8-Bit Kogge-Stone Adder

Shayaun Bashar (A15877671),
Zoom Chow (A16462757),
Kenneth O'Connor (A17241796)

Introduction:
Adder circuits are fundamental building blocks in digital systems , playing a crucial role in arithmetic operations such as addition and subtraction. The design and implementation of high-performance adders have been the focus of extensive research and development efforts. This paper addresses the design challenge of developing a high-speed 8-bit adder with improved performance compared to the synthesized adder in Lab 4. The objective is to surpass the limitations of conventional adder architectures and demonstrate the ability to create innovative designs that achieve faster operation.

The need for high-performance adders arises from the increasing demand for efficient arithmetic computations in various applications, including digital signal processing, microprocessors, and communication systems. As technology advances, there is a continuous push to enhance the speed and efficiency of adder circuits to meet the demands of modern computing systems. Therefore, the design challenge presented in this paper is significant in the context of improving overall system performance.

The design challenge presented in this paper provides an opportunity to delve into the intricacies of adder architecture design, utilizing custom schematic design techniques. By exploring innovative architectures and surpassing the limitations of conventional designs, this paper aims to contribute to the development of high-performance adder circuits, thereby enhancing the overall efficiency and speed of digital systems. The subsequent sections will detail previous architectures, the chosen architecture, innovative aspects of the design, and future directions for further improvements in adder performance.

Previous Architectures:
Traditional adder architectures, such as the simple ripple-carry design, have been widely used due to their simplicity and ease of implementation. However, these architectures suffer from limitations in terms of propagation delay, as the carry signals must propagate through each stage sequentially. This results in slower overall performance, especially for wider bit-width adders.

The carry-lookahead adder is an improvement over the RCA that reduces the carry propagation delay. It achieves this by precomputing the carry signals for each stage based on the input bits. While the CLA architecture offers improved speed by enabling parallel carry computation, it suffers from increased area overhead due to additional logic gates and the need for extra circuitry to generate the lookahead signals. This added complexity makes it less efficient in terms of power consumption.

The carry-skip adder, also known as the carry-bypass adder, aims to further reduce the carry propagation delay of the RCA. It achieves this by incorporating carry-skip logic, allowing the carry signal to bypass multiple stages and directly propagate to subsequent stages. While the CSA architecture provides faster operation than the RCA, it suffers from increased area overhead and complexity. The additional logic for generating and managing skip signals adds to the overall circuit complexity, impacting power consumption and design scalability.

To overcome the limitations of the RCA, CLA, and CSA, the parallel prefix adder (PPA) architecture has emerged as a promising solution. The PPA architecture addresses the challenge of carry propagation delays by utilizing a tree-like structure with multiple levels of carry computations. By leveraging parallel processing of carry signals, the PPA significantly reduces the critical path delay and enhances overall speed performance. This innovative architecture enables efficient carry signal propagation through various paths, allowing for faster addition of multi-bit numbers. The PPA offers improved performance and reduced latency compared to traditional adder architectures, making it an attractive choice for high-speed arithmetic operations.

Two noteworthy variations of traditional PPA architecture include the Brent-Kung adder and the Kogge-Stone adder. The Brent-Kung adder offers improved power efficiency because of its lowest area delay with large number of input bits. However, the large number of levels in this architecture reduces its operational speed [7]. The Kogge-Stone adder has minimal fan-out and logic depth. Thus, the Kogge-Stone adder architecture becomes one of the fastest adders that is favored in electronic technology. However, as a tradeoff the KSA has a larger area.

In the subsequent sections, we will explore the design and innovative aspects of the parallel prefix adder—in particular, the Kogge-Stone adder—and discuss its potential to overcome the limitations observed in previous designs.

Chosen Architecture and Advantages:
The Kogge-Stone adder (KSA) architecture stands out as a compelling alternative due to its distinct advantages. The KSA architecture exhibits superior performance by leveraging a parallel prefix computation approach, like the CLA, but without the associated area overhead. It reduces the critical path delay by enabling efficient carry propagation through a tree-like structure, allowing multiple carry signals to be computed in parallel, reducing latency, making the KSA an excellent choice for high-speed arithmetic operations. Furthermore, the KSA's structure allows for efficient scaling to accommodate larger bit-widths. The KSA strikes a balance between speed, area efficiency, and scalability, making it a captivating choice for designing high-performance adders [1]. In the subsequent sections, we will delve into the detailed architecture and innovative aspects of the Kogge-Stone adder.

Dynamic logic:
Dynamic logic is a modern design paradigm that offers a range of innovative techniques for efficient circuit design. It has gained traction due to its ability to address growing demand for higher performance and energy efficiency in integrated circuits. Unlike conventional static logic design, dynamic logic utilizes time-varying signals and transient behavior to achieve compact and power-efficient implementations.

