

8-bit Wallace Tree Multiplier Using Full-Adder Inversion Property

William Cunningham

School of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana
wrcunnin@purdue.edu

Akshath Raghav Ravikiran

School of Electrical and Computer Engineering
Purdue University
West Lafayette, Indiana
araviki@purdue.edu

Abstract—A Wallace Tree Multiplier (WTM) accelerates integer multiplication by reducing partial-products in parallel using carry-save-adder (CSA) chains, followed by a final ripple-carry-adder (RCA) stage. This project implements an 8-bit WTM at the transistor level, with full-adder cells capable of producing inverted sum-carry bits. Complete schematic, layout, DRC/LVS sign-offs and verification were performed using Cadence Virtuoso with the open-source GPDK45 library. Finally, an energy, latency and area analysis was completed to verify that the design met project specifications.

I. INTRODUCTION

Binary multiplication can be expressed as the iterative summation of intermediate products generated between the multiplier and multiplicand's bits. An N-bit multiplication produces N^2 partial products, whose weighted sum forms the $2N$ -bit product. The overall process can be broken down into (1) Partial Product Generation (PPG), (2) Partial Product Summation, and (3) final Carry-Sum Addition. A naive implementation would accumulate partial products sequentially, resulting in long signal propagation paths.

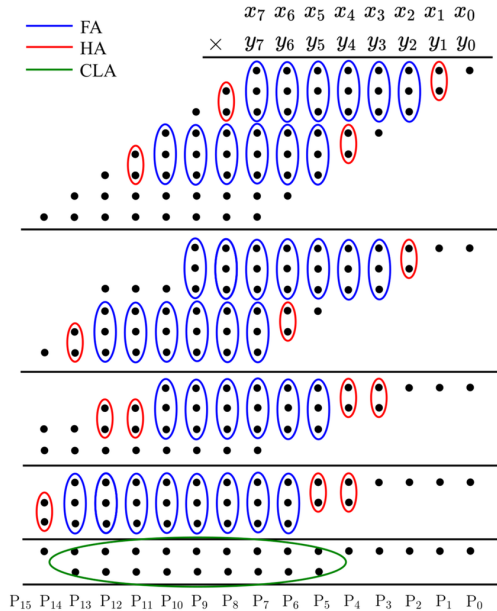


Fig. 1: 8-bit Wallace Tree Diagram [1]

The Wallace Tree Multiplier (WTM), developed by Chris Wallace, decomposes the summation stage and identifies an efficient way to parallelize certain computations. With the help of Full Adder (FA) and Half Adder (HA) units, the PPG outputs are reduced to only two rows for the final stage to handle. Each reduction level computes similar latency operations, reducing the critical path depth. These $\log_2(N)$ levels form the Partial Product Reduction (PPR) stage.

$$FA(A, B, C_{in}) \rightarrow (S, C_{out}) \quad (1)$$

$$FA(\overline{A}, \overline{B}, \overline{C_{in}}) \rightarrow (\overline{S}, \overline{C_{out}}) \quad (2)$$

$$FAInv(A, B, C_{in}) \rightarrow (\overline{S}, \overline{C_{out}}) \quad (3)$$

$$FAInv(\overline{A}, \overline{B}, \overline{C_{in}}) \rightarrow (S, C_{out}) \quad (4)$$

The Inversion Property [2] defines passing inverted inputs to a FA also inverts the outputs (1)(2). This property is characteristic of CMOS-based logic, given that symmetry is ensured in the output. Consequently, an alternative FA design can be used, such that passing inverted inputs produces non-inverted outputs and non-inverted inputs produces inverted outputs. (3)(4). Exploiting this feature allows for saving inverters within every FA unit, but requires re-architecting any existing logic design.

Although the WTM design generally improves overall logic propagation latency, its performance is subject to layout efficiency. The goal of this project was to implement an 8-bit WTM and evaluate post-layout energy, delay, and area trade-offs. The design needs to pass DRC and LVS sign-offs and be verified with inputs transitioning at 50 ps rise/fall times under a 2 fF output capacitance. The final deliverable needs to guarantee four items:

- worse-case propagation delay no more than 2.5 ns,
- energy consumption no more than 850 fJ (with the same inputs for worst delay case),
- correct functionality, and
- minimal area utilization.

II. DESIGN METHODOLOGY

The core development strategy was to identify and group common logic units into cells that could be laid out hierarchically. Each WTM stage can be decomposed into chains

of inverters, NAND gates and FAs. These units were first designed and optimized at the transistor level, then re-used across higher-level blocks. This approach allowed for optimizing the local circuitry and maximum re-use.

