**Efficient Logic Design Using Spin Transfer Torque Memory Technology & Lookup Tables**

**Brandon Leung, Yu Ting Huang, Fernando Lorenzo, Sergio Rodriguez, Janine Young**

**Advisor: Professor Mahmoodi**

**Student Mentor: Aliyar Attaran**

**Table of Contents**

[Abstract](#_dg8lkpolnhuz)

[1 Introduction](#_rgjem4zf5lgi)

[2 Background](#_1qcaf9o36j8b)

[2.1 STT-LUT Implementation](#_tphqbg2rvux2)

[2.2 Read and Write Operations](#_m7mglydbfpoa)

[2.3 Application of STT-LUT](#_m2l9jdmma4hw)

[3 Proposed Design Methodology: Hybrid Use of STT-LUT and Custom CMOS](#_40k08jrb531x)

[3.1 Hybrid Use of STT-LUT and Custom CMOS](#_5vat7mt5xh70)

[3.2 Case Study: 4-bit Adder](#_5pjutn2rul1)

[3.3 Single Gate Mapping](#_c9xhsgio0n0t)

[3.4 Multiple Gate Mapping](#_e14v0gj832b0)

[4 Results](#_lzqr5vtrymfd)

[4.1 Single Gate Mapping Results](#_dcv9edsx1uxq)

[4.2 Multiple Gate Mapping Results](#_u2q2cbjckill)

[5 Conclusion](#_24ia0h47am9s)

[Acknowledgements](#_mbahyfnnxfj3)

[Bibliography](#_oi7rfg4nvwt3)

**Table of Figures**

Figure 1. Perpendicular MTJ: (a) Parallel (low resistance) and anti-parallel (high resistance) states, (b) R-I characteristics

Figure 2. Physics behind MTJ: (a) Antiparallel to Parallel, (b) Parallel to Antiparallel 1

Figure 3. Fabric of static STT-LUT

Figure 4. XNOR gate devised by programming the content of LUT 1

Figure 5. 4-bit carry look-ahead adder gate schematic.

Figure 6. Various LUT mappings.

Figure 7. Gates with no delay overhead.

Figure 8. Delay at minimum period (individual mapping). The black bar represents the all custom CMOS. The red bars represent gates along the critical path. The multicolored bars represent the non-critical gates.

Figure 9. Active power at a 1 ns period (individual mapping)

Figure 10. Area of 4-bit adder (individual mapping)

Figure 11. PDP at minimum period (individual mapping)

Figure 12. Two possible sets of independently selected gates.

Figure 13. Two sets of dependently selected gates.

Figure 14. A set of parametric-aware dependent selected gates.

Figure 15. Delay at minimum period (multiple gate mapping).

Figure 16. Active Power at 1ns Period (multiple Gates)

Figure 17. Area of 4-bit adder (multiple gate mapping).

Figure 18. PDP at minimum period (multiple gate mapping).

**Abstract**

Spin Transfer Torque Random Access Memory (STT-RAM) is a promising technology for information storage in the form of magnetic fields compared to existing charge-based memories like static random-access memory (SRAM), dynamic random-access memory (DRAM) and flash. Charge-based memory is notorious for its volatility and its constant use of power. STT-RAM technology is advantageous for its non-volatility, complementary metal-oxide -semiconductor (CMOS) compatibility, and programmability and hardware security through lookup tables (LUTs). The contents of LUTs may be programmed and then simulated in a software called Synopsys to implement any arbitrary logic function. However, due to circuit complexity, LUTs occupy significantly more area than logic gates, so the amount of delay and power consumption will inevitably increase. The 4-bit adder 74283 benchmark circuit has been chosen as the case study in this project. The purpose of this paper is to maximize the number of LUTs implemented while minimizing delay and power consumption by replacing specific logic gates with LUTs in this case study. First, all logic gates of a 4-bit adder were individually mapped with LUTs and we measured the delay, active power, standby power, power delay product (PDP) and area. From the data extracted, we found that certain gates produced higher delay and power consumption but the majority of mapped gates do not have delay penalties. Next, three mapping optimization methods — dependent, independent and parametric-aware selection — were used to map certain gates with their respective LUTs at once. The outcome of these methods provides a tremendous insight on which set of gates selections are best to map in order to increase hardware reconfigurability and security which is the trend of STT-LUT in LUT applications. Using the optimal pathway, STT-LUT implementation can achieve maximum efficiency with a mix of complex logic gates, which means faster processing of reconfigurable hardware such as Field Programmable Gate Arrays (FPGA). The outcome of this implementation provides less power leakage and usage, and ease of reconfigurable non-volatile hardware.

# **1 Introduction**

Integrated Circuit (IC) design companies are becoming more vulnerable to increasing threats to their designs, which are at risk of being reverse engineered and be illegally reproduced. Spin Transfer Torque (STT) is a promising technology that recently gained prominence in the IC industry to increase the security of their chips by storing binary information in the form of magnetic fields rather than utilizing existing electron charge-based memories like SRAM, DRAM and flash. STT technology is advantageous for its non-volatility, reconfigurability, scalability, CMOS compatibility, and low leakage current, while offering enhanced hardware security, configurability and performance efficiency 1.

The three types of memories commonly used today are SRAM, DRAM, and Flash. SRAM is Static Access Random Memory, and can retain data in the form of binary bits of 0s and 1s as long as power is constantly supplied. It is used as a computer’s cache memory that programs instructions into the computer; these instructions need be accessed quickly by the processor, therefore the fast read and write operation required. SRAM has great read and write speeds but is stored in a large cell. In Dynamic Random Access Memory, data is stored in a capacitor within an Integrated Circuit. The capacitors can either be charged or discharged, which are used to determine binary units. However, DRAM requires a constant flow of power because the capacitor slowly discharges when it is not in use. This is known as power leakage. Flash memory, primarily in USB drives, is a non-volatile memory where information can be stored without requiring any power. Its ability to read and write information is slower than that of SRAM and DRAM, but allows information to be reprogrammed, written, or erased 1.

STT-RAM has the advantages of SRAM, DRAM, and Flash combined without sacrificing their key features. STT-RAM has the best attributes of these 3 memories. It has moderately fast read/write speeds while keeping CMOS compatibility for ease of integrating existing memory devices. It also offers nonvolatility which means that data is not lost after power is cut off and uses less voltage that does not require memory to be refreshed. This means more power efficiency, all with a small cell density that can make room for more memory processing per area. STT-RAM’s predecessor would be the SRAM, with a key difference in its volatility and larger cell size.

# **2 Background**

## **2.1 STT-LUT Implementation**

STT technology offers great advantages such as hardware reconfigurability and security when used in conjunction with lookup tables. For STT-LUT implementation, Magnetic Tunnel Junctions (MTJ) are utilized to store information that is used to determine the specific logic function of STT-LUT. Spin Transfer Torque is a phenomenon that occurs inside the MTJs and is used to read, write and retain information in the form of magnetic orientation 1,4.As shown in Figure 1(a) , MTJs consist of two ferromagnetic layers and an oxide barrier in between them. One layer, referred to as the fixed or pinned layer, has a fixed magnetic orientation. For the other layer, referred to as the free layer, the magnetic orientation can be altered by changing the direction of the current passing through it. Depending on the layers’ orientations, MTJs could have a state of logic “0” or a state of logic “1” (see MTJ Hysteresis in Figure 1(b)). If the magnetic orientations of the two layers are in a parallel state with respect to each other (low resistance), then the MTJ holds a logic state of 0. Otherwise, if the magnetic orientations are in an antiparallel state (high resistance), the MTJ holds a logic state of 19.

## 

## **2.2 Read and Write Operations**

In order to know the state of a MTJ or to switch the state of the MTJ, a current has to be applied. The applied current is different depending on the function that needs to be accomplished. If the desired function is to know the content of the cells (read operation), a relatively small current has to be applied to the MTJ. This current has to be smaller than the critical write current, which is the minimum current required to switch the state of an MTJ. The current will sense the resistance of the MTJ and the cell’s status as a state of 0 (low resistance) or a state of 1 (high resistance) will be read (see Figure 1 (b)). On the other hand, if the desired function is to switch the state of the MTJ (write operation), then a current equal to or greater than the critical write current has to be applied. 1