Our utilization of dynamic logic for implementing the gates in the 8-bit adder offers significant advantages over static complementary MOSFET logic (CMOS) found in the gsclib045 library. In contrast to CMOS, which necessitates separate pull-up and pull-down networks comprised of PMOS and NMOS transistors, dynamic logic achieves gate functionality with reduced transistor count, utilizing only one PMOS transistor in the pull-up network. Moreover, dynamic logic employs a carefully orchestrated sequence of pre-charge and conditional evaluation phases, driven by the clock signal, to accomplish its operations. This distinctive design approach contributes to our enhanced efficiency and performance in the implementation of

the 8-bit adder circuit compared to an adder generated after PnR in Lab 4 which can be viewed in the transient simulation **Figure 4a**.

Dynamic logic operates in a two-phase clocking scheme, consisting of the pre-charge and evaluation phase. During the pre-charge phase, the output node is pre-charged to either a high (logical one) or low (logical zero) state using the PMOS transistor. Subsequently, in the evaluation phase, the NMOS logic evaluates the input conditions and determines whether the output should remain at a low or high state. To enforce a single transition per clock trigger, an inverter is commonly placed at the output of the dynamic logic circuit, ensuring that only one state change occurs for each clock cycle. This two-phase operation and the incorporation of an inverter at the output effectively enable reliable and controlled behavior of dynamic logic circuits.

One notable concern associated with the implementation of dynamic logic gates is the potential occurrence of charge leakage or sharing between different stages. Such leakage can result in the undesired flow of charges stored on the load capacitance of the dynamic gate, which introduces noise back into the circuit. This excess charge can significantly impact the overall performance and reliability of the logic gate. To address this challenge in our design for a faster and more robust adder, we have employed a keeper transistor—a weak transistor positioned in parallel with the PMOS transistor in the pull-up network. The role of the keeper transistor is to supply a small current from the power supply to the dynamic output node, effectively reducing the noise margin within the circuit. By doing so, the presence of the keeper transistor enhances the overall functionality and performance of the dynamic logic circuit.

We choose to use dynamic logic over static complementary CMOS and other techniques since it offers us the chance to use less transistors, it requires N+2 transistors while static CMOS uses 2N and pseudo-NMOS uses N+1. Additional benefits compared are the smaller load capacitance, no static power, non-ratioed logic, it is also much faster since $t_{pLH} = 0$ and t_{pHL} is small and in pre-charge time is small. In general, about 1.3-2x faster than static CMOS. However, the downsides are high power consumption as well as noise leakage, charge sharing and inputs not being able to go from 1-0. We can still combat some of the issues by using a keeper transistor, extra pre-charge transistor and domino logic which we will get into. This influenced our design choice since we knew that we wanted to optimize for speed.

Domino Logic:
Domino logic has emerged as a prominent dynamic logic family, offering significant improvements in performance and efficiency for integrated circuits. Domino logic operates on a cascaded structure, utilizing the concept of pre-charging and dynamic evaluation to achieve rapid switching and propagation of logic signals.

Utilizing the cascading of multiple stages in domino logic offers a unique advantage in effectively leveraging transitions from high to low and low to high, facilitated by a single switch triggered by the clock for our design of an 8-bit adder. This characteristic enables inputs to transition between 0 and 1 or 1 and 0, ensuring proper functionality of our 8-bit adder. Additionally, the reduced input capacitance achieved by employing fewer transistors contributes to significantly enhanced circuit speed. This attribute proves particularly beneficial in the context of an 8-bit adder, where circuit speed plays a crucial role in ensuring optimal functionality.

High Skew Inverter
The utilization of a high skew inverter in our Kogge-Stone adder holds substantial purpose and significance. The high skew inverter, characterized by intentionally imbalanced rise and fall delays, introduces controlled delay variations to the signal propagation path. By strategically incorporating a high skew inverter within our Kogge-Stone adder's architecture, it becomes possible to synchronize the critical path delays, minimizing the overall delay and enabling faster carry propagation. This not only provides improvement to the performance and speed of the adder but also ensures an efficient utilization of resources. The high skew inverter plays a vital role in achieving optimal timing and enhancing the overall functionality of our Kogge-Stone adder, making it a valuable and viable component for advanced digital circuits. This innovative use of a high skew inverter can be found in our custom designed AND and OR logic gates detailed in a following section and shown in **Figure 2a** and **Figure 2b**.

Figure 1a illustrates the designed transistor level schematic used for our 8-bit Kogge-Stone adder.

