**CS 200: Computer Organization**

**Project 1: 3-bit Adder**

Shariq M. Jamil

Due: Wednesday, February 7, 2014

**Overview**

**Purpose**

This project required us to design a 3-bit adder using AND, OR or NOT gates using Logisim.

**Approach**

This project was a complete exercise in designing a circuit from scratch. We were provided an idea of how a full adder worked and, we used it to define input (X, Y, Carry In) and output variables (Carry out, Sum). A truth table was created using the variables and a Boolean function for each output was produced using it. We then used Karnaugh maps to simplify the functions and used the minimized functions to create a 1-bit full adder in Logisim. Once the 1-bit adder was assembled, copies of it were used as sub-circuits to create a 3-bit adder.

**Solution**

A truth table showing all combinations of inputs and outputs.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **X** | **Y** | **Cin** | **Sum** | **Cout** |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

Non-minimized Boolean functions for Carry-out and Sum, derived from the truth table.

Cout(X,Y,Cin) = XYCin’ + X’YCin + XY’Cin + XYCin

Sum = X’YCin’ + XY’Cin’ + X’Y’Cin + XYCin

Karnaugh Maps used to minimize both functions are displayed below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **C**out | **Y | Cin** |  |  |  |
| **X** | **00** | **01** | **11** | **10** |
| **0** | 0 | 0 | 1 | 0 |
| **1** | 0 | 1 | 1 | 1 |

Each one of the 1’s is grouped with the 1.

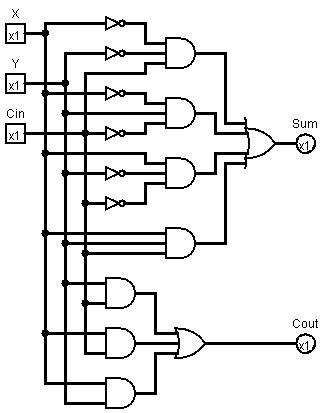
Cout(X,Y,Cin) = YCin + XCin + XY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sum** | **Y | Cin** |  |  |  |
| **X** | **00** | **01** | **11** | **10** |
| **0** | 0 | 1 | 0 | 1 |
| **1** | 1 | 0 | 1 | 0 |

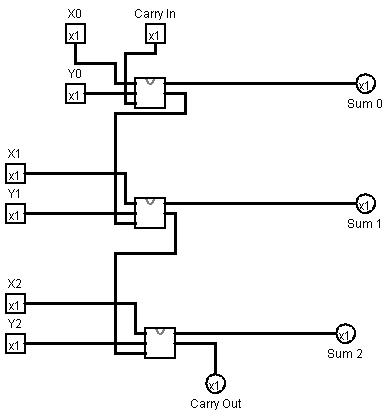
Sum(X,Y,Cin) = X’Y’Cin + X’YCin’ + XYCin + XY’Cin’

**Sample Output**

The 1-bit adder shown below was created first using the minimized functions derived using the K-maps.



The 1-bit adder was then used as a sub-circuit to create a 3-bit adder. The circuit is displayed below.



**Conclusion**

This was a great exercise for everything we have covered in class so far because it covered the use of truth tables, K-maps and circuit design. Creating truth tables and using K-maps to minimize Boolean functions was easy because of Homework 1 and Homework 2. It took some time to get used to Logisim’s interface. At first I accidentally used the input pins as output pins and it took me a while to figure out why my circuit was not working. It also took me time to figure out how to use the 1-bit adder as a sub-circuit because the tutorial was not very helpful. Overall, this was a great exercise and having the Logisim circuit confirm the truth table projections was very fulfilling.