# Efficient 4X4 Enhanced Folded Pipeline Multiplier Based with Various Optimization

Eyab Ghifari, Osama Qutait, Tala Alsweiti, and Dr. Khader Mohammad
*Electrical and Computer Engineering Department*
*Birzeit University, Palestine*
{1190999, 1191072, 1191068, khamadawwad}@birzeit.edu

***Abstract*—In the current era of data-driven technological advancements, arithmetic operations of binary numbers have become a cornerstone, particularly in domains such as artificial intelligence and machine learning. This paper introduces a novel technique to implement a high-speed, low-power multiplier based on the Sleepy Stack Technique, Transmission Gate Logic, and optimized for transistor count. The core contribution is successfully realizing a 4 x 4-bit multiplier solely utilizing basic combinations circuits. We further accentuate the multiplier's design with high-performance and transmission gate-based 1-bit full adders, bringing forth a reliable and efficient 4-bit array multiplier. Comprehensive simulations were carried out to assess its performance, particularly focusing on power and delay metrics at operational voltages of 5V for technological nodes at 300nm. Results manifest the proposed multiplier's significant edge in efficiency and speed over traditional CMOS-based designs.**

***Index Terms*—full adder, transmission gate, array multiplier, low power, high speed, CMOS**

## I. Introduction

IN the continuously evolving digital landscape, the burgeoning demand for reliable, low-power, high-speed digital circuits is accentuated, especially with the rise of the Internet of Things (IoT) in portable and wearable devices [1-3]. Central to these innovations are multipliers, with array multipliers, characterized by their combination of full adders, half-adders, and basic logic gates, drawing significant attention. The efficacy of these multipliers is deeply rooted in the full-adder type, pivotal in multiplication tasks [1][2] [3][4]. Stepping into this domain, our newly proposed multiplier, crafted to utilize fewer transistors than its traditional counterparts, promises not only reduced propagation delay and power consumption but also a minimized layout footprint.

## II. Sleepy Stack Approach

The Sleepy Stack method splits the original transistors into two smaller transistors, reminiscent of the stack strategy. Subsequently, sleep transistors are introduced, paralleling one of the divided transistors [5].

The structure is illustrated in Figure 1. In sleep mode, the sleep transistors are deactivated, with the stacked transistors curtailing leakage current and preserving the state. Each sleep transistor, aligned parallel to one of the stacked counterparts, diminishes the path's resistance, thereby optimizing delay during active mode. Nonetheless, this approach presents a notable area overhead, as each transistor is substituted by a trio of transistors. Moreover, supplementary wiring for S and S'—representing sleep signals—adds to the complexity.

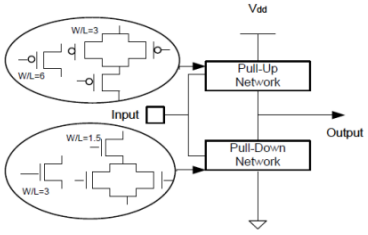



Fig. 1: Sleepy Stack Approach[5]

The technique diminishes noise and power consumption, prompting us to employ the Sleepy Stack approach within the AND and OR gates. This integration aims to devise a more streamlined 4x4 multiplier. An elementary representation of the Sleepy Stack method applied to the AND, OR and NOT gate can be seen in the provided figure.

The PMOS will have dimensions of a length of 2 and a width of 10. The NMOS, on the other hand, will have dimensions of length of 2 and width of 5.

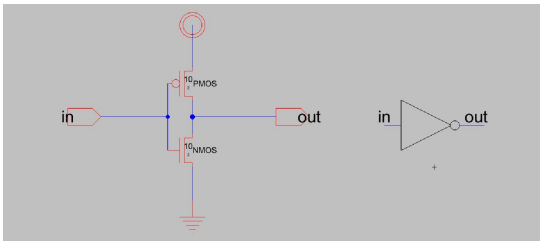



Fig. 2: design approach for NOT Gate

The AND gate is the NAND gate with its output coupled to an inverter, as demonstrated by the Boolean algebra principles in Figure 4. Due to the fact that there are two NMOSs connected in series, the NMOS's width will be expanded from 5 to 10 by multiplying the original width by a factor of 2.

