Brenden Moran

8 Bit Adder Created Using CMOS Transistor Design

Introduction:

This project is a basic design of an 8-bit adder created as an introduction into CMOS design for myself. For more information on how to design using CMOS logic gates I recommend this blog located at:
https://www.allaboutelectronics.org/cmos-logic-gates-explained/
This resource helped with my research and understanding of how to create different logic gates using CMOS design.

The first thing I want to introduce is the truth table for a 1-bit adder which is what helped me design the circuit as a whole. The truth table has 3 inputs two are the user inputs of A and B and the other is the carry in bit (C_{in}) from the previous 1-bit adder. The adder then has two outputs, S which is the value stored by A+B and C_{out} which is the carry out bit from the addition of A+B.

Table 1: Truth Table for 1-Bit Adder

C_{in}	A	B	S	C_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Piece by Piece:

I will now explain a small piece of the pie of how I created a one bit adder. In figure 1 below is the circuit for a 1-bit adder where it takes in input value A_1 and B_1 as well as the previous carry bit labeled $Carry_0$. The simplified function for the output S in the truth table above (Table 1) is $C_{in} \oplus (A \oplus B)$ or a 3 input XOR gate. The way I implemented it into my circuit is by having two XOR gates in series which then stores the value into OUT_1 (S).

The XOR gate works in CMOS design by having the two inputs in parallel with one another which are in series with the negation of the two same inputs which are connected to a pull up resistor as seen in the upper half portion of the gate in Figure 1. As for the lower half of the gate there are two inputs in series with one another which are then in parallel with the negation of the two inputs in series. All of which are connected to a pull down resistor as seen in the lower half of Figure 1.

As for how I designed the carry bit, the simplification of C_{out} output in Table 1 is $C_{in}A + C_{in}B + AB$. We can have this use slightly less transistors in our design by converting both of the OR gates into NAND gates by using De Morgan's Law. This is because an OR gate uses 6 transistors while NAND gates use 4 transistors. This leaves us with the Boolean expression $((AB)'(AC_{in})'(BC_{in})')'$. The implementation of this to calculate the carry bit can be seen in Figure 2.

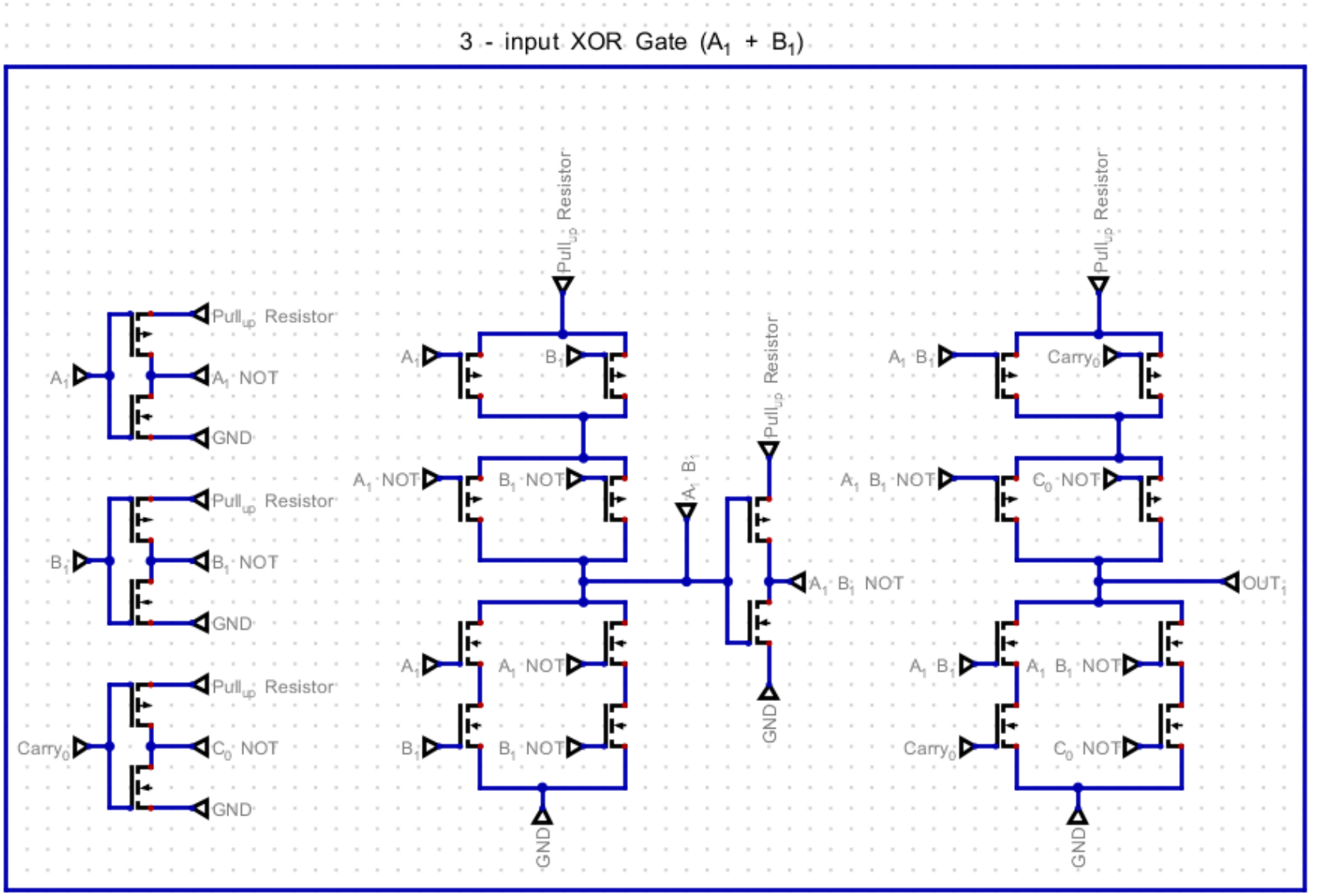

Figure 1: Adder circuit to add A_1 and B_1 and store the output into OUT_1.