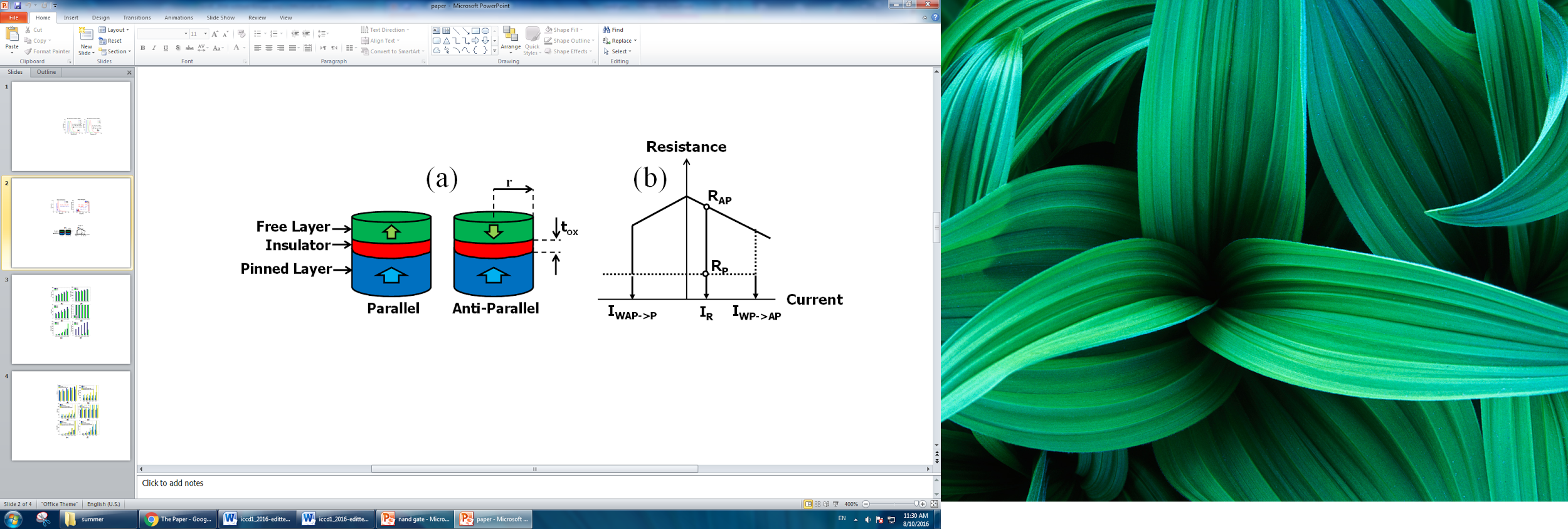
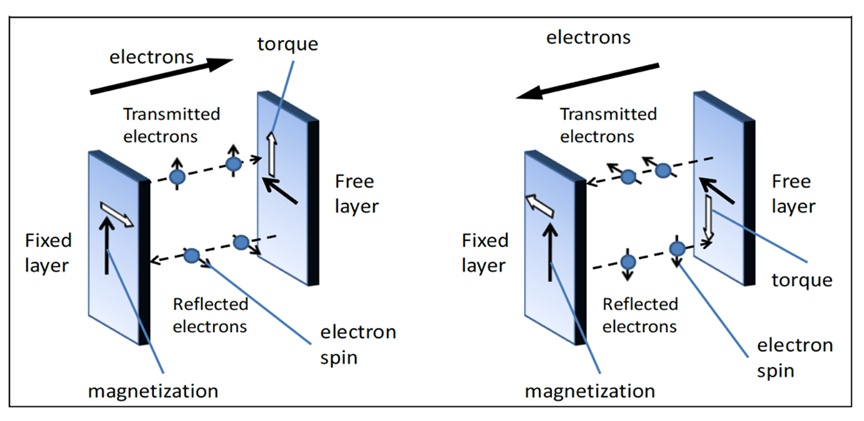


Figure 1. Perpendicular MTJ: (a) Parallel (low resistance) and anti-parallel (high resistance) states, (b) R-I characteristics

The write currents of an MTJ are different for logic 0 and logic 1. As shown in Figure 2a, to go from an antiparallel state to a parallel state, or write a 0 logic state to an MTJ, a current greater than the critical write current has to be applied. This current has to go from the free layer to the fixed layer, which means that electrons will flow from the fixed layer to the free layer. These electrons will apply a torque in the direction of the fixed magnetic field due to their spin polarization. Some electrons will be reflected back towards the fixed layer; however, the amount of reflected electrons is minimal, so their spin will not have a detrimental effect on the orientation of the fixed layer. As a consequence of the torque applied by the spin-polarized electrons on the free layer, the magnetic orientation of the free layer will take the magnetic orientation of the fixed layer, effectively changing the state of the MTJ to a parallel state of low resistance. In contrast, as shown in Figure 2b, to go from a parallel state to an antiparallel state—which is the same thing as writing a logic 1—a current has to be applied from the fixed layer to the free layer. In this case, electrons will flow from the free layer to the fixed layer, but the torque that the electrons applied on the fixed layer cannot switch the magnetic orientation of the fixed layer. A small percentage of electrons are reflected back to the free layer after taking an orientation opposite to the fixed layer. With a large enough current, the amount of reflected electrons will change the state of the free layer.

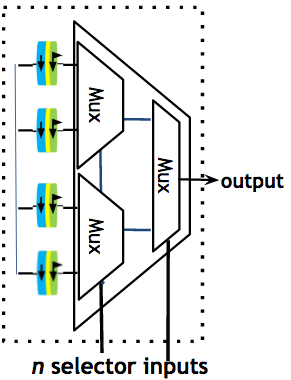


(a) (b)

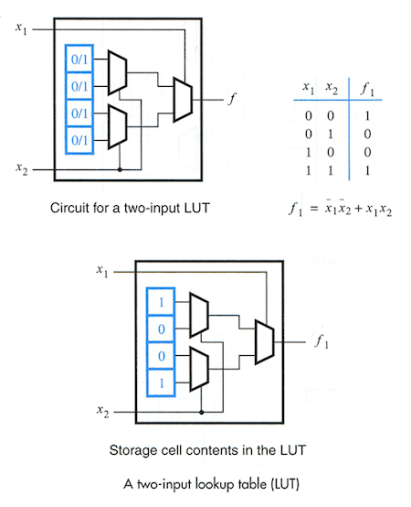
Figure 2. Physics behind MTJ: (a) Antiparallel to Parallel , (b) Parallel to Antiparallel1

## **2.3 Application of STT-LUT**

Since our design is static implementation, STT-LUT is composed of a number of multiplexers (MUX) and STTRAM (see Figure 3). Each STTRAM cell consists of two major components: MTJs and access transistors (CMOS circuit). The MTJs are used as storage cells to store logic “0” or “1” by the magnetic formation while the CMOS circuits are used to provide the switching current density to the MTJ and to select a particular memory cell 5,6,7,8. A MUX is a combinational circuit in which a set of selection inputs determines a specific input signal to be selected as output signal 2. The number of MUX-based LUT selector signals sets a limit on the number of input signals; in other words, 2n bit content requires n numbers of selector signals.

Figure 3. Fabric of static STT-LUT 

As before mentioned, each LUT works as a reconfigurable logic block. Therefore, we can map any logic function by programming the contents of the LUT. Take mapping an XNOR gate for instance (see Figure 4). To devise an XNOR gate, we implement a LUT with 2 inputs, x1 and x2, and program its 4-bit content according to the XNOR gate’s truth table, which is 1 0 0 1. More complicated logic functions with multiple inputs can be mapped securely by programing the content of LUTs.

Figure 4. XNOR gate devised by programming the content of LUT 1

# **3 Proposed Design Methodology: Hybrid Use of STT-LUT and Custom CMOS**

## **3.1 Hybrid Use of STT-LUT and Custom CMOS**

CMOS technology carries out its function with little delay and minimal power consumption but is unable to be reconfigured and can easily be reversed engineered. In contrast to CMOS, STT-LUT is reconfigurable with better protection against reverse engineering but requires more power and delay to operate. This report discusses optimal hybridization which utilizes the benefits of both STT-LUT and custom CMOS. The ideal hybridization would have similar delay and power consumption as CMOS technology while offering some degree of reconfigurability. Due to the fact that replacing gates of a circuit with STT-LUTs will increase performance costs, specific gates need to be selected in order to avoid significant penalties. For example, mapping gates along a pathway of a circuit that contains the most gates will result in an increase in delay penalty. This can be avoided by selecting logic gates outside of said pathway, though there may be some gates that will produce delay overhead when replaced with their respective STT-LUT. We investigate different scenarios of partial mapping logic gates to LUTs using a 4-bit adder as a case study.

## **3.2 Case Study: 4-bit Adder**

The 4-bit carry look-ahead adder netlist is created from a hardware description language called Verilog. This adder, whose gate schematic is shown in Figure 5 on the next page, adds two 4-bit binary numbers and produces a 4-bit binary sum and a 1-bit carry. It takes a certain amount of time for the adder to carry out its operation. This time, or delay, can be measured from the positive edge of the input pulses to the positive edge of the corresponding output pulse. The longest delay, also known as critical delay, is the most accurate delay of the design, and it is measured from the path from any input to any output ports that passes through most number of gates. Without implementing STT-LUT into the adder, the pathway with the most delay would be the path that contains the most gates. This path is known as the critical path. Our goal is to create a hybrid circuit by selecting a number of gates to be mapped with STT-LUTs, but before we can map multiple gates, we must first determine how each gate affects the circuit when mapped.

