8-bit Magnitude Comparator

Abstract

This project presents the design and implementation of an 8 bit magnitude comparator using digital logic gates. A magnitude comparator is a combinational circuit that compares two binary numbers and determines their relative magnitudes.This project presents the design, implementation, and verification of an 8-bit magnitude comparator using Verilog.

Introduction

A magnitude comparator is an essential component in digital systems, used in arithmetic operations, decision-making circuits, and control units. This project focuses on designing an 8bit comparator that takes two 8-bit binary inputs A and B and produces three outputs:

1. A > B
2. A < B
3. A = B

Background Analysis

An n-bit comparator works by:

1. Comparing the most significant bit (MSB) first.
2. If MSB values differ, decision is made immediately.
3. If equal, move to the next bit until all bits are checked.

For 8-bit binary numbers A7A6…A0 and B7B6…B0, the comparison follows:

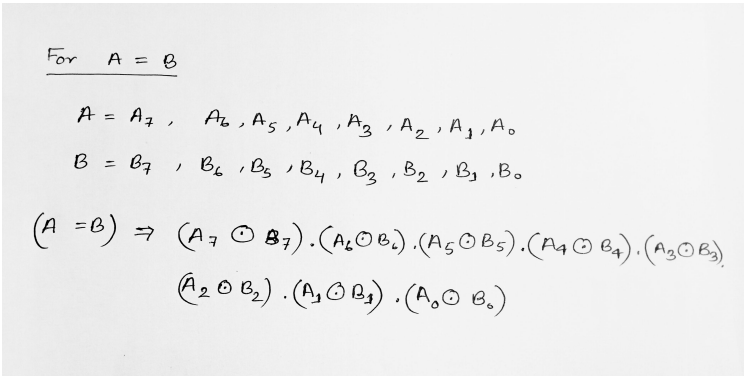
1.A > B → First position from MSB where A=1, B=0

