ECE 165 Project: Optimized Kogge Stone Adder

Jonathan Mi A15849618
Mikhail Rossoshanskiy A14967680

In this project we design an 8-bit adder from the transistor level up. This design challenge tests our understanding of transistor-level digital circuits. From the top-level down, we need to understand the functionality and implementation of an adder. Then, we implement the adder through Cadence Virtuoso software by designing the logic gates on a transistor level, paying close attention to parameters such as sizing and number of transistors. The goal is to produce the fastest possible adder with the power limit. Since layout is not required for this project, we can disregard any parasitics from routing. Finding the proper sizing of transistors and gates to minimize delays within the adder, most importantly the critical path of the adder, is key to achieving high-speed performance with a power-efficient design. Any kind of CMOS logic can be used, so tradeoffs between logic types, notably static CMOS vs dynamic logic, must be considered.

The most basic 8-bit adder is the ripple carry adder. This adder works by adding each bit of the two inputs as well as the carry-in from the previous stage. Each of these stages is an implementation of a full adder. Even with a perfectly optimized transistor-level full adder, the ripple carry adder is still the slowest architecture due to the long critical path from the carry-in bit to the last sum-out bit. Each stage of the full adder must wait for the carry-out of the previous full adder with the critical input. Further developments using a carry-look-ahead are able to increase the speed of a ripple carry adder by using propagate and generate logic. These adders, however, still have a considerable critical path that is very long.

Further developments using propagate and generate (PG) logic have resulted in tree adders. These adders are able to compute addition much faster due to highly optimized parallel computation of group propagate and generate bits. The particular architecture that we have chosen to implement is the Kogge Stone Architecture. The Kogge Stone architecture is one of the fastest tree adders. The adder works in three stages: parallel preprocessing, PG logic, and sum bit calculation. In the parallel preprocessing stage, the propagate (Pi) and generate (Gi) bits for each pair of bits of the input are calculated. Then, in the PG logic stage, the Kogge Stone architecture calculates successively wider group propagate and group generate bits. Due to the very efficient use of black and gray cells, the adder only has 3 rows of logic. The critical path of the adder only runs through two black and one gray cell from the output of the parallel bitwise PG output to the input of the sum bit calculation. The final stage of the Kogge Stone architecture is the parallel sum bit calculation. Here, the current propagate bit is XOR'd with the previous bit group generate. While this stage may appear to be calculated in parallel, they are actually only calculated as the previous bit group generates are propagated through the adder.

One of the largest downsides of the Kogge Stone Adder architecture is the long wires running between each of the black and gray cells. While this is a non-issue in a schematic-level implementation of the adder, when layout is implemented the long wires lead to significant delays due to the increased capacitances associated with the long wires. Luckily, we are not required to implement the layout of the adder for this project, only a schematic level implementation so this downside of the Kogge Stone can be overlooked. Other architectures like the Brent-Kung adder have slightly longer critical paths but significantly shorter wire runs for lower post-layout capacitances.

In order to best optimize our adder, we implemented full custom transistor-level logic gates, PG logic, transistor sizing, and bubble pushing. For the logic gates, we implemented custom sized NAND, XNOR, INV, and BUF gates. These gates are utilized in the parallel bitwise PG calculations (Fig. 2c), as well as the final sum calculation (Fig. 2d). For the PG logic we implemented custom transistor level black and gray cells (Fig.1ab, 2ab). Rather than use logic gates to implement the black and gray cells, which requires 18 total transistors for the black cell, we were able to reduce it to 12 cells, significantly reducing not only the propagation delay overhead, but also the power consumption. You may notice the absence of non-inverted logic gates. This is because of the bubble pushing we performed on the PG logic. By using odd cells which output inverted group/bitwise PG bits given non inverted inputs in conjunction with even cells, which output non inverted group PG bits given inverted inputs, we were able to reduce the inverter count significantly. Sizing of each of the gates was also calculated to minimize the logical effort of the adder stages, and buffers were added where appropriate. The full adder can be seen in Fig. 3.

All of this optimization led to a significantly faster adder as compared to the PnR adder (Fig. 5b). The transient output of our adder can be seen in Fig. 5a. The power consumption of our adder is also very small due to significantly fewer inverters as a result of the bubble pushing. A summary of the results can be seen in the table in Fig. 6.

One problem we encountered was a prominent "lag" between the input and output waveforms. This issue stemmed from the digital flip-flop at the output that synchronizes the data to the clock. A workaround would be to design a more "ideal" flip flop to improve the delay between input and outputs. Another problem we experienced was due to complications in "ratioed logic". Assuming we are dealing with non-ratioed logic, there should be freedom in sizing between the pull-down and pull-up network. However, we experienced incorrect results when trying to size the logic for the optimal delay.

