Schematic and Layout Design of A 6x6-bit Array Multiplier

Alan and Liu

Spring 2025

1 Introduction

In modern digital integrated circuit design, multipliers serve as core arithmetic components whose performance directly impacts overall system efficiency. Array multipliers are widely used in VLSI design due to their regular structure and straightforward implementation. This project aims to design and implement a 6×6-bit array multiplier using 0.18μm CMOS technology at 1.8V supply voltage, processing two 6-bit unsigned inputs to produce a 12-bit product output.

The motivation is to practice the complete CMOS VLSI design flow, from circuit architecture and transistor-level implementation to schematic simulation, physical layout, and post-layout performance evaluation. We ensure functional correctness while optimizing performance metrics such as delay and area, understanding the trade-offs between speed and area.

This report details the complete design process, covering architecture selection, transistor-level circuit design of basic units, simulation results, layout optimization, and performance analysis including delay and area metrics.

2 Problem Formulation

The project focuses on designing a 6x6-bit array multiplier in 180nm CMOS technology with 1.8V supply. Each 12-bit output drives a 50-fF capacitive load. The main objectives are to implement full multiplier functionality and optimize for minimal delay and area through post-layout simulation.

Our optimization approach includes: (1) architectural optimization to minimize inverter count, (2) transistor sizing through parameter scanning for balanced area-delay performance, and (3) compact layout design to reduce overall area.

3 CMOS Design and Simulation Results

3.1 Transistor Size Design

When designing static CMOS circuits, it is recommended to make the PMOS section wider than the NMOS section to equalize the driving strength of the pinspotter if it is desirable to maximize the noise tolerance and obtain symmetrical characteristics. The PMOS and NMOS transistors sized so that an inverter switching threshold is at the midpoint of the supply voltage have a derived equation [1].

$$\frac{(W/L)_p}{(W/L)_n} = \frac{k'_n V_{DSATn}(V_M - V_{Tn} - V_{DSATn}/2)}{k'_p V_{DSATp}(V_{DD} - V_M + V_{Tp} + V_{DSATp}/2)} \tag{1}$$

Combining the parameters from our 0.18μm CMOS process, and the assumption of a 1.8V supply voltage, we finalized $\frac{(W/L)_p}{(W/L)_n} = 2.5$, which is applied to all of our transistor sizing designs.

CMOS circuits can also equate all the capacitance to an aggregate capacitance CL from the output to ground, the switching delay can be approximated by t = 0.69R * CL, and it can be determined that CL is proportional to the delay. CL is comprised of parasitic capacitances from transistors, including gate overlap capacitance, junction capacitance, and other parasitic capacitances. If the larger the size of the transistor, the larger the W , L, then the higher the capacitance, the delay will be high. As for the determination of W/L, if the width length ratio is higher, the equivalent resistance of the tube is lower, and the delay is smaller. So width to length ratio and width and length should be different variables.

In general, when we consider RC delay, W/L is large, on-resistance is small; W*L is large, load is large. And since we want to trade-off area and delay, we always want smaller values of W and L to get small area, so

the first thing that can be determined without any doubt is to choose the minimum L value in 0.18μm process, all the transistor lengths in this design have been determined to be 0.18μm. For the determination of the width value, we will perform a wide range of parameter scans under $\frac{(W/L)_p}{(W/L)_n} = 2.5$ to see how the combined area and delay behave and whether the curves can converge.

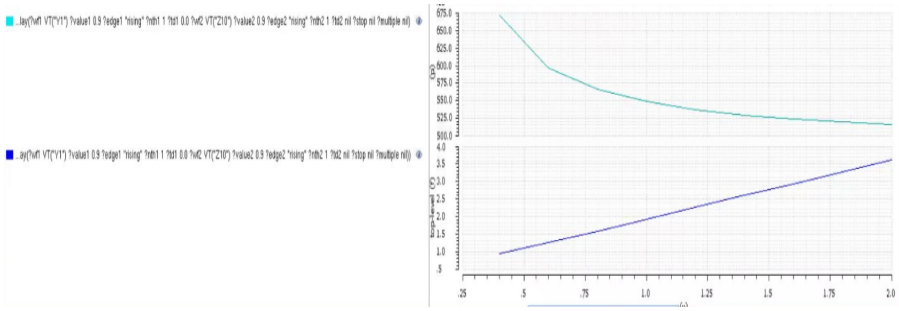


Figure 1: Parameter scan to determine transistor size.

As shown in the Figure 1, the NMOS transistor width was set to variable w_n, a 20 points linear step scan was performed between 0.4um and 2.0um. The PMOS transistor width was set to w_p = 2.5 * w_n. A test input of 111111*000000 varied to 111111*111111 was used. The first curve records the delay value t that was extracted by the The first curve records the delay value delay of the critical path extracted by calculator, the second curve multiplies ((VAR("w_n") + VAR("w_p") + bias) on top of delay. Bias is a prediction value representing the value of laying VDD, GND at the top and bottom of the layout, and laying interface electrical between PMOS and NMOS to reserve space between them for the metal needed for interface interconnections, etc. And with a defined structure, the area of the entire layout will be proportional to ((VAR("w_n") + VAR("w_p") + bias).