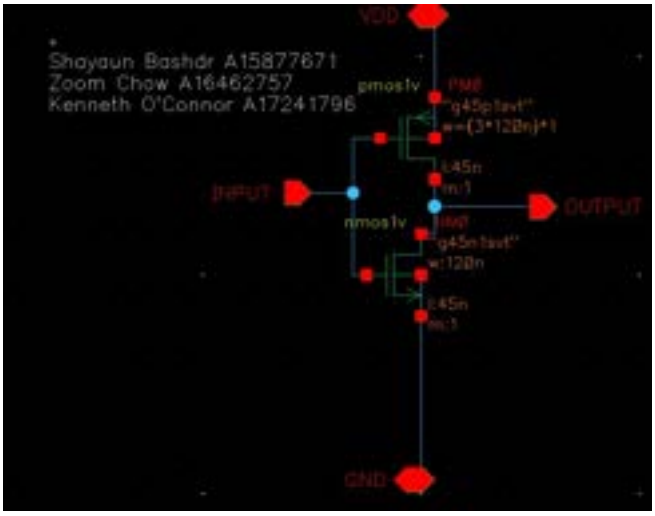

Figure 1a: High Skew Inverter

In following sections, we will highlight the use of such concepts to design and create a robust and high-speed adder from the transistor level schematic to the full 8-bit adder schematic.

PG Cell
The Propagate-Generate Cell (PG Cell) is a crucial component among the three distinct cells employed in the construction of our 8-bit Kogge-Stone adder. This cell serves as a pre-processing stage, responsible for computing generate and propagate signals for each pair of input bits, denoted as A and B. The generation of these signals follows a well-defined logic, governed by the following equations:

$$p_i = A_i \oplus B_i$$
$$g_i = A_i \,.\, B_i$$

Figure 1b represents the logic diagram that implements the described logic equations. This diagram illustrates the circuitry responsible for computing the generate and propagate signals for each bit pair in the

PG Cell. It provides a clear visual understanding of how these signals are generated, contributing to the operation of our 8-bit Kogge-Stone adder.

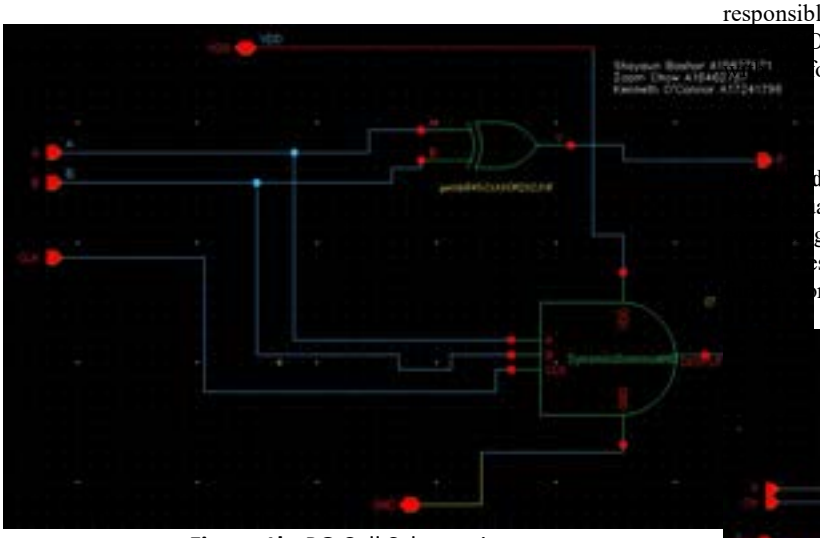

Figure 1b: PG Cell Schematic

FCO Cell

The Fundamental Carry Operator Cell (FCO Cell) serves as the second key component within our 8-bit Kogge-Stone adder. It can be conceptualized as a block comprising a group propagate and generate structure, or as a carry look-ahead network. These intermediate signals, denoted as group propagate and generate, play a pivotal role in the adder's operation, and are determined by the following logic equations:

$$P_{i:j} = P_{i:k+1} \,.\, P_{k:j}$$
$$G_{i:j} = G_{i:k+1} \vee (P_{i:k+1} \,.\, G_{k:j})$$

Figure 1c represents the logic diagram that implements the described logic equations. This diagram illustrates the circuitry responsible for computing the group generate and propagate signals from prior PG Cells. It provides a clear visual understanding of how these signals are generated, further contributing to the operation of our 8-bit Kogge-Stone adder.