Reducing redundant black cells to create a modified Kogge Stone adder architecture can further decrease the delay associated with the critical path of the adder. Further improving the sizing to minimize logical effort can also be explored. To make the project more practical. The most important next step would be to implement the physical layout of the adder to analyze the parasitics of the adder.

Circuit schematic design and validation were done by Jonathan while optimization and documentation were performed by Mikhail.

Acknowledgments:
We would also like to thank Austin and Riley for their insightful discussion.

References:
W. N. H. E. and D. M. Harris, *CMOS VLSI Design: A circuits and systems perspective.* Noida: Pearson, 2015.

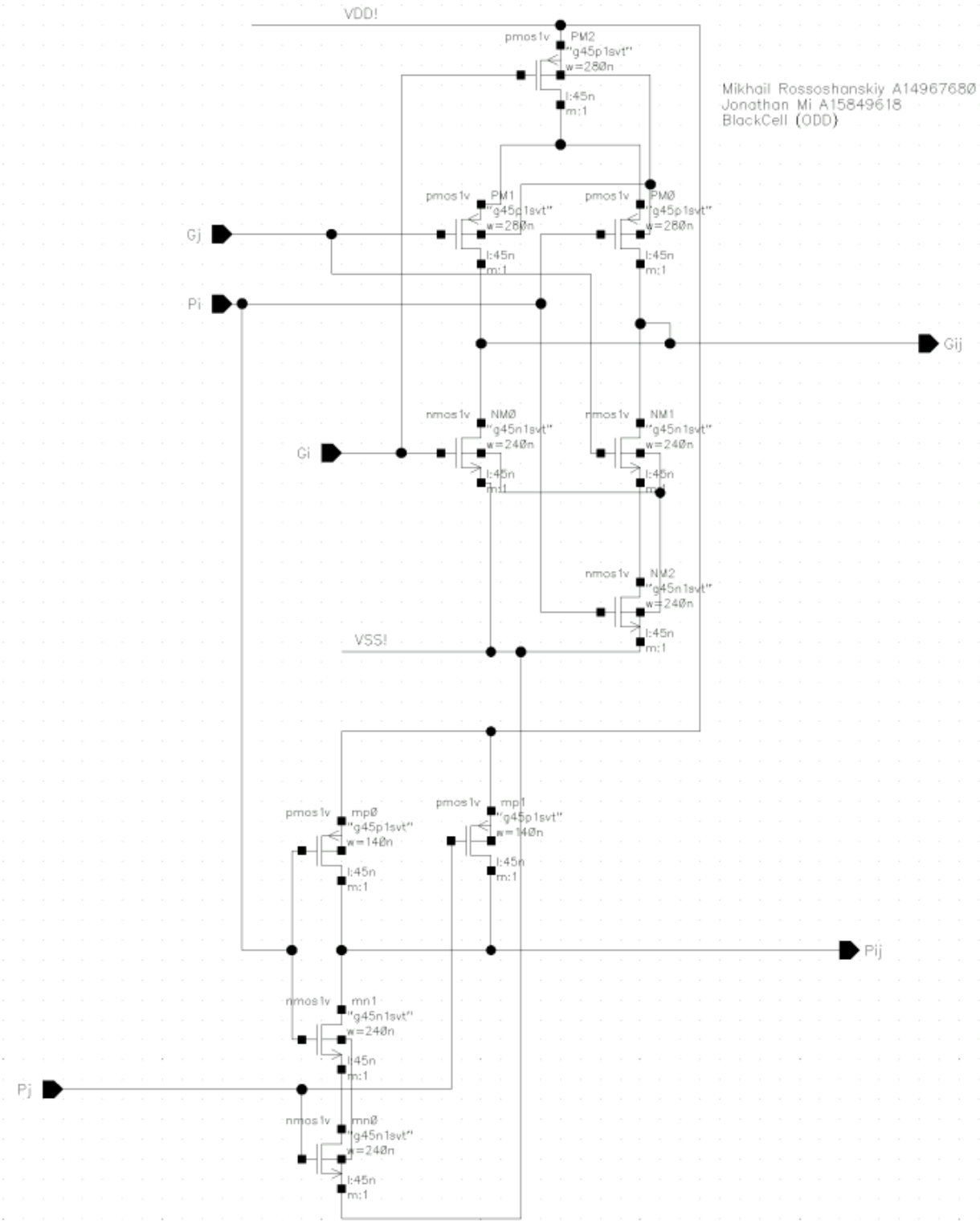


Figure 1a : Schematic of the largest repeated cell (Odd Black Cell). Features custom transistor level implementation to reduce the total number of transistors required.