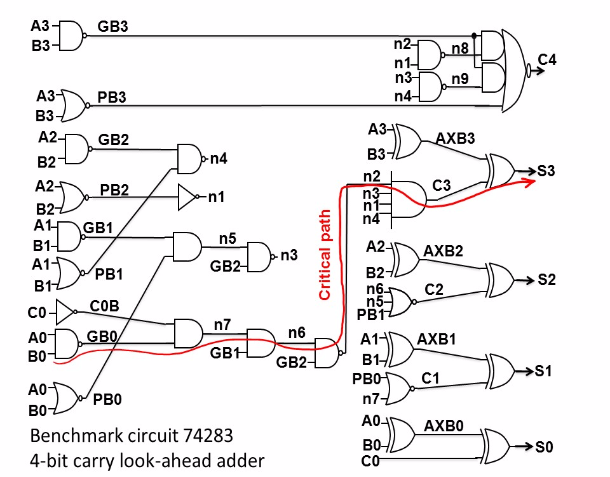


Figure 5. 4-bit carry look-ahead adder gate schematic 10

## **3.3 Single Gate Mapping**

STT-LUT implementation provides better security and programmability but it also runs the risk of decreasing performance parameters. These parameters are minimum delay, active and standby power consumption, power delay performance, and area. The minimum delay is the amount of time taken for the adder to complete its operation as quickly as possible without failing. Active power consumption is the amount of power used by the adder when in use. Unlike active power, standby power is the power used by the adder when idle. Power delay product is simply the product of the minimum delay and active power at the minimum delay. The area is the amount of transistors used to create the adder, which is in direct proportion with active power.

With these performance parameters in mind, the gates are replaced one at a time with its respective STT-LUT and then simulated using Synopsys tool. This process is carried out thirty times due to the fact that the adder contains thirty gates. We have written a code to calculate the delay and the amount of power consumed for the adder to carry out its operation. The extracted data is then compared to the data of the all custom CMOS version, also known as the circuit without STT-LUT implementation. When comparing the two types of data, we look for data that is similar if not the same as each other. This tells us that we can implement this LUT while minimizing overall delay overhead and power consumption.

## **3.4 Multiple Gate Mapping**

After mapping each gate and finding how they affect the performance of the circuit, the next step is to map multiple gates at once and extract the results thus creating a more secure hybrid reconfigurable circuit. Due to the absurd amount of gate combinations, it is unreasonable to map all the cases. To find the best gates combination, three selection algorithms were used to determine which gates should be mapped. The algorithms are independent selection, dependent selection, and parametric-aware dependent selection, introduced in 3.

Independent selection selects gates that are not along each other’s pathways, hence the name independent. The idea behind this selection is to map as many gates as possible while avoiding any delay overhead. Although the delay will be at a minimum, the security of the circuit is at risk. An attacker can potentially decrypt the function of the LUT by simply using trial and error; also known as brute force method. This is possible due to the fact that the preceding and subsequent gates of the LUT is not mapped, giving the attacker an estimated function estimated function of the partially mapped logic design.

Unlike independent selection, dependent selection selects gates that share the same pathways. This protects the circuit from potential threats by making it more difficult to find the function of the LUTs. Since the mapped gates are along the same pathways, this method could increase the amount of time needed to carry out the function. This selection methodology trades speed for security.

Parametric-aware dependent selection selects gates that do not produce extreme delay regardless of whether or not they share the same path. This selection methodology prioritizes the maximum amount of gates that can be mapped without leading to large amounts of delay overhead. Unlike independent and dependent selection, the gates are mapped in no specific sequence as long as mapping it does not produce a significant delay penalty.

# **4 Results**

Through netlisting in Synopsys tool, results of the delay, active power (both at minimum period, which is 26ps, and at a 1ns period), the power delay product (PDP), the stand-by power, and the area of the adder can be obtained. To evaluate the effectiveness of STT-LUT implementation, there must be a comparison of the results in which the adder’s gates are not reconfigured. This is called an all custom CMOS logic implementation. The critical output, which is the indication of where the critical path ends and if the critical path has changed at all, was observed

## **4.1 Single Gate Mapping Results**

For individual gate mapping, the majority of the gates were replaced with LUT2, NOT gates N1 and C0B were replaced with LUT1, NOR gate C2 was replaced with LUT3, AND gate C3 was replaced with LUT4, Gate C4, which is a combination of two AND gates and a NOR gate, was also replaced with LUT4. Though this gate appears to have 5 inputs, there are only 4 unique inputs due to the output of gate GB3 becoming an input to each of the two AND gates. A visual is shown in Figure 6 (shown on the next page).

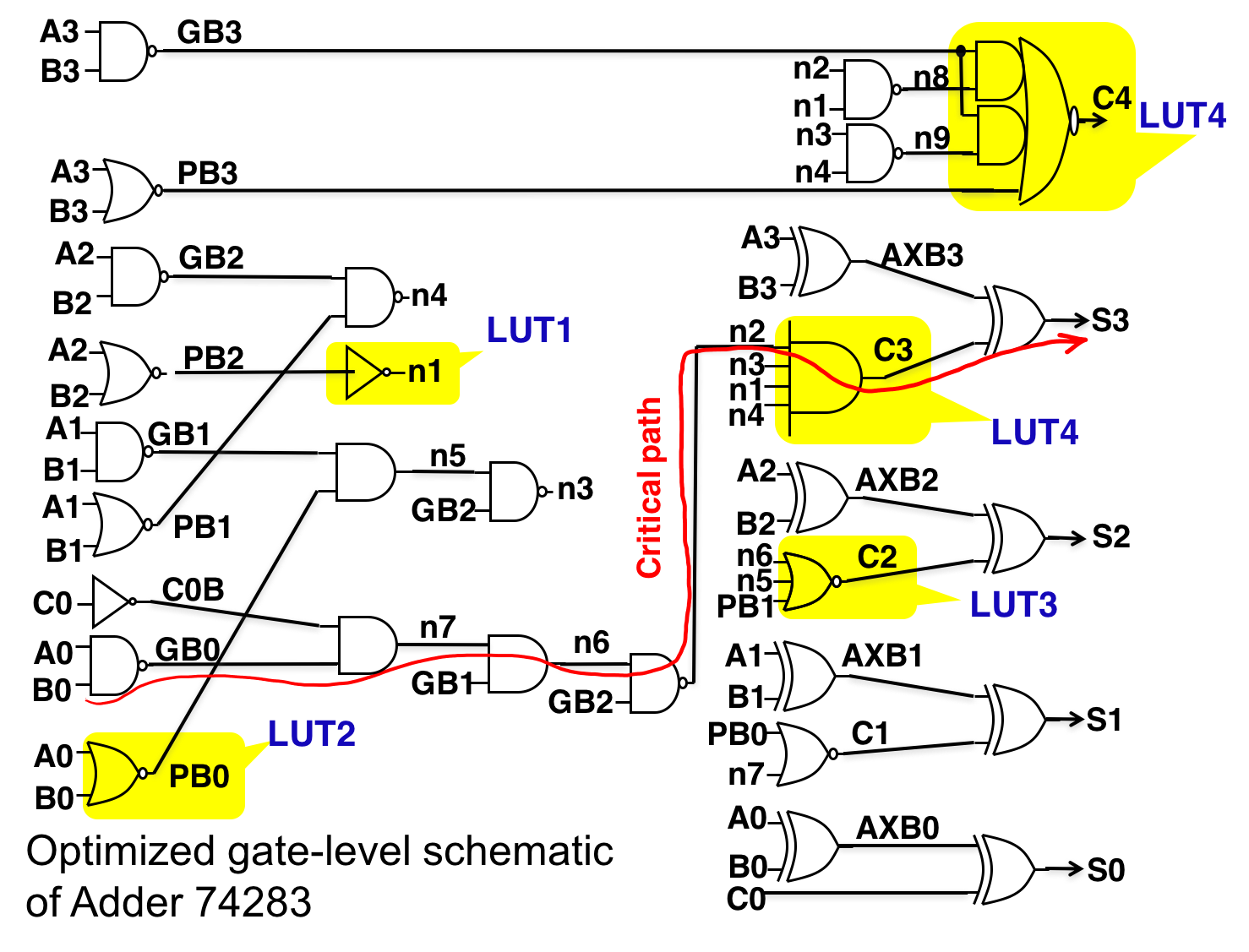


Figure 6. Various LUT mappings.

As expected from STT-LUT characteristics, STT-LUT replacement of a gate may result in delay overhead. The minimum period is the smallest delay of the design where it is operating without failure. When none of the gates are replaced with LUTs, the minimum period of the all custom CMOS is 26 ps. The identified gates that do not produce overhead are shown in Figure 7.