It can be seen that the delay decreases as w_n increases, but the order of magnitude of this change has much less effect on the increase in layout height (i.e., area). The second curve, which reflects the overall surface of the area times delay, is an approximately straight line, and in conjunction with the final layout layout considerations, we finalized the transistor sizes of the circuits in the cell bank so that their t_{plh} and t_{phl} are the same as those of the inverters with the following W / L sizes: 1.25μm / 0.18μm for PMOS and 0.5μm / 0.18μm for NMOS.

Adding devices in series slows down the circuit, so the devices must be designed wider to avoid performance degradation. When sizing a multi-input gate transistor, we should consider combinations of inputs that could lead to worst-case situations. And because the PMOS transistor widths of the INV , NAND , NOR, and XOR gates are all multiples of 1.25um, and the NMOS transistor widths are all multiples of 0.5um, we have adopted a multi-finger structure in the final layout to ensure that all gates have the same height, resulting in a more compact area.

Specifically, in the FA design, we derived the structure with the lowest number of transistors directly from the logic function and determined the transistor width of 1.25um for all pmos and 0.5um for nmos transistors, consistent with the overall layout.

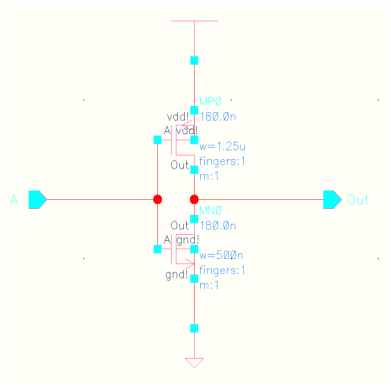
3.2 Standard cell library

There are four standard cells in our design in total, which are INV gate, NAND gate, NOR gate and XOR gate. The AND gate is made by connecting the INV gate to the back of the NAND gate, and the OR gate is made by connecting the INV gate to the back of the NOR gate. Based on the Standard cell library, we have designed two kinds of structural On the basis of the Standard cell library we have designed two half adder a special full adder to accommodate the final array multiplier architecture.

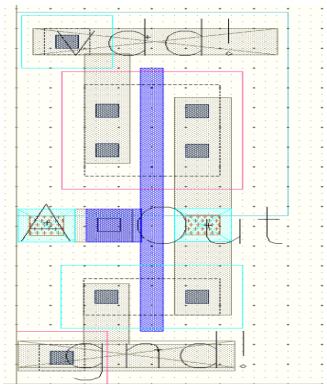
3.2.1 INV gate

Logical function: Out = A'

The INV gate functions to invert the input signal. If the input is a logic 1, the output is a logic 0; if the input is a logic 0, the output is a logic 1. Number of transistors: 2 (1 PMOS and 1 NMOS).



(a) Schematic of INV gate.



(b) Layout of INV gate.

Figure 2: Schematic and Layout of INV gate

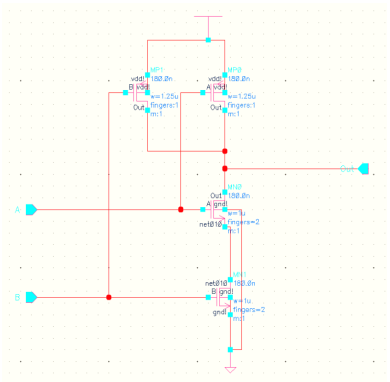
Table 1: The sizes of all the transistors in INV gate

transistor	Width(μm)	Length(μm)
MP0	1.25	0.18
MN0	0.5	0.18

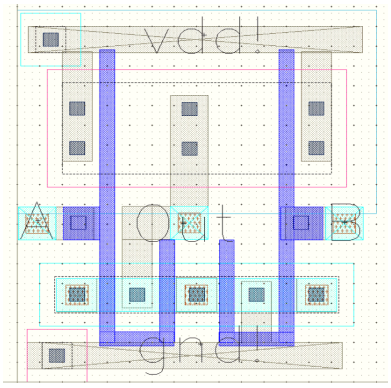
3.2.2 NAND gate

Logical function: Out = Out = (AB)'

The function of a NAND gate is that the output is a logic 0 only if all inputs are logic 1s; otherwise the output is a logic 1. Number of transistors: For an n-input NAND gate, 2n transistors (n PMOS and n NMOS) are required. Therefore, a 2-input NAND gate requires 4 transistors.



(a) Schematic of NAND gate.



(b) Layout of NAND gate.

