CENTER OF NANOELECTRONICS & DEVICES

Ministry of Communications and Information Technology

eme egypt makes electronics

CND 111: Introduction to Digital Design

Assignment #: 2

Section #: 16

Submitted by:

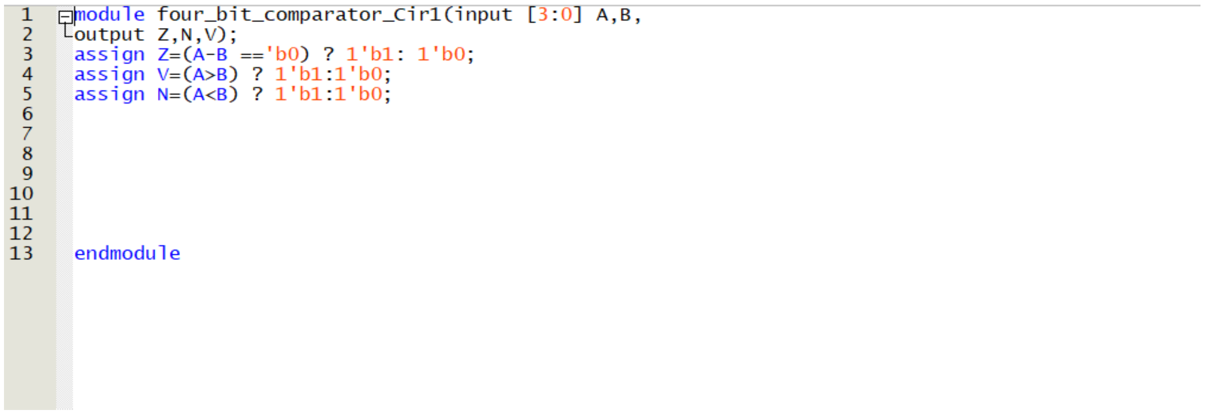
Student Name	ID
Aya Ahmed Abdelrahman	23010284
Karim Mahmoud Kamal	V23010174

Submitted to TA: Ahmed Elshafey

Date: 3/10/2023

➢ Assignment (Comparator)

Comparator cir1 (behavioral implementation)



```
module four_bit_comparator_Cir1(input [3:0] A,B,
output Z,N,V);
assign Z=(A-B =='b0) ? 1'b1: 1'b0;
assign V=(A>B) ? 1'b1:1'b0;
assign N=(A<B) ? 1'b1:1'b0;







endmodule
```

Figure 2 RTL of comparator circuit using behavioral structure.

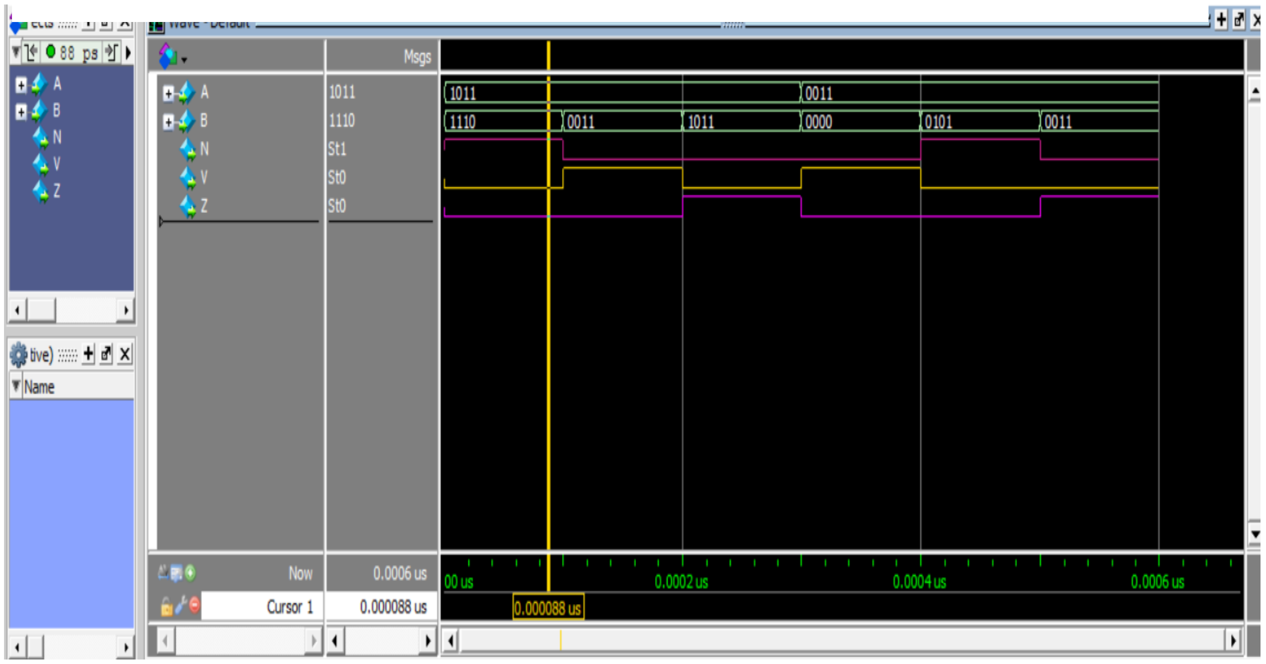


Figure 1 wave form that shows the simulation of comparator cir1.

