ICON



SCHEMATIC



SIMULATION

```
vdd Vdd 0 DC 5
vin Inp0 0 DC 5 pulse(0 5 10f 0.5f 0.5f 20n 40n)
vin2 Inp1 0 DC 5 pulse(0 5 10f 0.5f 0.5f 40n 80n)
cload Out 0 50fF
.tran 0 100ns
.include tsmc180.txt
```

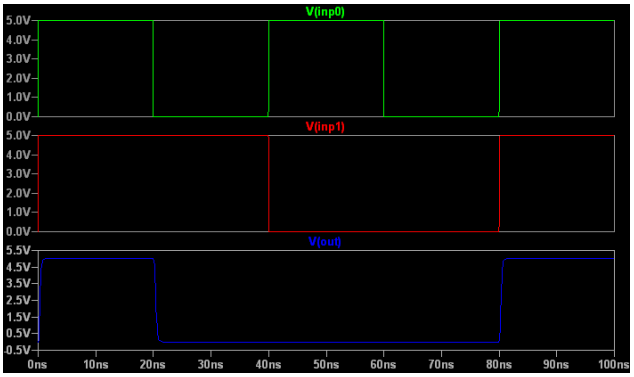


Fig 4. AND Gate LTSpice Graph Inp0, Inp1 vs. Output.

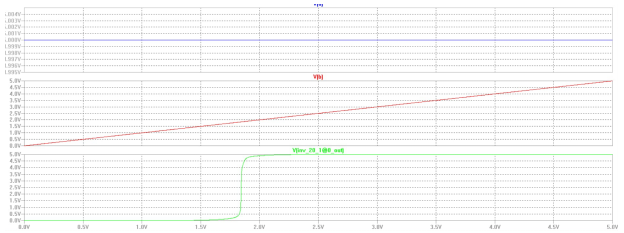


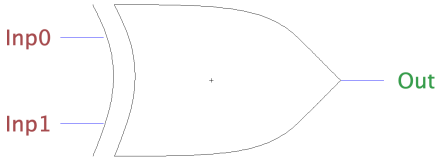
Fig 5. AND Gate DC Sweep Analysis

XOR

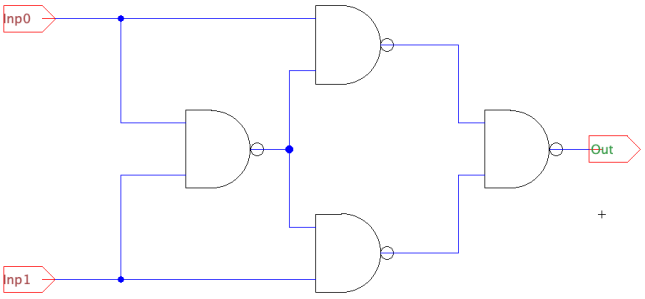
A	B	A **XOR** B
0	0	0
0	1	1
1	0	1
1	1	0

Table 1. XOR Logic Expressions

ICON



SCHEMATIC



SIMULATION

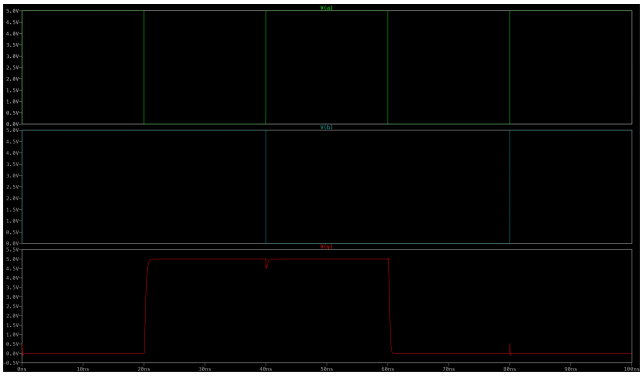
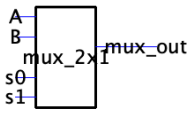


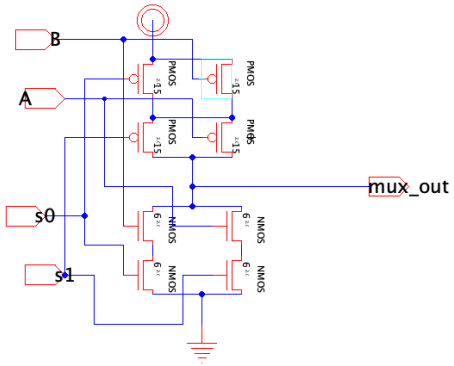
Fig 6. XOR LTSpice Analysis

MUX 2x1

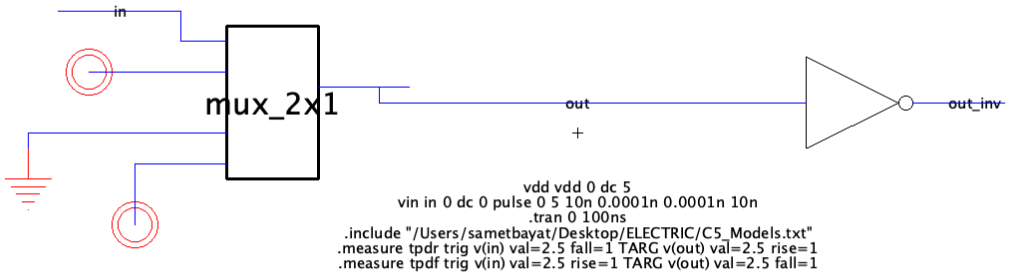
ICON



SCHEMATIC



SIMULATION



* For my individual experiment 180nm model and C5_Model simulated separately, and the results observed the same.

