**CS 303 Pre-Lab #3 Report**

Zeynep Yıldız

00029325

I started off with making Boolean algebraic statements for A>B, A<B and A=B

The logic diagram looked like this:

Diagram, schematic

Description automatically generated

* Picture is taken from <http://users.encs.concordia.ca/~asim/COEN_6511/Projects/final6511report.pdf>

I turned every gate into algebraic statements and what turned out is:

*assign w5 = (~(A30 ^ B30));*

*assign w6 = (~(A2 ^ B2));*

*assign w7 = (~(A1 ^ B1));*

*assign w8 = (~(A0 ^ B0));*

*assign w1 = (A30 & (~B30));*

*assign w2 = (A2 & (~B2) & w5);*

*assign w3 = (A1 & (~B1) & w5 & w6);*

*assign w4 = (A0 & (~B0) & w5 & w6 & w7);*

*assign w9 = ((~A30) & B30);*

*assign w10 = ((~A2) & B2 & w5);*

*assign w11 = ((~A1) & B1 & w5 & w6);*

*assign w12 = ((~A0) & B0 & w5 & w6 & w7);*

w is for wire and every wire is represented in the picture with s1,s2,…s8. There is not s9…s12 but they are for A<B. They represent those gates.

And I gathered those wires together for final gates that would determine the result.

// C0 = "A>B"

// C1 = "A=B"

// C2 = "A<B"

assign C0 = (w1 | w2 | w3 | w4);

assign C1 = (w5 & w6 & w7 & w8);

assign C2 = (w9 | w10 | w11 | w12);

But this solution does not work for signed integers. So I negate the first bit (A3 and B3). So I can compare the numbers more accurately.

Table

Description automatically generated

The picture and the idea are taken from:

<https://electronics.stackexchange.com/questions/519178/boolean-algebra-how-do-i-implenent-a-4-bit-signed-comparator>

As wanted, I followed the instructions given for the Xilinx and Vivado. I did simulations which worked perfectly. A screenshot of a computer

Description automatically generated with medium confidence

// **C0 = "A>B" C1 = "A=B C2 = "A<B"**

simulation code is below:

// A = 0001 B = 0010

A0 = 1; A1 = 0; A2 = 0; A3 = 0;

B0 = 0; B1 = 1; B2 = 0; B3 = 0;

#10;

// A = 0010 B = 0000

A0 = 0; A1 = 1; A2 = 0; A3 = 0;

B0 = 0; B1 = 0; B2 = 0; B3 = 0;

#10;

// A = 1001 B = 0001

A0 = 1; A1 = 0; A2 = 0; A3 = 1;

B0 = 1; B1 = 0; B2 = 0; B3 = 0;

#10;

// A = 1010 B = 1001

A0 = 0; A1 = 1; A2 = 0; A3 = 1;

B0 = 1; B1 = 0; B2 = 0; B3 = 1;

#10;

// A = 1110 B = 1101

A0 = 0; A1 = 1; A2 = 1; A3 = 1;

B0 = 1; B1 = 0; B2 = 1; B3 = 1;

#10;

// A = 0000 B = 0000

A0 = 0; A1 = 0; A2 = 0; A3 = 0;

B0 = 0; B1 = 0; B2 = 0; B3 = 0;

#10;

// A = 1010 B = 1010

A0 = 0; A1 = 1; A2 = 0; A3 = 1;

B0 = 0; B1 = 1; B2 = 0; B3 = 1;

#10;

// A = 0110 B = 0110

A0 = 0; A1 = 1; A2 = 1; A3 = 0;

B0 = 0; B1 = 1; B2 = 1; B3 = 0;

#10;

I did synthesis and implementation. Those files should be included inside the zip file.