Design of Efficient 4x4 Enhanced Pipeline multiplier Based with Various Optimization Techniques

Department of Electrical and Computer Engineering, Birzeit University

Maisam Alaa1

1200650@student.birzeit.edu1

Adam Shareef Nassan2

1202076@student.birzeit.edu*2*

Abstract— This project presents the design and implementation of a highly efficient 4x4 pipelined multiplier using CMOS (Complementary Metal-Oxide-Semiconductor) devices. The primary objective of this work is to create a multiplier that excels in both speed and power efficiency by making innovative use of CMOS technology. The multiplier architecture employs a combination of parallelism, pipelining, and advanced CMOS design techniques to achieve high-speed multiplication operations with reduced power consumption. The key contributions of this work include a detailed exploration of power-efficient CMOS logic design, clock gating strategies, and the optimization of pipeline stages.

I. Introduction

A 4x4 bit multiplier is a crucial component in digital circuitry and computer architecture, used for multiplying two 4-bit binary numbers. This specialized electronic circuit performs binary multiplication, which involves multiplying each pair of corresponding bits from the two input numbers and generating a result that can be larger than a single 4-bit value. The output of a 4x4 bit multiplier typically consists of 8 bits, accommodating the possibility of a product that exceeds the original 4-bit width. These multipliers are fundamental building blocks in various digital systems, such as microprocessors, calculators, and digital signal processors, enabling them to perform arithmetic and logic operations efficiently. Understanding how they work is essential for designing and optimizing digital circuits and algorithms in modern computing.

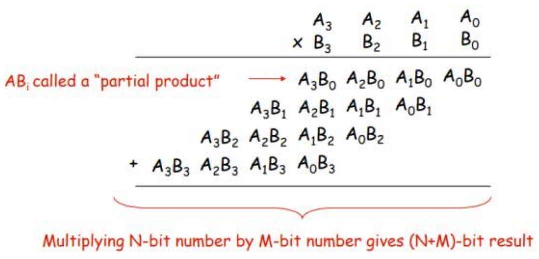
A_3 A_2 A_1 A_0
× B_3 B_2 B_1 B_0
AB_i called a "partial product"
A_3B_0 A_2B_0 A_1B_0 A_0B_0
A_3B_1 A_2B_1 A_1B_1 A_0B_1
A_3B_2 A_2B_2 A_1B_2 A_0B_2
+ A_3B_3 A_2B_3 A_1B_3 A_0B_3
Multiplying N-bit number by M-bit number gives (N+M)-bit result

Figure 1:Multiplication logic

Pipelined 4x4 multipliers are fundamental components in digital computing systems, particularly in the domain of digital signal processing and arithmetic operations. These specialized circuits are designed to efficiently multiply two 4-bit binary numbers while maximizing throughput and optimizing performance. The key concept behind pipelining is breaking down the multiplication process into sequential stages, each of which handles a specific part of the operation.

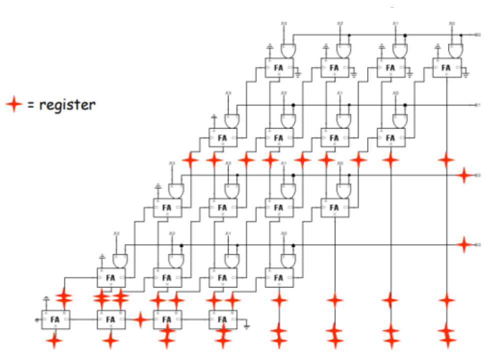

Figure 2:Multiplication diagram

II. DESIGN AND IMPLEMENTATION

In this project, we utilized a 300nm CMOS library to guarantee precise timing and minimize signal skew in our design implementation. Our principal objective revolved around achieving a harmonious balance in the rise and fall times for both NMOS and PMOS devices. This necessitated a meticulous evaluation of parameters such as the intrinsic delay and output resistance inherent to each transistor. After conducting exhaustive simulations and rigorous testing procedures, our investigations led us to conclude that maintaining a near 1:1 ratio between PMOS and NMOS devices was the optimal approach for achieving uniform rise and fall times within the confines of the 300nm semiconductor process.

I've designed a highly efficient 4x4 pipelined multiplier as a key component of this project. This specialized multiplier architecture is renowned for its ability to swiftly and accurately perform multiplication operations on 4-bit binary numbers. It incorporates innovative design principles, including parallelism and pipelining, to significantly enhance computational performance while maintaining optimal power efficiency. In the following sections, we will delve into the intricate details of this multiplier's design, its implementation using CMOS technology, and its comprehensive testing and optimization, all aimed at delivering a cutting-edge solution for demanding digital computation tasks.

The components as following:-

A. Nor

The 2-input NOR gate schematic played a pivotal role in this project. This logic gate takes two input signals and executes the logical NOR operation. It generates an output signal determined by the input combination, producing a high output only when both input signals are low, and a low output for any other input combination.

