LAB # 12

Fall 2020

CSE304L Computer Organization & Architecture Lab

Submitted by: **ASHLEY ALEX JACOB**

Registration No: **18PWCSE1649**

Class Section: **A**

"On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work."

Student Signature: ______________

Submitted to:

Engr. Amaad Khalil

March 18, 2021 (Thursday)

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

Lab Tasks:

Task:

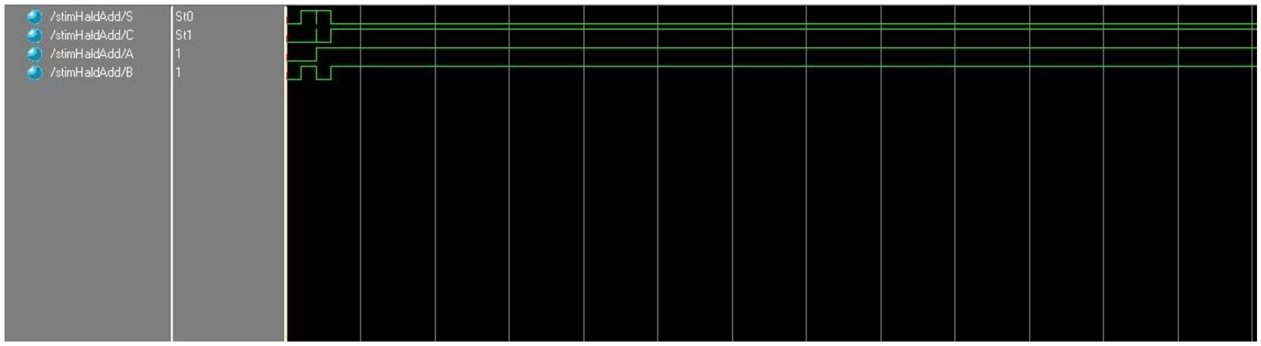
Write a Verilog code for Half Adder using Gate Level modeling.

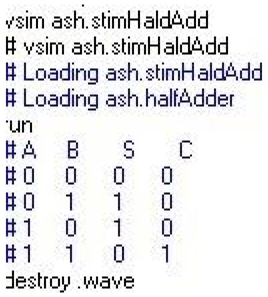
Code:

```
module halfAdder( output s,c,input a,b);

and(c,a,b);

xor(s,a,b);

endmodule




module stimHaldAdd();

wire S,C;

reg A,B;



halfAdder ha(S,C,A,B);



initial

begin
```

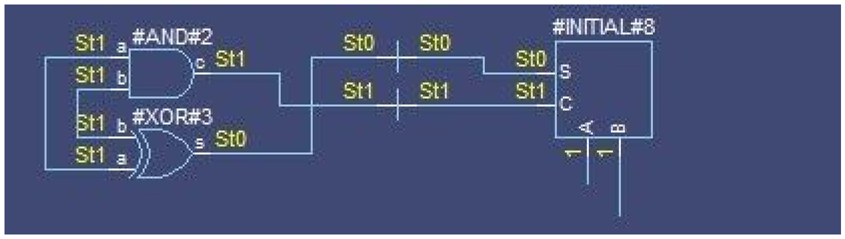
```
$display("A     B       S       C");

A=0;

B=0;

#1 $display("%b     %b      %b      %b",A,B,S,C);



A=0;

B=1;

#1 $display("%b     %b      %b      %b",A,B,S,C);



A=1;

B=0;

#1 $display("%b     %b      %b      %b",A,B,S,C);



A=1;

B=1;

#1 $display("%b     %b      %b      %b",A,B,S,C);



end
```

```
endmodule
```





```
vsim ash.stimHaldAdd
# vsim ash.stimHaldAdd
# Loading ash.stimHaldAdd
# Loading ash.halfAdder
run
# A   B   S   C
# 0   0   0   0
# 0   1   1   0
# 1   0   1   0
# 1   1   0   1
destroy .wave
```



Task:

Write a Verilog code for Full Adder using Gate Level modeling.