Figure 3: Schematic and Layout of NAND gate

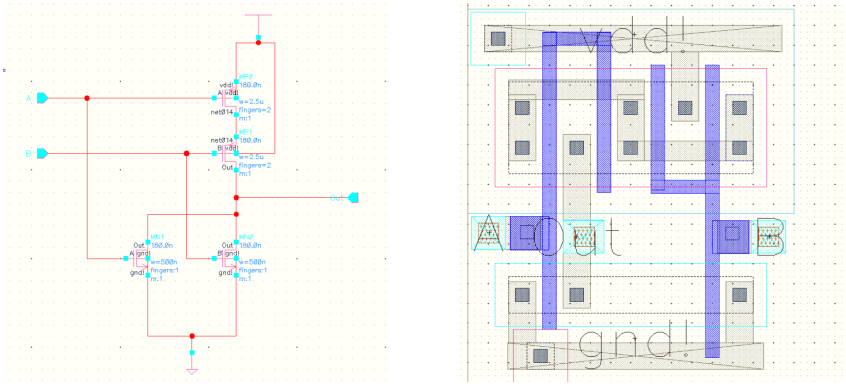
Table 2: The sizes of all the transistors in NAND gate

transistor	Width(μm)	Length(μm)
MP0	1.25	0.18
MP1	1.25	0.18
MN0	1	0.18
MN1	1	0.18

3.2.3 NOR gate

Logical function: Out = Out = (A + B)'

The function of a NOR gate is that the output is a logic 1 only if all inputs are logic 0s; otherwise the output is a logic 0. Static CMOS Logic Transistor Count: For an n-input NOR gate, 2n transistors (n PMOS and n NMOS) are required. Therefore, a 2-input NOR gate requires 4 transistors.



(a) Schematic of NOR gate. (b) Layout of NOR gate.

Figure 4: Schematic and Layout of NOR gate

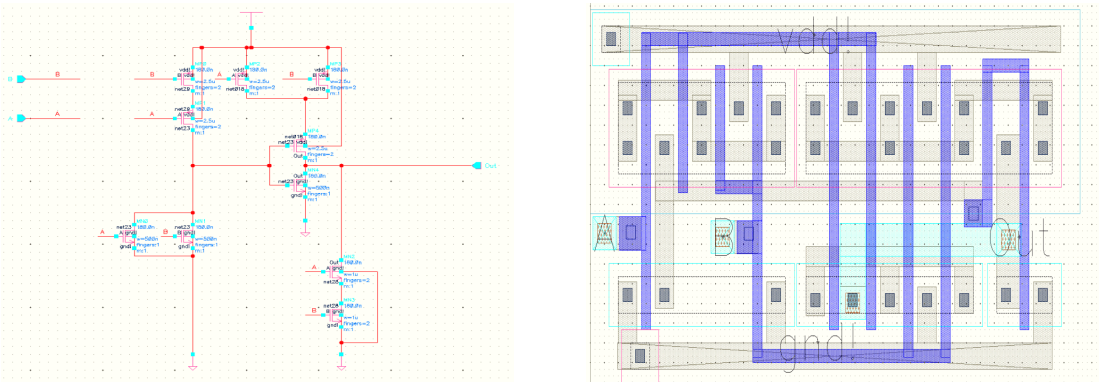
Table 3: The sizes of all the transistors in NAND gate

transistor	Width(μm)	Length(μm)
MP0	2.5	0.18
MP1	2.5	0.18
MN0	0.5	0.18
MN1	0.5	0.18

3.2.4 XOR gate

Logical function: Out = AB' + A'B = (A + B)' + AB)'

The function of the XOR gate is that the output is a logic 1 if and only if the input signals are different; if the input signals are the same, the output is a logic 0. There are various ways of implementing the XOR gate and the number of transistors will vary accordingly. In our design, the In our design, the static CMOS logic XOR gate consumes 10 transistors.



(a) Schematic of XOR gate. (b) Layout of XOR gate.

Figure 5: Schematic and Layout of XOR gate.

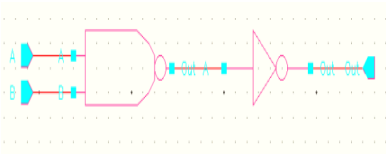
Table 4: The sizes of all the transistors in XOR gate

transistor	Width(μm)	Length(μm)
MP0	2.5	0.18
MP1	2.5	0.18
MP2	2.5	0.18
MP3	2.5	0.18
MP4	2.5	0.18
MN0	0.5	0.18
MN1	0.5	0.18
MN2	1	0.18
MN3	1	0.18
MN4	0.5	0.18

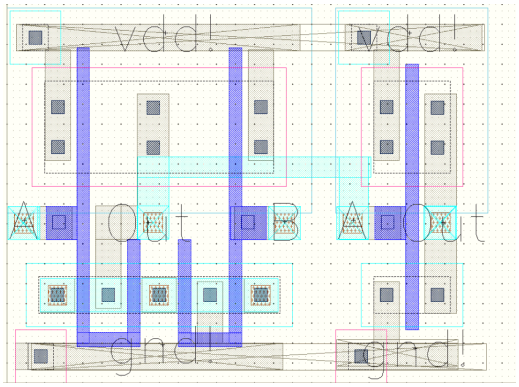
3.2.5 AND gate and OR gate

Logical function: AND: Out = AB; OR: Out = A + B.