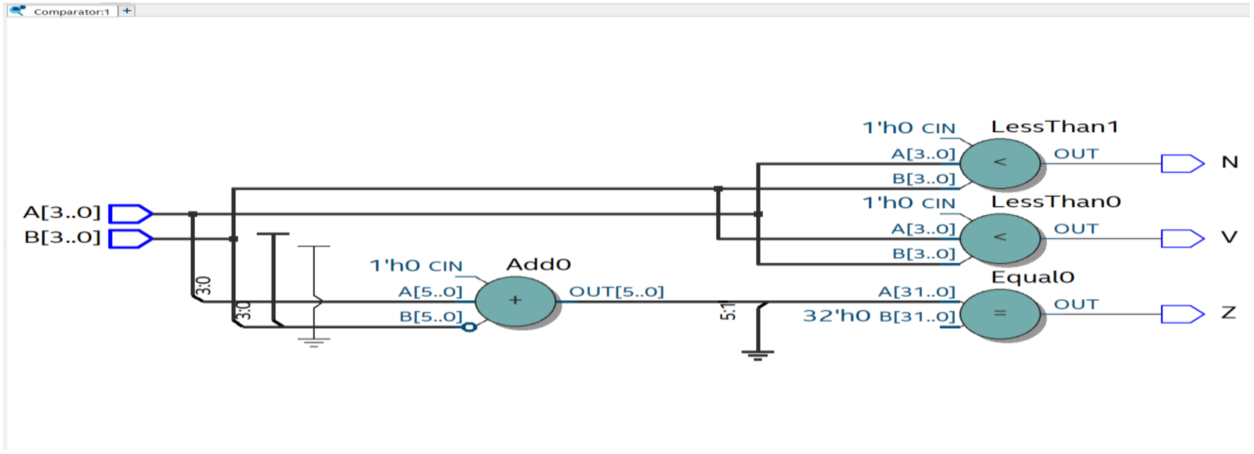
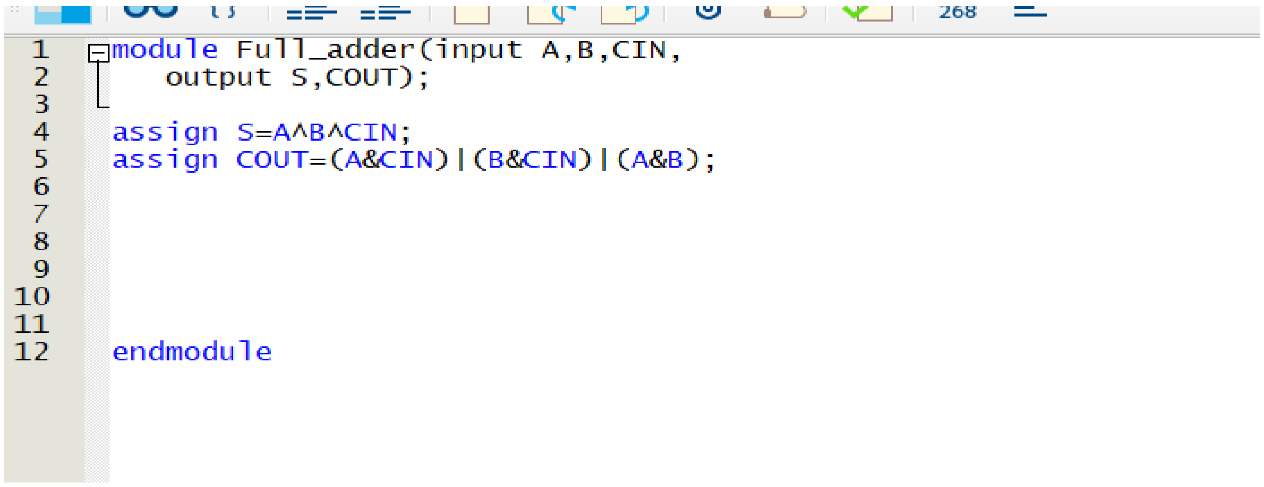