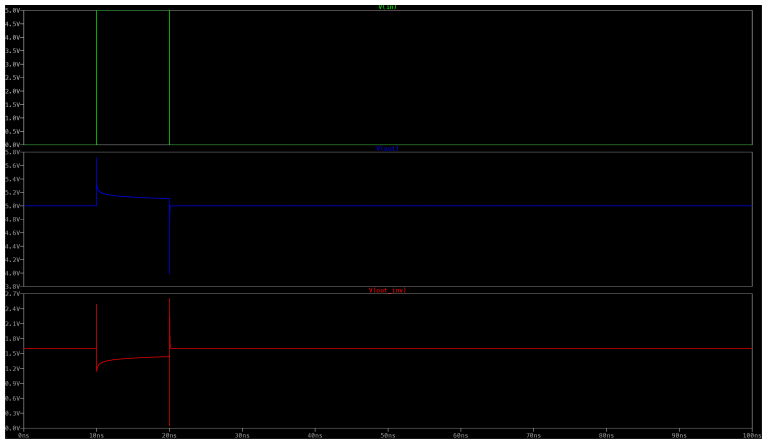


Figure 7. LTSpice MUX 2x1 Behavior

D FLIP-FLOP

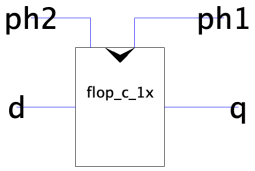
Breakdown of each element

- **D**: This represents the Data input. The value present at this pin determines what the flip-flop will output.
- **PH1 & PH2**: These are control signals often referred to as "Preset" and "Clear" respectively. They are active high signals, meaning a logic 1 (high voltage) on either input forces the flip-flop's output (Q) to a specific state.
- **CLK**: This is the Clock signal. The flip-flop captures the value at the Data (D) input only on the rising or falling edge (depending on the flip-flop type) of the clock pulse.
- **Q**: This represents the Output of the flip-flop. It reflects the data captured based on the D input and the clock signal, while also being affected by the Preset (PH1) and Clear (PH2) signals.

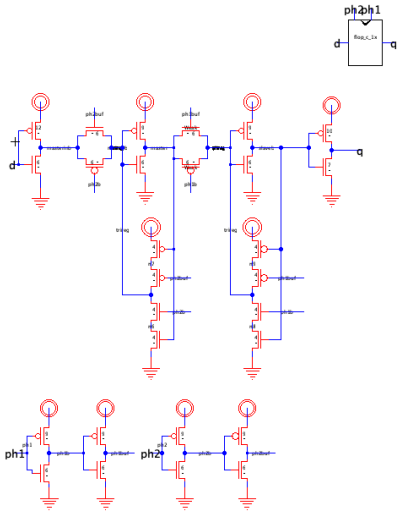
How the D flip-flop with these control signals typically functions:

1. **Preset (PH1) & Clear (PH2) Priority**: If either PH1 or PH2 is high (logic 1), the output (Q) is overridden regardless of the clock or data input. A high on PH1 usually sets the output to 1 (Q = 1), while a high on PH2 resets it to 0 (Q = 0). These Preset and Clear functions are generally used for initialization or emergency situations.
2. **Data Capture on Clock Edge**: When both PH1 and PH2 are low (logic 0), the flip-flop enters its normal operating mode. The value present at the Data (D) input is captured and transferred to the output (Q) on the rising or falling edge of the clock (CLK) pulse, depending on the specific flip-flop type.

ICON



SCHEMATIC



SIMULATION

```
* Spice Code nodes in cell cell 'flop_sim{sch}'
vdd vdd 0 DC 5
vin d 0 DC 5 pulse(0 5 10f 0.5f 0.5f 20n 40n)
vin2 ph2 0 DC 0
vin3 ph1 0 DC 0
cload q 0 50f
.tran 0 100ns
.include "/Users/sametbayat/Desktop/ELECTRIC/tsmc180.txt"
.end
.END
```