The AND gate is made by connecting a INV gate to the back of a NAND gate, and the OR gate is made by connecting a INV gate to the back of a NOR gate.



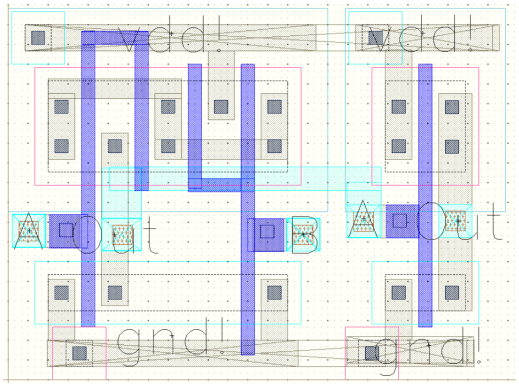
(a) Schematic of AND gate.



(b) Layout of AND gate.

Figure 6: Schematic and Layout of AND gate

(a) Schematic of OR gate.



(b) Layout of OR gate.

Figure 7: Schematic and Layout of OR gate

3.2.6 Propagation delay

Among them, the most basic gate elements are INV, NAND, XOR, and NOR, where each transistor's L takes the minimum value of 180nm. The following two tables show their pre-simulation and post-simulation delays.

Table 5: Propagation delay of different gate circuits (Schematic simulation)

Types	C_{out} = 15fF		C_{out} = 120fF		C_{out} = 960fF	
	t_{PLH} (ns)	t_{PHL} (ns)	t_{PLH} (ns)	t_{PHL} (ns)	t_{PLH} (ns)	t_{PHL} (ns)
INV	0.0857	0.1034	0.3763	0.4467	2.6765	3.1731
NAND	0.0723	0.1129	0.2914	0.4563	2.0347	3.1994
XOR	0.1518	0.1565	0.4840	0.4476	3.1543	2.7589
NOR	0.1140	0.1175	0.4539	0.4092	3.1227	2.7156

Table 6: Propagation delay of different gate circuits (Post-layout simulation)

Types	C_{out} = 15fF		C_{out} = 120fF		C_{out} = 960fF	
	t_{PLH} (ns)	t_{PHL} (ns)	t_{PLH} (ns)	t_{PHL} (ns)	t_{PLH} (ns)	t_{PHL} (ns)
INV	0.0926	0.1112	0.3802	0.4503	2.6859	3.1965
NAND	0.0825	0.1283	0.3003	0.4709	2.0425	3.2048
XOR	0.2221	0.2119	0.5566	0.5047	3.2287	2.8174
NOR	0.1164	0.1110	0.4531	0.4021	3.1216	2.7076

As the external load capacitance increases, the delay of the gate circuit increases. This is because the gate circuit must charge or discharge a larger capacitor, which takes more time.

When moving from simple CMOS inverters to more complex gate circuits (such as with-non gates, or non-non gates, or heterodyne gates), complex gate circuits have more internal nodes and transistors than inverters, which results in higher intrinsic capacitance due to diffusion capacitance of the additional transistors as well as gate-source/drain capacitance. Due to the increased intrinsic capacitance in complex gate circuits, the effect of external load capacitance on total delay become less pronounced relative to inverters.

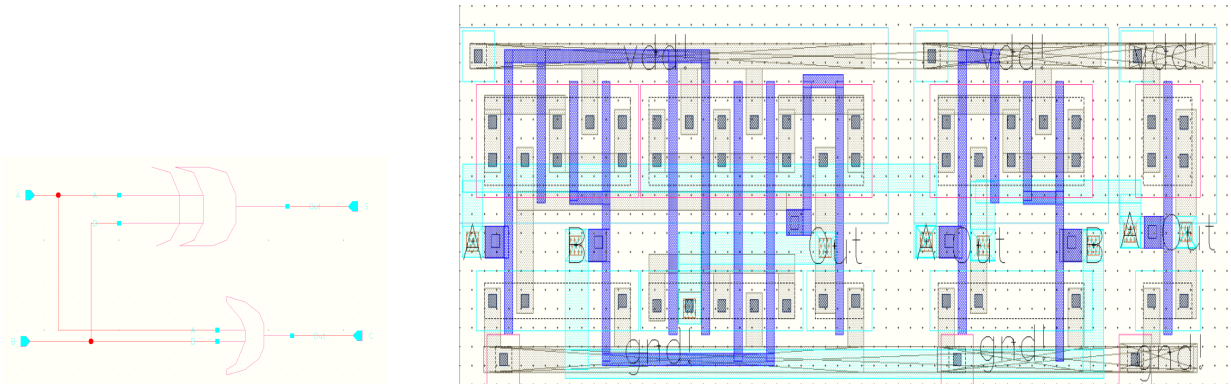
3.3 Half Adder

In this multiplier design, we employed two different types of Half Adders (HA) to accommodate design requirements and optimize specific logic paths.