A. Primitive Cells

Appropriate transistor sizes in the Pull-Up (PUN) and Pull-Down Networks (PDN) for these standard cells were determined analytically to maintain near-equal rise and fall delays ($\tau_{rise} \approx \tau_{fall}$). This assists in balancing propagation delay and dynamic power. From the GPDK45nm library, we chose a 2 : 1 PMOS-to-NMOS width ratio.

1) *Cell 1 – Inverter*: The fundamental building block of many logic circuits is the CMOS Inverter. It is used to invert a signal and also acts as a buffer when dealing with long interconnects with high-fanout. This cell takes up an area of $0.618\mu\text{m}^2$.

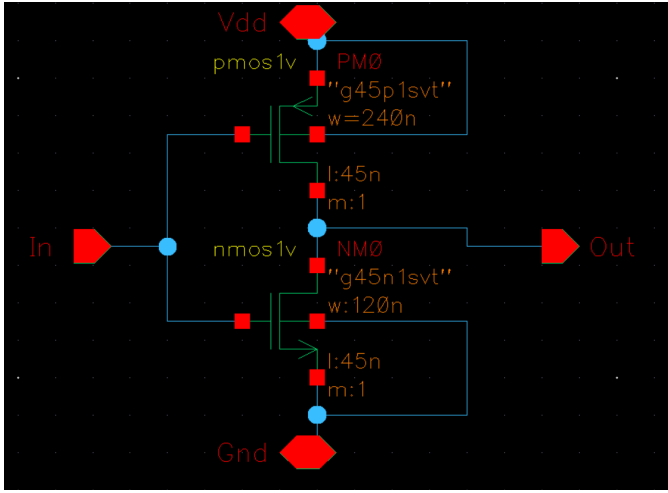


Fig. 2: Inverter Schematic

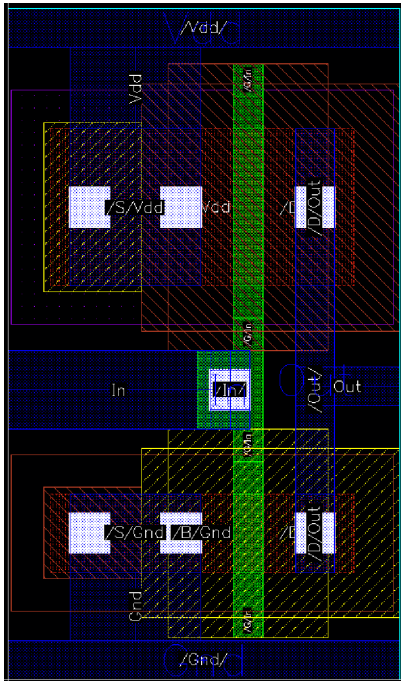


Fig. 3: Inverter Layout

2) *Cell 2 – NAND*: The NAND gate forms the core component of the Partial Product Generation stage. The partial products are typically generated using an array of AND gates, a combination of Inverter and NAND cells. In our design, we forego the inverter and pass the inverted partial products into the PPR stage. Each NAND gate takes up an area of $1.107\mu\text{m}^2$.