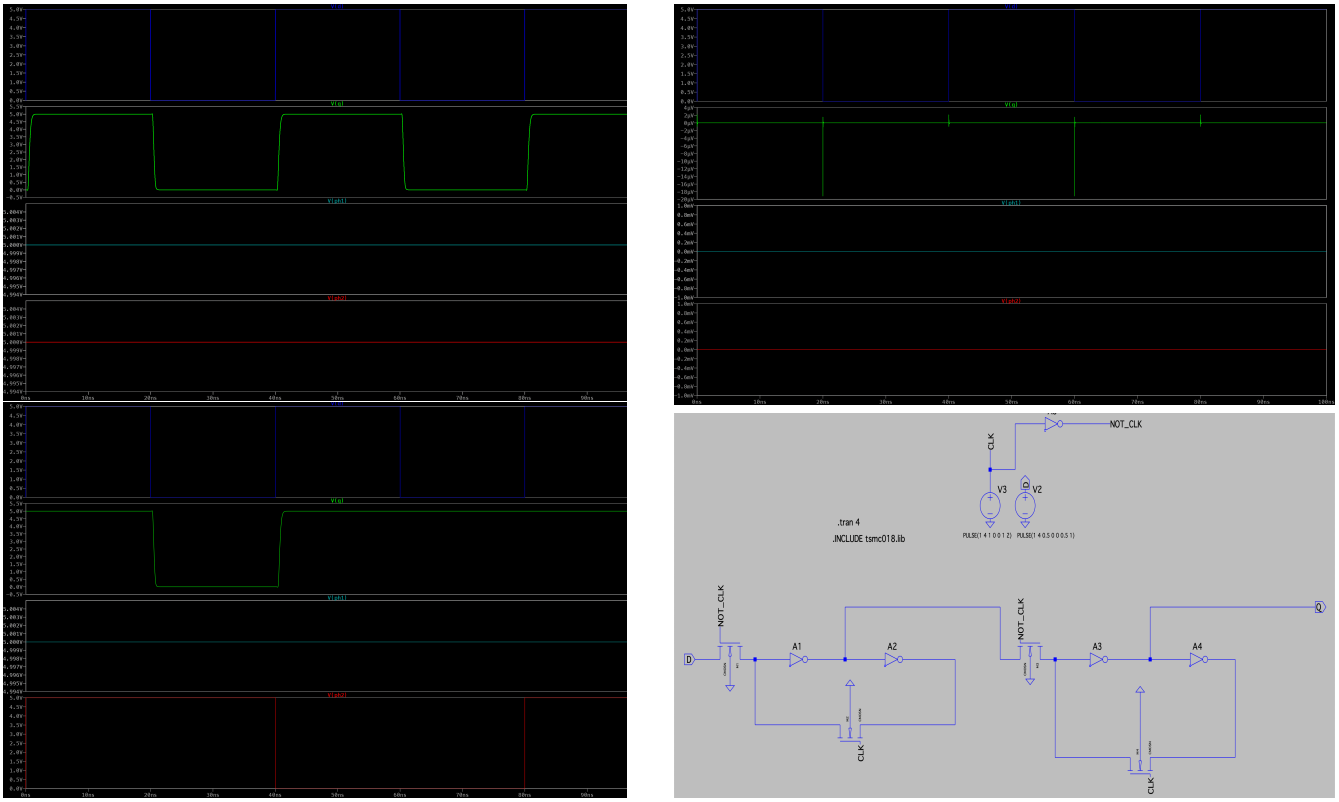
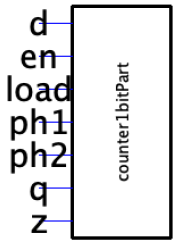


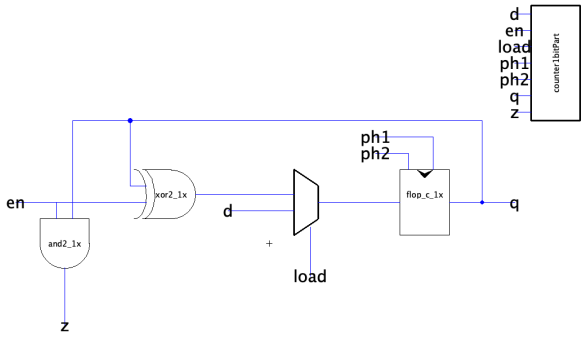
Fig 8a-b-c-d. D Flip-Flop response for various ph1-ph2 scenarios (dark-background graphs) & D Flip-Flop LTSpice Schematic

PLCNT1 (1-bit Unit Cell of Counter)

ICON

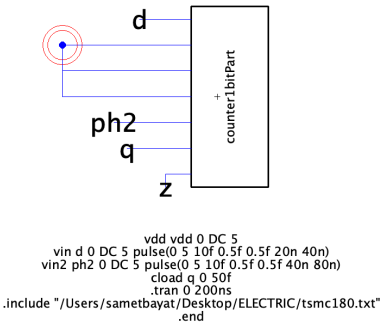


SCHEMATIC



P1.ii.

SIMULATION



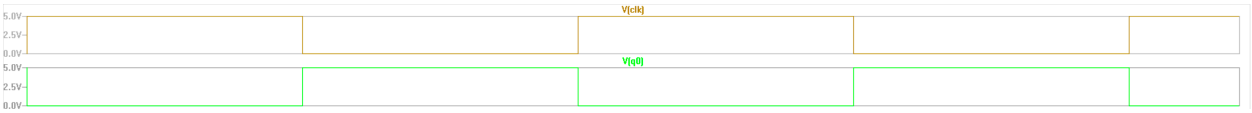


Fig 9. PLCNT1 LTSpice Simulation.

PLCNT4 (4-bit Cell of Counter)

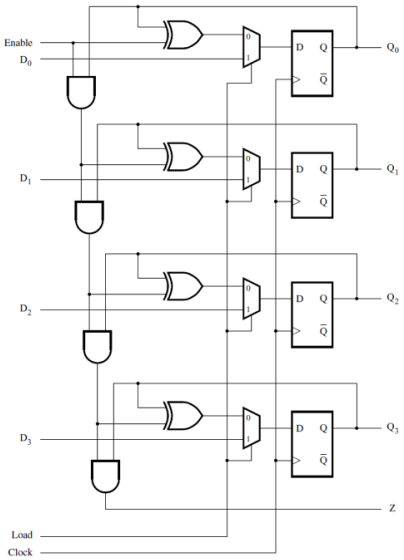
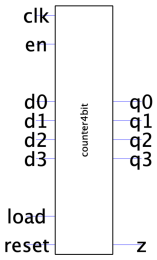
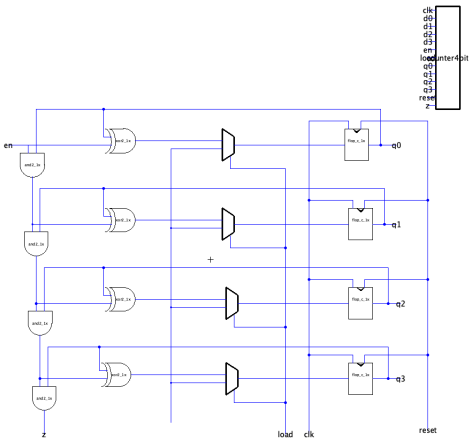