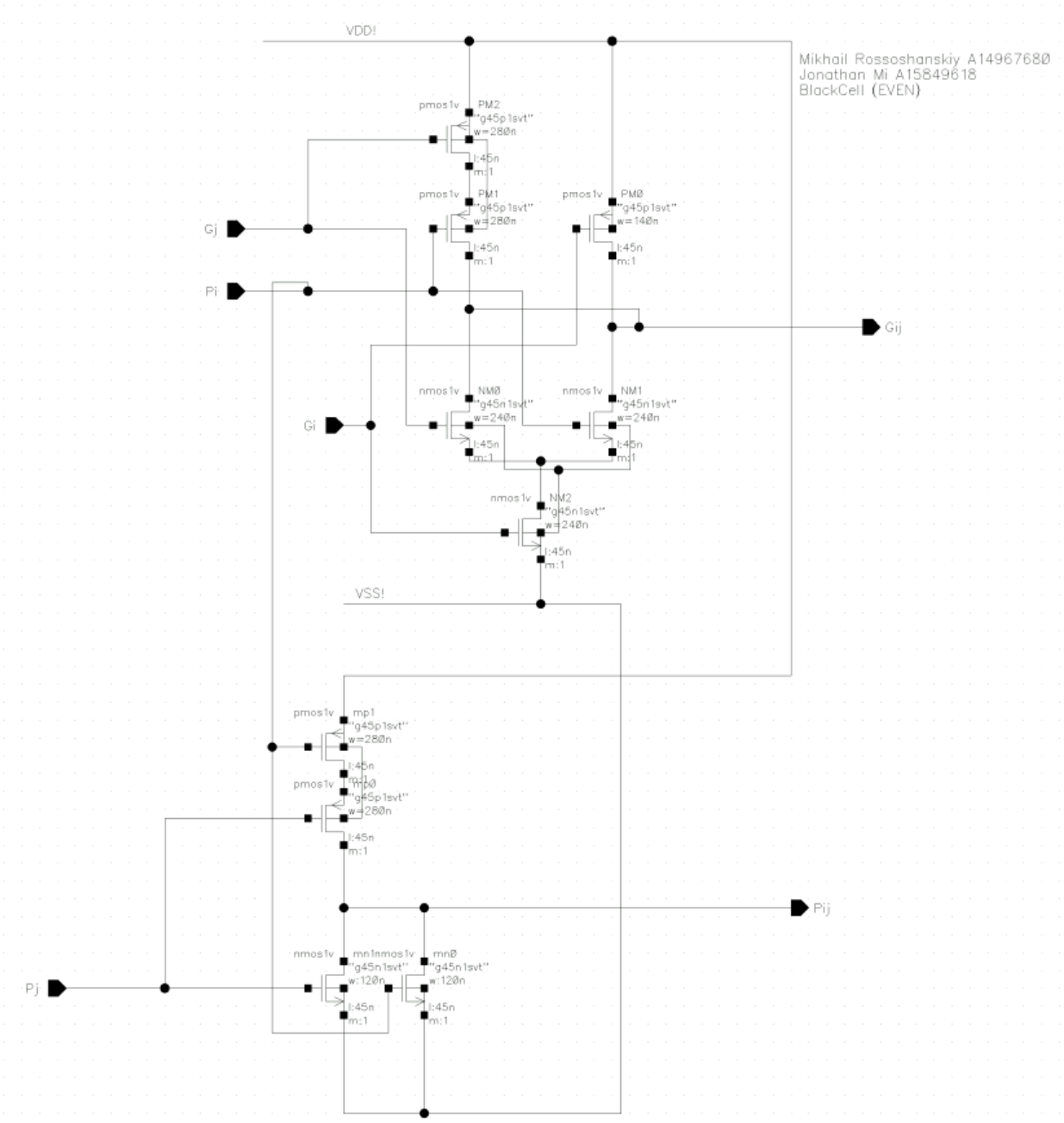


Figure 1b: Even Black Cell

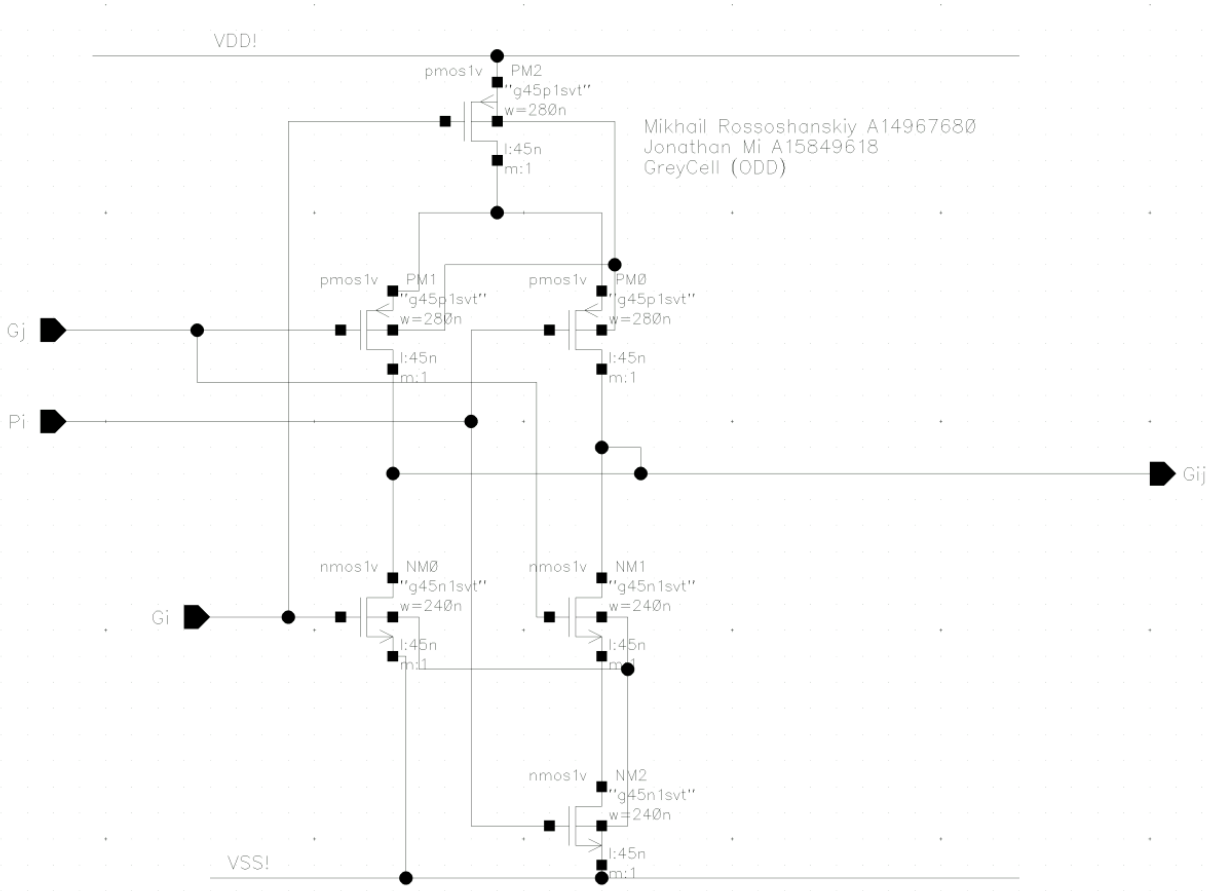


Figure 2a: Custom transistor