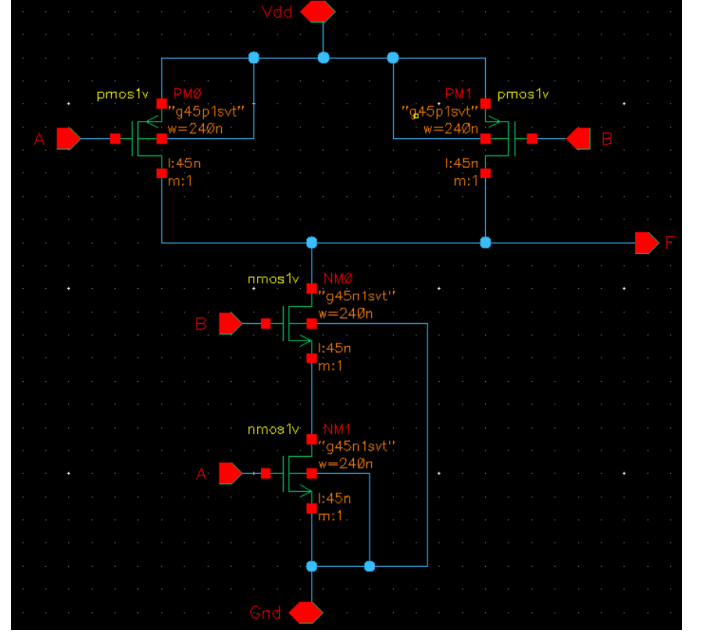


Fig. 4: NAND Schematic

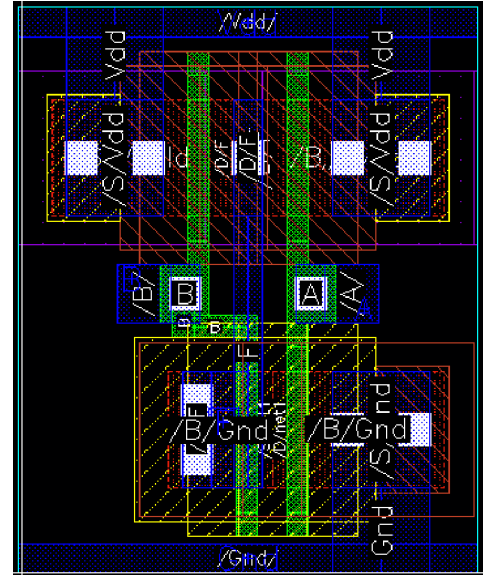


Fig. 5: NAND Layout

3) *Cell 3 – Inverted Full Adders*: Non-Inverting CMOS Full Adders require 28 transistors. By utilizing the Inversion Property, we implemented the "Mirror Adder" topology (FAInv) [3] which uses only 24 transistors. By avoiding the output inverters, these cells naturally produce inverted outputs. By calculating transistor widths to optimize the logical effort,

the delay from C_{in} to C_{out} was minimized at an expense of effort to S . Each inverting Full Adder takes up an area of $16.357\mu m^2$.

With this design choice, we had to alternate reduction stages between "Positive" and "Negative" logic intermediates, eliminating the need for intermediate inverters. It also required us to exclude HA's from the WTM, as they do not satisfy the inversion property. Instead, we substituted HA with FAInv, setting one input to VDD if the other two inputs were inverted and GND if they were not.

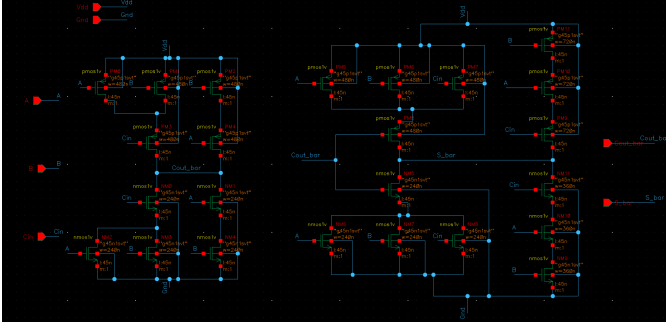


Fig. 6: Inverted Full Adder Schematic

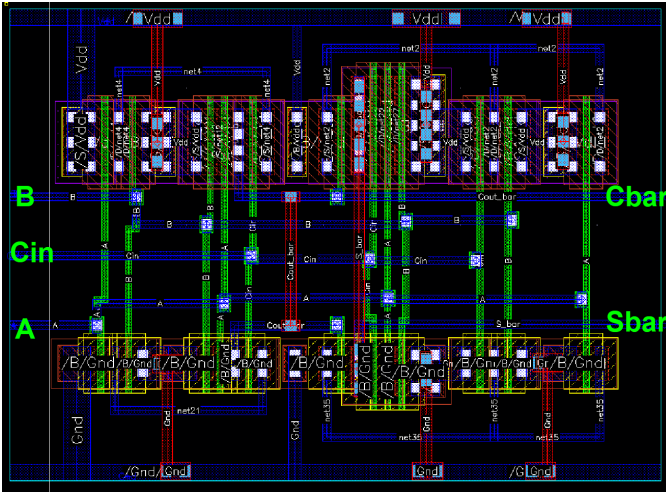


Fig. 7: Inverted Full Adder Layout

4) *Cell 4 – Inverted Full Adder Chains:* To simplify routing all the individual FAInv cells in the reduction stages, we chose to identify the smallest set of repeating chains and re-use them. Each such unit takes up an area of $120.16\mu m^2$.

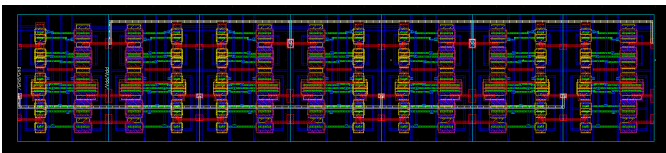


Fig. 8: 7-FAInv-Chain Layout

B. Partial Product Generation

This stage generates the Partial Product (PP) matrix. For an 8-bit multiplier, this results in 64-PP-bits that are grouped into 8-bit buses. The generation logic consists of NAND cells to

generate the inverted partial products, which is fed in to the PPR stage.

Even here, we modularize and exploit the repeating pattern of 8-bit packing to make the Partial Product Row. Then, we duplicate it 8 times – to a combined an area of $512.76\mu m^2$ – to complete the PPG.