Fig 10. A counter with parallel-load capability. [2]

ICON



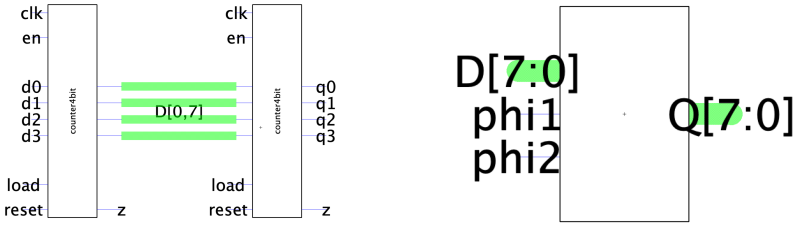
SCHEMATIC



PLCNT8

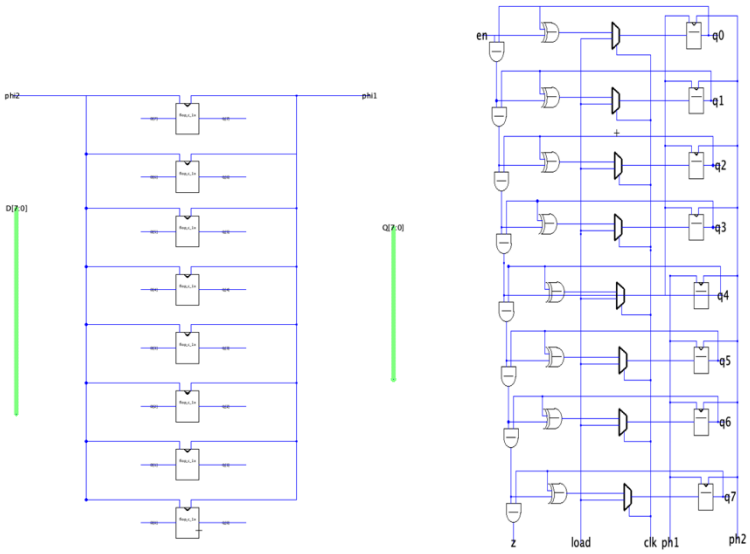
P1.iii-iv-v.

ICON



PREVIOUS RECENT

SCHEMATIC



APPROACH1 (ARRAY) APPROACH2 (PLCNT1x8)

SIMULATION

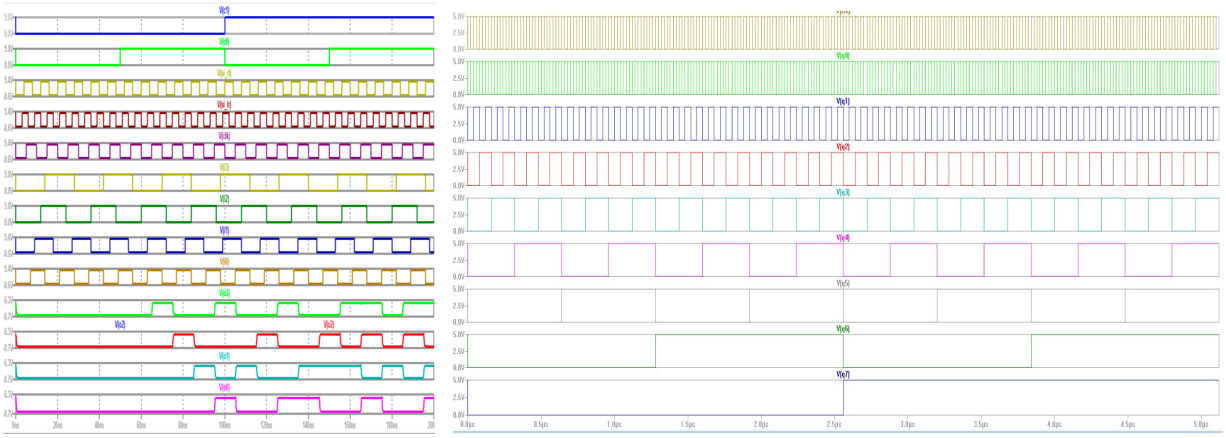
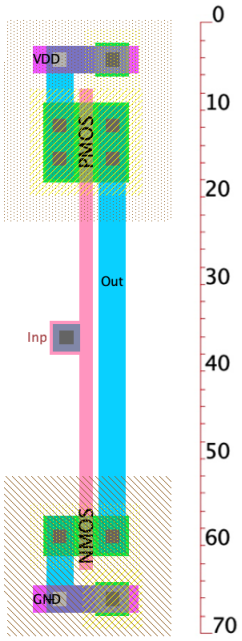


Fig 11. 8-bit Counter LTSpice Simulations wit.