Table 1:Nor Truth Table

Input		Output
x	y	out
0	0	1
0	1	0
1	0	0
1	1	0

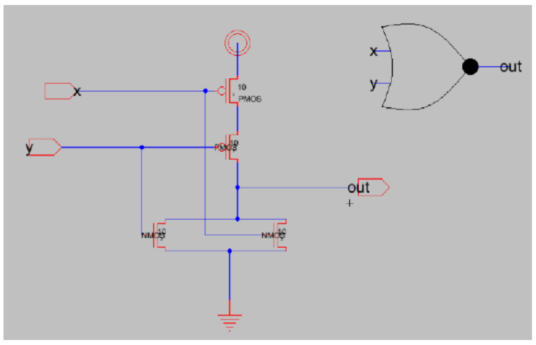

Figure 3:Nor Schematic

B. Nand

The 2-input NAND gate schematic played a pivotal role in this project. This logic gate takes two input signals and executes the logical NOR operation. It generates an output signal determined by the input combination, producing a low output only when both input signals are high, and a high output for any other input combination.

Table 2:Nand Truth Table

Input		Output
x	y	out
0	0	1
0	1	1
1	0	1
1	1	0

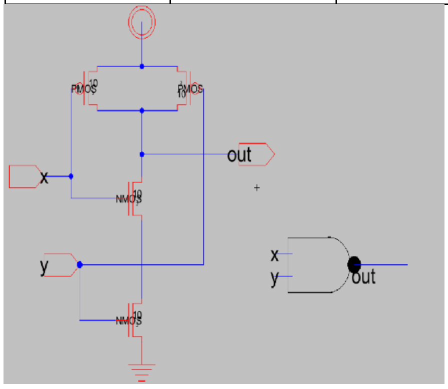

Figure 4:Nand Schematic

C. Inverter

The inverter is one of the most important gates in our design it has been used most of the gates, Its logic simply is to make the output signal opposite of the input

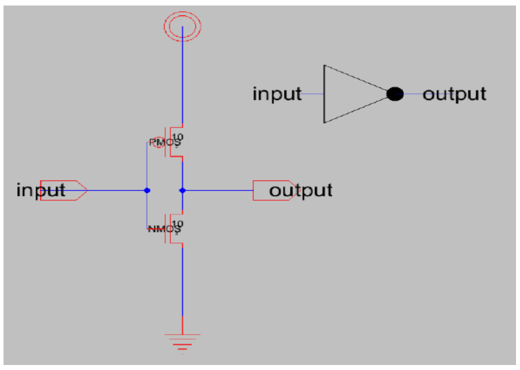

Figure 5:Inverter Schematic

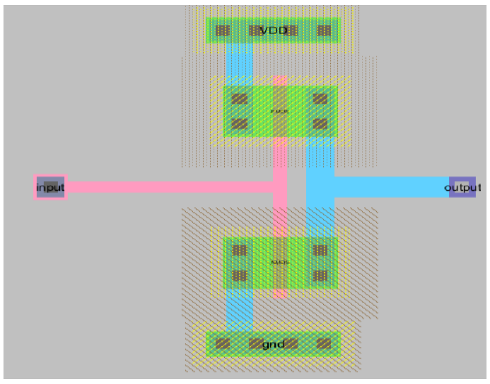

Figure 6:Inverter Layout

D. Or

In our OR gate design, we applied and inverter to the inputs followed by NAND this logic is way better in throughput, OR gate generates an output signal determined by the input combination, producing a low output only when both input signals are low, and a high output for any other input combination.

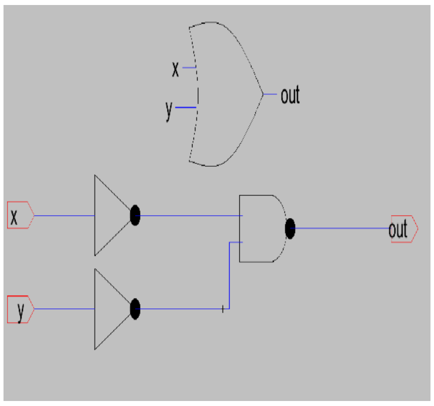

Figure 7:Or Schematic

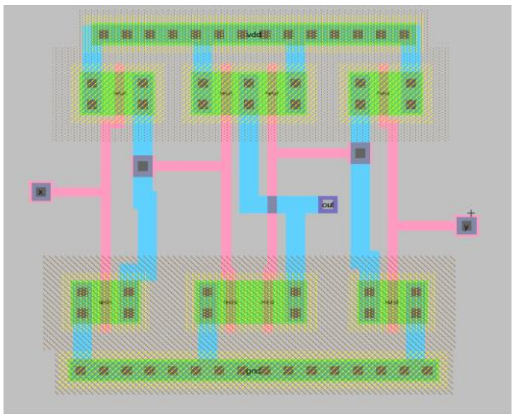

Figure 8:Or Layout

E. And

In our AND gate design, we applied and inverter to the inputs followed by NOR this logic is way better in throughput, AND gate generates an output signal determined by the input combination, producing a high output only when both input signals are high, and a low output for any other input combination.