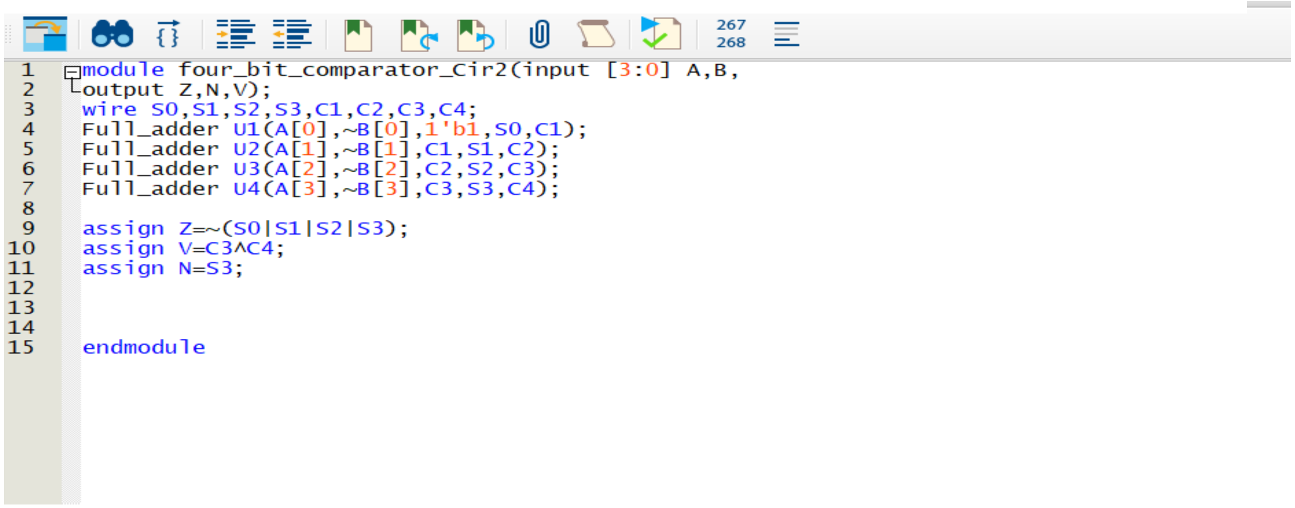
Figure 3 circuit 1 of comparator schematic.

Comparator cir2 (structural implementation)



```
module Full_adder(input A,B,CIN,
    output S,COUT);

assign S=A^B^CIN;
assign COUT=(A&CIN)|(B&CIN)|(A&B);






endmodule
```

Figure 4 full adder circuit implementation



```
module four_bit_comparator_Cir2(input [3:0] A,B,
output Z,N,V);
wire S0,S1,S2,S3,C1,C2,C3,C4;
Full_adder U1(A[0],~B[0],1'b1,S0,C1);
Full_adder U2(A[1],~B[1],C1,S1,C2);
Full_adder U3(A[2],~B[2],C2,S2,C3);
Full_adder U4(A[3],~B[3],C3,S3,C4);

assign Z=~(S0|S1|S2|S3);
assign V=C3^C4;
assign N=S3;



endmodule
```

Figure 5 structural implementation of comparator

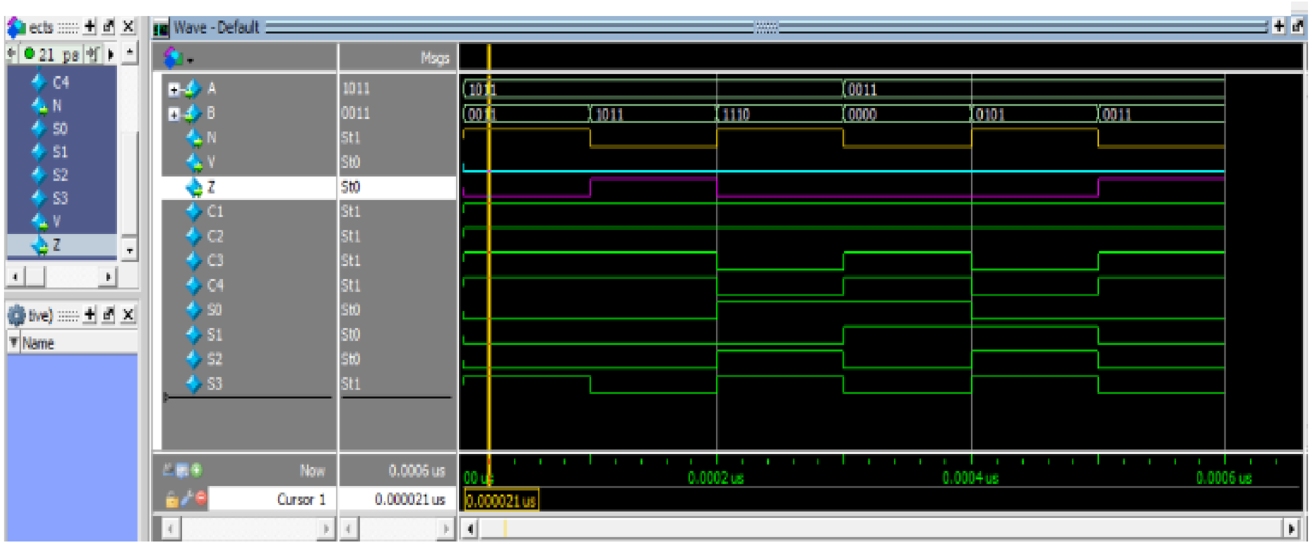


Figure 6 wave form of circuit comparator using structural implementation.