3. 8-bit COUNTER LAYOUT

INVERTER

LAYOUT



SIMULATION

```
vdd Vdd 0 DC 5
vin Inp 0 DC 5 pulse(0 5 10f 0.5f 0.5f 20n 40n)
cload Out 0 50fF
.tran 0 100ns
.include tsmc180.txt
```

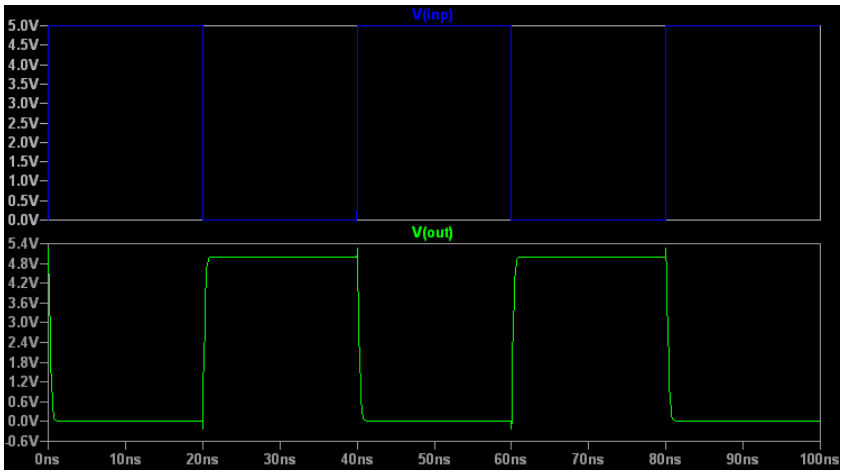
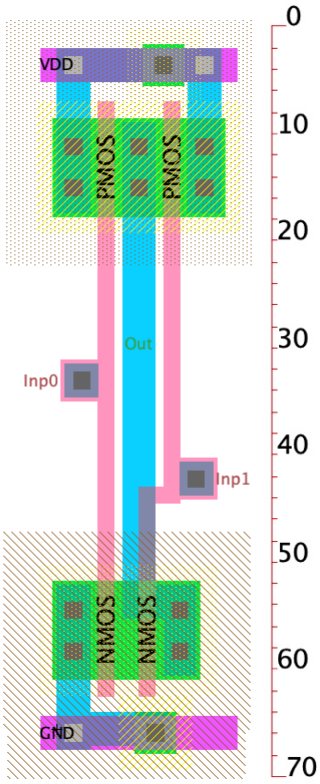


Fig 12. LTSpice Inverter Graph Input vs. Output

NAND

LAYOUT



SIMULATION

```
vdd Vdd 0 DC 5
vin Inp0 0 DC 5 pulse(0 5 10f 0.5f 0.5f 20n 40n)
vin2 Inp1 0 DC 5 pulse(0 5 10f 0.5f 0.5f 40n 80n)
cload Out 0 50fF
.tran 0 100ns
.include tsmc180.txt
```

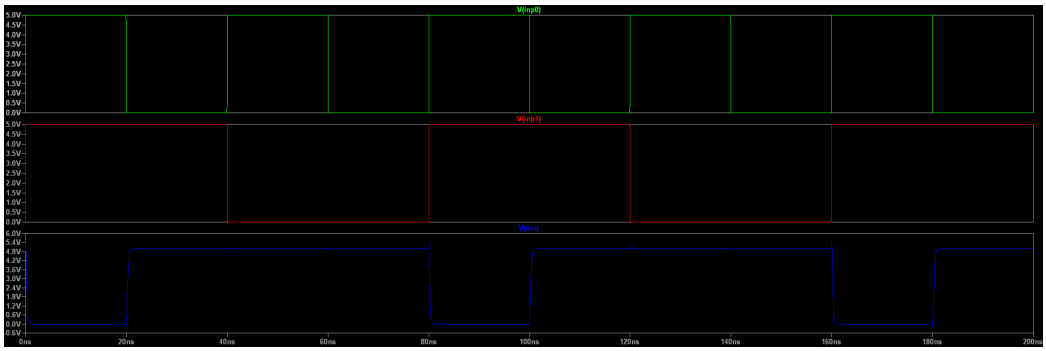
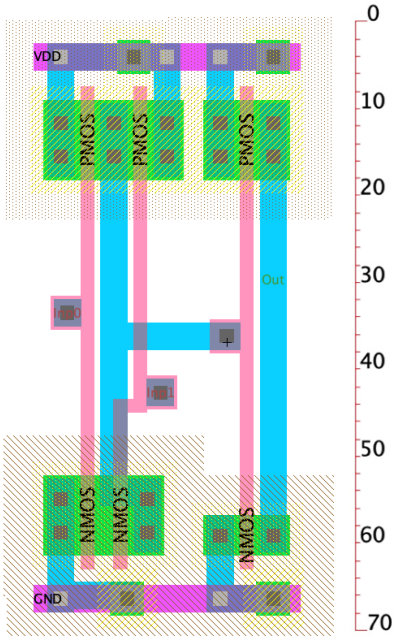