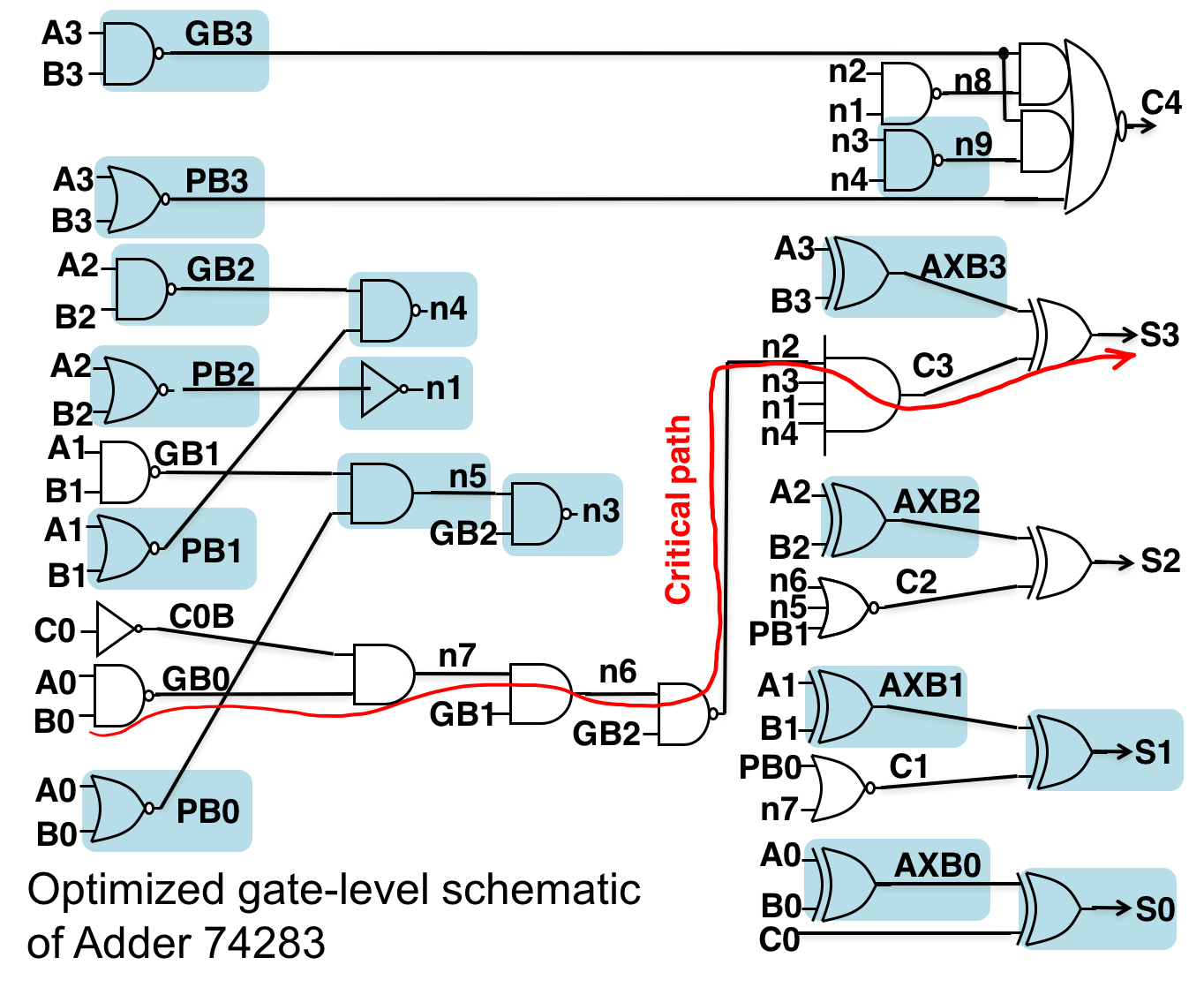


Figure 7. Gates with no delay overhead.

The individual gate mapping results are shown in a bar graph in Figure 8. As observed from the results, all gates in the critical path and some gates that are not in the critical path (referred to as non-critical gates) cause delay.

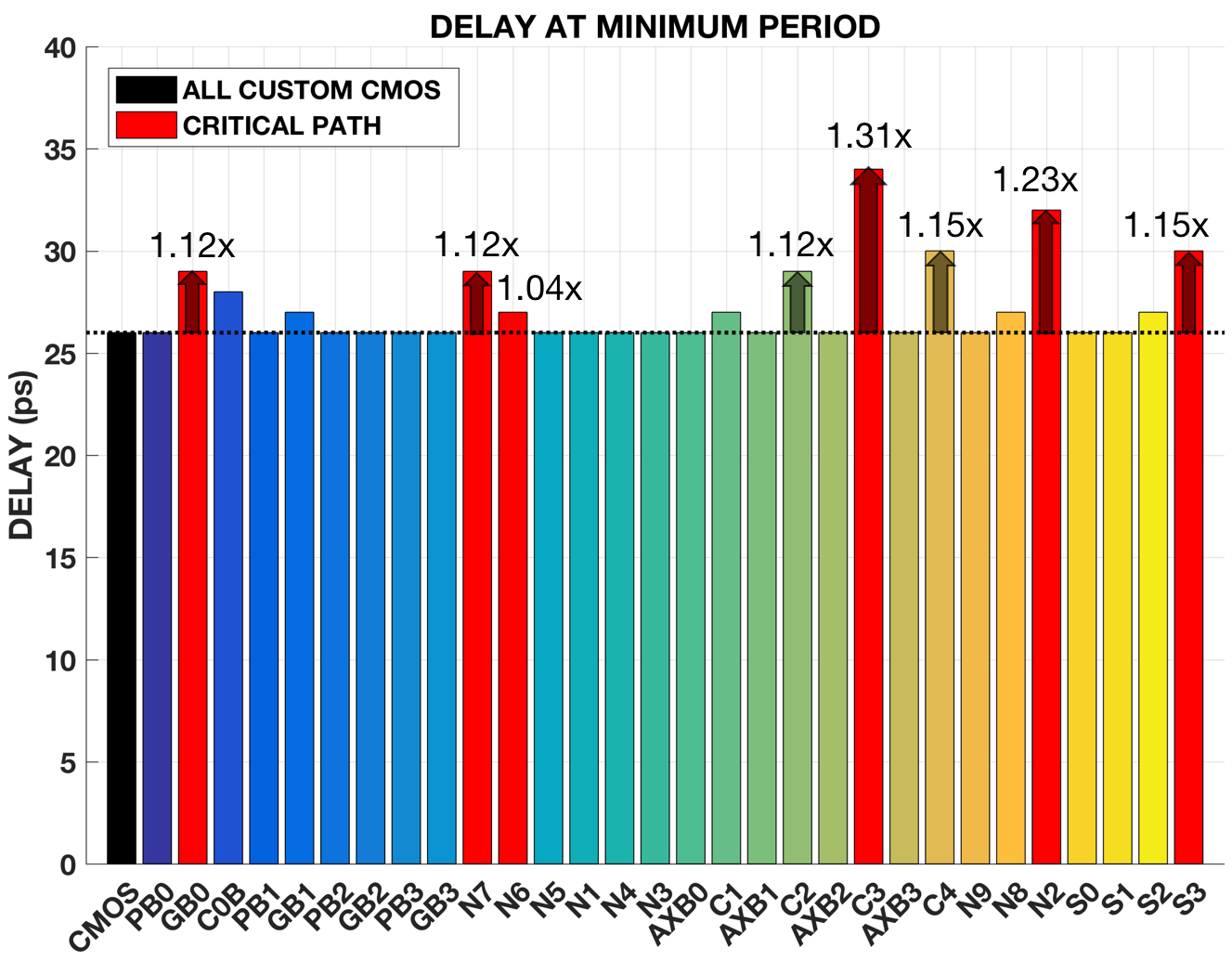


Figure 8. Delay at minimum period (individual mapping). The black bar represents the all custom CMOS. The red bars represent gates along the critical path. The multicolored bars represent the non-critical gates.

The mapping of critical gate C3 produced the highest delay of 34 ps, which is 1.31 times greater than the delay of the all custom CMOS case. Another critical gate, N2, produced the second highest delay of 32 ps, which is 1.23 times greater than the all custom CMOS delay. The mapping of a non-critical gate, such as C4, displayed the third highest delay of 29 ps, which is 1.15 times greater than the all custom CMOS delay. Gate C4 resulted in the same minimum period as the critical gate S3.

As for the critical output, only three gates displayed a change in the critical path: gates C2, C4, and S2. Interestingly, gate C2 did not have a single critical output, but had multiple outputs which were S0, S2, and S3. Gates C4 and S2 are respectively a carry and sum output gate so their critical path changed to their own gates. The critical path changed for gate C4 due to the large LUT implementation and cause in delay. As for gate S2, a possibility as to why the critical path changed was due to its C2 input. Gate C2 is a 3-input gate and one of its inputs is an output from the critical path in which the delay may have carried over.

The active power is measured in two different time periods: at its minimum period and at a 1 ns period. Comparison of the active power at the 1 ns period is more reasonable because the minimum period of each gate may be different from other gates.

As expected from STT-LUT characteristics, STT-LUT replacement of any gate results in active power, area, and PDP overhead. LUTs have a larger area, which is due to a more complex design of multiplexers and a larger number of transistors compared to bulk CMOS logic gates. For a gate with a higher number of inputs, there is a consequence of larger area thus more active power is required. The PDP is the product of the delay and active power at the minimum period of each gate’s LUT implementation. PDP is the energy consumed by the adder when the gates are mapped with LUTs. The results of active power, area, and PDP are shown in a set of bar graphs in Figures 9, 10, and 11, respectively.