Because there are two PMOSs connected in series, their widths will be multiplied by a factor of two, allowing us to create the OR gate by connecting the NOR gate we created with an inverter as seen in Figure 6.

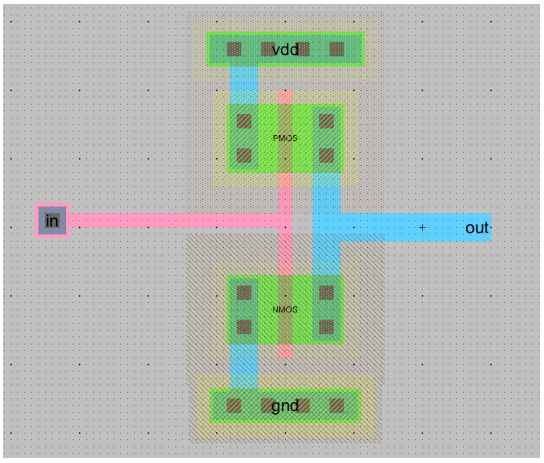



Fig. 3: design approach for NOT Gate layout

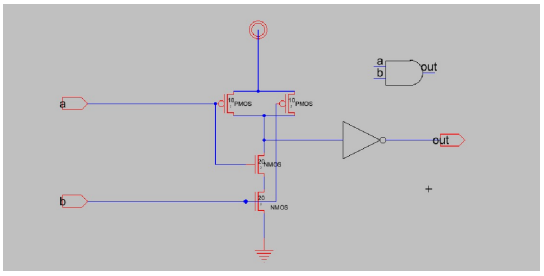



Fig. 4: design approach for AND Gate

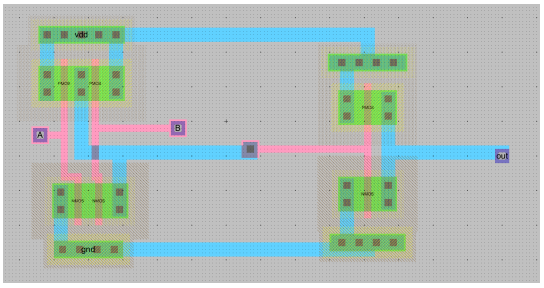



Fig. 5: design approach for AND Gate layout

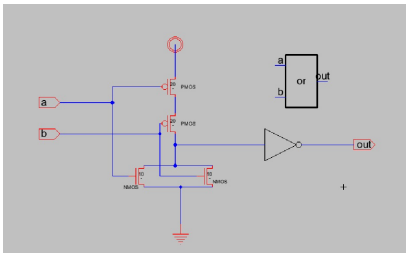



Fig. 6: design approach for OR Gate

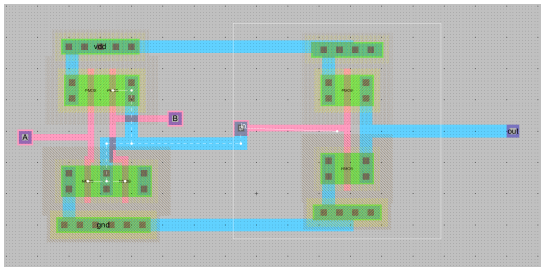
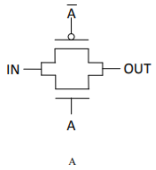



Fig. 7: design approach for OR Gate layout

## III. Transmission Gate Logic

The CMOS Transmission Gate utilizes Transmission Gate Logic to enable advanced logic functions with a minimal number of complementary transistors, effectively addressing low logic levels by employing both PMOS and NMOS transistors. This gate is characterized by its low-resistance, low-capacitance switch, resulting in a less-ratio logic gate with DC characteristics that remain unaffected by input levels. It is constructed by connecting the source to source and drain to drain terminals of NMOS and PMOS transistors, where the NMOS transistor passes a strong signal '0,' while the PMOS transistor passes a strong signal '1' to the output [6].




A

| A | IN | OUT |
|---|---|---|
| H | H | H |
| H | L | L |
| L | X(don't care) | Z(high impedance) |

B

Fig. 8: (A) Transmission Gate (B) Truth Table [6]

The accompanying schematic diagram Figure 8 is equipped with clear labels for IN and OUT, ensuring consistent circuit operation even if the labels are interchanged. This design delivers genuine bidirectional functionality without compromising input signal integrity, as depicted in the Transmission Gate symbol and truth table shown in Figure 8 [6].