level implementation of odd gray cell

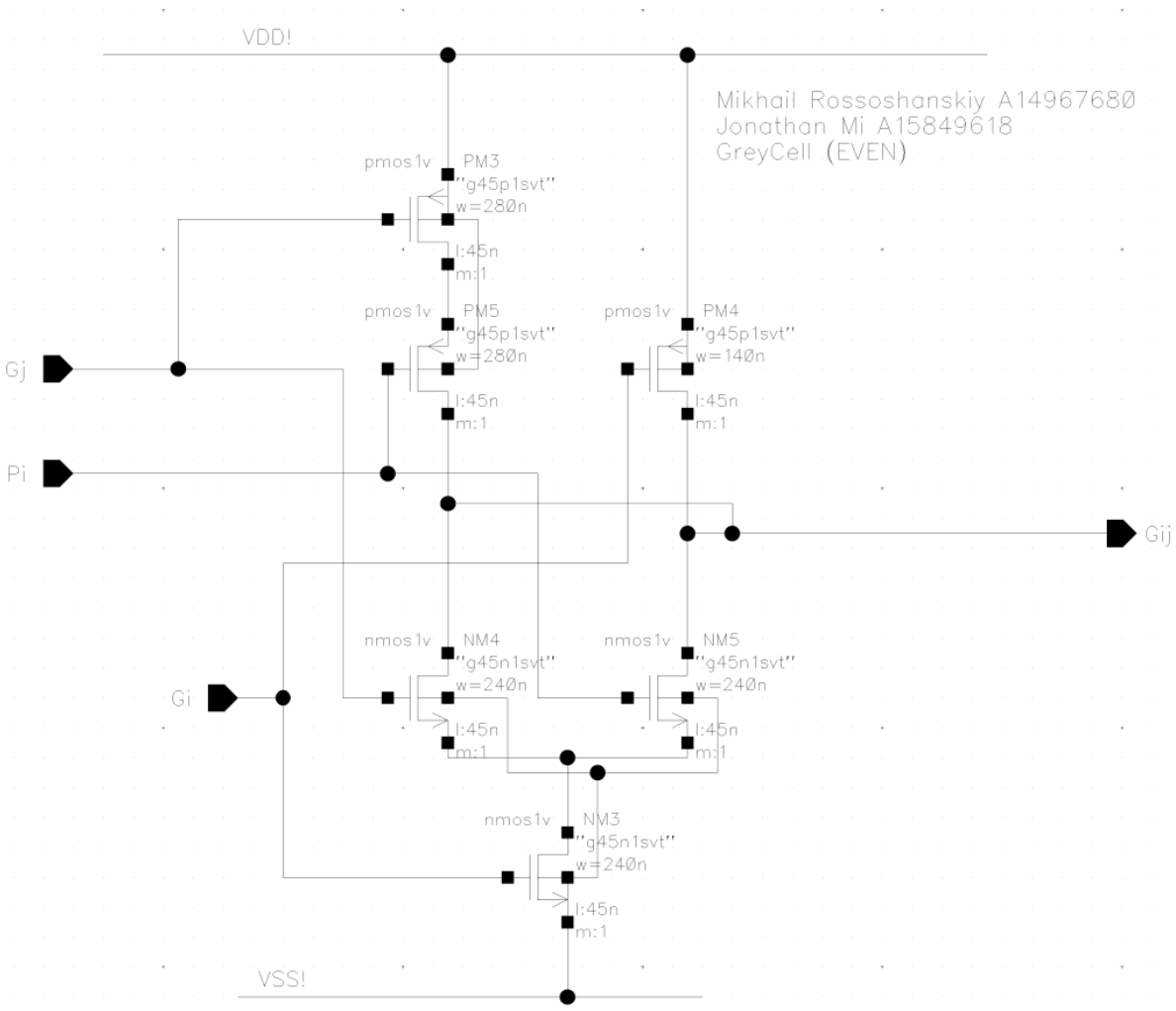


Figure 2b: Custom transistor level implementation of even gray cell