2.A < B → First position from MSB where A=0, B=1

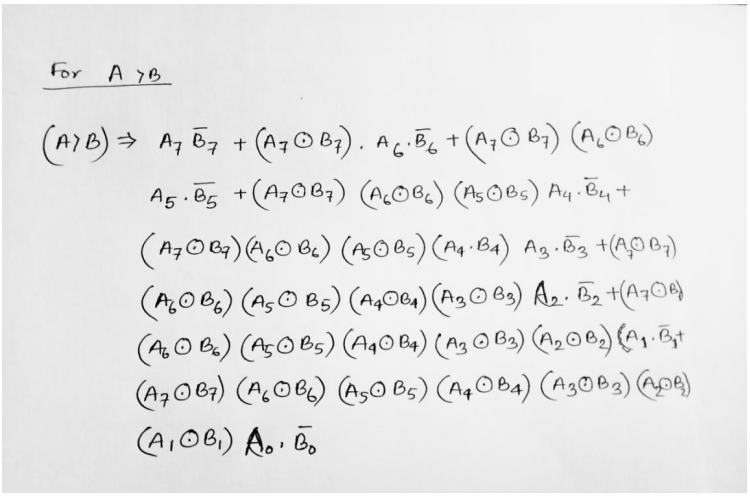
3.A = B → All bits equal

Logic Development

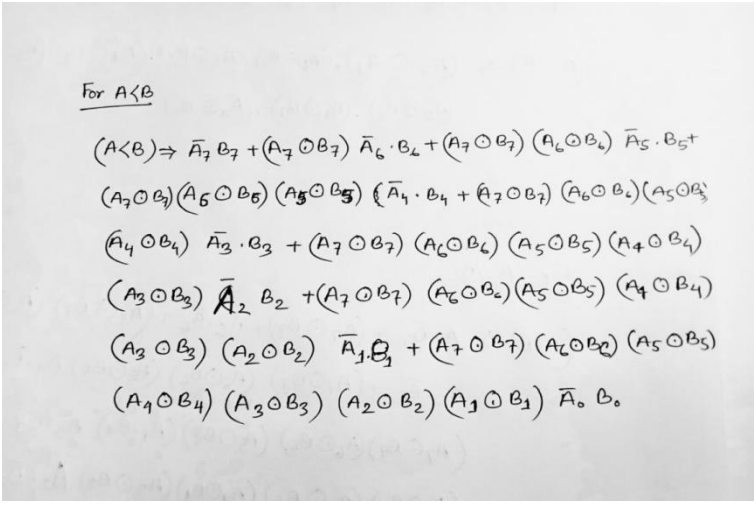
1.When A= B the logic will be,



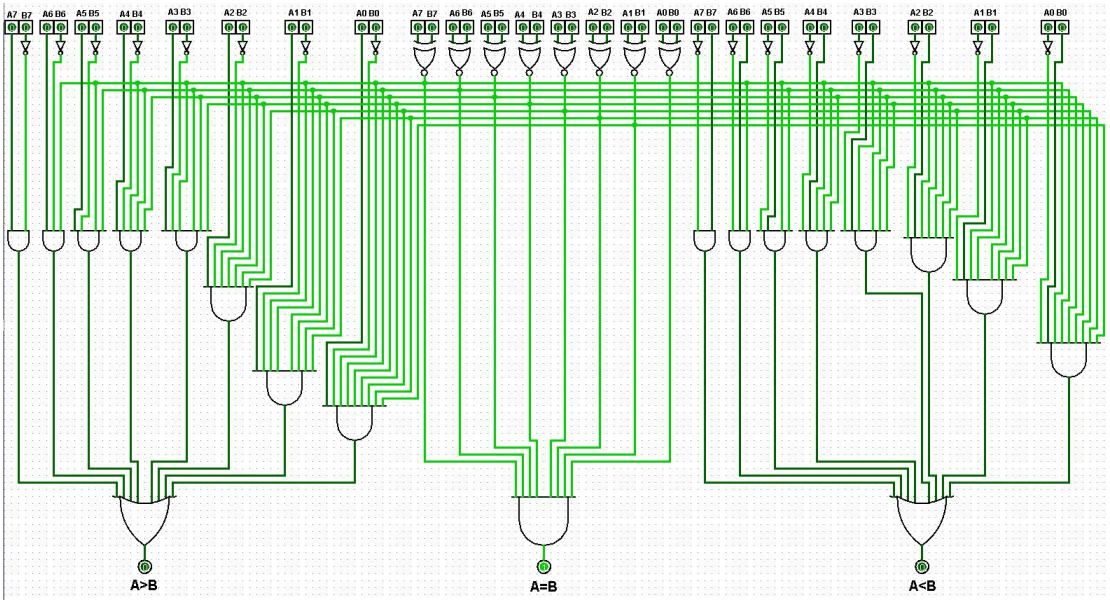
2. When A > B the logic will be,



3. When A < B the logic will be,



Circuit Diagram



Verilog Code

```
module 8bitmag(
   A7,A6,A5,A4,A3,A2,A1,A0,
   B7,B6,B5,B4,B3,B2,B1,B0,
   AeqB,AgtB,AltB
);
   input A7,A6,A5,A4,A3,A2,A1,A0;
   input B7,B6,B5,B4,B3,B2,B1,B0;
   output AeqB,AgtB,AltB;

   wire e7,e6,e5,e4,e3,e2,e1,e0;
   wire g7,g6,g5,g4,g3,g2,g1,g0;
   wire l7,l6,l5,l4,l3,l2,l1,l0;
   wire x0,x1,x2,x3,x4,x5,x6;
   wire y0,y1,y2,y3,y4,y5,y6;

   //xnor operation to check if A and B are equal
   xnor(e7, A7, B7);
   xnor(e6, A6, B6);
   xnor(e5, A5, B5);
   xnor(e4, A4, B4);
   xnor(e3, A3, B3);
   xnor(e2, A2, B2);
   xnor(e1, A1, B1);
   xnor(e0, A0, B0);
```

```
  //A AND B' to check if A is greater at each position
  and(g7, A7, ~B7);
  and(g6, A6, ~B6);
  and(g5, A5, ~B5);
  and(g4, A4, ~B4);
  and(g3, A3, ~B3);
  and(g2, A2, ~B2);
  and(g1, A1, ~B1);
  and(g0, A0, ~B0);

  //A' AND B to check if A is greater at each position
  and(l7, ~A7, B7);
  and(l6, ~A6, B6);
  and(l5, ~A5, B5);
  and(l4, ~A4, B4);
  and(l3, ~A3, B3);
  and(l2, ~A2, B2);
  and(l1, ~A1, B1);
  and(l0, ~A0, B0);

  //A = B
  and(AeqB, e7, e6, e5, e4, e3, e2, e1, e0);

  //A > B
  and(x0, e7, g6);
  and(x1, e7, e6, g5);
  and(x2, e7, e6, e5, g4);
  and(x3, e7, e6, e5, e4, g3);
  and(x4, e7, e6, e5, e4, e3, g2);
  and(x5, e7, e6, e5, e4, e3, e2, g1);
  and(x6, e7, e6, e5, e4, e3, e2, e1, g0);
  or(AgtB, g7, x0, x1, x2, x3, x4, x5, x6);

  // A < B
  and(y0, e7, l6);
  and(y1, e7, e6, l5);
  and(y2, e7, e6, e5, l4);
  and(y3, e7, e6, e5, e4, l3);
  and(y4, e7, e6, e5, e4, e3, l2);
  and(y5, e7, e6, e5, e4, e3, e2, l1);
  and(y6, e7, e6, e5, e4, e3, e2, e1, l0);
  or(AltB, l7, y0, y1, y2, y3, y4, y5, y6);

endmodule
```

Conclusion

The 8-bit magnitude comparator was successfully designed, implemented, and tested. The Verilog code was synthesized without errors, and simulation confirmed that the outputs were correct for all tested inputs.