We used Transmission Gate Logic (TGL) for the XOR gate in our project since it uses fewer transistors than the Sleepy Stack Approach's implementation of the XOR gate, from Figure 9 and 10 show the variations between them.

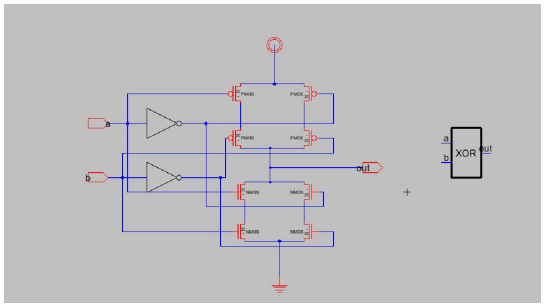



Fig. 9: XOR Gate using Sleepy Stack Approach

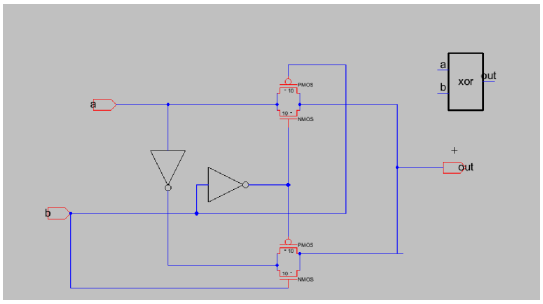



Fig. 10: XOR Gate using Transmission Gate Logic

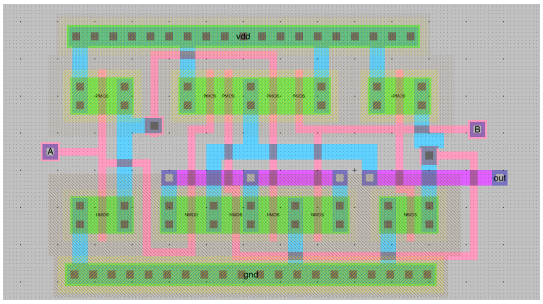



Fig. 11: design approach for XOR Gate layout

As we have already created an 8-T XOR and 4-T AND gate, a half adder may now be created with just 12 transistors. The 12-T half-adder block diagram is shown in Figure 12.

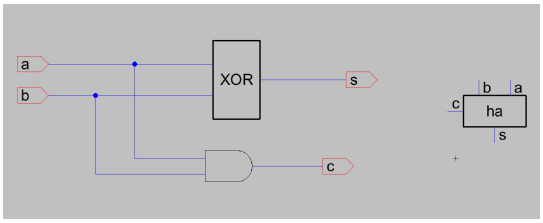



Fig. 12: design approach for Half adder

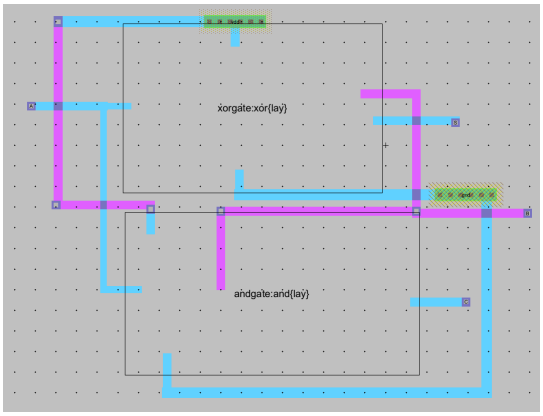



Fig. 13: design approach for Half adder layout

A combinational circuit known as a full adder performs numerous arithmetic operations on three logic bits. The block diagram of a full adder is depicted in Figure 14. It is composed of 3 blocks (a half adder, and an OR gate).