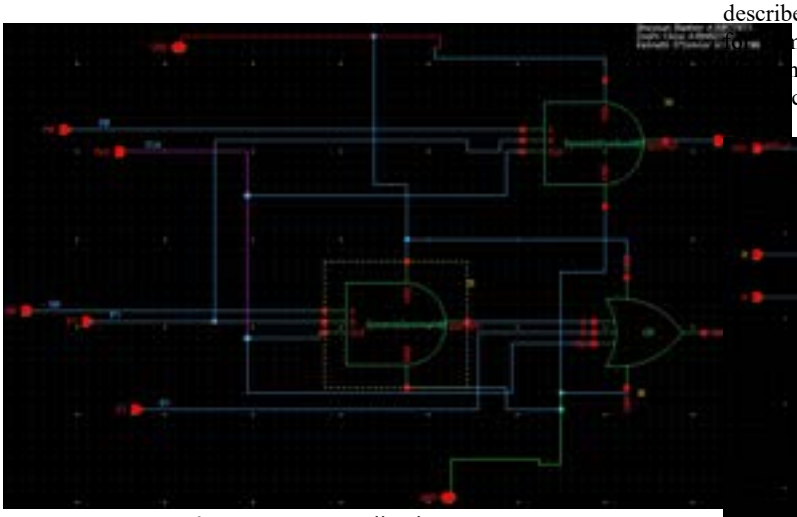

Figure 1c: FCO Cell Schematic

PPC Cell

The Post Processing Carry cell (PPC Cell) is one halve of the third and final distinct cell to complete the construction of our 8-bit Kogge-Stone adder. This cell serves as a part of the post-processing stage, responsible for the carry out values given the P and G values from the O stage as well as the carry in. The signals can be illustrated ollowing logic equation:

$$C_i = (P_i \,.\, C_{in}) \vee G_i$$

d represents the logic diagram that implements the described ation. This diagram illustrates the circuitry responsible for g the carry out from prior FCO cells along with the carry in. s a clear visual understanding of how the signal is generated, ntributing to the operation of our 8-bit Kogge-Stone adder.

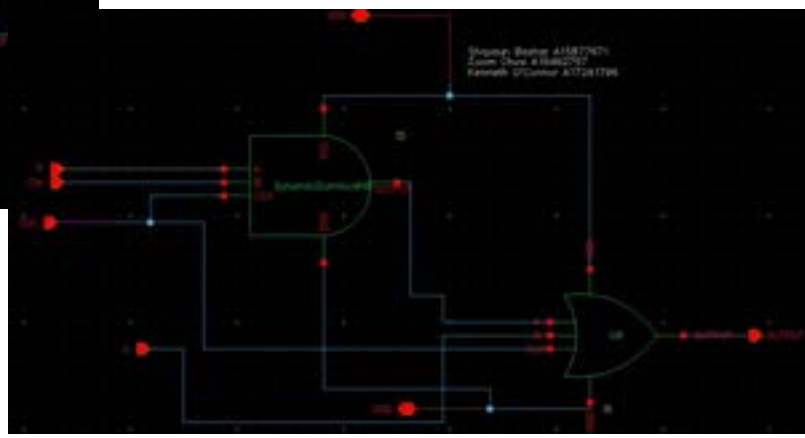

Figure 1d: PPC Cell Schematic

PPS Cell

The Post Processing Sum Cell (PPS Cell) is the second halve of the third and final distinct cell to complete the construction of our 8-bit Kogge-Stone adder. This cell serves as a part of the post-processing stage, responsible for the output of the sum values. The signals can be illustrated with the following logic equation:

$$Sum_i = p_i \oplus C_{i-1}$$

Figure 1e displays the transistor level schematic that implements the described logic gate. This diagram illustrates the circuitry responsible nputing the sum from the current propagate and the previous n signals. It provides a visual understanding of how the final computed for the output of the Kogge-Stone adder.

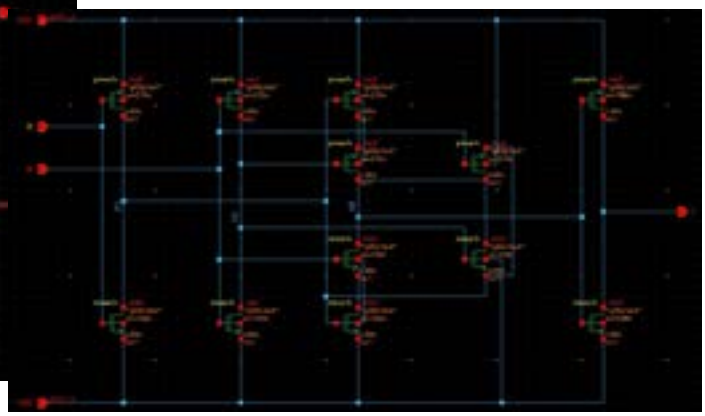

Figure 1e: PPS Cell Schematic

Important Circuits

OR Gate

Figure 2a shows the transistor level schematic of the OR gate utilizing dynamic and domino logic to provide lower logical effort and therefore faster operation. The OR gate is found in multiple cells that are crucial for the operation of the 8-bit adder such as the PPC and FCO cell. This custom designed gate also utilizes a high skew inverter, favoring a critical rising edge on the output, and using less logical effort compared to an un-skewed inverter.