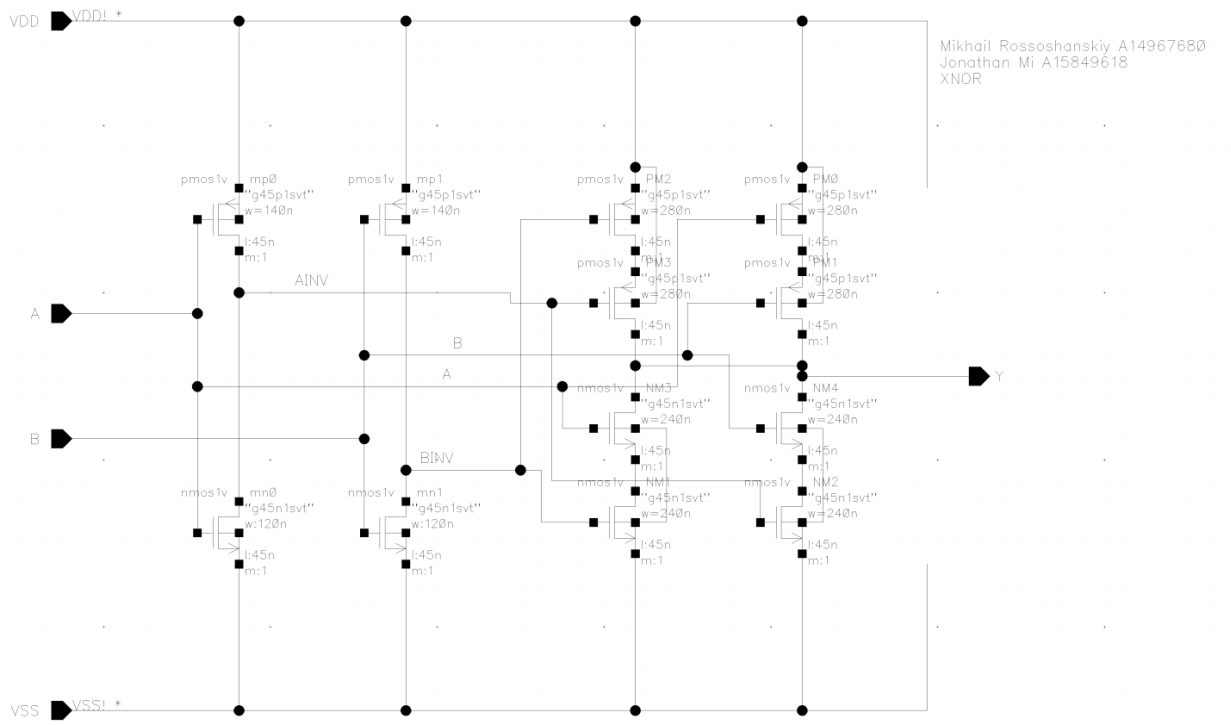


Figure 2c: Sum XNOR with custom sized transistor level implementation for minimum logical effort