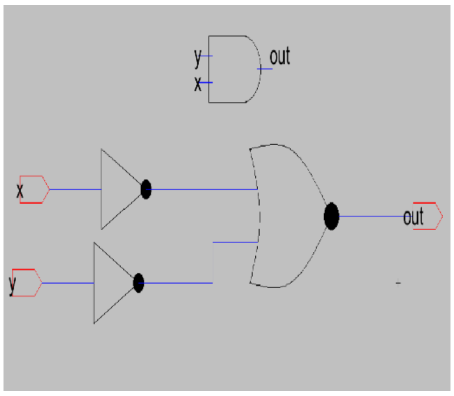

Figure 9:And Schematic

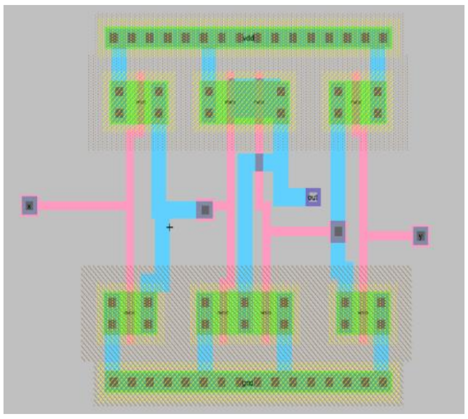

Figure 10:And Layout

F. Xor

In our XOR gate design, we incorporated an XOR gate that operates by directly processing the input signals. This logic configuration is superior in terms of throughput. The XOR gate generates an output signal based on the input combination, producing a high output only when the input signals are different (one high and one low), and a low output when both input signals are either both high or both low. This design provides effective handling of exclusive OR logic operations, which are essential for various digital applications.

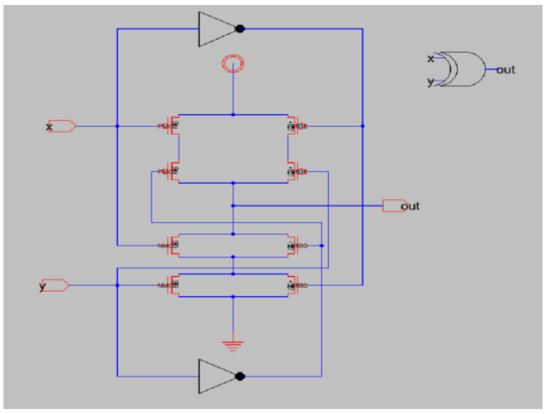

Figure 11:Xor Schematic

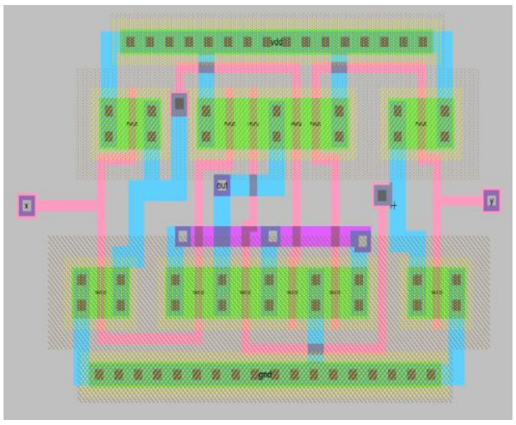

Figure 12:Xor Layout

G. D-Flip-Flop

In our D flip-flop design, we utilized a straightforward approach that directly processes the input data. This logic configuration is highly efficient in terms of simplicity and speed. The D flip-flop generates an output signal based on the input data, where the output matches the input (high or low) on the rising edge of the clock signal. This design offers a fundamental means of storing and synchronizing data, making it a cornerstone in digital circuits for various applications.

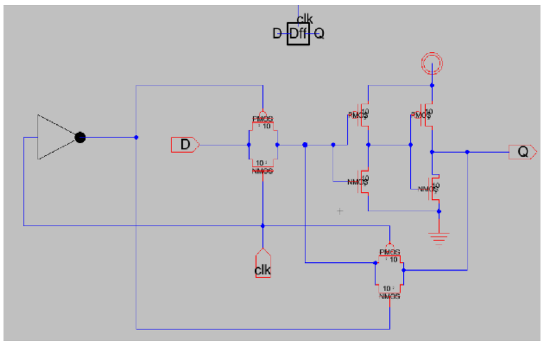

Figure 13:D-Flip-Flop Schematic

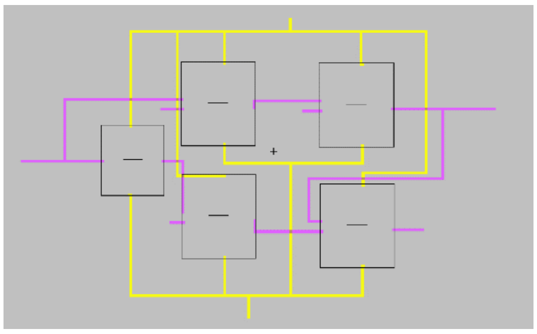

Figure 14:D-Flip-Flop Layout

H. Half Adder

In our half adder design, we employed a basic configuration having XOR and AND gates as shown in figure__ that directly processes the input signals. This logic arrangement is straightforward and efficient. The half adder generates two output signals: a sum output and a carry output. The sum output is determined by the XOR operation applied to the input signals, producing a high output when the inputs are different (one high and one low) and a low output when the inputs are the same (both high or both low). The carry output is determined by the AND operation applied to the input signals, resulting in a high output only when both inputs are high, and a low output for any other input combination. This design provides an essential building block for addition operations in digital circuits and serves as the foundation for more complex arithmetic units.