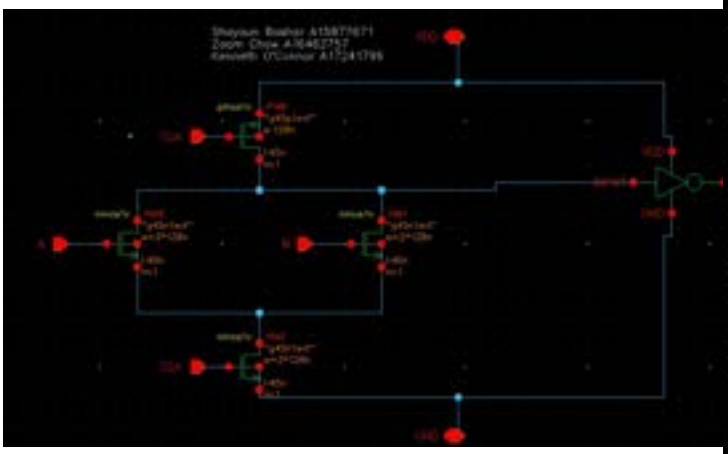

Figure 2a: OR Gate with High Skew Inverter and Dynamic & Domino Logic

AND Gate

Figure 2b shows the transistor level implementation of the AND gate found in the PG, PPC and FCO cell. This gate is utilizing dynamic and domino logic, giving way to faster operation. We have placed a high-skew inverter at the output of the AND gate to favor a rising edge transition to be able to take advantage of the lower logical effort needed for the gate.

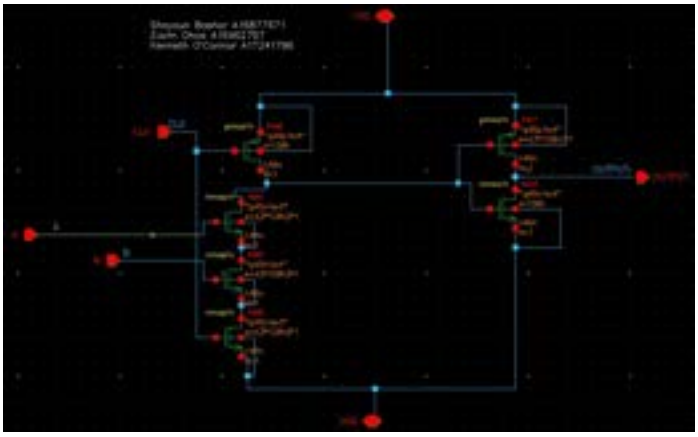

Figure 2b: AND Gate with High Skew Inverter and Dynamic & Domino Logic

8-Bit Adder

The purpose of an 8-bit adder is to, unsurprisingly, perform accurate and efficient binary addition for 8-bit data. It serves as a fundamental component in various digital systems and can be scaled to perform arithmetic of larger quantities. The advantageous architecture of our Kogge-Stone adder combines parallelism, scalability, and regularity to enhance its performance. The architecture incorporates distinct cells, comprised of the aforementioned innovative logic gates, that play crucial roles in achieving the desired efficient carry and propagation and overall functionality. Shown in **Figure 3** are the PG Cells serving as the pre-processing stage, the FCO Cells as an intermediary stage, and the PPC and PPS Cells as the post-processing stage which ultimately yields our final result. From integrating these cells to complete the Kogge-Stone adder architecture it can easily be seen as an attractive choice for high-speed arithmetic units in modern digital systems.