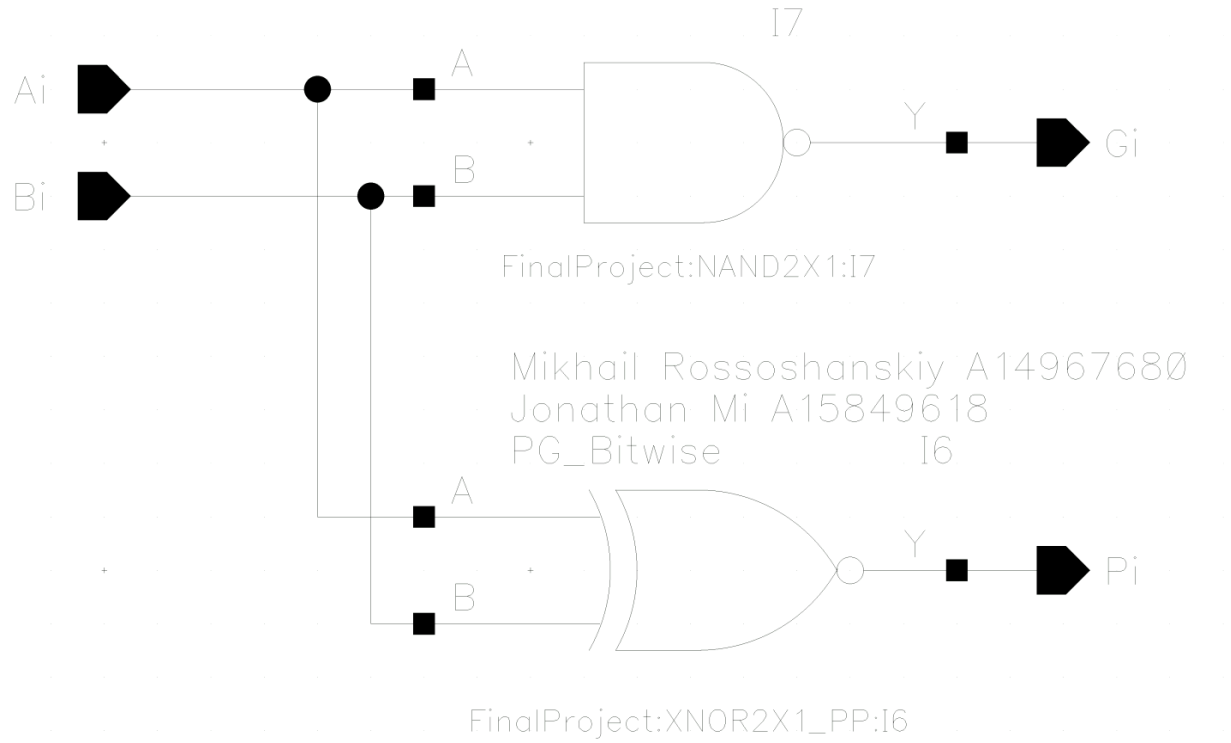


Figure 2d: Bitwise Pbar and Gbar calculation for parallel preprocessing with custom-sized transistor level implementation for minimum logical effort