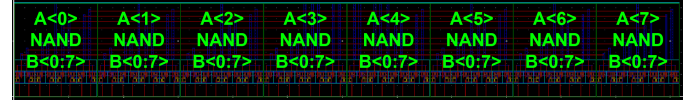


Fig. 9: Partial Product Generator

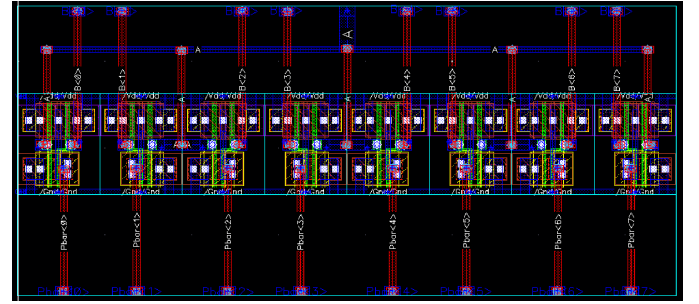


Fig. 10: Partial Product Row

C. Partial Product Reduction

This design makes use of 53 FAInv cells and 27 inverters to implement the following reduction stages. FAInv cells were used to replicate the behavior of HA units as mentioned earlier.

Stages 1 and 2 bear the largest overall logic effort, and are visualized as two columns of FAInv chains. These stages also generate the $M < 4 : 0 >$ final output bits.

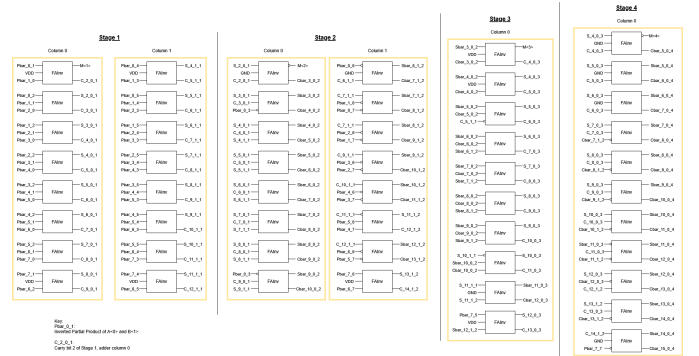


Fig. 11: Reduction Overview w/ FAInv Cells

1) Reduction Stage 1:

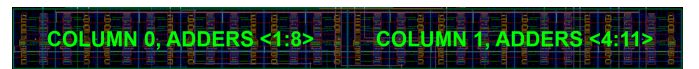


Fig. 12: Stage 1 Layout

2) Reduction Stage 2:

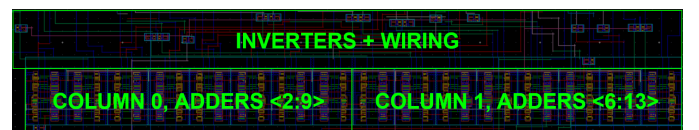


Fig. 13: Stage 2 Layout

3) Reduction Stage 3:



Fig. 14: Stage 3 Layout

4) Reduction Stage 4:



Fig. 15: Stage 4 Layout

D. Carry-Sum Addition

This final stage [Fig. 16] is an 11-bit Ripple Carry Adder (RCA) that sums the remaining outputs ($M < 15 : 5 >$) from the reduction tree. While other units were laid out horizontally and produced outputs vertically, the RCA propagates outputs horizontally to minimize layout. There is a potential reduction of inverter usage obtainable by inverting the C_{out} of the adders in the chain instead of inverting the other two inputs and the output, a reduction of 15 inverters to 10 inverters. However, this puts all 10 inverters on the critical path. We determined this may negatively impact the worst-case delay and decided to opt for more inverters.

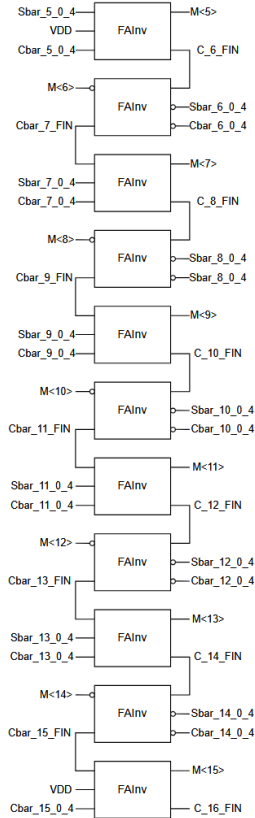


Fig. 16: Final Stage Diagram

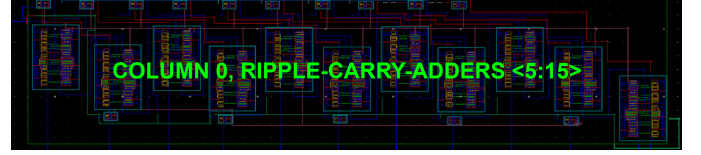


Fig. 17: Final Stage Layout

E. Top Level Integration

Finally, the top-level floorplan integrates the PPG, four PPR, and the final RCA. This implementation's P&R boundary has a width of $61.475\mu m$ and height of $53.77\mu m$, resulting in an area of $3305.51\mu m^2$.