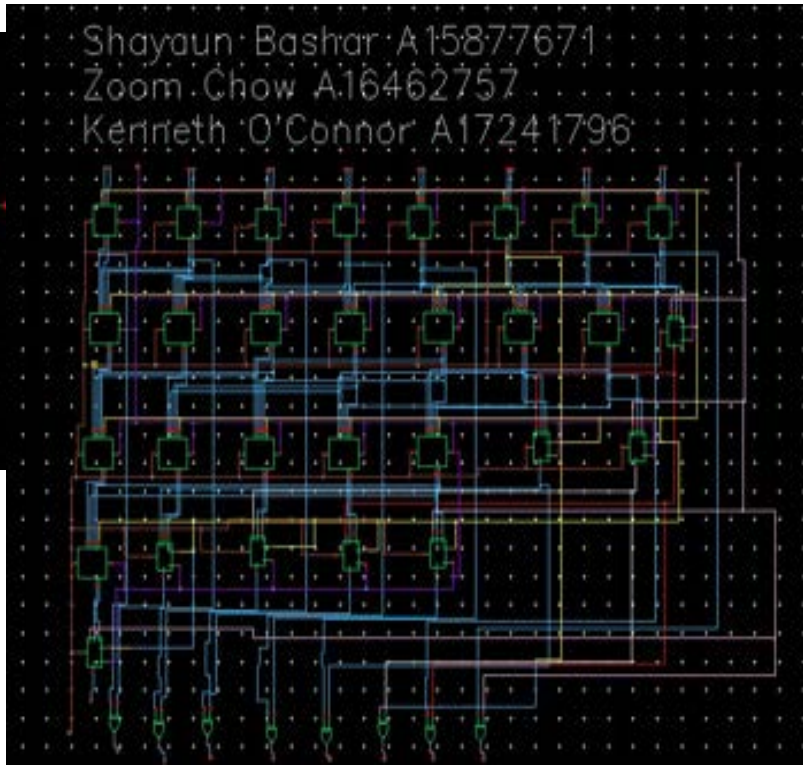

Figure 3: 8-bit Kogge-Stone adder schematic

Transient Simulation

As claimed prior, our innovative design implementing dynamic logic, domino logic, and high skew inverters enables a faster clocked 8-bit adder in comparison to the one generated in Lab 4. To be specific, the transient simulation of the Place and Route Adder from Lab 4 can be seen in **Figure 4a** with a maximum frequency of 2.2GHz. Following this is **Figure 4b** depicting the transient simulation for our custom designed 8-bit Kogge-Stone adder with a maximum frequency of 3GHz. **Figure 4c-4e** show the transient current for each adder at each clock frequency we measured for.

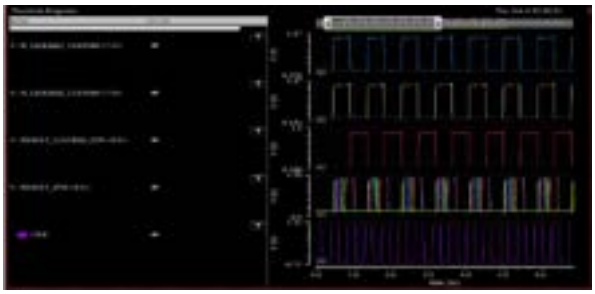

Figure 4a: Transient Simulation of Place and Route Adder @2.2GHz (Max Frequency)

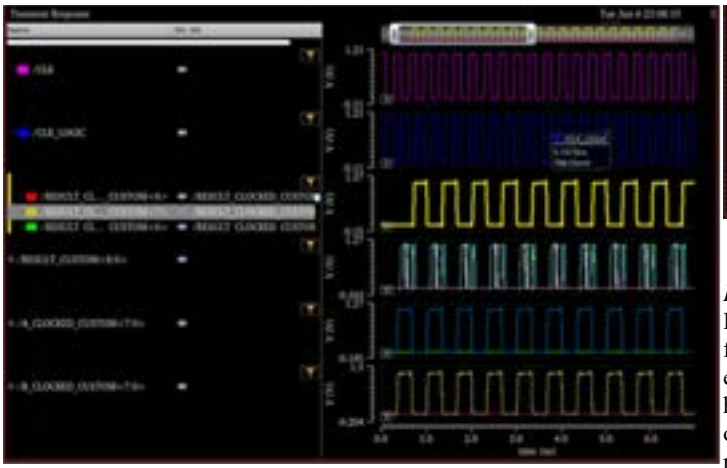

Figure 4b: Transient simulation of Custom Kogge-Stone Adder @3GHz (Max Frequency)