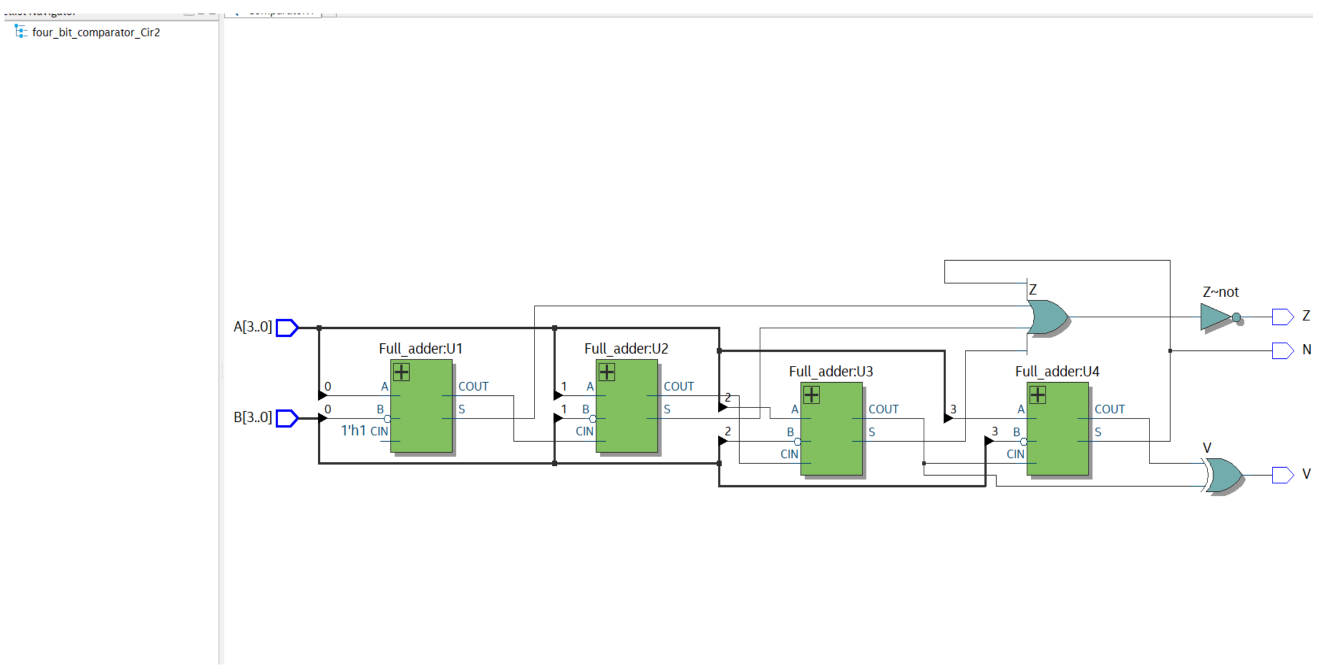


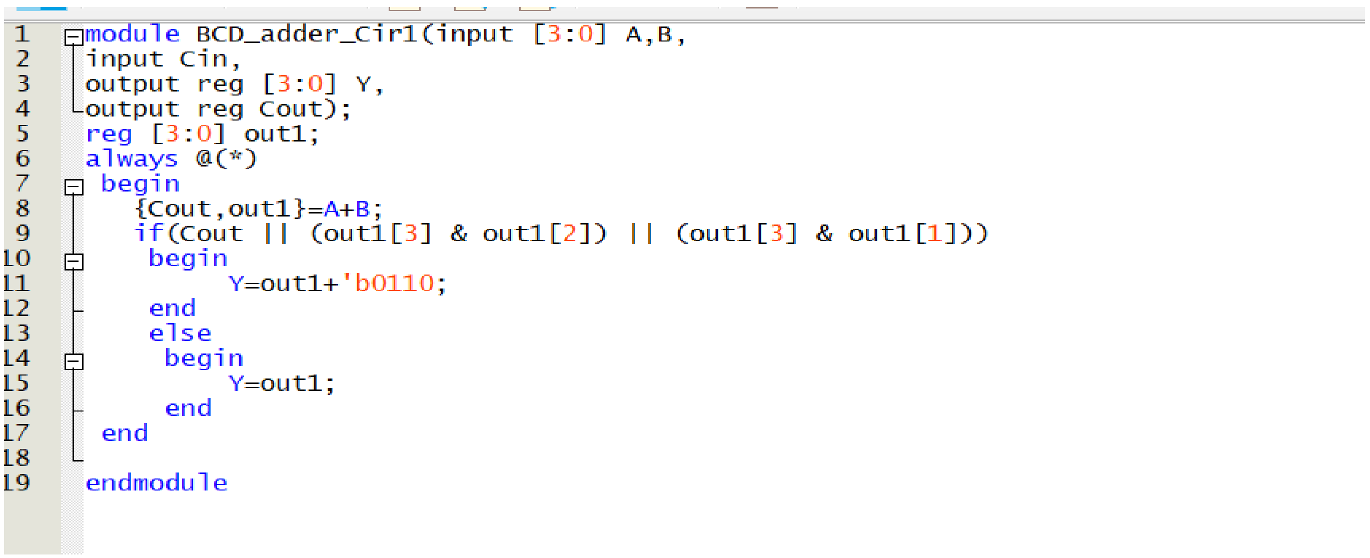
Figure 7 circuit 2 of comparator schematic.

comments:

comparator is a circuit that compares two input numbers to each other and gives flags to indicates that the numbers are equal or one of them is greater or less than the other number. In cir1 and cir2, we implement the comparator that when A is greater than B the output V will equal to 1, when A is equal to B zero flag will be 1 and when A is less than B the output N will be 1.Note the circuit behave correctly in Cir1 but in Cir2 the output overflow didn’t behave as expected and I searched more about it and didn’t get the correct answer and why we should use xor circuit for C3,C4 to get the value of the flag.