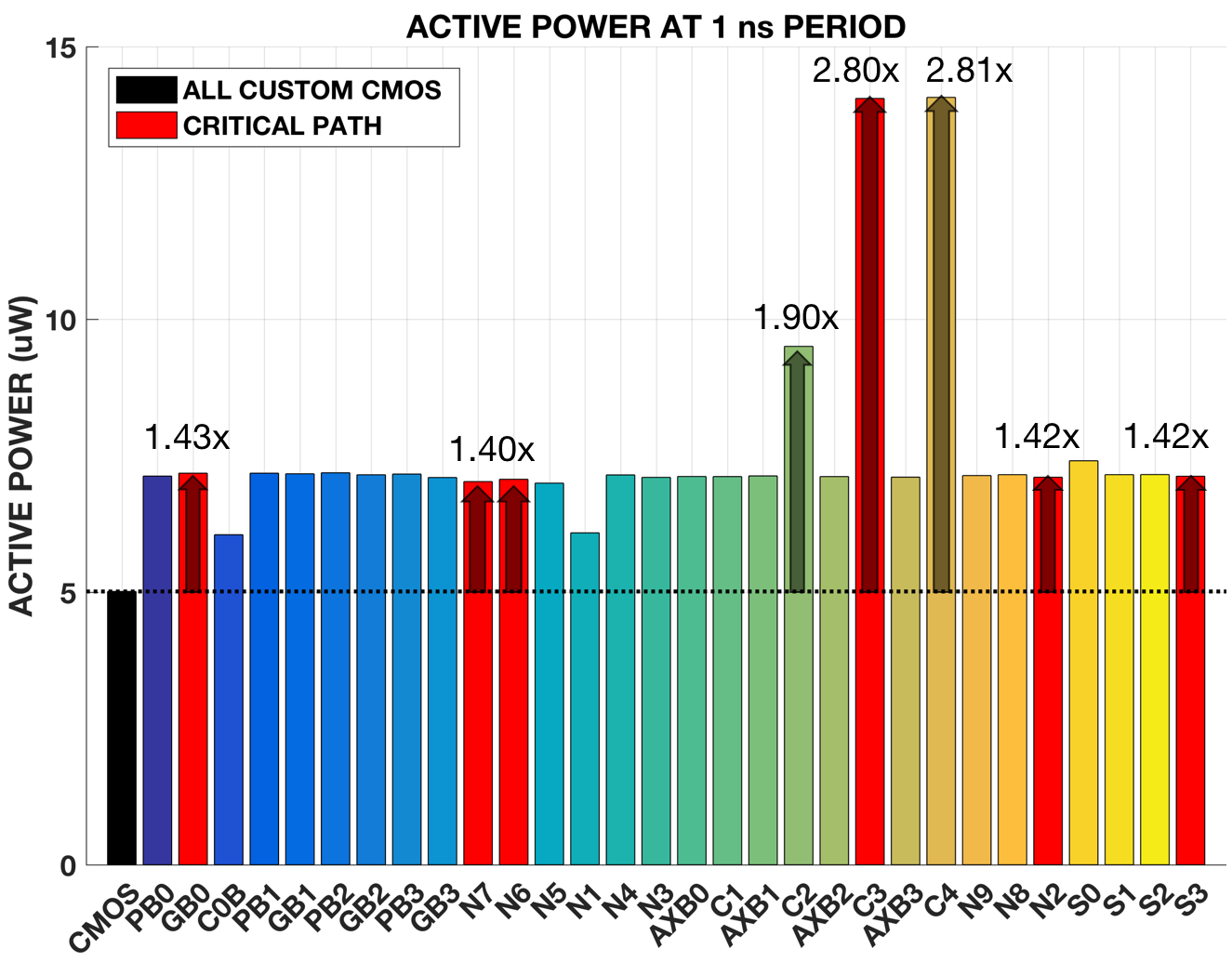


Figure 9. Active power at a 1 ns period (individual mapping).

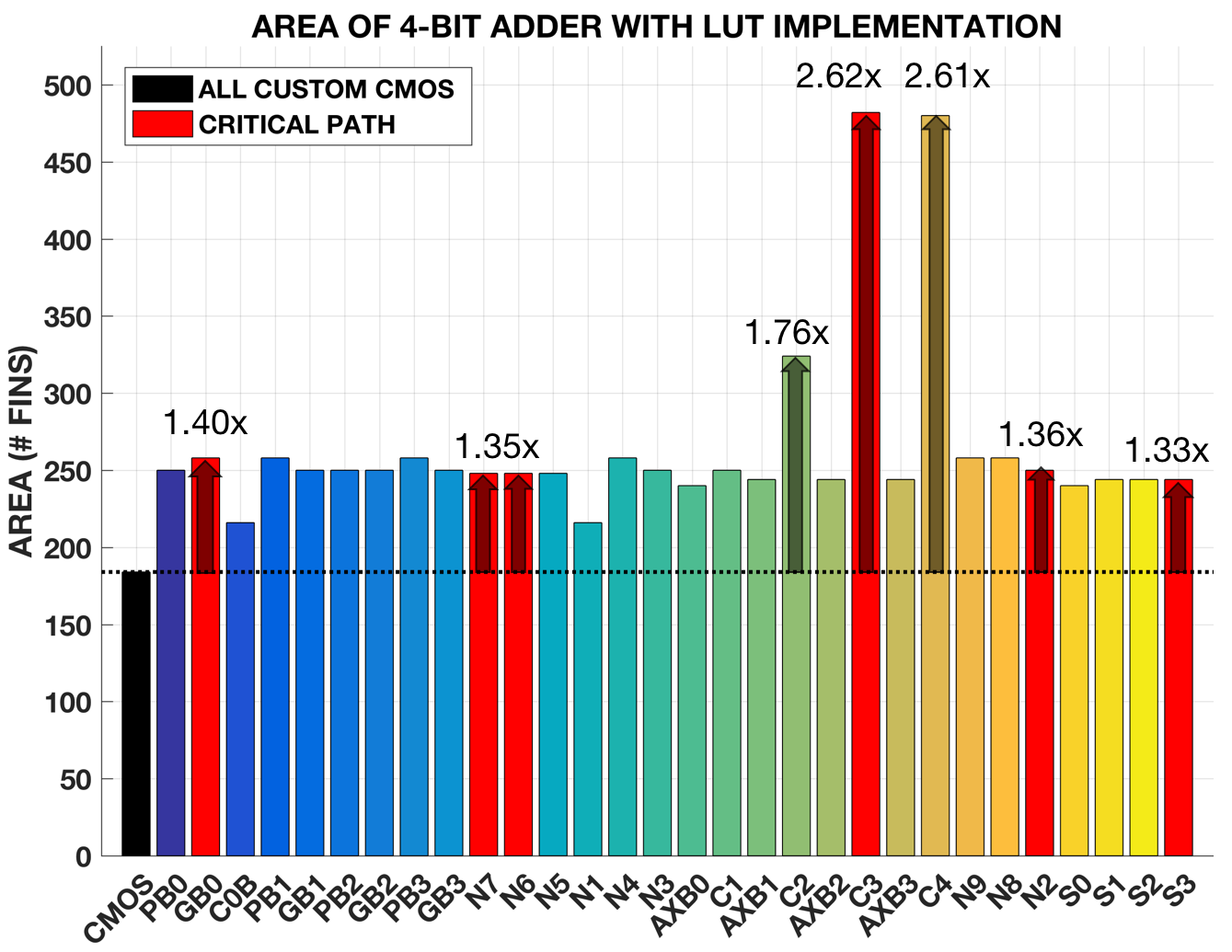


Figure 10. Area of 4-bit adder (individual mapping).