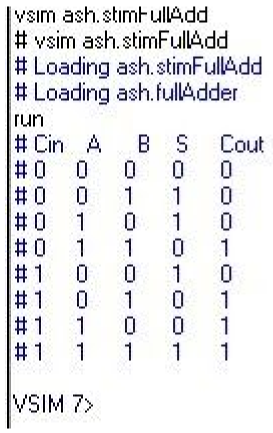
Code:

```
module fullAdder( output s,cOut,input a,b,cIn);

xor(n1,a,b),(s,cIn,n1);

and(n2,n1,cIn),(n3,a,b);

or(cOut,n2,n3);
```

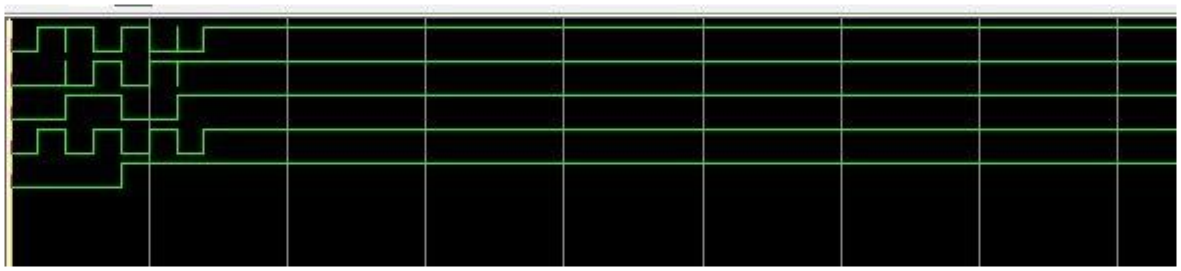
```
endmodule



module stimFullAdd();

wire S,Cout;

reg A,B,Cin;


fullAdder fa(S,Cout,A,B,Cin);


initial

begin


$display("Cin    A      B    S     Cout");

Cin=0;

A=0;

B=0;

#1 $display("%b     %b      %b      %b      %b",Cin,A,B,S,Cout);

Cin=0;
```