The size of the transistors in the half-adder is consistent with the modules used in the cell library.

3.3.1 First Type Half Adder: Based on XOR and OR Gates

The first half adder (HA1) consists of an XOR gate and an OR gate with logical expressions: $S = A \oplus B$, $C = A + B$. When both inputs A and B are inverted, the sum S remains unchanged while the carry C becomes inverted.



(a) Schematic of HA1 (XOR + OR).

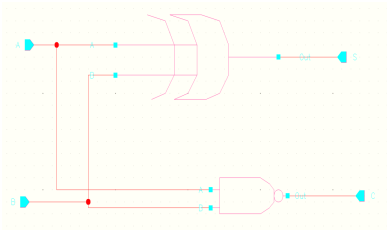
(b) Layout of HA1.

Figure 8: Schematic and Layout of First Type Half Adder

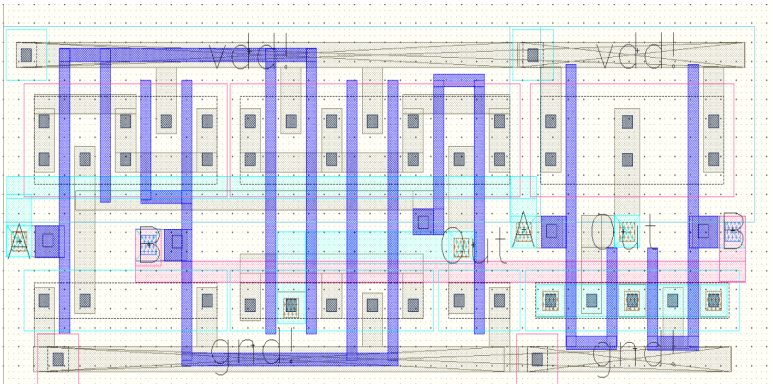
3.3.2 Second Type Half Adder: Based on XOR and NAND Gates

The second half adder (HA2) consists of an XOR gate and a NAND gate: $S = A \oplus B$, $C_{out} = \overline{A \cdot B}$. This design provides complementary carry output compared to HA1, enabling flexible control over carry signal

polarity.



(a) Schematic of HA2 (XOR + NAND).



(b) Layout of HA2.

Figure 9: Schematic and Layout of Second Type Half Adder

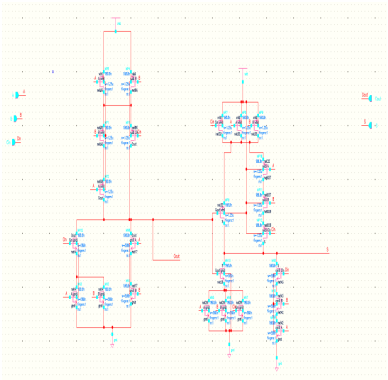
By using HA1 and HA2 selectively, we can ensure correct sum calculation regardless of input polarity while maintaining flexible control over carry signal polarity throughout the multiplier array.

3.4 Full Adder

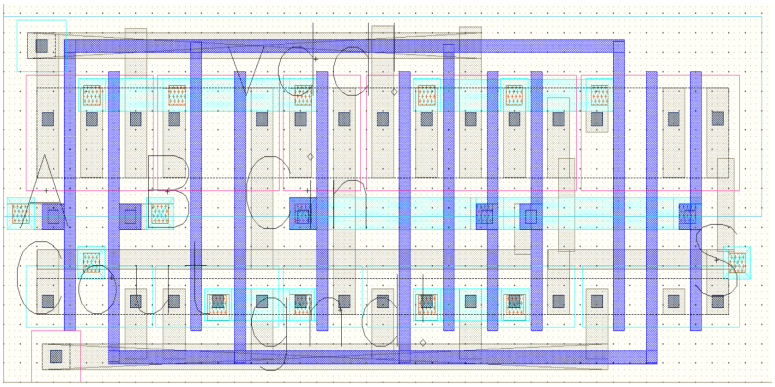
This design employs a 24T (24-transistor) full adder structure optimized for the array multiplier implementation. The circuit is based on a conventional full adder design but with the output inverters removed, resulting in inverted outputs. This modification reduces transistor count and power consumption while maintaining proper functionality within the multiplier array.

The full adder performs three-input binary addition with inputs A, B, and carry-in (C_{in}), producing sum (S) and carry-out (C_{out}) outputs according to: $S = A \oplus B \oplus C_{in}$, $C_{out} = AB + C_{in}(A \oplus B)$.