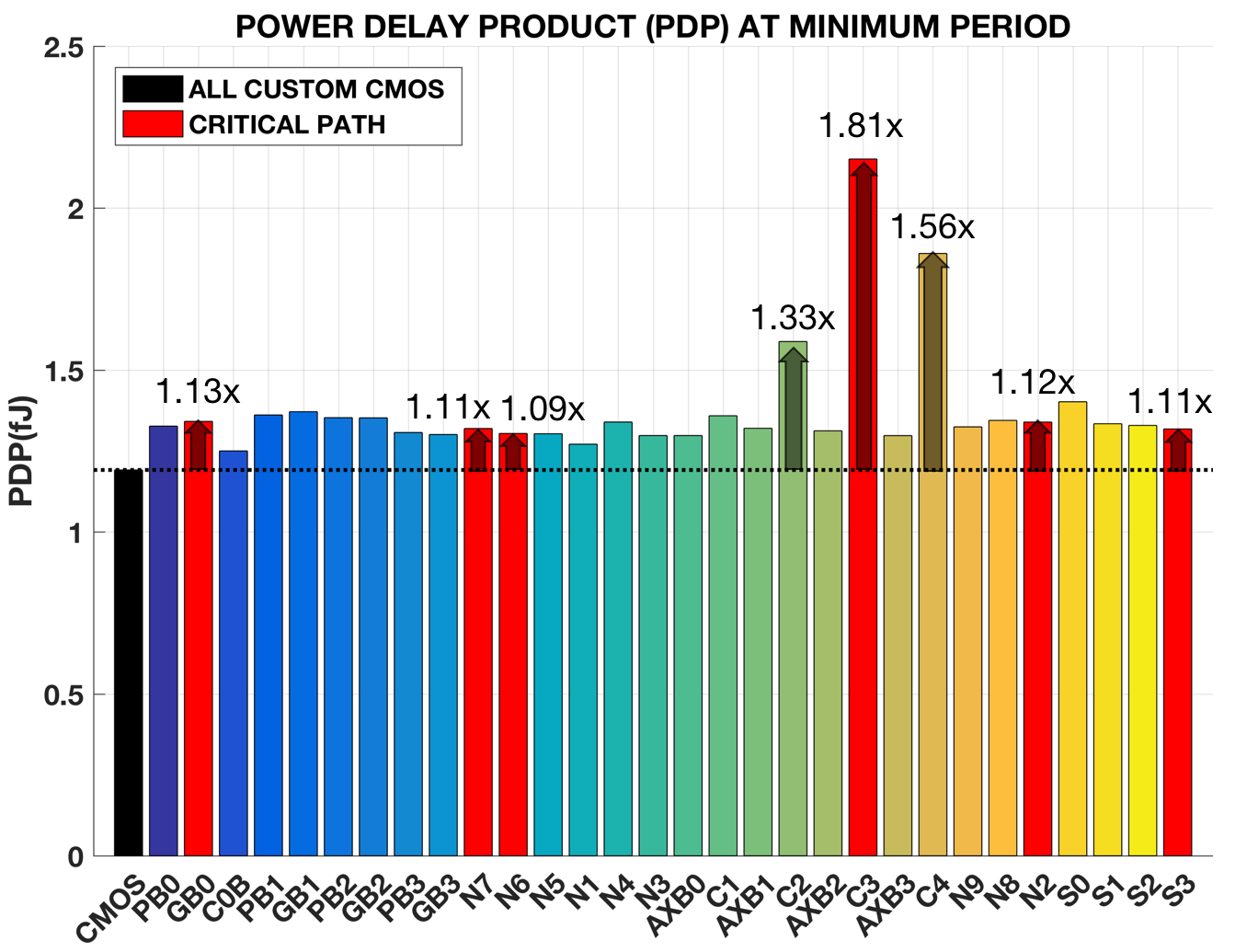


Figure 11. PDP at minimum period (individual mapping)

The mapping of gates C3 and C4 resulted in an active power 2.80x and 2.81x greater than the all custom CMOS and their area overhead are 2.62x and 2.61x greater. When gate C4 is implemented, it has two less transistors than gate C3, thus causing only the slightest difference in area and active power. Since gate C3 has a higher delay than gate C4, the PDP will also have a proportional overhead of 1.81x and 1.56x greater than the PDP of the all custom CMOS. The mapping of gate C2 results in an active power and a PDP of 1.90 and 1.33 times greater than the all custom CMOS and an area overhead of 1.76x larger.

## **4.2 Multiple Gate Mapping Results**

For multiple gate mapping, three optimization algorithms ­– independent selection, dependent selection, and parametric-aware dependent selection – are used to select a set of gates to be replaced with LUTs.

Independent selection chooses gates that do not intersect with each other nor are subsequent to each other. The algorithm applied to this selection chose specific gates with the 4-bit data inputs (A0 to A3, B0 to B3) and/or a sum output gate (S0 or S1). For inputs (A0, B0), gate AXB0 or the sum output gate S0 can be chosen. For inputs (A1, B1), gate AXB1 or the sum output gate S1 can be chosen. For inputs (A2, B2), only gate AXB2 can be chosen. For inputs (A3, B3), gates GB3, PB3, or AXB3 can be chosen. There are 12 possible combinations. Two examples of combinations are shown in Figure 12.

|  |  |
| --- | --- |
| 1. One possible combination: gates GB3, AXB2, AXB1, and S0. | (b) Another possible combination: gates PB3, AXB2, AXB0, and S1. |

Figure 12. Two possible sets of independently selected gates.

The dependent selection are gates along the same path and can be subsequent to each other. The algorithm applied to this selection chose specific gates with delay overhead. Two examples of the selection are shown in Figure 13. Figure 13(a) refers to the first case selection of gates GB0, N7, C2, and S2. This selection shows the dependency of gates when all have delay overhead and when the path is nonconsecutive. Gate N7 is dependent on gates GB0 and C0B. Gate C2 is dependent on one of three inputs, none of which were implemented with an LUT, and the output gate S2 is dependent on gate C2. In other words, this nonconsecutive path contains a break in dependency. Figure 13(b) refers to the second case selection of gates GB1, N5, N3, N9 and C4. Both the input and output gates, GB1 and C4, have delay overhead. This selection shows the dependency of gates when there is a mixture of gates with and without delay overhead and when the path is consecutive.

|  |  |
| --- | --- |
| 1st_dep_crit_path.png   1. First case selection: gates GB0, N7, C2, and S2. | 2nd_dep_crit_path.png  (b) Second case selection: gates GB1, N5, N3, N9 and C4. |

Figure 13. Two sets of dependently selected gates.

The parametric-aware dependent selections are gates that do not severely impact the delay, regardless of dependency. The algorithm applied to this selection chose all gates that do not produce delay overhead except for two 2-input gates, GB1 and GB0, and the carry output gate C4. The example of the selection is shown in Figure 14.



Figure 14. A set of parametric-aware dependent selected gates.

Figure 15 shows the delay at minimum period in all three selection methods and the all custom CMOS case. As can be concluded from Figure 15, LUT replacement of multiple gates does not result in delay overhead for independently selected gates. Based on the individual mapping results, the following gates chosen for this selection method did not produce delay overhead as well. LUT replacement of multiple gates results in delay overhead for both dependent and parametric-aware dependent selections. The first case of dependent selection resulted in a delay 1.23 times greater than the all custom CMOS. The second case of dependent selection resulted in a delay 1.15 times greater and the parametric-aware dependent selection resulted in a delay 1.12 times greater. There is more overhead in the dependent selection, despite having lesser gates implemented than the parametric-aware dependent. When gates in the same pathway are replaced with LUTs, the input of one gate is dependent on the other, thus it is possible for more delay to carry over.

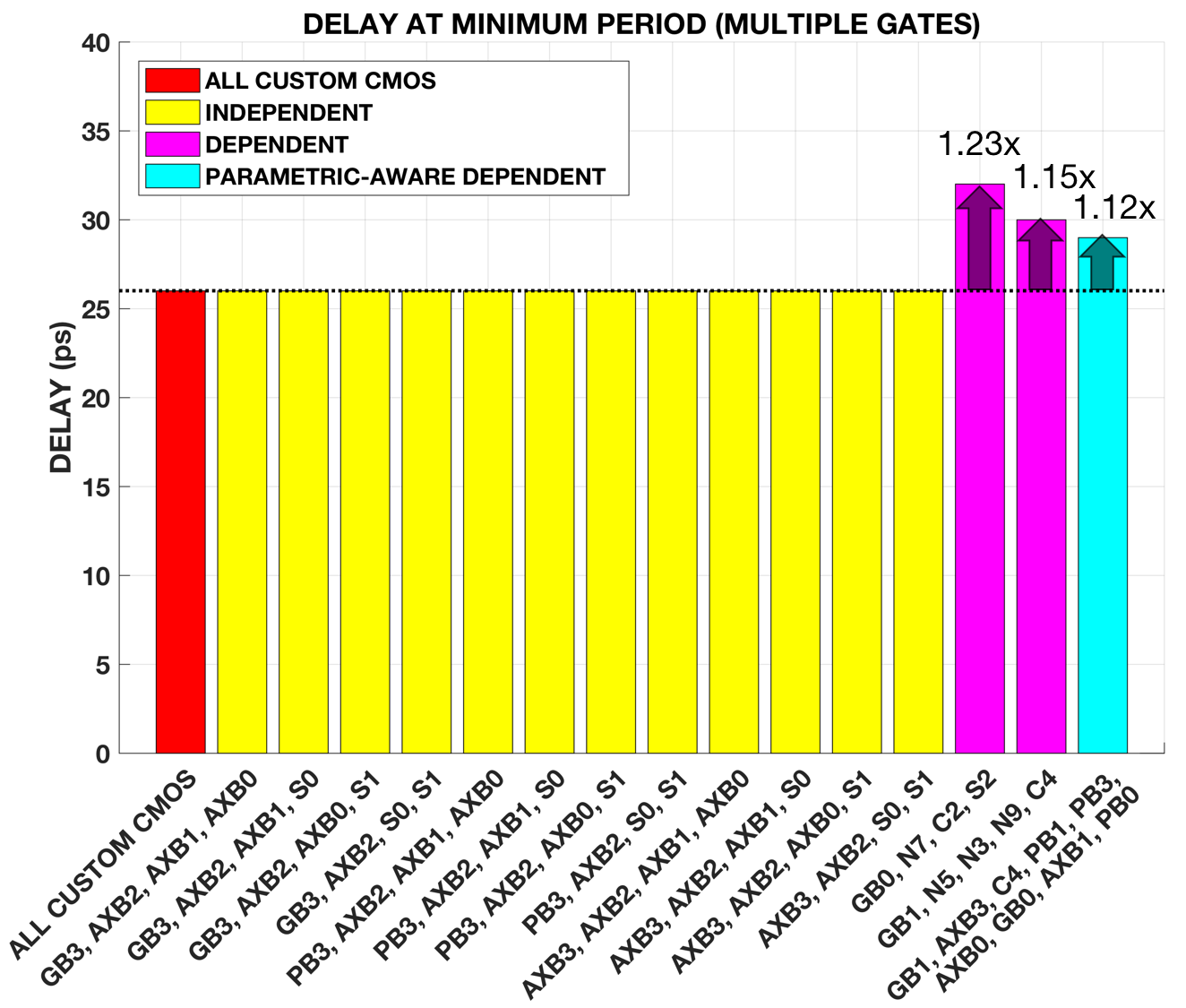


Figure 15. Delay at minimum period (multiple gate mapping).