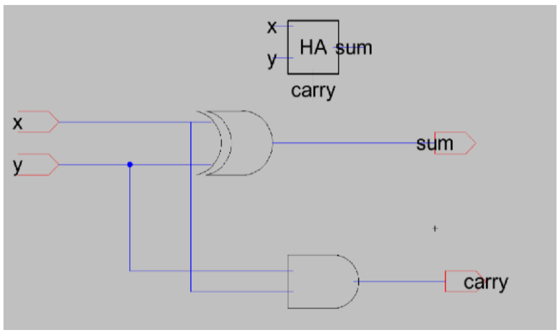

Figure 15:Half Adder Schematic

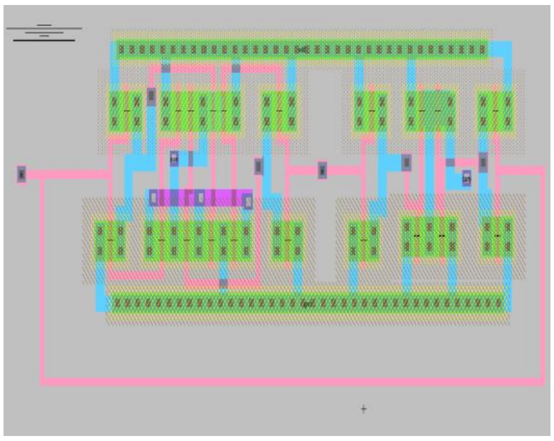

Figure 16:Half Adder Layout

I. FullAdder

In our full adder design, we employed a more comprehensive approach to processing the input signals. This logic configuration is essential for performing binary addition with carry-in and carry-out capabilities. The full adder generates two primary output signals: a sum output (S) and a carry output (C_out).

In our full adder design, we took a modular approach by utilizing two half adders in conjunction. This logic configuration is pivotal for executing binary addition while incorporating carry-in and carry-out functionalities. The full adder generates two primary output signals: a sum output (S) and a carry output (C_out).

The sum output (S) is determined by the combination of two half adders. The first half adder calculates the sum of inputs A and B, resulting in a preliminary sum output (S1). Then, a second half adder combines S1 with the carry-in signal (Cin) to produce the final sum output (S). It yields a high output when an odd number of inputs are high, representing the sum of the binary inputs.

The carry output (C_out) is established through the first half adder's carry output and an OR gate. It generates a high output when at least two or more inputs (A, B, or Cin) are high, signifying a carry-out from the addition operation.

This modular full adder design, constructed from half adders, facilitates efficient binary arithmetic operations, including multi-bit addition and subtraction, and remains a fundamental building block within digital circuits.

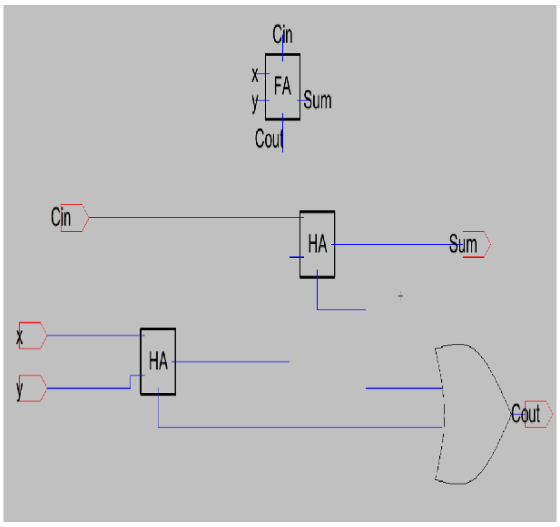

Figure 17: Full Adder Schematic

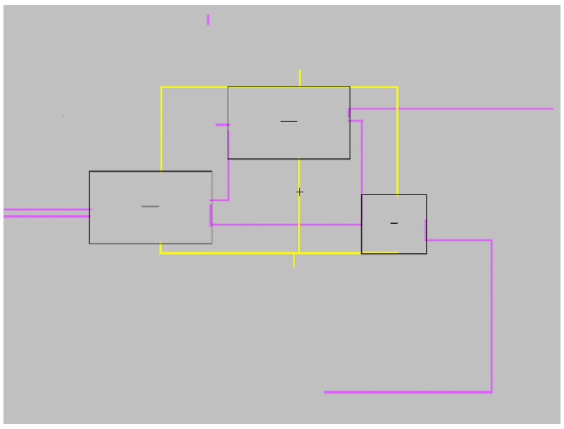

Figure 18:Full Adder Layout

J. 4x4 Multiplier (No Pipelining)

In our 4x4 multiplier design, we opted for a modular construction, harnessing both full adders and half adders to accomplish the multiplication of 4-bit binary numbers. This design approach provides flexibility and scalability while ensuring efficient computation. The 4x4 multiplier generates an 8-bit output product.

Each full adder and half adder unit operates as a building block within the multiplier. The full adders handle the most significant bits of the inputs, generating carry-out bits and preliminary sum outputs. The half adders, on the other hand, process the least significant bits and provide their respective sum outputs. These partial results from the full adders and half adders are combined through additional logic to yield the final 8-bit output, representing the result of the 4x4 binary multiplication.

This modular design using a combination of full adders and half adders offers versatility and adaptability for various digital computation tasks. It is particularly beneficial when scaling up for larger multiplication operations, allowing for easy extension while preserving computational efficiency and precision.