The 24T structure utilizes transmission gate logic combined with complementary CMOS logic to efficiently implement the full adder functionality. By eliminating output inverters, the circuit produces inverted outputs that are compatible with the overall multiplier architecture. All transistors in the full adder use uniform sizing: PMOS width (W_p) of 1.25 μm, NMOS width (W_n) of 0.5 μm, and channel length (L) of 180 nm.



(a) Schematic of Full Adder.



(b) Layout of Full Adder.

Figure 10: Schematic and Layout of Full Adder

3.5 6x6-bit Array Multiplier

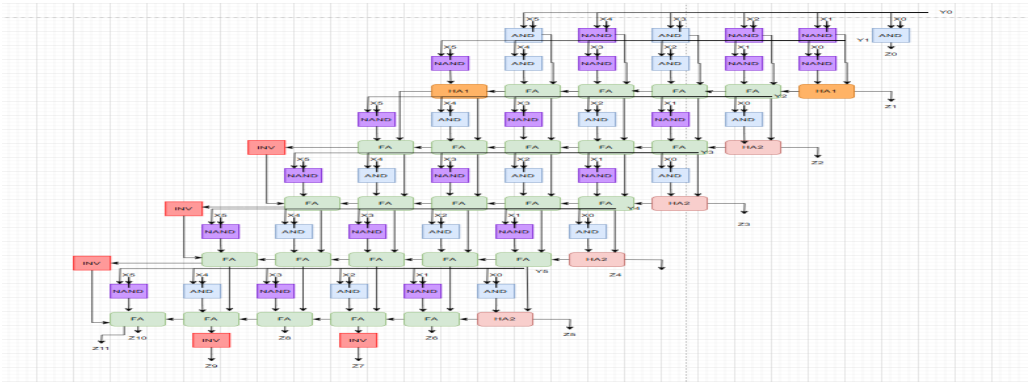


Figure 11: 6x6-bit Array Multiplier Architecture.

Our 6x6-bit array multiplier design prioritizes area compression while maintaining computational accuracy. The architecture of our array multiplier is illustrated in Figure 11. The array multiplier works based on the principle of add and shift algorithm. In this Multiplier, the partial products can be generated using AND gates and the summation of partial products can be performed using Full Adders and Half Adders. In an n * n array multiplier, n * n and gates computes the partial products and the addition of partial products can be performed by using n * (n - 2) full adders and n half adders [2].

The multiplier architecture utilizes a custom 24-transistor (24T) full adder design where the outputs are logically inverted compared to standard full adders. This design choice eliminates the need for output inverters in the full adders, significantly reducing the overall silicon area. Additionally, due to the inverted output characteristics of our full adders, some input AND gates can be replaced with NAND gates, further reducing the requirement for inverters and optimizing both area and delay performance.

To provide flexible control over the carry output (C) of half adders and meet the specific input polarity requirements of subsequent full adders, we implemented two distinct types of half adders. The first type consists of an XOR gate combined with an OR gate, while the second type uses an XOR gate paired with a NAND gate. Both half adder configurations ensure correct sum (S) output under normal or inverted input conditions, with their carry outputs (C) being logically complementary to each other. By replacing the traditional OR gate with a NAND gate in the second half adder type, we avoid the need for additional inverters to adjust carry polarity, thereby saving area while potentially reducing path delays.

After testing, the Cout and S delays of the full adder are greater than the C and S delays of the two half adders. Therefore, the longest path should start from the first half adder and not pass through the half adder again, all the way to the output Z11. After a simple recursive calculation of the layout of the intermediate full adders, it can be concluded that there are a total of 323 critical paths.

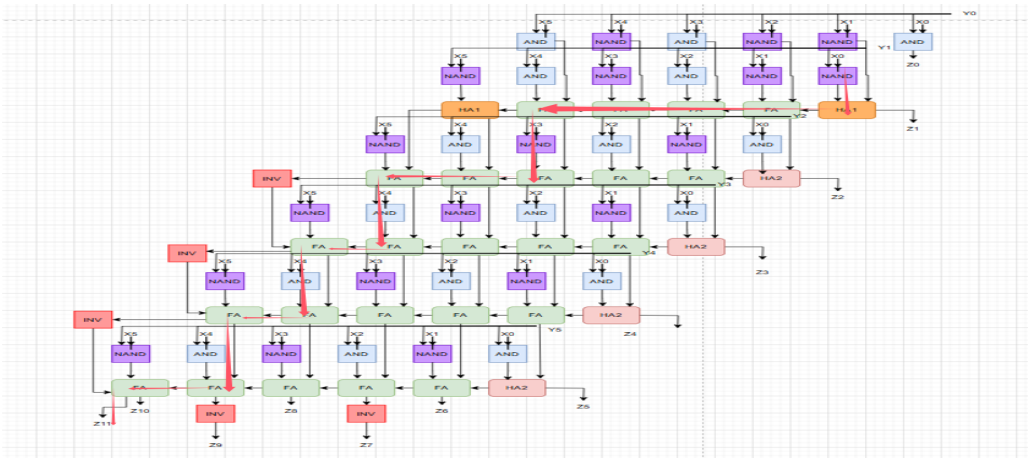


Figure 12: A Critical Path of 6x6-bit Array Multiplier.