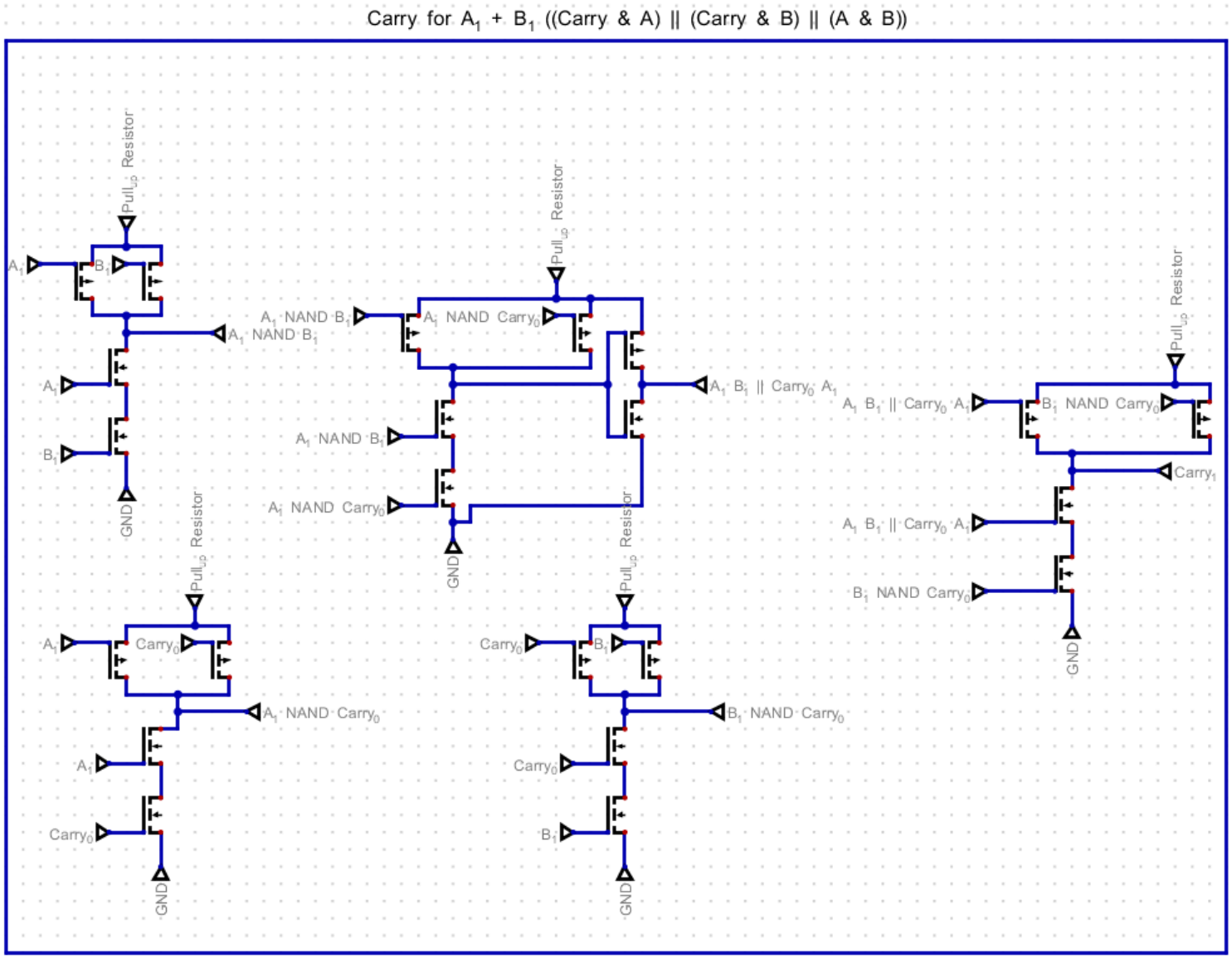

Figure 2: Circuit used to calculate the carry out of A+B.

Next Steps:

The final step is to use this same design for all other bits 2-7 making sure to connect the previous carry bit to the new adder circuit. It is important to note that I use a different circuit for bits 0 because there is no previous carry bit to worry about which makes the circuit much simpler as we can assume that the C_{in} value is 0.

Conclusion:

This project was a great beginner project to help introduce me into CMOS design and how many IC chips are created. This gave me a hands-on look at how IC chips may be designed and the steps taken in order to create larger circuits from small components. This project also helped with my skills of working backwards as I began with an adder from logic gates which I then created into CMOS design and had to optimize that design such that my design would use the least amount of transistors needed.