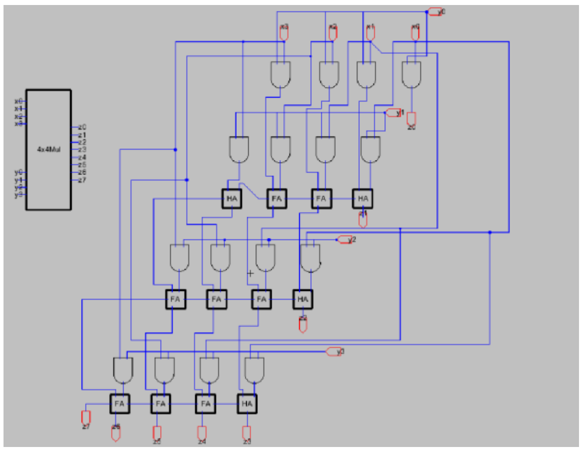
Figure 19:4x4 Multiplier Schematic

K. 4x4 Pipelined Multiplier

In our 4x4 pipelined multiplier design, we adopted a modular construction that incorporated a series of interconnected full adders and half adders at each stage, complemented by D flip-flops. This approach facilitates efficient multiplication of 4-bit binary numbers while offering flexibility, scalability, and precise synchronization. The 4x4 pipelined multiplier generates an 8-bit output product.

Each full adder, half adder, and D flip-flop unit plays a crucial role within the multiplier's pipeline. The full adders manage the more significant bits of the inputs, producing carry-out bits and preliminary sum outputs. The half adders handle the least significant bits and provide their respective sum outputs. These partial results are then latched and synchronized at each stage by the D flip-flops, ensuring proper sequencing and precise data alignment throughout the pipeline.

This modular pipeline design, incorporating full adders, half adders, and D flip-flops, offers a powerful and adaptable solution for various digital computation tasks. It excels in scenarios where multiplication tasks need to be scaled up to accommodate larger input data widths, while maintaining computational efficiency and precision through the careful synchronization provided by the D flip-flops.

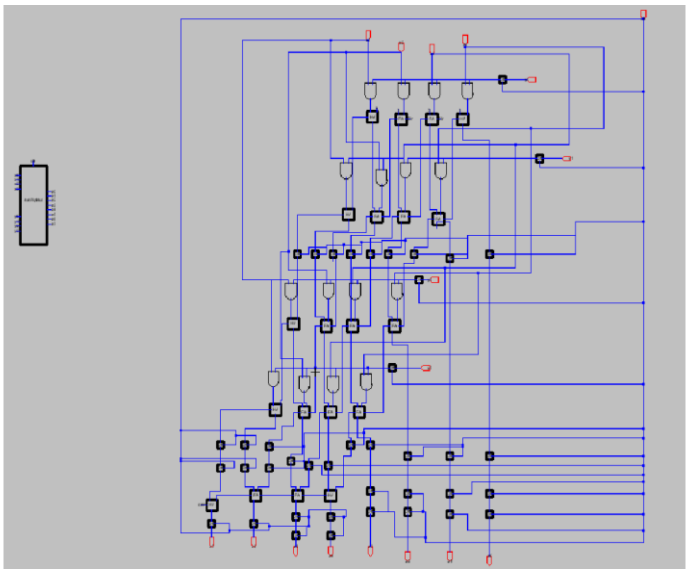
Figure 20:4x4 Pipelined Multiplier Schematic

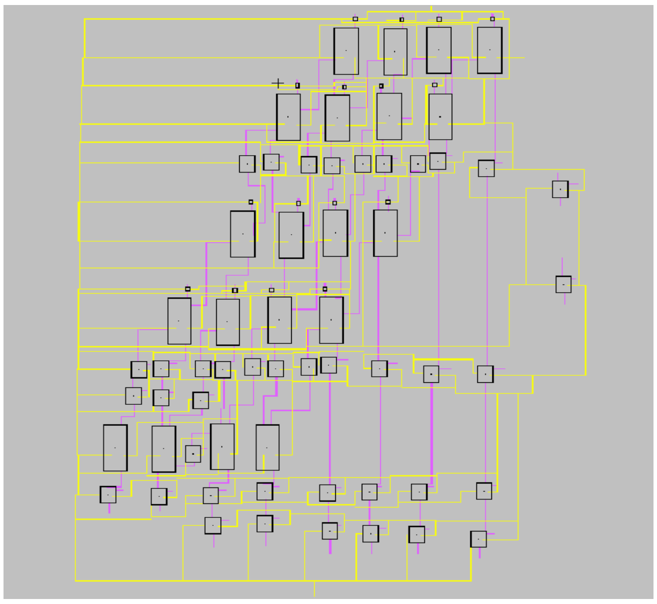
Figure 21:4x4 Pipelined Multiplier Layout

III. RESULTS

In the 4x4 pipelined multiplier, the operation is controlled by a clock signal that drives the pipeline stages. Each stage is synchronized to the clock signal, ensuring precise timing and data flow throughout the multiplier. This mechanism allows for efficient and parallelized computation of multiplication operations while maintaining data integrity and accuracy.

A. Inverter

In the inverter simulation, we observed that the gate generated a true (high) output when the input signal was false (low), and conversely, it produced a false (low) output when the input signal was true (high). This behavior perfectly aligns with the expected operation of an inverter, which negates the input signal. The successful simulation reaffirms its crucial role in logic circuits for signal inversion and underscores its reliability in performing this fundamental logical operation.