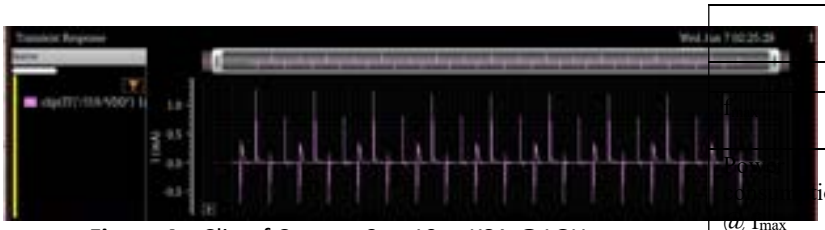

Figure 4c: Clip of Current 3ns-19ns KSA @1GHz

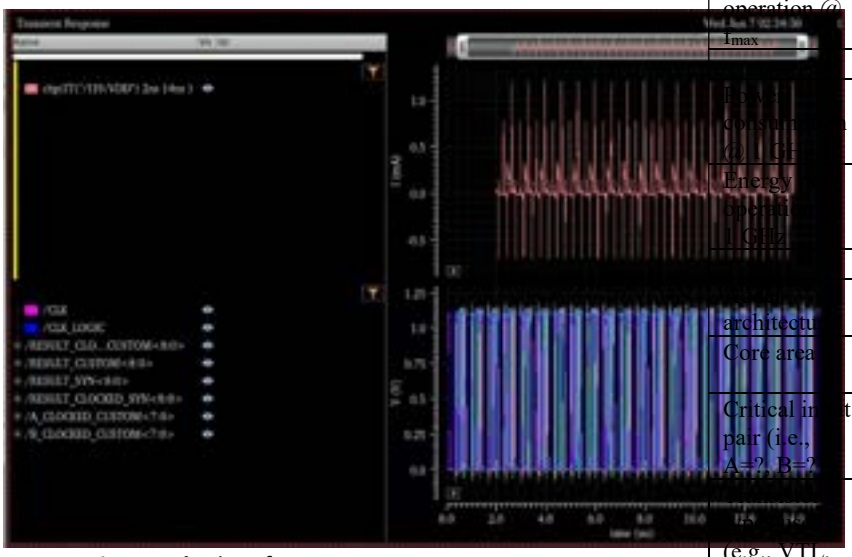

Figure 4d: Clip of Current 2ns-14ns KSA @3GHz

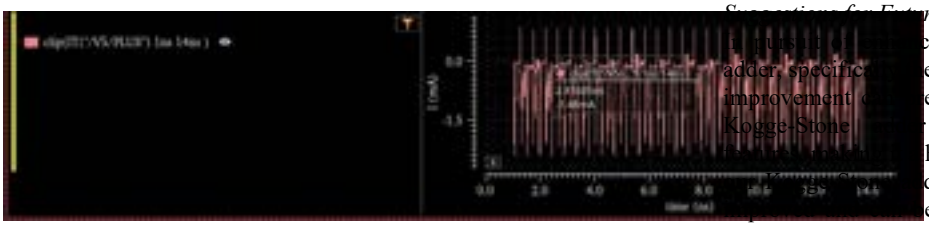

Figure 4e: Clip of Current 1ns-14ns PnR @2.2GHz

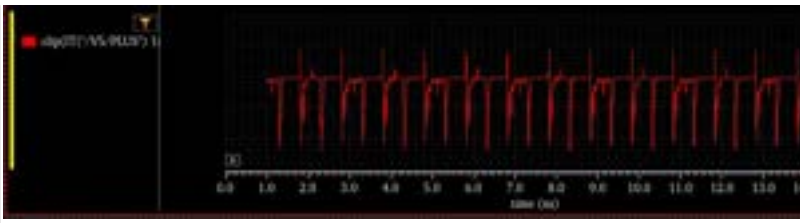

Figure 4f: Clip of Current 1ns-14ns PnR @1GHz

Results

In table 1, we can see that the custom adder can work at a higher clock frequency at a much lower power consumption energy per operation at either of the adder's individual maximum frequency or at 1GHz. We have confirmed that if we are competent designers that we can outperform the PnR adder by great margins. KGS adder is about 85% more efficient at its respective max clock frequency than PnR adder.

Table 1

	Place+route schematic	Custom design schematic	Place+route (optional)	Custom desi (optional)
	Performance for VDD = 1.1V			
	2.2Ghz	3GHz	NA	NA
ion @ 1max	529uW/per cycle	78.64uW/per cycle	NA	NA
Energy per operation @ 1max	240.5fJ/per cycle	26.9fJ/per cycle	NA	NA
	Performance for VDD = 1.1V, f_{CLK} = 1 GHz			
consum @ 1G	267uW/per cycle	29.12uW/per cycle	NA	NA
Energy operati GHz	267fJ/per cycle	29.12fJ/per cycle	NA	NA
	Other important parameters			
architectu	Ripple-carry	Kogge-Stone Parallel Prefix	Ripple-carry	Kogge-Stone Parallel Prefi
Core area	----------	----------	NA	NA
Critical in t pair (i.e., A=?, B=?	A=00000001 B=00111111	A=00000001 B=00111111	NA	NA
(e.g., VTL, VTG)	Normal Threshold Voltage	Normal Threshold Voltage	NA	NA

Suggestions for Future Iterations:

In this [illegible] cing future iterations of the 8-bit parallel prefix adder, specific[illegible]e Kogge-Stone adder, identifying areas for further improvement [illegible]esent certain challenges. The architecture of the Kogge-Stone [illegible] already incorporates several advantageous [illegible] highly efficient design. However, when designing [illegible]der, we realized that there are areas that can be [illegible]e noted as suggestions for future designs of the adder. The first suggestion that we encountered would be the sizing of the transistors used in each gate. While we used the given sizes in the gpdk045 process design kit, simulating each transistor using parametric analysis to find the optimal sizing would be the most ideal. Additionally, rerouting wires and combining the FCO blocks in the adder can result in a faster adder. For example, we can figure that the generate bit of 6 and 7 need generate bits from the calculation of bit 0

to 5 but instead, we can give it the generate bit calculated using bit 0 to 6 and will obtain the same result, since the generate bit from 6 is already included but is being added again from the computation of generate bit of 0 to 6 and this does not affect the result. Using this technique, we can increase the speed of operation of the overall 8-bit adder.