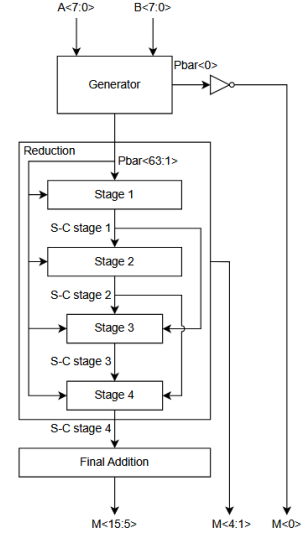


Fig. 18: Top Level Diagram

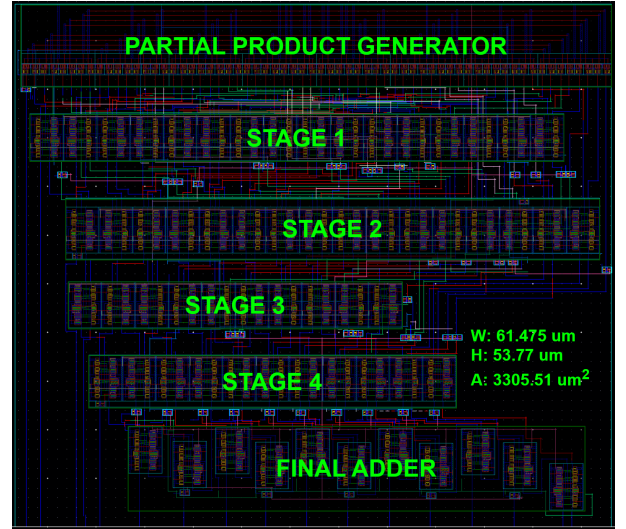


Fig. 19: Top Level Layout

F. Post-Layout Verification

1) *DRC/LVS Sign-Offs*: Following physical design, the layout was verified to pass manufacturing rule-checks (metal spacing, interconnect width, etc.) and have the extracted transistor netlist to match the schematic design.

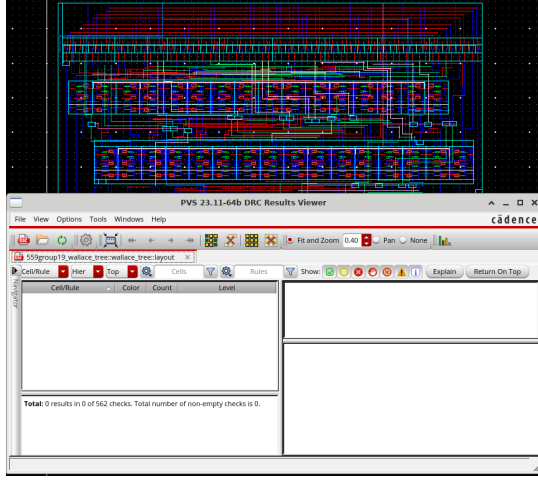


Fig. 20: DRC Passing Confirmation

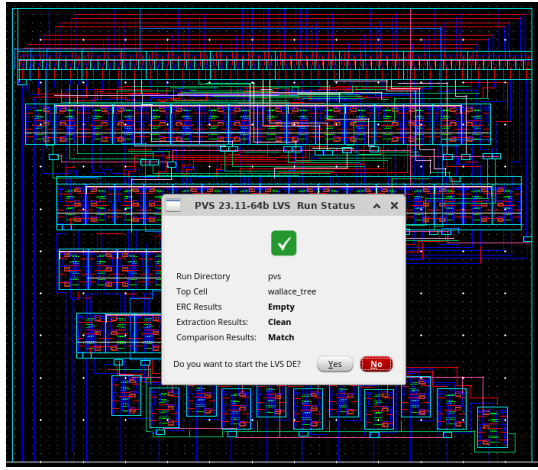


Fig. 21: LVS Passing Confirmation

2) *Parasitic Extraction*: Finally, the RC parasitics (interconnect resistance and capacitance) were extracted to generate a netlist for final functional verification.