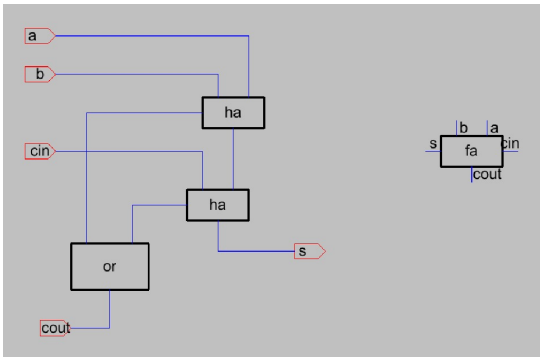



Fig. 14: design approach for 1-bit Full adder

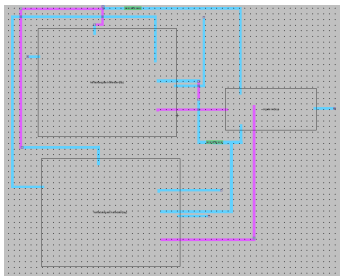


Fig. 15: design approach for Full adder layout

## IV. Array Multiplier

The array multiplier(Carry Ripple Multiplier) is a basic form of a parallel multiplier. It employs the conventional ”add and shift” method to execute multiplication tasks based on its algorithms[7]. Figure 16 showcases a 4-bit array multiplier’s structure. To multiply the multiplicand by each bit of the multiplier, the partial products generator uses the ’n’ number of ’AND’ gates. These partial products are then shifted based on their sequence, and their summation is achieved using both full and half adders. Specifically, a 4x4 array multiplier uses 4x4 AND gates to produce partial products, along with 4x(4-2) full adders and 4 half adders for summation [8] [9] [10].

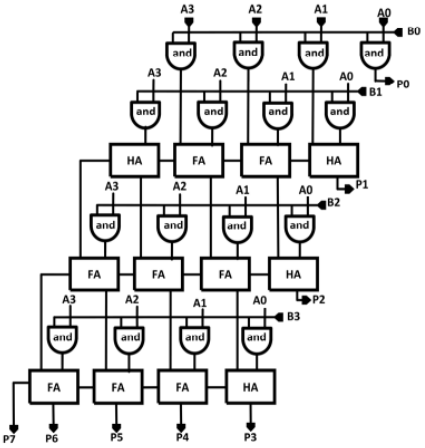



Fig. 16: design approach for Array Multiplier

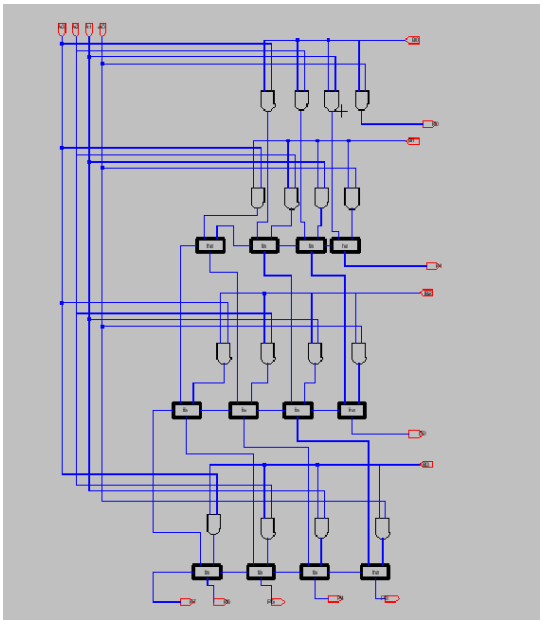


Fig. 17: our implementation for Array Multiplier

## V. SIMULATION RESULTS & ANALYSIS

In this section, we present the output of every circuit and component we constructed. We begin with the NOT gate, our first design. Please refer to Figure 2 for its schematics and Figure 3 for its layout.

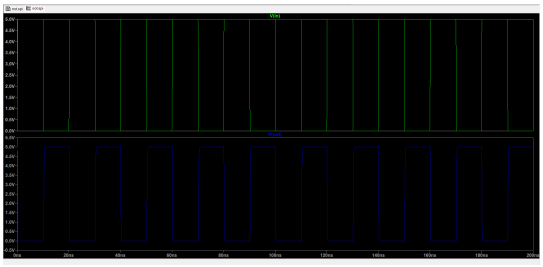


Fig. 18: NOT gate inputs and output

The results from the NOT gate, as demonstrated in Figure 18, were consistent and reliable. Following this, we designed the OR gate. The schematics for this can be found in Figure 6, and its layout is shown in Figure **??**.