Fig 13. LTSpice NAND Graph for Inp0, Inp1 vs. Output

AND

LAYOUT



SIMULATION

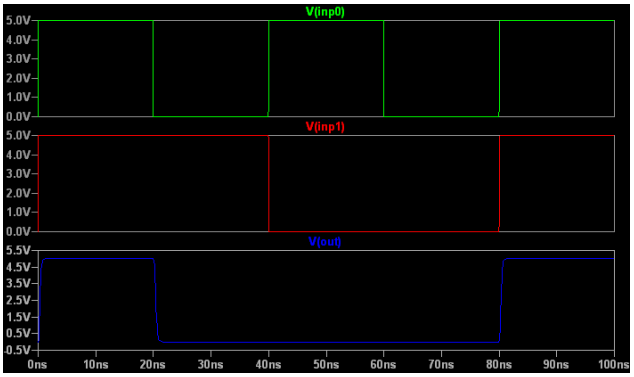
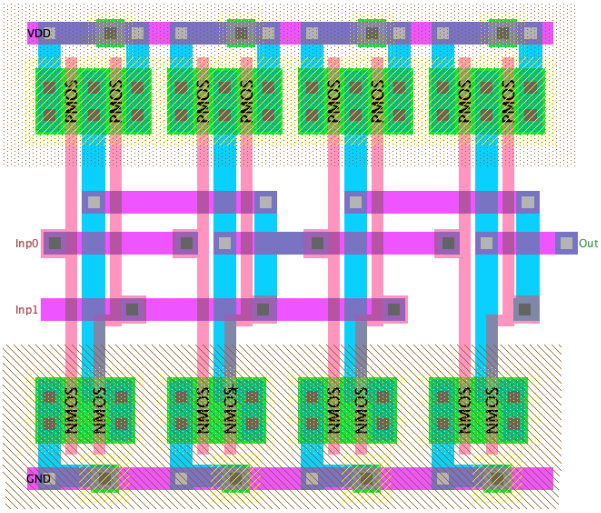


Fig 14. AND LTSpice Sim.

XOR

LAYOUT



SIMULATION

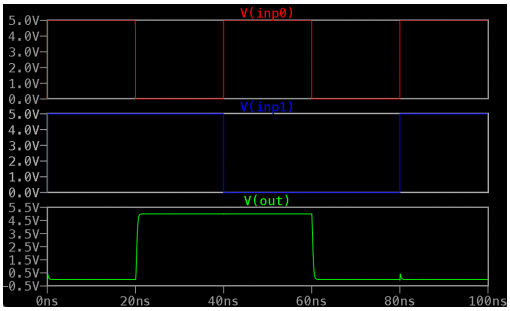
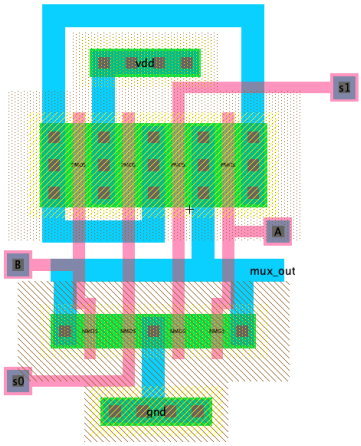


Fig 15. XOR Characteristic

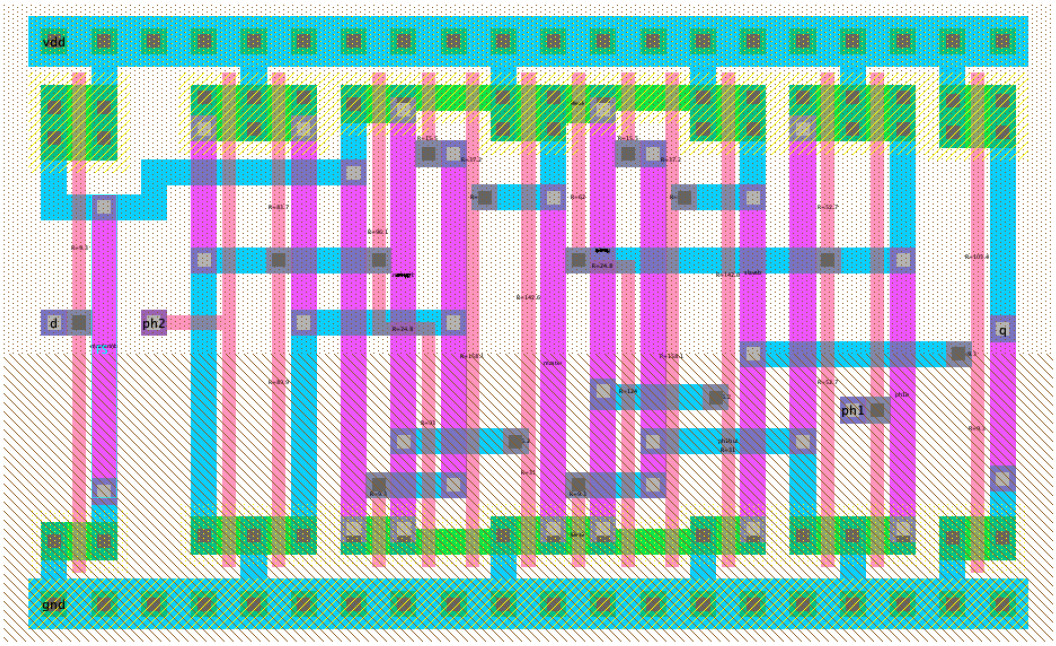
MUX 2x1

LAYOUT



D Flip-Flop

LAYOUT



SIMULATION

```
vdd vdd 0 DC 5
vin CLK 0 DC 5 pulse(0 5 10f 0.5f 0.5f 20n 40n)
vin2 D 0 DC 5 pulse(0 5 10f 0.5f 0.5f 40n 80n)
cload out 0 50f
.tran 0 100ns
.include "/Users/sametbayat/Desktop/ELECTRIC/tsmc180.txt"
.end
```