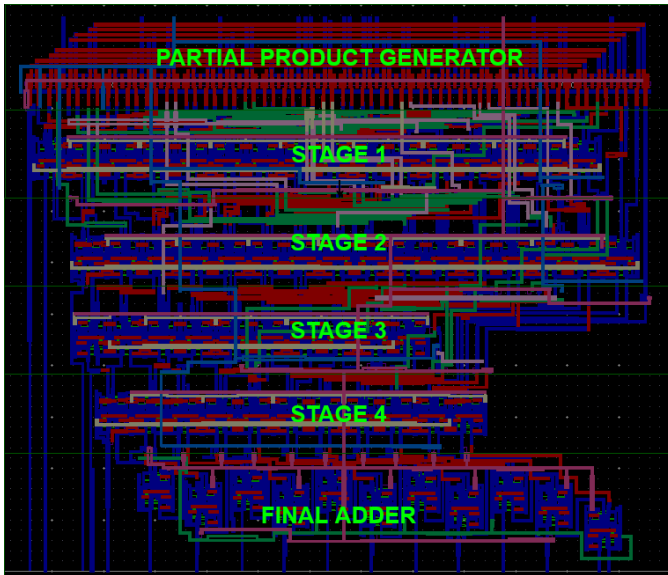


Fig. 22: Post-Parasitic-Extraction Layout View

G. Functional Verification

To guarantee correct operation of this WTM implementation, a testbench schematic – with 2 fF output capacitances on each output – was created. The following $A < 7 : 0 >$ and $B < 7 : 0 >$ input vectors were chosen to test a few scenarios.

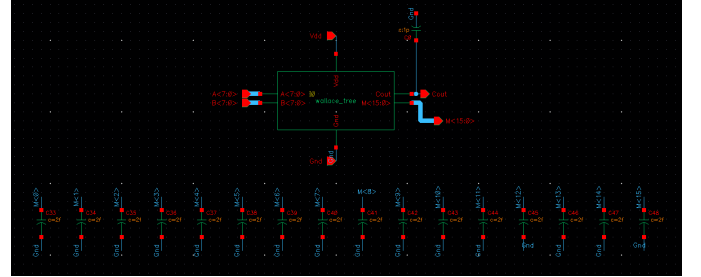


Fig. 23: Post-Parasitic-Extraction Testbench

- **Test 1 – Random Vector:** $163 \times 70 = 11410$ [Fig. 24].
- **Test 2 – Random Vector:** $194 \times 32 = 6208$ [Fig. 24].
- **Test 3 – Worst Case Delay:** $129 \times 255 = 32895$. A ($129 = 0x81$) and B ($255 = 0xFF$) were chosen such that $M < 15 >$ would be computed from the final RCA instead of the partial product reduction. The product 32895 is $0b100000000111111$ in binary [Fig. 25].
- **Test 4 – Worst Case Energy:** $255 \times 255 = 65025$. Both A and B vectors transition from $0x00$ to $0xFF$ and back. This forces maximum switching activity and forces transitions on all the primitive cells [Fig. 26].
- **Test 5 – Swapping A and B :** $85 \times 170 = 170 \times 85 = 14450$. A and B swap their values to show M remains at the same product [Fig. 27].

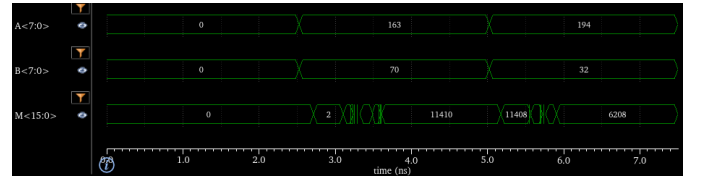


Fig. 24: Test 1 & 2 Digital Waveforms

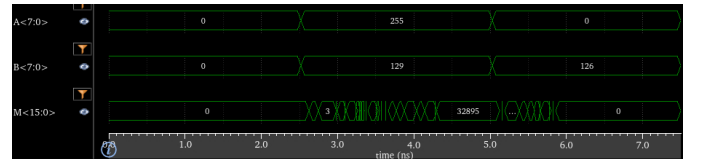


Fig. 25: Test 3 Digital Waveforms

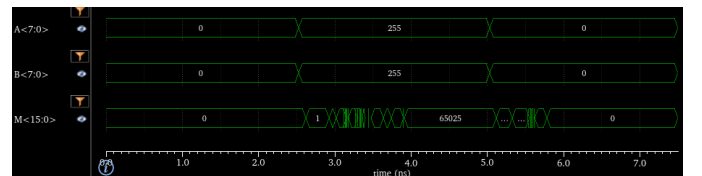


Fig. 26: Test 4 Digital Waveforms

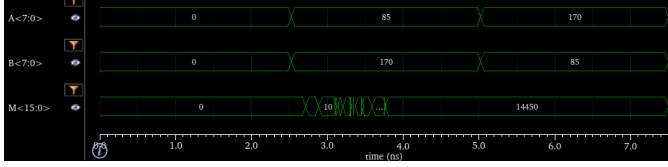


Fig. 27: Test 5 Digital Waveforms

III. RESULTS

When testing our WTM implementation, voltages of 1V and 1.1V were used to measure the worst-case delay and energy consumption.

A. Delay

Test 3 covers the worst-case latency. In Table I, we summarize our results clearly. With all delay numbers residing comfortably under 2.5ns, the implementation passes the delay constraint.