➢ Bonus assignment

BCD adder cir1 (Behavioral implementation)



```
module BCD_adder_Cir1(input [3:0] A,B,
input Cin,
output reg [3:0] Y,
output reg Cout);
reg [3:0] out1;
always @(*)
 begin
    {Cout,out1}=A+B;
    if(Cout || (out1[3] & out1[2]) || (out1[3] & out1[1]))
     begin
          Y=out1+'b0110;
     end
     else
      begin
          Y=out1;
      end
 end

endmodule
```

Figure 8 RTL implementation (behavioral) of BCD adder.

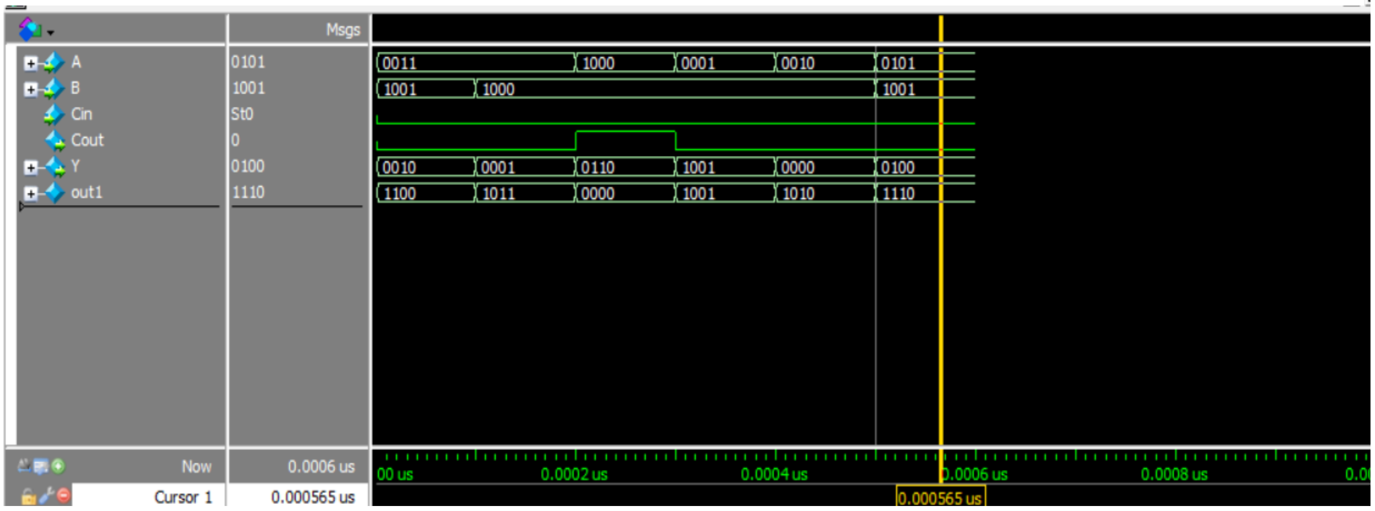


Figure 9 wave form of BCD adder Cir1

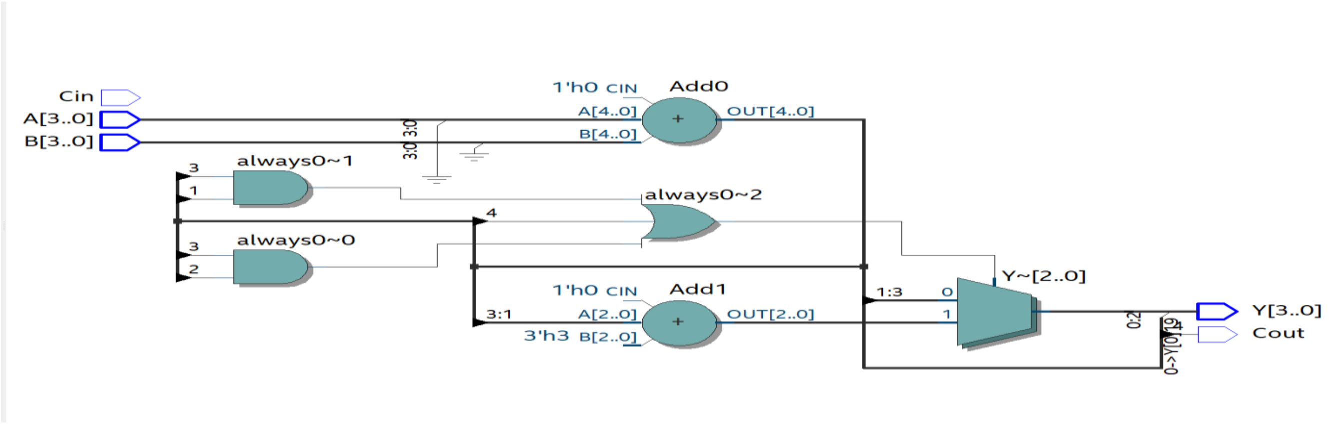
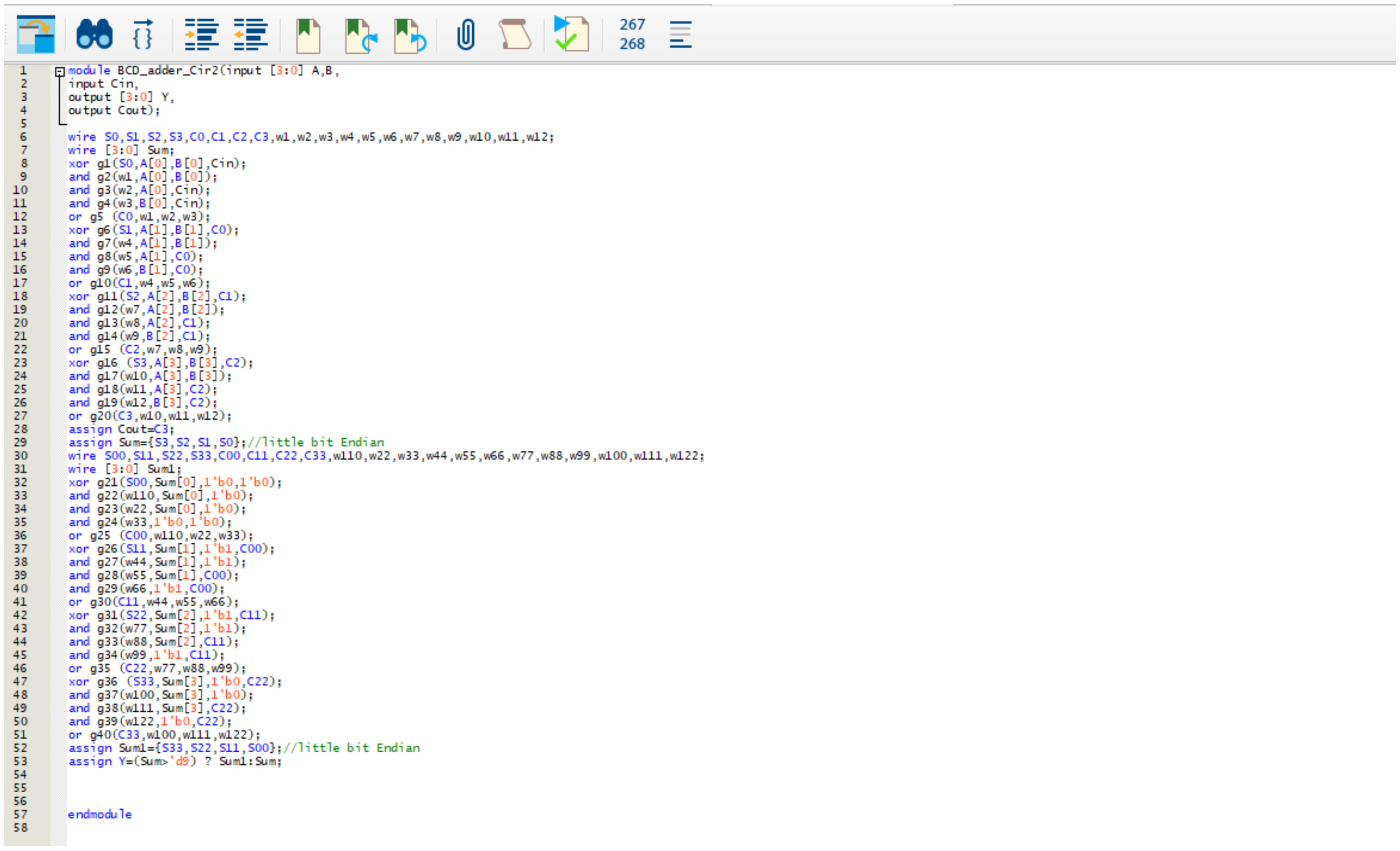