The critical output was also observed for the multiple gate mapping. The independent selections did not have a change in critical path, as their critical output still ends at gate S3. For the first dependent selection, the critical path changed to gate S2 due to the large delays produced by mapping gates GB0, N7, C2, and S2, as seen in Figure 14(a). For the second dependent selection, the critical path changed to gate C4 due to the delays produced by mapping gates GB1 and C4, as seen in Figure 14(b). For the parametric-aware dependent selection, the critical path also changed to gate C4 due to the delays produced by mapping gates GB1, C4, and GB0, as seen in Figure 15.

LUT replacement of multiple gates produces active power, area, and PDP overhead. The results are shown in a set of bar graphs at Figures 15, 16, and 17, respectively. The results are extracted in a similar way with the individual mapping data. All independent selections are approximately 2.68x greater in active power than the all custom CMOS. All independent selections are approximately 2.34x and 1.40x greater in area and PDP, respectively, than the all custom CMOS. The reason for this trend is because all the selected gates are 2-input gates and all are replaced with LUT2, which provides optimal power, area, and PDP overhead.1 LUT2 occupies less area compared to LUT3 and LUT4. The first dependent selection has an LUT3 implementation on gate C2. The second dependent and parametric-aware dependent selection have an LUT4 implementation on gate C4. However, higher area is not directly proportional to higher delay.

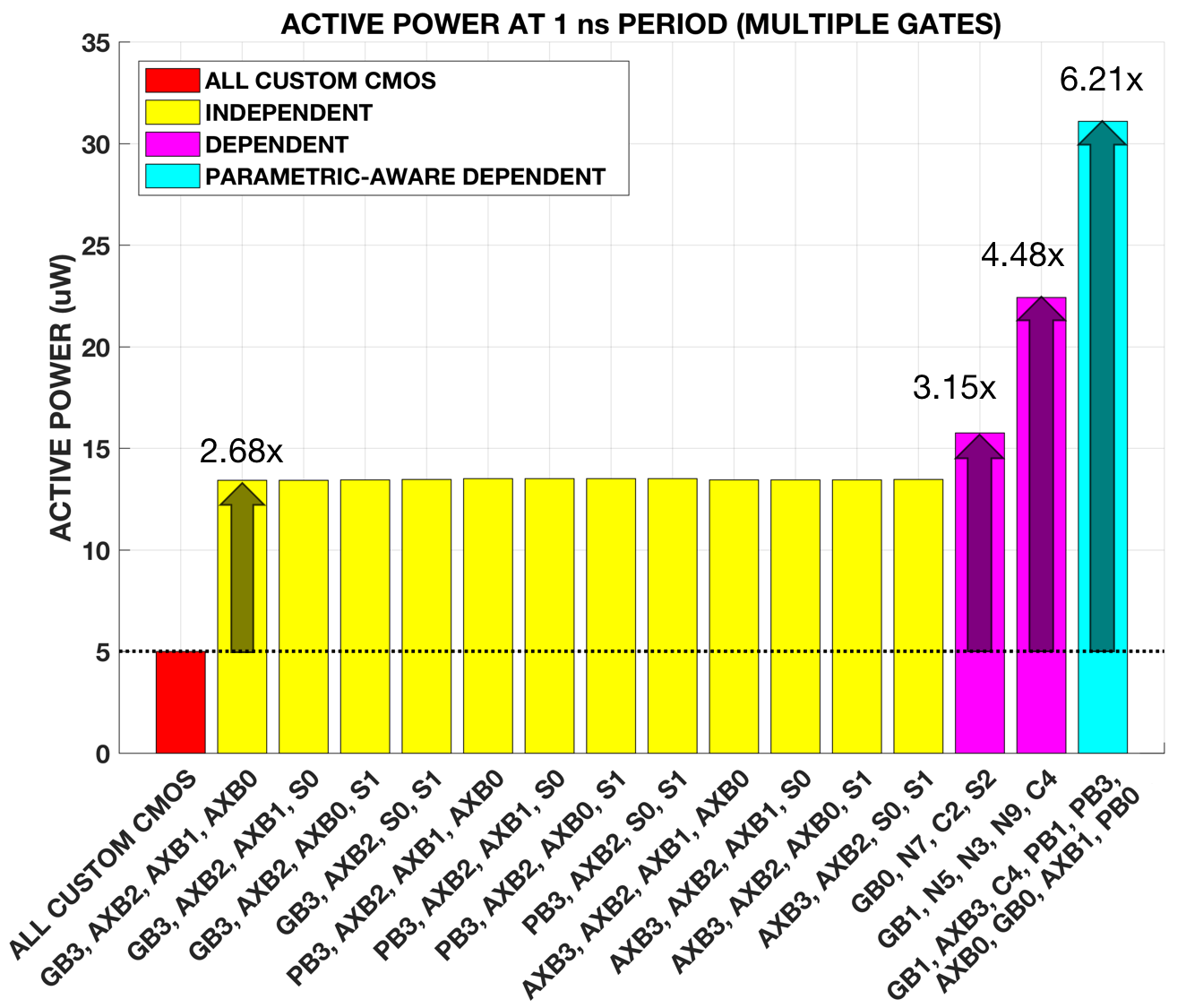


Figure 16. Active power at a 1 ns period (multiple gate mapping).

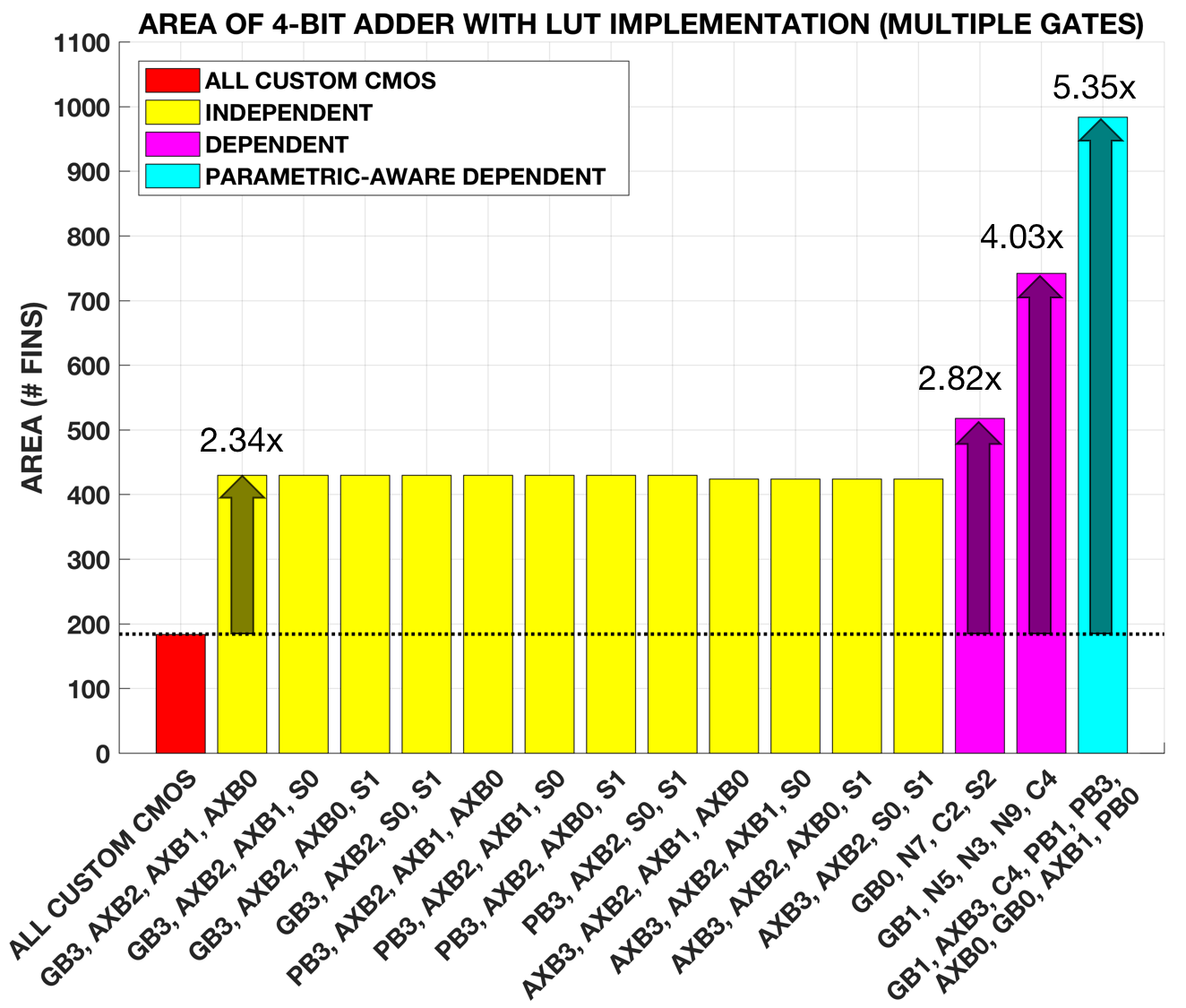


Figure 17. Area of 4-bit adder (multiple gate mapping).