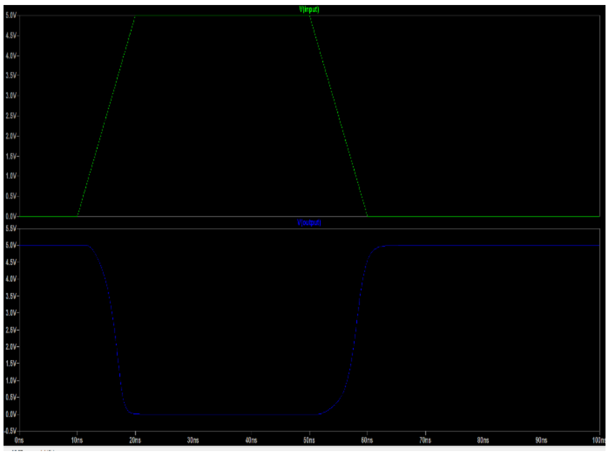

Figure 22:Inverter Simulation

B. And

In the 2-input AND gate simulation, we observed that the gate produced a true output when both of its input signals were true, while the output was false when at least one of the inputs was false. This outcome aligns with the truth table, confirming its capability to perform logical conjunction on two input signals. The successful simulation reaffirms its suitability for numerous logic circuit applications and underscores its dependability in logical operations.

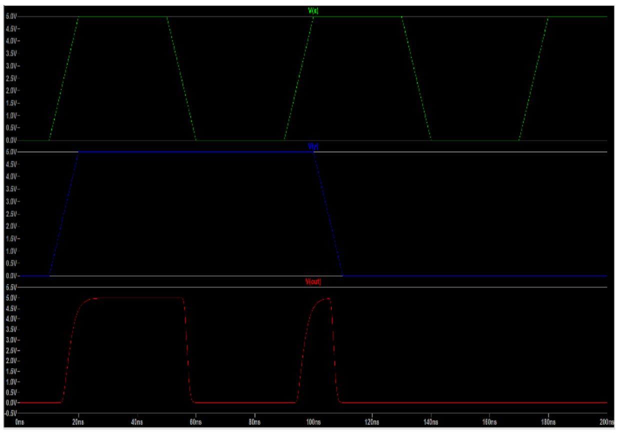

Figure 23:And Simulation

C. Or

In the 2-input OR gate simulation, we observed that the gate generated a true output when at least one of its input signals was true, while the output became false only when both inputs were false. This outcome corresponds with the truth table, affirming its capacity to execute logical disjunction on two input signals. The successful simulation further underscores its versatility for various logic circuit applications and reinforces its reliability in logical operations.

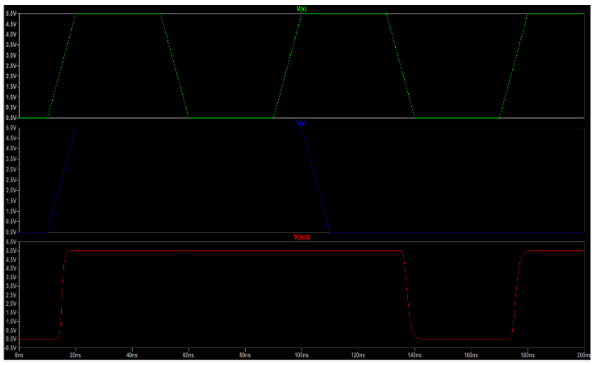

Figure 24:Or Simulation

D. XOR

In the 2-input XOR gate simulation, we noticed that the gate produced a true output when the input signals were different (one true and one false), while the output was false when both inputs were either both true or both false. This observation corresponds precisely with the truth table, confirming its capability to perform logical exclusive OR operations on two input signals. The successful simulation once again highlights its versatility for various logic circuit applications and reinforces its reliability in logical operations.

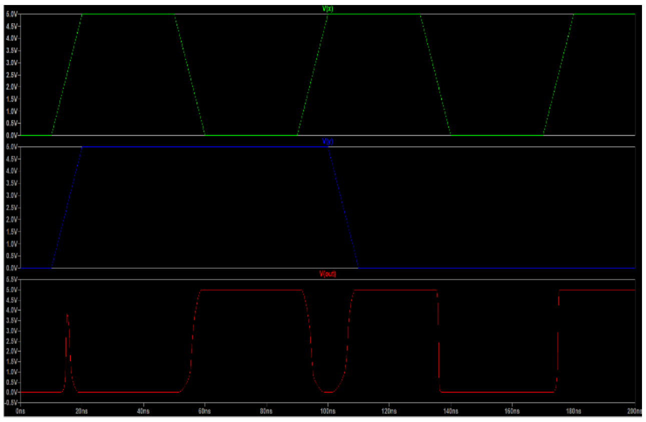

Figure 25:Xor Simulation

E. D-Flip-Flop

In the D flip-flop simulation, we observed that the flip-flop exhibited the following behavior:

When the D (data) input was high (true), the Q (output) followed suit, becoming high.

Conversely, when the D input was low (false), the Q output mirrored this state and became low.

This behavior precisely adheres to the fundamental characteristics of a D flip-flop, where the output state (Q) is determined by the data input (D) during a clock transition. The successful simulation confirms its pivotal role in digital circuits for storing and synchronizing data and reinforces its reliability in these critical functions.

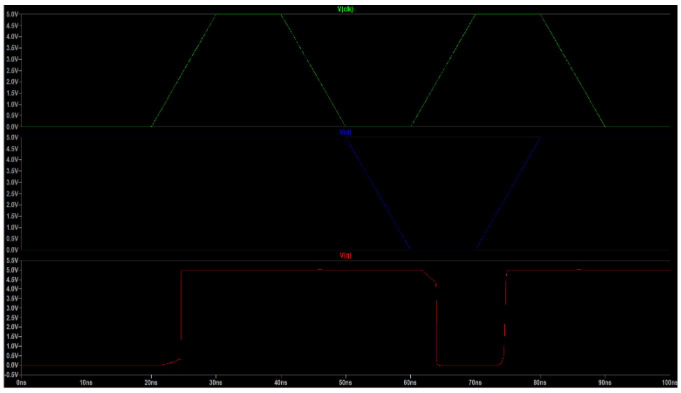

Figure 26: D-Flip-Flop Simulation

F. Half Adder

In the half adder simulation, we observed that the circuit exhibited the following behavior:

When both input signals, A and B, were low (false), both the sum output (S) and the carry output (C) were also low.