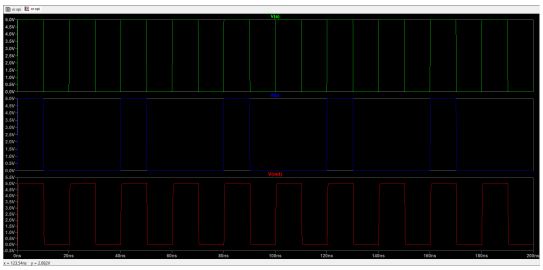


Fig. 19: OR gate inputs and output

After achieving positive outcomes with the OR gate, as shown in Figure 19, our next endeavor was the AND gate. It is illustrated in Figure 4, with its layout provided in Figure 5.

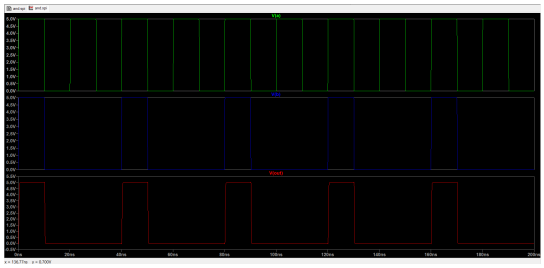


Fig. 20: AND gate inputs and output

Our progression led us to the XOR gate next, the schematics for which are depicted in Figure 10 and its layout in Figure **??**.

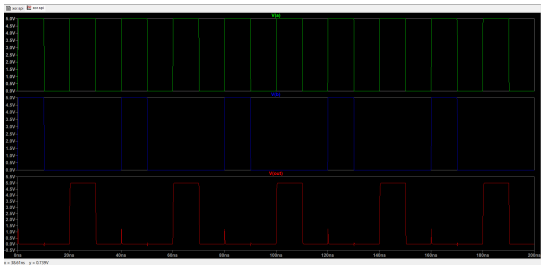


Fig. 21: XOR gate inputs and output

Building upon these foundations, we then developed the half-adder gate. Its schematics and layout are displayed in Figure 12 and Figure 13, respectively.

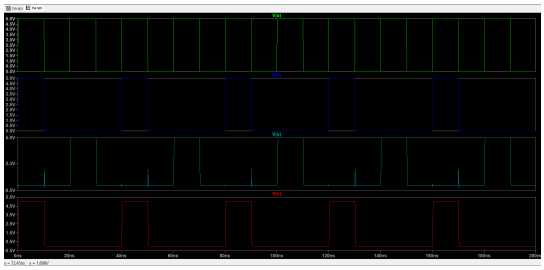


Fig. 22: half adder inputs and output

Finally, with all the prior components in place, we constructed the full adder gate. Refer to Figure 14 for its schematics and Figure 15 for its layout. Its performance, evident from Figure 23, solidified our confidence in the designs.