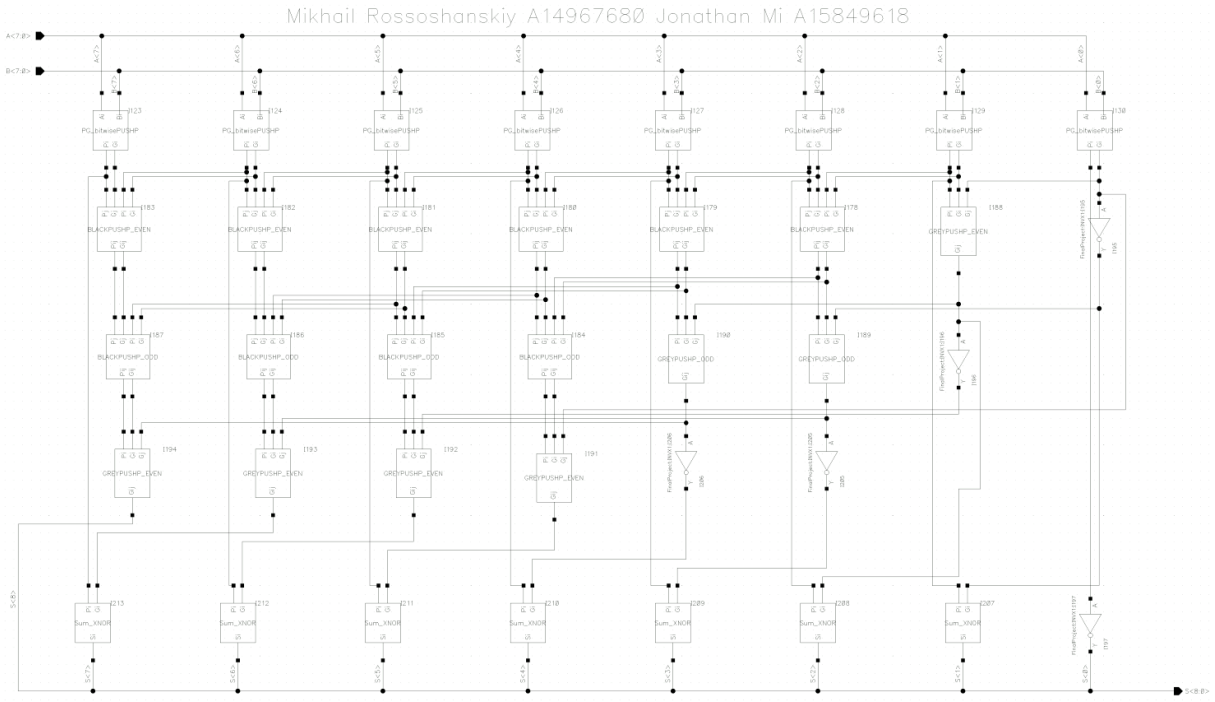
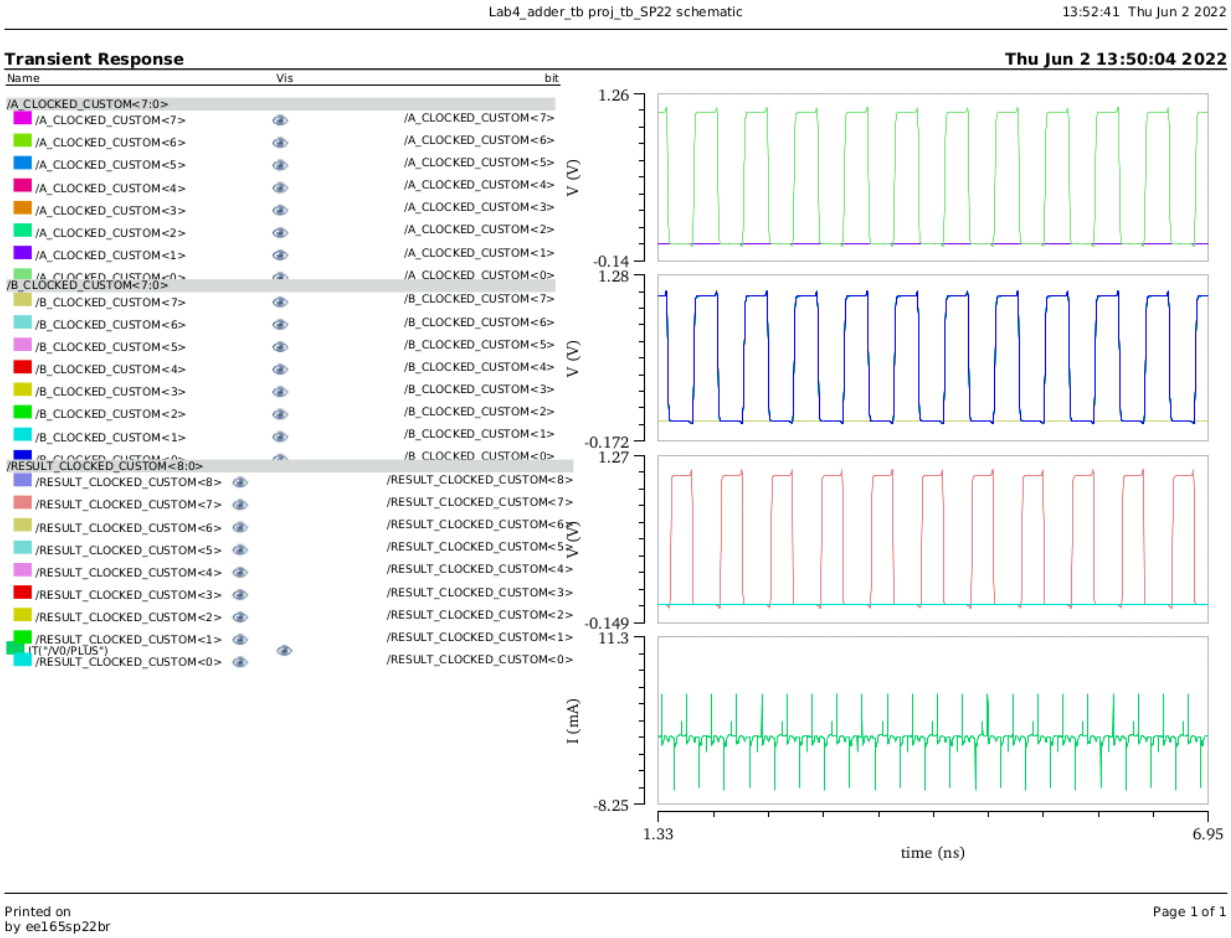