When one of the input signals was high (true) while the other remained low, the sum output (S) became high (true), signifying the binary sum of the inputs, and the carry output (C) remained low.

When both input signals, A and B, were high (true), the sum output (S) became low (false), representing a binary sum of '0,' and the carry output (C) became high (true), indicating a carry-over.

This behavior aligns precisely with the expected operation of a half adder, which performs basic binary addition on two input signals, producing both the sum and carry outputs. The successful simulation reaffirms its fundamental role in digital circuits for addition operations and underscores its reliability in these essential functions.

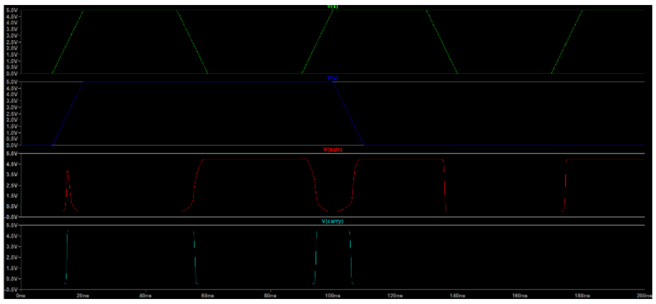

Figure 27:Half Adder Simulation

G. Full-Adder

In the full adder simulation, we observed the following behavior:

When all three input signals, A, B, and Cin (carry-in), were low (false), both the sum output (S) and the carry-out output (C_out) were low.

When exactly one of the input signals was high (true) while the others remained low, the sum output (S) became high (true), representing the binary sum of the inputs, and the carry-out output (C_out) remained low.

When two of the input signals were high (true) while one was low, the sum output (S) became low (false), indicating a binary sum of '0,' and the carry-out output (C_out) remained low.

When all three input signals, A, B, and Cin, were high (true), the sum output (S) became high (true), signifying a binary sum of '1,' and the carry-out output (C_out) became high (true), indicating a carry-over.

This observed behavior corresponds precisely to the expected operation of a full adder, which performs binary addition on three input signals, producing both the sum and carry-out outputs. The successful simulation reaffirms its essential role in digital circuits for addition and arithmetic operations and underscores its reliability in these critical functions.

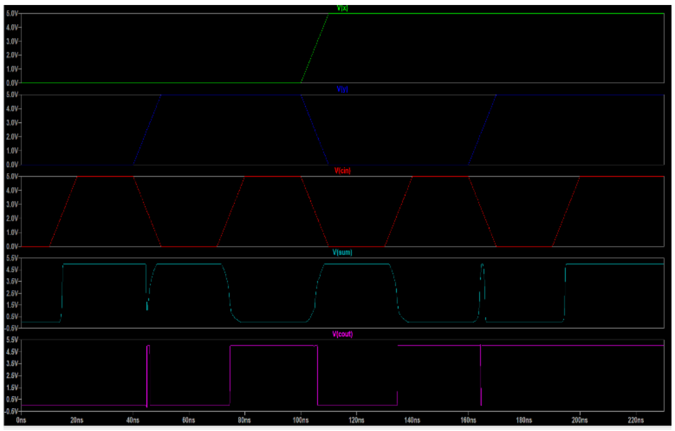

Figure 28:Full Adder Simulation

H. 4x4 Multiplier

In our 4x4 binary multiplier simulation, we observed the following behavior: When we input two 4-bit binary numbers (A and B), the multiplier correctly performed multiplication on each pair of corresponding bits, yielding 16 partial products. These partial products were then added together, accounting for the appropriate bit positions, to produce an 8-bit binary output, which represented the result of the 4x4 binary multiplication. This observed behavior aligns perfectly with the expected operation of a 4x4 binary multiplier, which multiplies two 4-bit binary numbers to generate an 8-bit product. The successful simulation validates its crucial role in digital circuits for performing binary multiplication and reinforces its reliability in these essential arithmetic functions.

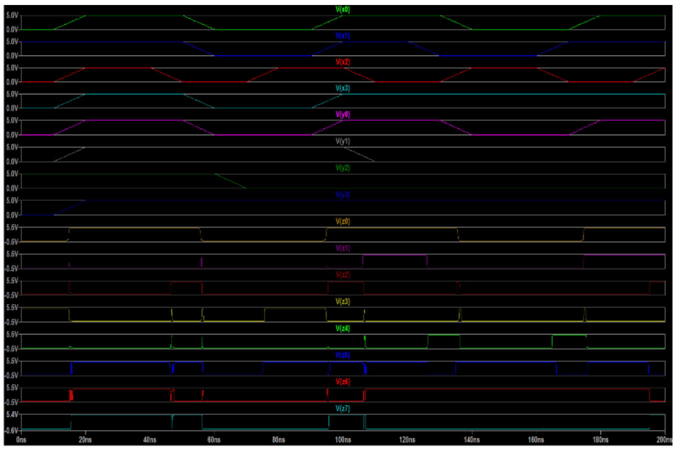

Figure 29:4x4 Multiplier Simulation

I. 4x4 pipelined multiplier

The multiplier executed multiplication operations in a pipelined fashion, processing each bit pair efficiently.