Figure 10 Schematic view of BCD adder

BCD adder cir2 (structural implementation)



```verilog
module BCD_adder_Cir2(input [3:0] A,B,
input Cin,
output [3:0] Y,
output Cout);

wire S0,S1,S2,S3,C0,C1,C2,C3,w1,w2,w3,w4,w5,w6,w7,w8,w9,w10,w11,w12;
wire [3:0] Sum;
xor g1(S0,A[0],B[0],Cin);
and g2(w1,A[0],B[0]);
and g3(w2,A[0],Cin);
and g4(w3,B[0],Cin);
or g5 (C0,w1,w2,w3);
xor g6(S1,A[1],B[1],C0);
and g7(w4,A[1],B[1]);
and g8(w5,A[1],C0);
and g9(w6,B[1],C0);
or g10(C1,w4,w5,w6);
xor g11(S2,A[2],B[2],C1);
and g12(w7,A[2],B[2]);
and g13(w8,A[2],C1);
and g14(w9,B[2],C1);
or g15 (C2,w7,w8,w9);
xor g16 (S3,A[3],B[3],C2);
and g17(w10,A[3],B[3]);
and g18(w11,A[3],C2);
and g19(w12,B[3],C2);
or g20(C3,w10,w11,w12);
assign Cout=C3;
assign Sum={S3,S2,S1,S0};//little bit Endian
wire S00,S11,S22,S33,C00,C11,C22,C33,w110,w22,w33,w44,w55,w66,w77,w88,w99,w100,w111,w122;
wire [3:0] Sum1;
xor g21(S00,Sum[0],1'b0,1'b0);
and g22(w110,Sum[0],1'b0);
and g23(w22,Sum[0],1'b0);
and g24(w33,1'b0,1'b0);
or g25 (C00,w110,w22,w33);
xor g26(S11,Sum[1],1'b1,C00);
and g27(w44,Sum[1],1'b1);
and g28(w55,Sum[1],C00);
and g29(w66,1'b1,C00);
or g30(C11,w44,w55,w66);
xor g31(S22,Sum[2],1'b1,C11);
and g32(w77,Sum[2],1'b1);
and g33(w88,Sum[2],C11);
and g34(w99,1'b1,C11);
or g35 (C22,w77,w88,w99);
xor g36 (S33,Sum[3],1'b0,C22);
and g37(w100,Sum[3],1'b0);
and g38(w111,Sum[3],C22);
and g39(w122,1'b0,C22);
or g40(C33,w100,w111,w122);
assign Sum1={S33,S22,S11,S00};//little bit Endian
assign Y=(Sum>'d9) ? Sum1:Sum;



endmodule
```

Figure 11 RTL implementation (structural) of BCD adder.