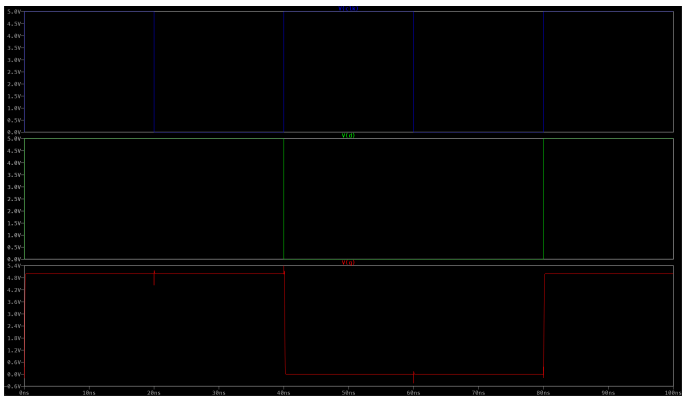


Fig 16. D Flip-flop characteristics.

8-bit Counter Layout

LAYOUT

P2.i-ii.

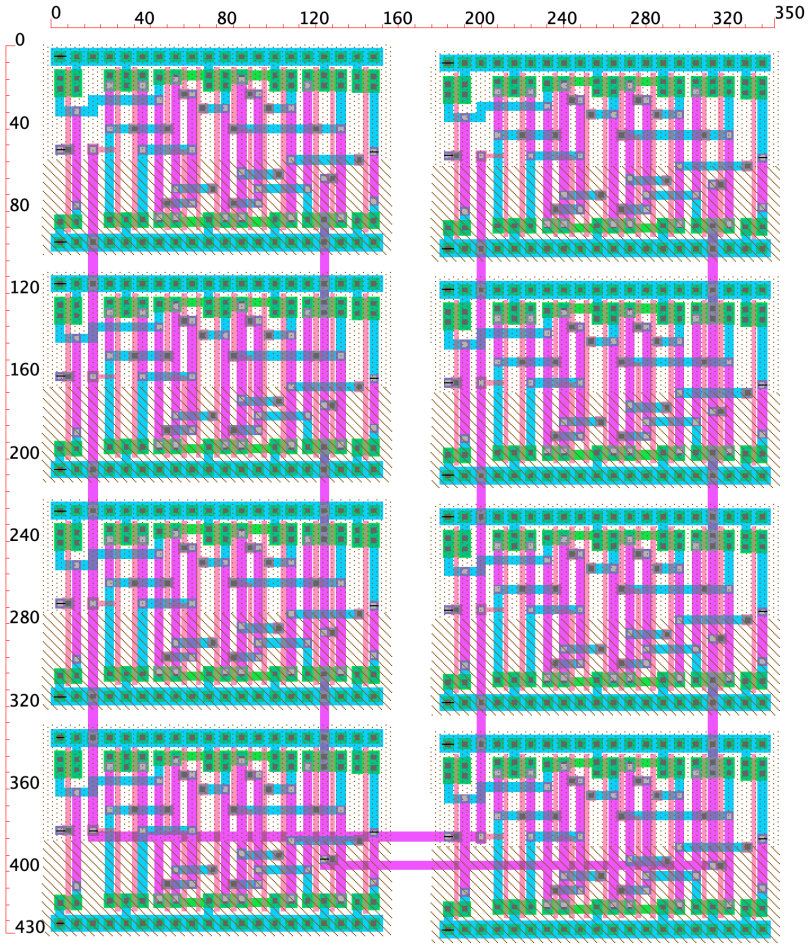


Fig 17. 8-bit Counter (2x4)

DRC, Well & NCC Check Log

P2.iii.

```
================================2================================
Running DRC with area bit on, extension bit on, Mosis bit
Checking again hierarchy .... (0.007 secs)
Found 35 networks
0 errors and 0 warnings found (took 0.031 secs)
================================3================================
Checking Wells and Substrates in 'VLSI-Project-r-apr20:PLCNT8{lay}' ...
   Geometry collection found 48 well pieces, took 0.023 secs
   Geometry analysis used 8 threads and took 0.014 secs
NetValues propagation took 0.001 secs
Checking short circuits in 344 well contacts
   Additional analysis took 0.005 secs
No Well errors found (took 0.045 secs)
================================4================================
Hierarchical NCC every cell in the design: cell 'PLCNT8{sch}'  cell 'PLCNT8{lay}'
Comparing: VLSI-Project-r-apr20:flop_c_1x{sch} with: VLSI-Project-r-apr20:flop_c_1x{lay}
  exports match, topologies match, sizes not checked in 0.03 seconds.
Comparing: VLSI-Project-r-apr20:PLCNT8{sch} with: VLSI-Project-r-apr20:PLCNT8{lay}
  exports match, topologies match, sizes not checked in 0.005 seconds.
Summary for all cells: exports match, topologies match, sizes not checked
NCC command completed in: 0.052 seconds.
```