Lessening power consumption and area of the adder is another way we can improve our design if, for example, the adder was going to be used commercially. By rerouting the FCO blocks and combining redundant blocks in the tree section of the adder we are effectively making the area of the adder smaller.

Another method to improve future iterations is the implementation of timing-driven placement techniques during the physical layout design phase. Although not particularly explored in this report for our 8-bit KSA, the physical placement of the components within the adder has a significant impact on the overall timing and performance. By considering timing constraints and optimizing the relative placement of the cells, critical paths can be shortened, reducing propagation delays, and enabling a faster signal propagation.

Lastly, designing our own custom XOR gate would be a suggestion for future designs of the 8-bit adder. Using hi-skew inverters, domino logic and dynamic logic to implement the gate would be a way to lessen the logical effort of the gate and thus increasing the speed of our design. Simulating each transistor to find the optimal size would also decrease the delay through the gate.

Conclusion

In conclusion, our implementation of an 8-bit Kogge-Stone adder using innovative techniques like dynamic logic and domino logic for our custom logic gates, along with high skew inverters, has demonstrated remarkable potential in achieving higher performance while ensuring robust functionality. By leveraging dynamic logic and domino logic, the adder benefits from reduced transistor count and improved speed, enabling efficient binary addition. Furthermore, the integration of high skew inverters ensures optimal timing synchronization, allowing our adder to achieve a maximum clock frequency of 3GHz; roughly a 36% increase from the adder generated from Lab 4 clocking at 2.2GHz. Had our design choice been to utilize the standard cells provided in the gsc045lib process design kit it would have negatively affected the adder's performance in regard to speed due to the static CMOS nature of the cells. Thus, our careful consideration and evaluation of the PDK were crucial in forging our path towards an enhanced Kogge-Stone adder to meet our desired specification for a faster adder.

Contributions:

In the design and implementation of an 8-bit Kogge-Stone adder contributions were made by Shayaun Bashar, Zoom Chow, and Kenneth O'Connor. The contributions encompass both conceptual advancements and practical implementations, providing valuable insights and improvements to the field of high-performance digital arithmetic circuits.

Research done prior to deciding design approaches and implementations was done by Shayaun Bashar, Zoom Chow, and Kenneth O'Connor. The logic gates OR and AND were designed and created Zoom Chow and Kenneth O'Connor respectively. The cells used to assemble the KSA—PPC, FCO, and PG—were designed and created by Shayaun Bashar, Zoom Chow, and Kenneth O'Connor respectively. The entirety of the 8-bit KSA schematic was assembled by Shayaun Bashar. Debugging was done by Shayaun Bashar with assistance from Zoom Chow, Kenneth O'Connor, and Professor Patrick Mercier. Performance evaluation via running simulations was completed by Shayaun Bashar and verified by Kenneth O'Connor. Shayaun Bashar wrote out results in **Table 1,** a section in *Domino Logic and the Results response*. Zoom Chow wrote out the *Introduction*, *Previous Architecture*, *Chosen Architecture and Advantages*, *High Skew Inverter*, *PG Cell*, *FCO Cell*, *PPC Cell*, *PPS Cell*, *8-Bit Adder*, *Transient Simulation*, *Conclusion*, and *Contributions* sections. Kenneth O'Connor wrote out the *OR Gate* and *AND Gate* sections. Zoom Chow and Kenneth O'Connor wrote out the *Dynamic Logic*, *Domino Logic*, and *Suggestions for Future Iterations* sections. Shayaun Bashar provided all images used as figures in this report.

Acknowledgments:

Professor Patrick Mercier and Teaching Assistants Shihkai Kuo, Hongyu Lu, and Abhishek Vasudevan.

References:

[1] https://www.diva-portal.org/smash/get/diva2:19722/fulltext01
[2] https://iopscience.iop.org/article/10.1088/1742-6596/1049/1/012077/pdf
[3] https://www.cl.cam.ac.uk/research/srg/han/ACS-P35/8-bit-KoggeStone-Adder.pdf
[4] https://www.youtube.com/watch?v=ivry9K_7HzM
[5] https://www.youtube.com/watch?v=tgY4UIufLd0
[6] https://link.springer.com/chapter/10.1007/978-1-4757-3705-9_4
[7] https://www.researchgate.net/publication/336825176_Comparitive_Analysis_of_Brent-Kang_Kogge-Stone_Parallel-Prefix_Adder_for_Their_Area_Delay_Power_Consumption
[8] http://www.ijeetc.com/v3/v3n1/13_A0141_(p.116-121).pdf