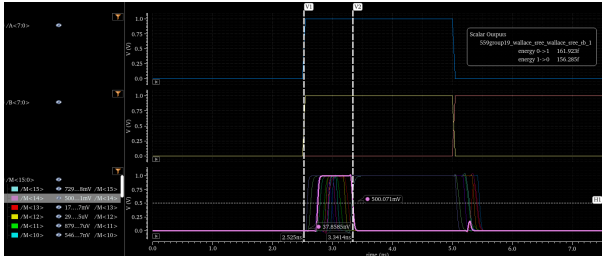


Fig. 28: Worst Case Delay (1V, Schematic)

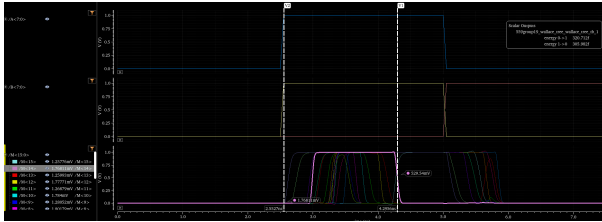


Fig. 29: Worst Case Delay (1V, PEX-Layout)

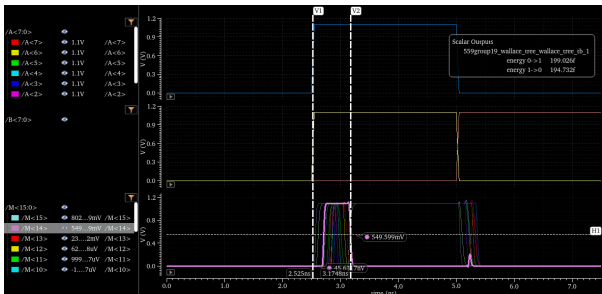


Fig. 30: Worst Case Delay (1.1V, Schematic)

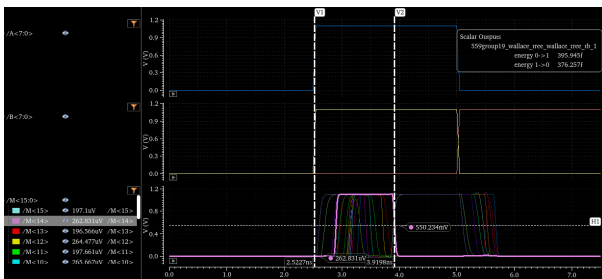


Fig. 31: Worst Case Delay (1.1V, PEX-Layout)

Voltage (V)	Schematic Delay (ns)	Layout Delay (ns)	Delay Delta (ns)
1.0	0.8164	1.742	+0.9256
1.1	0.6498	1.3948	+0.7450

TABLE I: Worst-Case Delay

B. Energy

The worst-case energy consumption resulted from both inputs A and B transitioning from 0x00 to 0xFF for the schematic [Fig. 34], and for the layout 0xFF to 0x00 [Fig. 35]. In Table II, we summarize our results values clearly. When increasing the supply voltage, we notice an increase in all energy consumption values across the board.

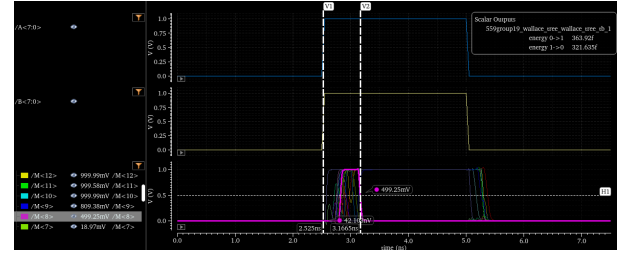


Fig. 32: Worst Case Energy (1V, Schematic)

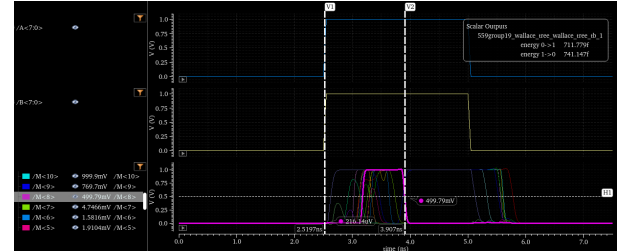


Fig. 33: Worst Case Energy (1V, PEX-Layout)

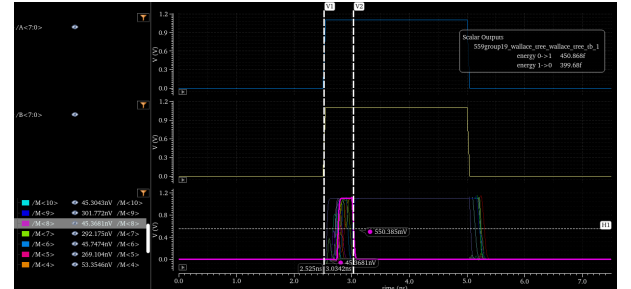


Fig. 34: Worst Case Energy (1.1V, Schematic)