One of the critical delay paths (from 111111 * 000000 to 111111 * 111111) is selected, as shown in Figure 12. The propagation delay of the multiplier is measured through schematic-level simulation. Assuming each

product output drives an extrinsic load capacitance of 50 fF, the measured delays are t_{PLH} = 0.81772 ns and t_{PHL} = 0.6368 ns, as shown in Figure 13.

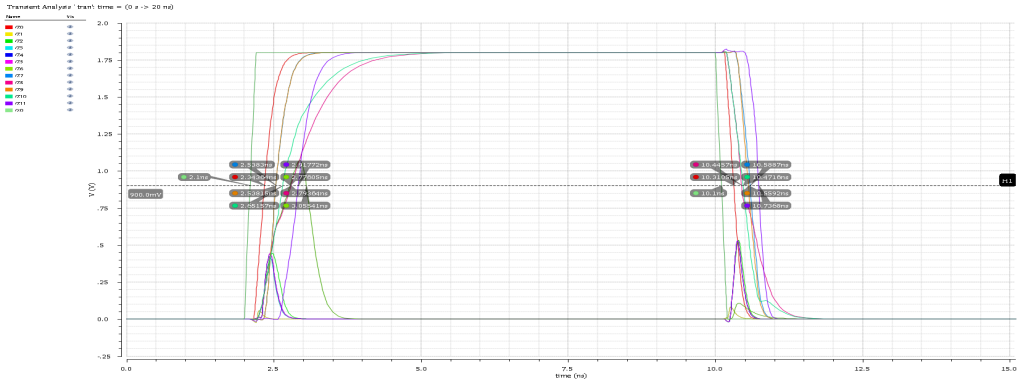


Figure 13: Propagation delay of 6x6-bit Array Multiplier(Schematic simulation).

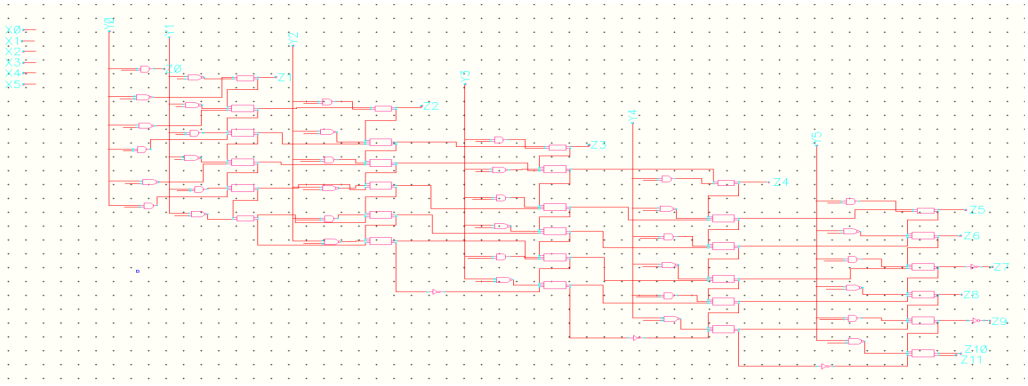


Figure 14: Schematic of 6x6-bit Array Multiplier.

3.6 6x6-bit Array Multiplier Layout Area and Delay

The physical layout of our 6x6-bit array multiplier is presented in Figure 15. Total layout area measured $104.892 \times 30.961 = 3247.561\,\mu m^2$

Post layout simulation of the extracted layout can yield the measured delays are t_{PLH} = 1.2245 ns and t_{PHL} = 1.0014 ns, as shown in Figure 16.