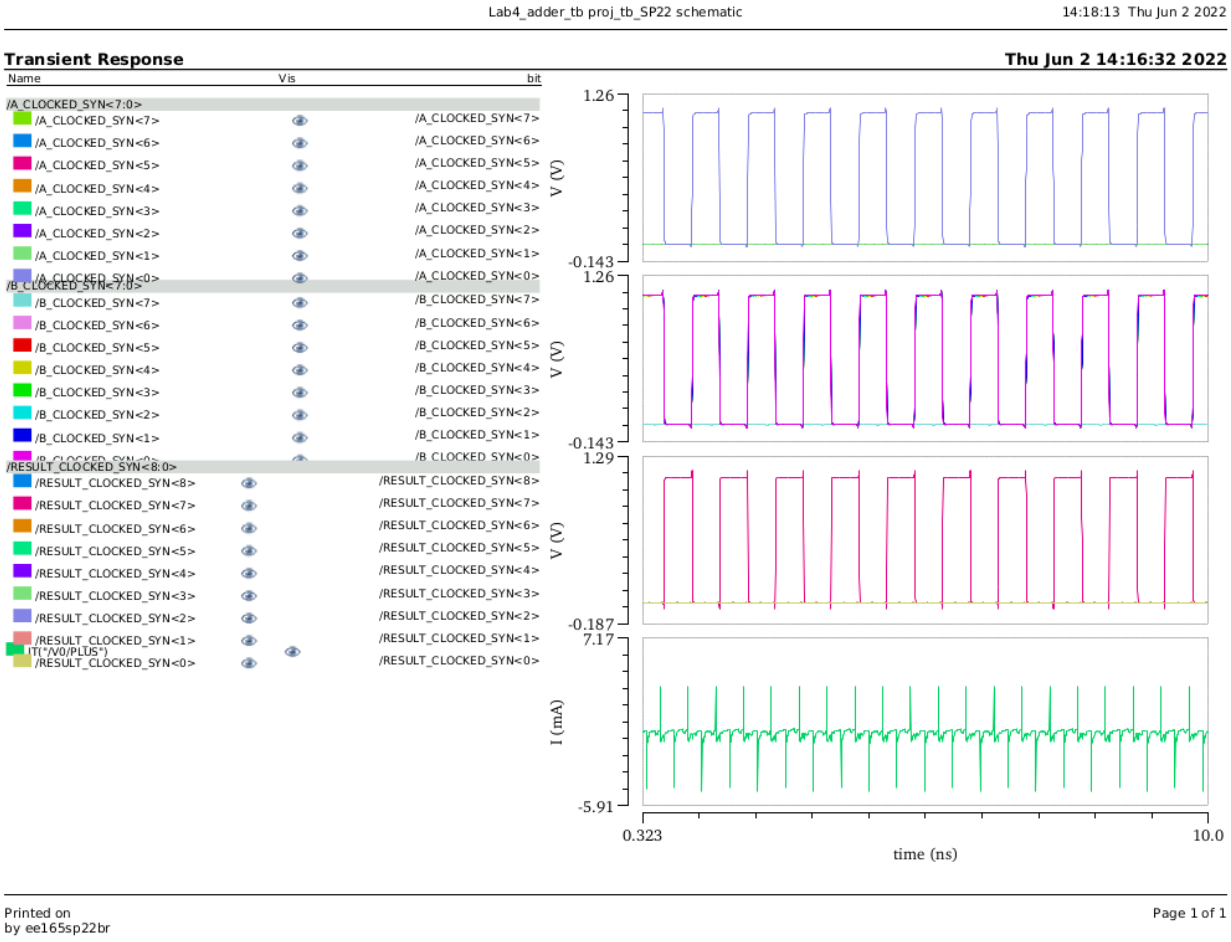
Figure 3: Schematic of custom 8-bit Kogge Stone Adder



A: 0000 0001
B: 0111 1111
S: 1000 0000
Cout: 0
fclk (max): 3.9Ghz
10 cycles:
T_start: 0.5n
T_end: 6n
Avg Power: 512uW

Figure 5a: Transient simulation of custom adder at the max working frequency (3.9GHz)

There are multiple critical inputs that have the same delay. Any input that passes through 2 black and one gray cell will be a critical input.



A: 0000 0001
B: 0111 1111
S: 1000 0000
Cout: 0
fclk (max): 2.1Ghz
10 cycles:
T_start: 0.5n
T_end: 10n
Avg Power: 289uW

Figure 5b: Transient simulation of PnR adder at max working frequency (2.1 Ghz)

	Place+route schematic	Custom design schematic
Performance for VDD = 1.1V		
f_max	2.1GHz	3.9GHz
Power cons. @ f_max	289uW	512uW
Energy cons. @ f_max	137fJ	131fJ
Performance for VDD 1.1V, f_clk = 1GHz		
Power cons. @ 1GHz	62uW	54uW
Energy cons. @ 1GHz	62fJ	54fJ
Other important parameters		
Architecture	Ripple Carry	Kogge Stone
Core area	n/a	n/a
Critical input pair	00000001 01111111	00000001 01111111
Transistor types used	p/nmos1v	p/nmos1v

Figure 6: Table of results