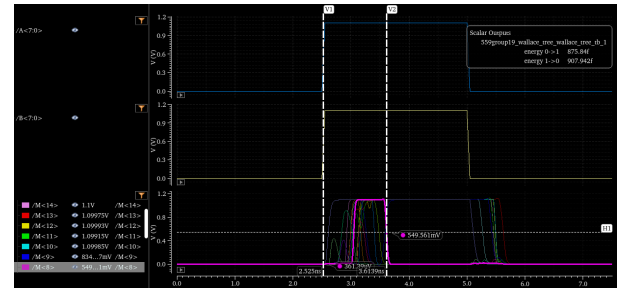


Fig. 35: Worst Case Energy (1.1V, PEX-Layout)

However, for the sake of this project, the constraints were set with regard to the consumption during the *worst-delay* analysis. Table III summarizes the values extracted from the simulations explained in section III-A. With every configuration's energy consumption being under 850 fJ, the custom WTM implementation passes the second project requirement as well.

Voltage (V)	Schematic Energy (fJ)	Layout Energy (fJ)	Energy Delta (fJ)
1.0	363.920	741.147	+377.227
1.1	450.868	907.942	+457.074

TABLE II: Worst-Case Energy Consumption

Voltage (V)	Schematic Energy (fJ)	Layout Energy (fJ)	Energy Delta (fJ)
1.0	161.923	320.712	+158.789
1.1	199.026	395.945	+196.919

TABLE III: Worst-Case Delay's Energy Consumption

C. Area Savings & Transistor Count

The conventional CMOS FA uses 28 transistors while similar AND-gates use 6 transistors. With these numbers, the standard WTM typically requires a total of 2176 transistors. Even though this design uses the same number of FA and AND-gate units, the use of the Inversion Property reduces the total inverter usage to 1878 transistors. Inverters had to be added specifically between a few cells, but the general exploitation of alternating positive and negative logic levels helped avoid 298 transistors.

The Non-Inverting design uses 192 inverters while the Inverting one uses only 43 (39 inverters and 2 buffers). This results in a saving of 149 inverters. Using the area numbers defined in section II-A, this design optimizes area by $92.082\mu\text{m}^2$. As mentioned in section II-E, the final area of the design was $3305.51\mu\text{m}^2$, which we believe is minimized sufficiently.

	Num. Inst.	Transistors/Inst.	Total Transistors
FAInv	64	24	1536
Inverter	43	2	86
NAND	64	4	256
Inverted Total			1878
FA	64	28	1792
AND	64	6	384
Non-Inverted Total			2176

TABLE IV: Counts for Inverting and Non-inverting designs

IV. CONCLUSION

This project successfully implemented an 8-bit Wallace Tree Multitplier using the inversion property to minimize total area. Post-layout validation using the GPD45nm process library verified that the design met all core requirements with (1) a worst-case delay of 1.742 ns at 1V and 1.3948 ns at 1.1V, and (2) an energy consumption of 320.712 fJ at 1V and 395.945 fJ at 1.1V for the worst-case delay. The exploitation of the inversion property also reduced the overall transistor count

by 298, leaving the final chip area measuring at $3305.51\mu\text{m}^2$. Parasitic extraction resulted in near doubling of the worst-case delay and energy consumption results, but the metrics remained well within the specifications, having a positive slack of 0.758 ns & 529.288 fJ at 1V and 1.1052 ns & 454.055 fJ at 1.1V.

V. CONTRIBUTIONS

Akshath took primary responsibility of the transistor-level schematic design, test-benching and report writing. William held primary responsibility over floor-planning, layout generation and creation of report diagrams. Both authors contributed to individual portions of layout and schematic, as well as the final functional verification.

REFERENCES

- [1] Jiménez, Abimael & Muñoz, Antonio. (2025). Very-Large-Scale Integration (VLSI) Implementation and Performance Comparison of Multiplier Topologies for Fixed- and Floating-Point Numbers. Applied Sciences. 15. 4621. 10.3390/app15094621.
- [2] (No date) COMP103- L13 Adder Design.1 Comp 103 lecture 13 adder design. Available at: [http://www.cs.tufts.edu/comp/103/notes/Lecture13\(Adders\).pdf](http://www.cs.tufts.edu/comp/103/notes/Lecture13(Adders).pdf) (Accessed: 13 December 2025).
- [3] N. H. Weste and D. M. Harris, CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. Boston, MA: Addison-Wesley, 2011.

VI. APPENDIX

WTM Diagrams (draw.io)