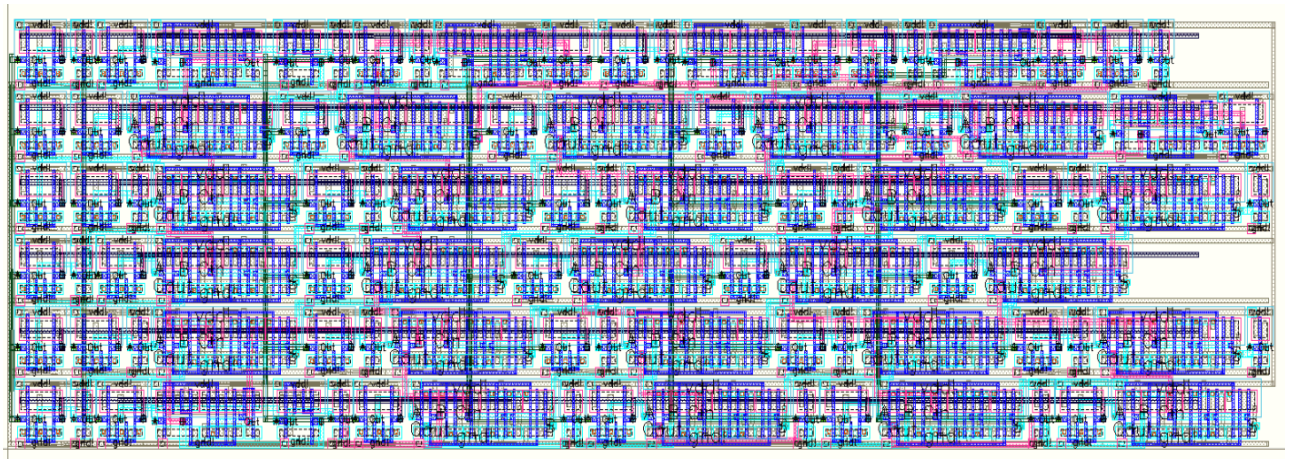


Figure 15: Layout of 6x6-bit Array Multiplier.

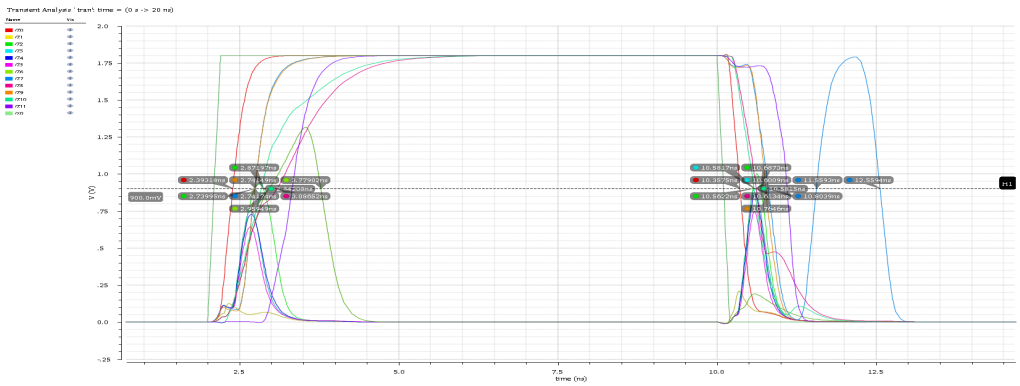


Figure 16: Propagation delay of 6x6-bit Array Multiplier(Post-layout simulation).

The differences between schematic-level and post-layout simulations primarily arise from parasitic resistances and capacitances that are not accounted for in ideal schematic simulations. Schematic-level circuit simulation tools have inherent limitations as they typically assume ideal interconnects with negligible parasitic effects. In contrast, post-layout simulations incorporate extracted parasitic elements from the actual physical layout, including metal line resistances, interlayer capacitances, and coupling effects between adjacent conductors. Accurate resistance and capacitance extraction is crucial for reliable delay estimation in CADENCE tools, as these parasitic elements can significantly impact signal propagation delays, especially in high-frequency or timing-critical applications. The extracted parasitic network provides a more realistic representation of the circuit behavior, enabling designers to verify that the implemented design meets timing specifications under real operating conditions.

4 Conclusions

In this project we have designed 6x6-bit Array Multiplier. We have designed the circuit connections, layout design, successfully passed the Design Rule Check (DRC) and Layout Vs Schematic (LVS) verification. The final layout area is 3247.561 μm^2 and the average delay for Schematic simulation are t_{PLH} = 0.81772 ns and t_{PHL} = 0.6368 ns and the average delay for post layout simulation are t_{PLH} = 1.2245 ns and t_{PHL} = 1.0014 ns, under the given simulation test data. The circuit performs well in terms of propagation delay and area of the combined performance is outstanding.

In our optimized 6x6-bit Array Multiplier architecture, only 24 INV gates, 40 NAND gates, 2 NOR gates, 6 XOR gates, and 24 full adders(24T structure) are used. A total of 852 transistors significantly reduces the number of transistors required for the architecture. During the design process of transistor sizing, area and delay trade-offs were considered, and the final transistor sizing parameters were determined from theory and combined with actual parameter scans. In addition, we continuously optimized the layout of each module and the final 6x6-bit Array Multiplier to maximize space savings and parasitic capacitance and resistance. This project gave us a deeper understanding of the digital CMOS IC design process.

References

[1] Rabaey, J. M., Chandrakasan, A., & Nikolic, B. (2002). Digital integrated circuits (Vol. 2). Englewood Cliffs: Prentice hall.

[2] Sabeetha, S., Ajayan, J., Shriram, S., Vivek, K., & Rajesh, V. (2015, February). A study of performance comparison of digital multipliers using 22nm strained silicon technology. In 2015 2nd international conference on electronics and communication systems (ICECS) (pp. 180-184). IEEE.