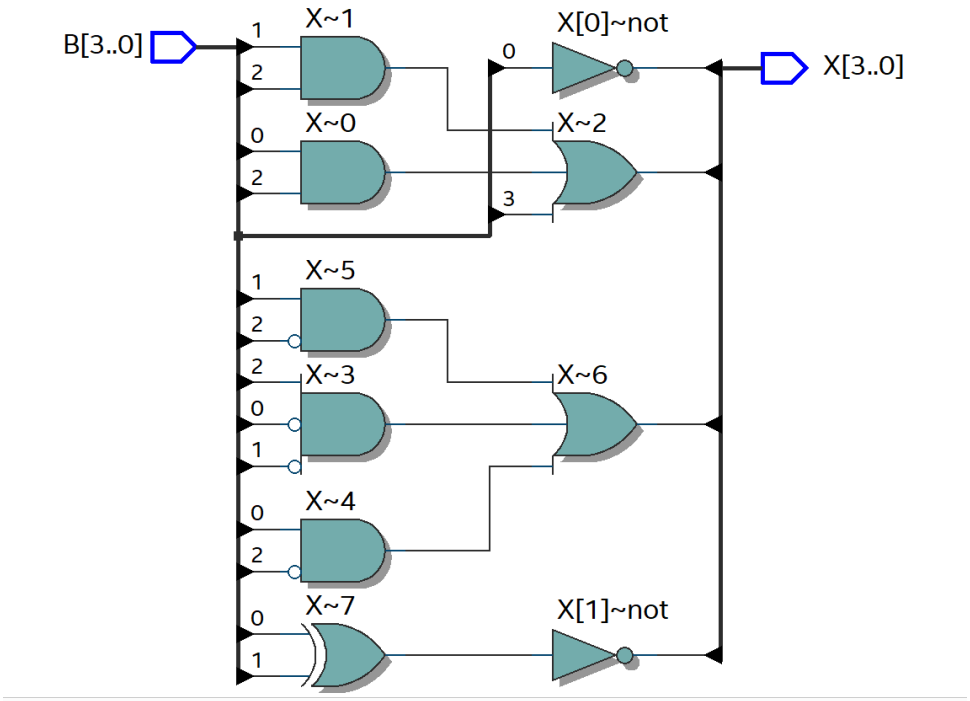
Original code from Ron Kalin

```
// Name:  Ron Kalin, Date: 5-29-24, Design: Lesson 3A1: BCD_to_Xess3
// Group: Ron Kalin/Lamin Jammeh _____
module BCD_to_Xess3 ( input [3:0] B,
                      output [3:0] X ); //B=BCD, X=Xcess3
wire notB0, notB1, notB2, notB3; //declare wires
wire and1X3, and2X3;
wire and1X2, and2X2, and3X2;
wire xnorX1;

//not not0 (notB0, B[0]); //not gates
//not not1 (notB1, B[1]);
//not not2 (notB2, B[2]);
//not not3 (notB3, B[3]);

//and and1 (and1X3, B[0], B[2]);  //and gates
//and and2 (and2X3, B[1], B[2]);
//and and3 (and1X2, notB1, notB0, B[2]);
//and and4 (and2X2, B[0],notB2);
//and and5 (and3X2, B[1],notB2);

//xnor xnor1 (xnorX1, B[0], B[1]); //xnor gate

//or orX3 (X[3], and1X3, B[3], and2X3);  //or gates to give output bits
//or orX2 (X[2], and1X2, and2X2, and3X2);
//assign X[1] = xnorX1;
//assign X[0] = notB0;

//equations below  can be used in lieu of gate logic above
//X3 = (B0B2) + B3 + (B1B2)
assign X[3] = (B[0] & B[2]) |  B[3] | (B[1] & B[2]);
//X2 = (B1'B0'B2) + (B0B2') + (B1B2')
assign X[2] = (!B[1]& !B[0] & B[2]) | (B[0] & !B[2]) | (B[1] & !B[2]);
//X1 = B0'B1' + B0B1 = B0 XNOR B1
assign X[1] = B[0] ^~ B[1];
//X0 = B0'
assign X[0] = !B[0];


endmodule
```

Netlist for the above code



Using the Netlist to design the test bench (note no intermediate output has been monitored)

```
/*-------------------------------------------------------------------------
Name Lamin Jammeh
CLass: EE417 Summer 2024
Lesson 03 HW Question 1
Group: Ron Kalin/ Lamin Jammeh
Main design was done by Ron Kalin
TestBench was done by Lamin Jammeh
-------------------------------------------------------------------------*/

//Step1 define a name for the test-bench
module BCD_to_Xess3_tb;

//Step2 define the inputs as registers and outputs as wires
reg [3:0] B;
wire [3:0] X;

//Step3 define the Unit under test UUT with all inputs and outputs
BCD_to_Xess3 UUT (
                        .B(B),
                        .X(X)
                        );

//Step4 open an initial block and define all the possible input conbination the BCD to Xess3 from 0-9
initial
   begin
            B = 4'b0000;                //define the input B as 4 bits with all zero binary at the start
       #100 B = 4'b0001;
       #100 B = 4'b0010;
       #100 B = 4'b0011;
       #100 B = 4'b0100;
       #100 B = 4'b0101;
       #100 B = 4'b0110;
       #100 B = 4'b1000;
       #100 B = 4'b1001;
       end
initial
   begin
         $monitor("At time %t, B = %b, X = %b", $time, B, X);
    end
endmodule
```

RTL Simulation Output with Bit waveforms (Note the output matches the expected values, since Excess3 = BCD + 4’b0011)