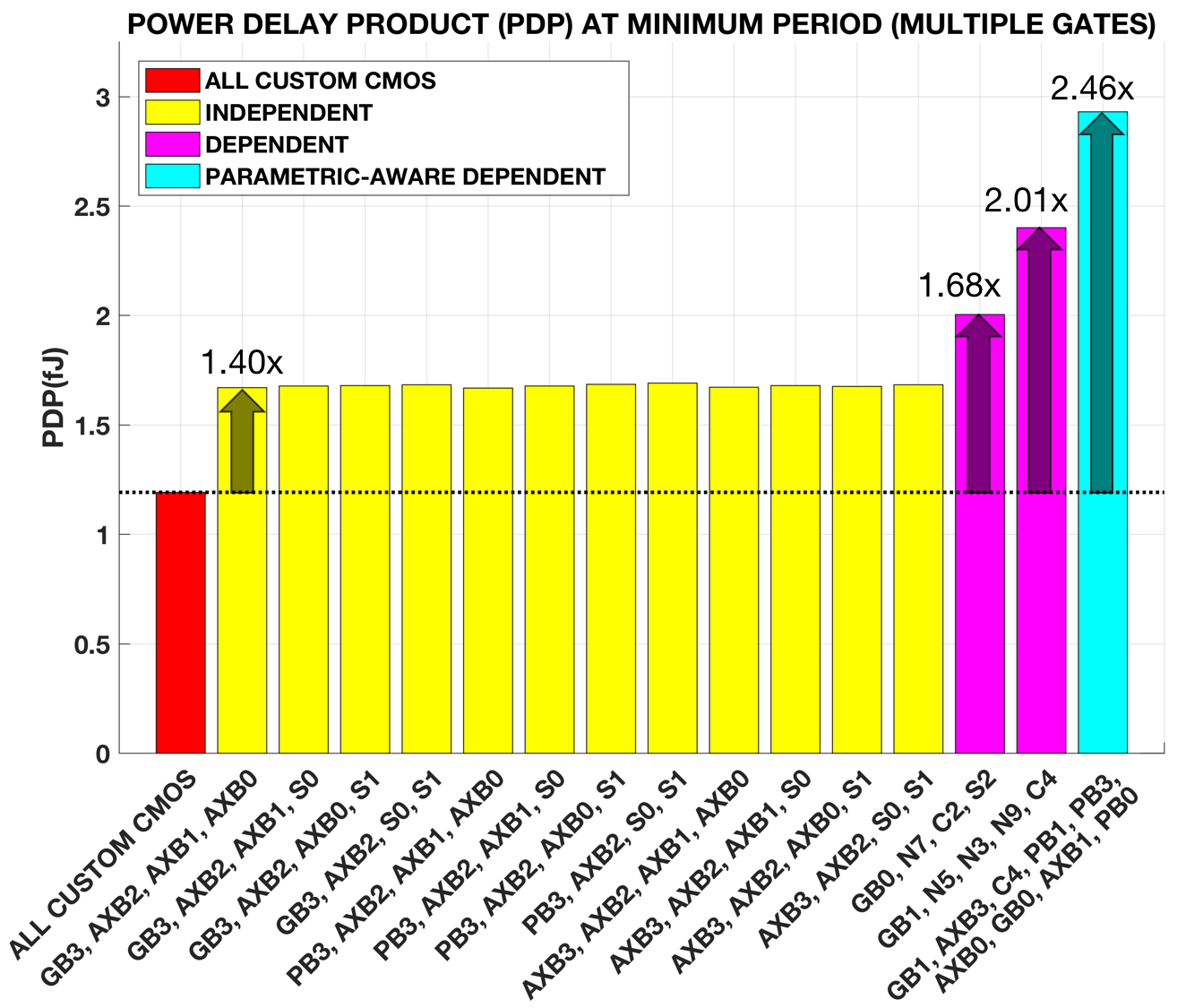


Figure 18. PDP at minimum period (multiple gate mapping).

With increasing area, there will also be an increase in active power and PDP. The first and second dependent selection and the parametric-aware selection had an area overhead of 2.82x, 4.03x, and 5.35x greater, respectively, as seen in Figure 17. The same trend is observed in Figure 16 for the active power, which are 3.15x, 4.48x, and 6.21x greater, and in Figure 18 for the PDP, which are 1.68x, 2.01x, and 2.46x greater than the all custom CMOS.

# **5 Conclusion**

Spin Transfer Torque random access memory is an emerging technology that has the potential to replace traditional memory like SRAM, DRAM, and flash because it is non-volatility, fast read and write speeds, little to no leakage current, and most importantly, it is more difficult to reverse engineer. However, in STT-LUT implementation, some of the performance parameters such as delay, active power, PDP, standby power and area may suffer due to its complexity. In this project, we explored STT-LUT implementation on the 4-bit adder case study and create a hybrid adder which consists of both STT-LUT and custom CMOS while minimizing performance penalties. Before implementing multiple LUTs, we had to retrieve data on how each implemented LUT would individually affect the adder. By individually replacing each gate with its respective LUT, we observed that all critical gates and some non-critical gates produced delay overhead while all gates produced active power, PDP, and area overhead. After determining how each gate would affect the adder performance characteristics, the next step is to map multiple gates simultaneously. In order to achieve the optimum set of gate selections to be mapped with minimum delay penalty, three algorithms were used when selecting gates to replace with their respective LUTs: independent selection, dependent selection, and parametric-aware dependent selection. We simulated a variety of combinations using these algorithms, extracted the data on performance penalties, and compared them with the all custom CMOS case and each other. In terms of delay and power consumption, independent selection would be ideal due to the fact that it avoided delay overhead entirely. As for hardware security, dependent selection would be more effective since it is more difficult for an attacker to find the function of the circuit. Parametric-aware dependent selection offers more hardware security than independent selection and less delay than dependent selection but requires the highest power consumption of the three. Depending on the needs of the manufacturer, they can choose certain algorithms to prioritize performance or security.

# **Acknowledgements**

This project was partly supported by the US Department of Education through Minority Science and Engineering Improvement Program (MSEIP, Award No. P120A15014), and the Hispanic-Serving Institution for Science, Technology, Engineering, and Mathematics (HSI STEM) Program, Award No. P031C110159.

# **Bibliography**

[1] Almasi, D., *Enhancing Hardware Security Using Spin Transfer Torque Logic*, in *Engineering of Embedded Electrical and Computer Systems*. 2015, San Francisco State University: San Francisco.

[2] Mano, M. Morris and Kime. Charles R. *Logic and Computer Design Fundamentals*. New Jersey: Pearson Education, 2008. Book.

[3] Winograd, T., et al. Preventing Design Reverse Engineering with Reconfigurable Spin Transfer Torque LUT Gates, in 2016 17th International Symposium on Quality Electronic Design. (ISQED). 2016.

[4] N. Nishimura et. al.“Magnetic tunnel junction device with perpendicular magnetization films for high-density magnetic random access memory,” Journal of Applied Physics, vol. 91, no. 8, pp. 5246-5249, 2002.

[5] L. Y. Loh. Mechanism and Assessment of Spin Transfer Torque (STT) Based Memory. Master's thesis, Massachusetts Institute of Technology, 2007.

[6] Jian-Gang Zhu, “Magnetoresistive Random Access Memory: The Path to Competitiveness and Scalability,” Proceedings of the IEEE, vol. 96, 2008, pp. 1786-1798.

[7] J. Katine and E.E. Fullerton, “Device implications of spin-transfer torques,” Journal of Magnetism and Magnetic Materials, vol. 320, Apr. 2008, pp. 1217-1226.

[8] M. Hosomi, H. Yamagishi, T. Yamamoto, K. Bessho, Y. Higo, K. Yamane, H. Yamada, M. Shoji, H. Hachino, C. Fukumoto, H. Nagao, and H. Kano, “A novel nonvolatile memory with spin torque transfer magnetization switching: spin-ram,” Electron Devices Meeting, 2005. IEDM Technical Digest. IEEE International, 2005, pp. 459-462

[9] Dorrance, R. W. (2011). Modeling and Design of STT-MRAMs. Electrical Engineering, University of California, Los Angeles. Master of Science

[10] D. Almasi, H. Mahmoodi, H. Homayoun, H. Salmani. “Device and Circuit Optimizations for Spin Transfer Torque Reconfigurable Logic”