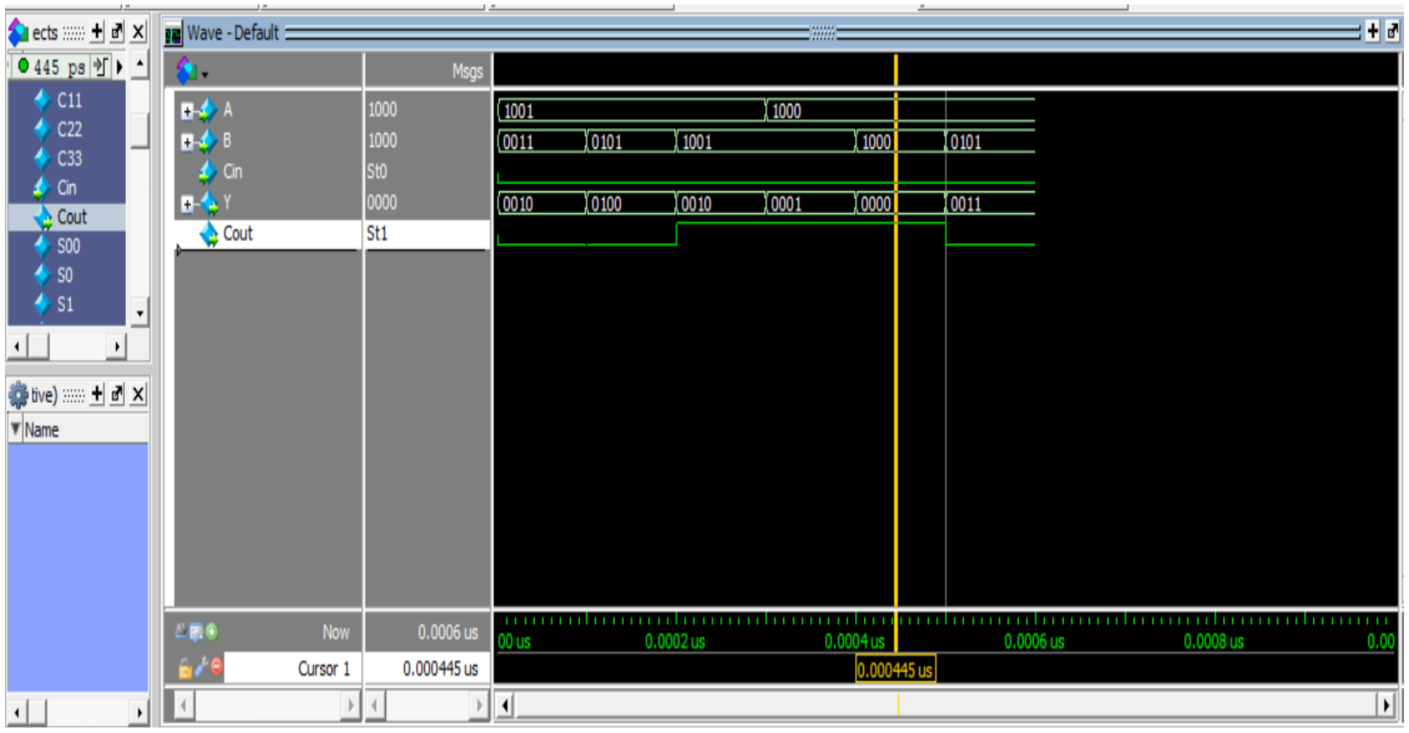


Figure 12 wave form of BCD adder Cir2

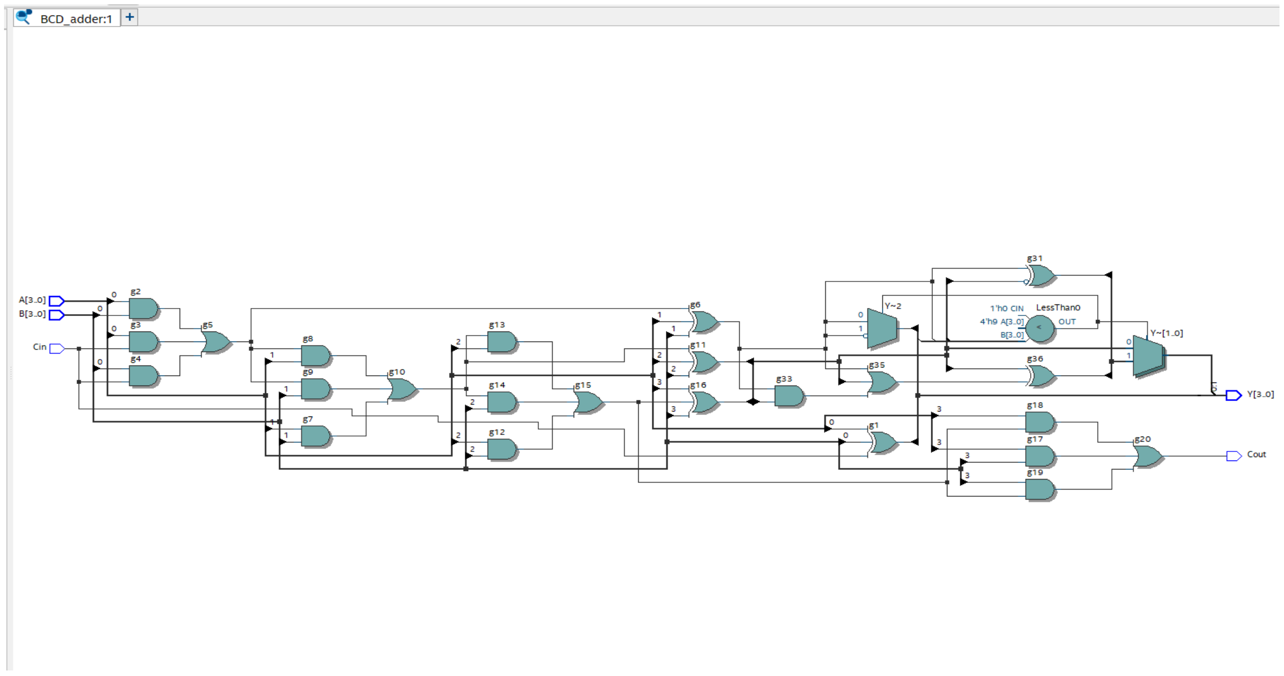


Figure 13 Schematic view of BCD adder cir2

comments:

BCD adder is a circuit that add two BCD numbers and the output must be also BCD (decimal values from 0 to 9 and are represented in 4 bits). As shown in cir1 and cir2 when we add two BCD numbers if the output is less than 9 the output is BCD and will be the same value but when the output is greater than 9 firstly, I will take the value of Cout if exits then if the remained value is greater than 9, I will add 6 to it and takes the LSB bits and considered it as the output to be a BCD value.

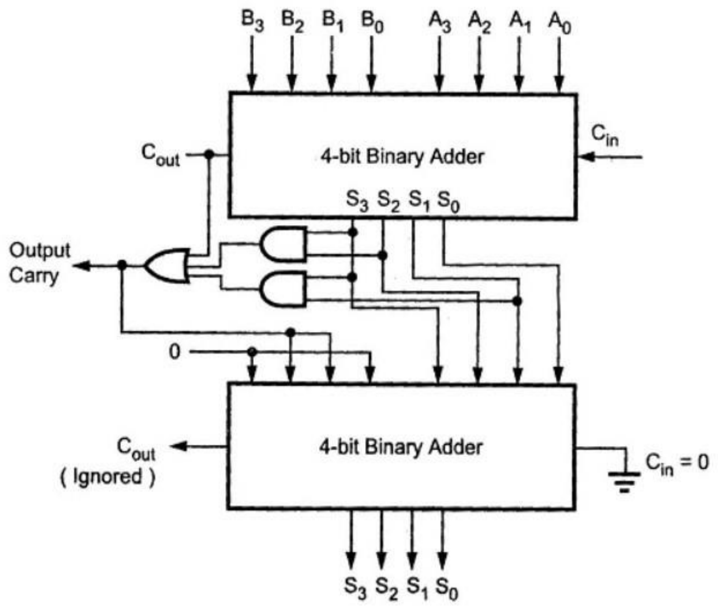


Figure 14 BCD adder circuit.