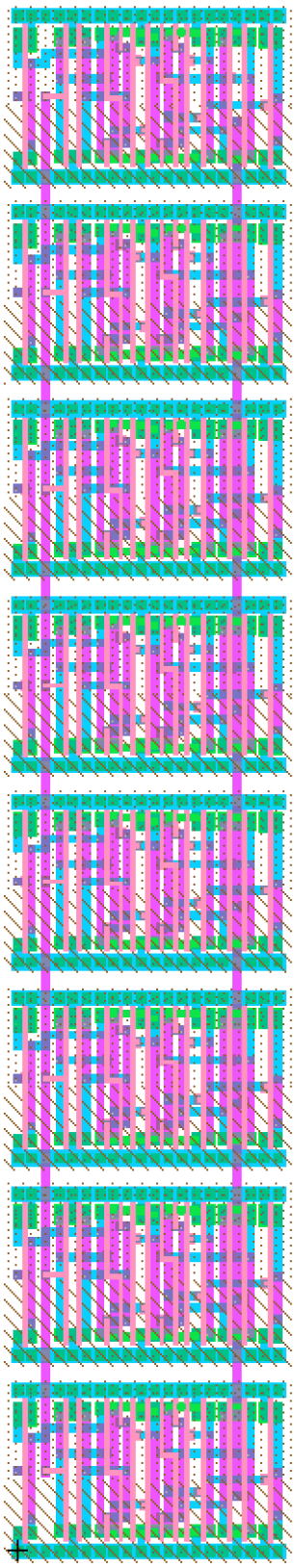


Fig 18. 8-bit Counter (1x8)

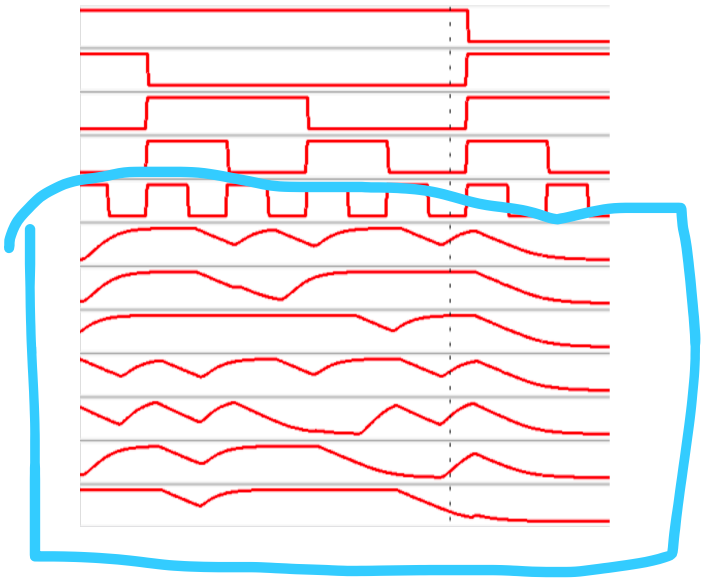


Fig 19. 8-bit Counter Advance

* Results obtained via Electric-VLSI Testing Tool

The counter operates synchronously, ensuring all flip-flops update their outputs in unison with a common clock signal. This synchronization eliminates the propagation delays inherent in asynchronous counters, leading to more predictable behavior at high frequencies.

Furthermore, the design includes parallel load functionality. This allows the counter to be directly loaded with a desired initial value, bypassing the counting sequence and enabling a wider range of applications. Additionally, an asynchronous reset is implemented to immediately force the counter's output to zero, independent of the clock signal, providing a mechanism for initialization or error correction.

Here the problem is counter degrades for MSBs (most-significant bits) part. Of course, the graph and the approach can be considered as vauge, still right after 4^{th} bit, the degration condition cannot be neglected.

CONCLUSION

The design of an 8-bit synchronous counter with parallel load and asynchronous reset was implemented using EDA tools. The counter successfully functioned for a 7-bit configuration and a 4-bit configuration when the layout dimensions were changed from 1x8 to 2x4. However, an issue arose where the most significant bit (MSB) did not function correctly in the 8-bit design.

Possible explanations for the malfunctioning MSB:

- **Layout issues:** When transitioning from a 1x8 layout to a 2x4 layout, routing for the clock signal or carry logic might have introduced unintended delays or mismatches. These delays could be affecting the propagation time to the MSB flip-flop, causing it to miss the clock edge.
- **Incomplete design:** Verification for the complete 8-bit counter might not have been thorough, leading to a logical error specific to the MSB logic or its interaction with other bits.

Recommendations for future work:

- **Detailed layout examination:** Analyze the routing paths for the clock signal and carry logic, particularly for the MSB flip-flop. Ensure proper signal integrity and minimize potential delays.
- **Comprehensive verification:** Perform rigorous simulation testing for the entire 8-bit counter, including various input scenarios and clock frequencies. This will help identify any logical errors specific to the MSB functionality.
- **Consider alternative layouts:** Explore different layout configurations (e.g., keeping a 1x8 structure) to determine if the issue is layout-related.

By addressing these points, the design of the 8-bit synchronous counter can be refined to achieve full functionality across all eight bits.

---

* All links malware checked.

REFERENCES

1. A. P. Douglas and E. Kamran, Basic VLSI Design, 3rd ed., Prentice Hall, 1994, pp. 72-76.

2. https://my.eng.utah.edu/~kalla/ECE3700/verlogic3_chapter5_portrait.pdf Access Date: Apr 12, 24.

3. https://cmosedu.com/cmos1/electric/electric.htm Access Date: Mar 2, 24.

4. Baker, R. J. (2019). *CMOS circuit design, layout, and simulation* (4th ed.). John Wiley & Sons.

5. https://www.ee.columbia.edu/~kinget/EE4303_S02/ Access Date: Mar 21, 24. (For LTSpice Demos)

6. Rabaey, J. M., Chandrakasan, A. P., & Nikolić, B. (2003). Digital Integrated Circuits: A Design Perspective. Pearson Education.

7. https://github.com/refaay/CMOS_Standard_Cell_Library (for .jelib examples) Access Date: Apr 14, 24.