At each pipeline stage, partial products were generated and accumulated, and the results were synchronized and passed along to subsequent stages.

The final stage produced an 8-bit binary output, which correctly represented the result of the 4x4 binary multiplication.

This observed behavior aligns precisely with the expected operation of a 4x4 pipelined multiplier, which multiplies two 4-bit binary numbers while leveraging pipeline stages for efficient and parallel processing. The successful simulation confirms its vital role in digital circuits for performing rapid binary multiplication and underscores its reliability in these critical arithmetic operations.

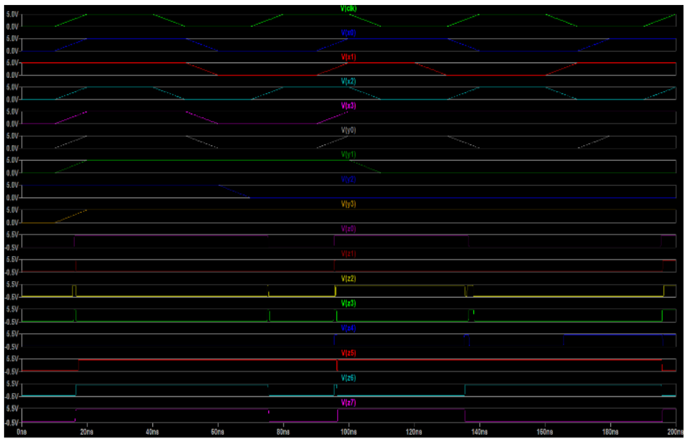

Figure 30:4X4 Pipelined Multiplier Simulation

IV. AREA, POWER, AND DELAY OPTIMIZATION

We have implemented a hierarchical design approach to ensure the necessary modularity and scalability of our single 4x4 pipelined multiplier. This design incorporates a structured layout composed of identical blocks, facilitating seamless integration into established systems.

To enhance the speed of the 4x4 multiplication process, we have introduced pipeline stages at strategic points within the design. This innovative approach is expected to yield a significant reduction in computation time.

Our primary objective remains the reduction of computation time, and we achieve this through the implementation of a folding and pipelining algorithm tailored to the single 4x4 multiplier. Importantly, this approach allows us to conserve chip area by utilizing the same 4-bit multiplier block. This design philosophy follows a structured methodology, ensuring flexibility for reconfiguration as necessary.

It's important to note that introducing pipelining inevitably introduces some latency, which is essential for achieving higher clock frequencies. Typically, increasing the depth of the pipeline enables higher clock frequencies. However, this also necessitates additional registers for intermediate data storage, thereby potentially increasing result generation latency.

Table 3:AREA, POWER, AND DELAY OPTIMIZATION

	Power	Delay
4x4 Multiplier	280Uw	22.4ps
4x4 Pipelined Multiplier	179.375uW	18.35ps
Transistor (L=2,W=10)	1.62uW	0.072ps

V. CONCLUSION

In conclusion, the design and implementation of a 4x4 bit pipelined multiplier using CMOS technology, along with the integration of half adders and full adders, represent a significant achievement in the realm of microelectronics and digital circuitry. The use of CMOS technology ensures low power consumption and compact design, making it well-suited for modern electronic devices. By breaking down the multiplication process into stages and processing multiple bits simultaneously, this approach offers substantial advantages in terms of reduced latency and improved throughput. This project has not only demonstrated the practical application of integrated circuit technology but also highlighted its critical role in modern computing systems.

VI. REFERENCES

[1] Introduction to Parallel Computing(understanding the concept of pipelining) - https://www.geeksforgeeks.org/instruction-level-parallelism/ [Accessed 4-Sep-23]

[2] What is 4×4 Array Multiplier and Its Working - https://www.elprocus.com/4x4-array-multiplier-and-its-working/ [Accessed 4-Sep-23]

[3] Half adder and full adder truth table and logic diagram - https://www.elprocus.com/half-adder-and-full-adder/ [Accessed 4-Sep-23]

[4] IEEE MICROWIND Performance Analysis of a Low-Power High-Speed Hybrid 1-bit Full Adder Circuit - https://www.youtube.com/watch?v=WxSR2Yhnqk4&t=30s [Accessed 4-Sep-23]

[5]understanding the concept of multiplier- https://www.diva-portal.org/smash/get/diva2:22435/fulltext01.pdf [6] Jan M. Rabaey, Anantha Chansrakasan, Borivoje Nikolic, "Digital Integrated Circuits", 2nd Edition, Prentice Hall Electronics and VLSI series.

[7] Jung-Yup Kang, Jean-Luc Gaudiot, "A fast and well-structured multiplier", proceedings of the EUROMICRO systems on Digital System Design (DSD'04) IEEE.

[8] Anantha P. Chandrakasan and Robert W. Brodersen,"Minimizing Power consumption in digital CMOS circuits" proceedings of the IEEE Vol. 83 No. 4 1995.

[9] Christian Piguet," LOW-POWER CMOS CIRCUITS"- https://dokumen.tips/documents/low-power-cmos-circuits-1.html?page=28

[10] C. M. Lee, H. Soukup, "An algorithm for CMOS timing and area optimization" https://ieeexplore.ieee.org/document/1052221