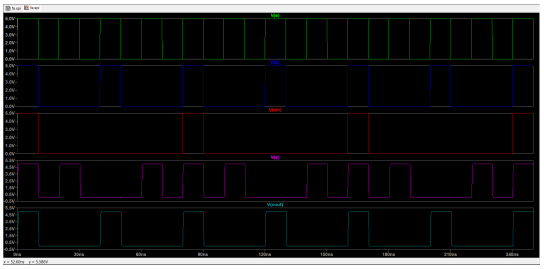


Fig. 23: full adder inputs and output

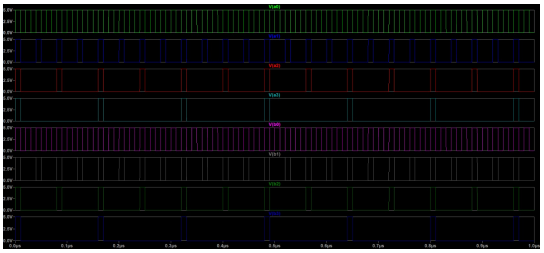


Fig. 24: 4*4 Multiplier inputs

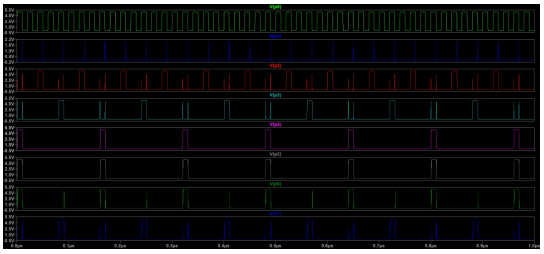


Fig. 25: 4*4 Multiplier output

## VI. Optimization

The Ripple Carry Multiplier boasts several noteworthy features. It is simple and straightforward in design, which makes it easily understandable. This simplicity is particularly beneficial in educational contexts. Furthermore, its regular structure is efficient in terms of space, requiring a smaller area compared to advanced multipliers. An added advantage is its flexibility, which permits scaling for different word lengths. Delving into the project optimization and challenges we faced: We initially accomplished our primary objective, which was the construction of 4-bit multipliers. As we ventured into optimizations, our initial focus was on decreasing wire lengths since longer wires escalate electron travel distance, introducing unwanted delays. In our design process, we adopted the lambda design rule for our layout. This approach not only yielded a more compact design than we initially envisaged but also surfaced some issues, especially when tracking bugs in the layout. To further streamline our process, we introduced a modular component method where we treated gates as building blocks. This methodology expedited error detection and rendered the design process more user-friendly and efficient for us as developers.

## VII. Conclusion

In this work, a 4-bit multiplier is presented. The CMOS technology components were the very first pieces of the project that we built. Then, utilizing the fundamental gates, we began to construct more intricate gates. Ultimately, this project deepened our understanding of the ”electric tool” IC program. We gained insights into crafting diverse gates solely using NMOS and PMOS components. Furthermore, we learned to determine the optimal dimensions for CMOS technology to achieve the most accurate outcomes. This experience also enhanced our skills in assessing power, area, and system delay. The ensuing table provides an overview of our key findings.

TABLE I: Measurements

| | |
|---|---|
| Area | 999.5 5$\mu m^2$ |
| Power | 0.189mW |
| Delay | 8ps |

## References


[1] M. Rafiee, N. Shiri, and A. Sadeghi, “High-performance 1-bit full adder with excellent driving capability for multistage structures,” *IEEE Embedded Systems Letters*, vol. 14, no. 1, pp. 47–50, 2021.

[2] Wikipedia Contributors, *Application-specific integrated circuit Wikipedia, the free encyclopedia*, [Online; September 3, 2023]. [Online]. Available: https://en.wikipedia.org/wiki/Application-specific_integrated_circuit.

[3] S. M. Kang and Y. Leblebici, *CMOS digital integrated circuits*. MacGraw-Hill New York, 2003.

[4] K. Barr, *ASIC Design in the Silicon Sandbox: A Complete Guide to Building Mixed-signal Integrated Circuits*. New York: McGraw-Hill, 2007, ISBN: 978-0-07-148161-8.

[5] A. Venkatesh, N. Rathan, and R. Saranya, "Design of power efficient 4x4 multiplier based on various power optimizing techniques," *Asian Journal of Applied Science and Technology (AJAST)*, vol. 1, no. 2, pp. 115–119, 2017.

[6] A. Dixit, S. Khandelwal, and S. Akashe, "Power optimization of 8: 1 mux using transmission gate logic (tgl) with power gating technique," *International Journal of Computer Applications*, vol. 99, no. 5, pp. 37–42, 2014.

[7] S. Vaidya and D. Dandekar, "Delay-power performance comparison of multipliers in vlsi circuit design," *International Journal of Computer Networks & Communications (IJCNC)*, vol. 2, no. 4, pp. 47–56, 2010.

[8] O. A. Albadry, M. M. El-Bendary, F. Z. Amer, and S. M. Singy, "Design of area efficient and low power 4-bit multiplier based on full-swing gdi technique," in *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, IEEE, 2019, pp. 328–333.

[9] D. Praneeth, R. Varshitha, S. S. Prashanth, V. Mahesh, L. R. Thoutam, and J. Ajayan, "Design and development of reliable low power high-speed 4-bit array multiplier using high-performance 1-bit full adders," in *2022 IEEE International Conference on Nanoelectronics, Nanophotonics, Nanomaterials, Nanobioscience & Nanotechnology (5NANO)*, IEEE, 2022, pp. 1–6.

[10] *Multiplication in fpgas*, Last modified: March 02, 1999; Retrieved September 3, 2023, 1999. [Online]. Available: http://www.massmind.org/techref/logic/multiply.htm?ke%20y=Memory%20Methods.