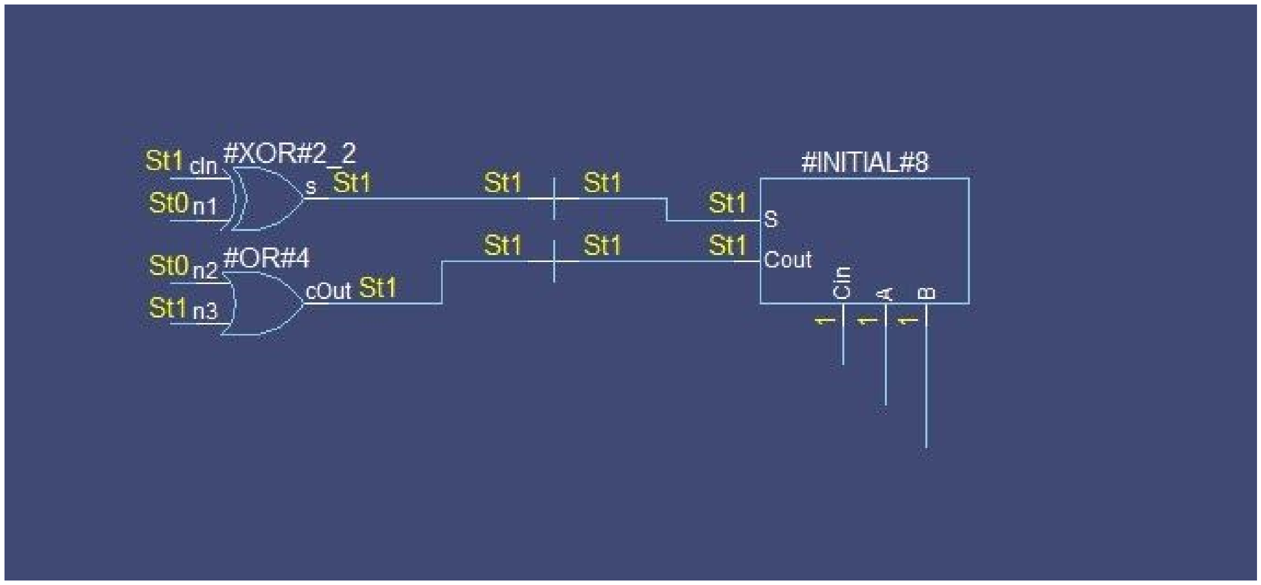
```
A=0;

B=1;

#1 $display("%b     %b      %b      %b      %b",Cin,A,B,S,Cout);

Cin=0;

A=1;

B=0;

#1 $display("%b     %b      %b      %b      %b",Cin,A,B,S,Cout);

Cin=0;

A=1;

B=1;

#1 $display("%b     %b      %b      %b      %b",Cin,A,B,S,Cout);

Cin=1;

A=0;

B=0;

#1 $display("%b     %b      %b      %b      %b",Cin,A,B,S,Cout);

Cin=1;

A=0;

B=1;
```

```
#1 $display("%b     %b      %b      %b      %b",Cin,A,B,S,Cout);

Cin=1;

A=1;

B=0;

#1 $display("%b     %b      %b      %b      %b",Cin,A,B,S,Cout);

Cin=1;

A=1;

B=1;

#1 $display("%b     %b      %b      %b      %b",Cin,A,B,S,Cout);




end

endmodule
```



```
vsim ash.stimFullAdd
# vsim ash.stimFullAdd
# Loading ash.stimFullAdd
# Loading ash.fullAdder
run
# Cin   A    B   S    Cout
# 0    0    0    0    0
# 0    0    1    1    0
# 0    1    0    1    0
# 0    1    1    0    1
# 1    0    0    1    0
# 1    0    1    0    1
# 1    1    0    0    1
# 1    1    1    1    1

VSIM 7>
```





Task:

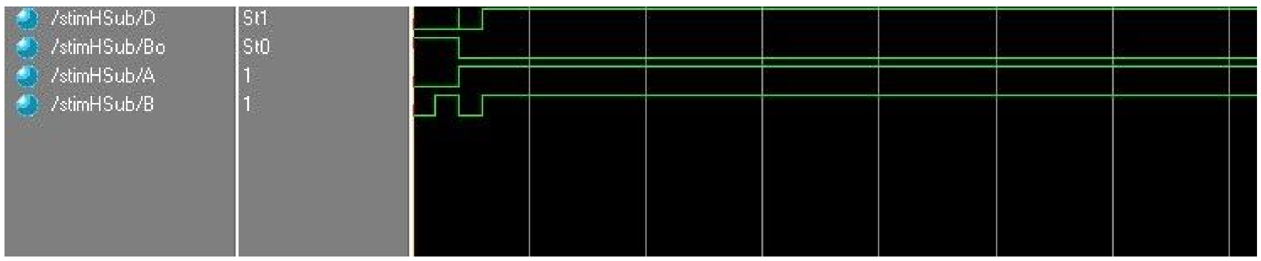
Write a Verilog code for Half Subtractor using Gate Level modeling.

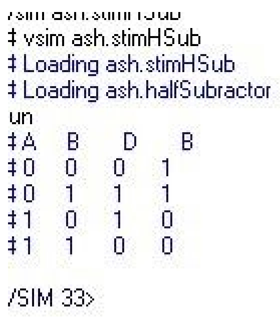
Code:

```
module halfSubractor(output d,bo,input a,b);

xor(d,a,b);

not(a_,a);

and(bo,a_);
```

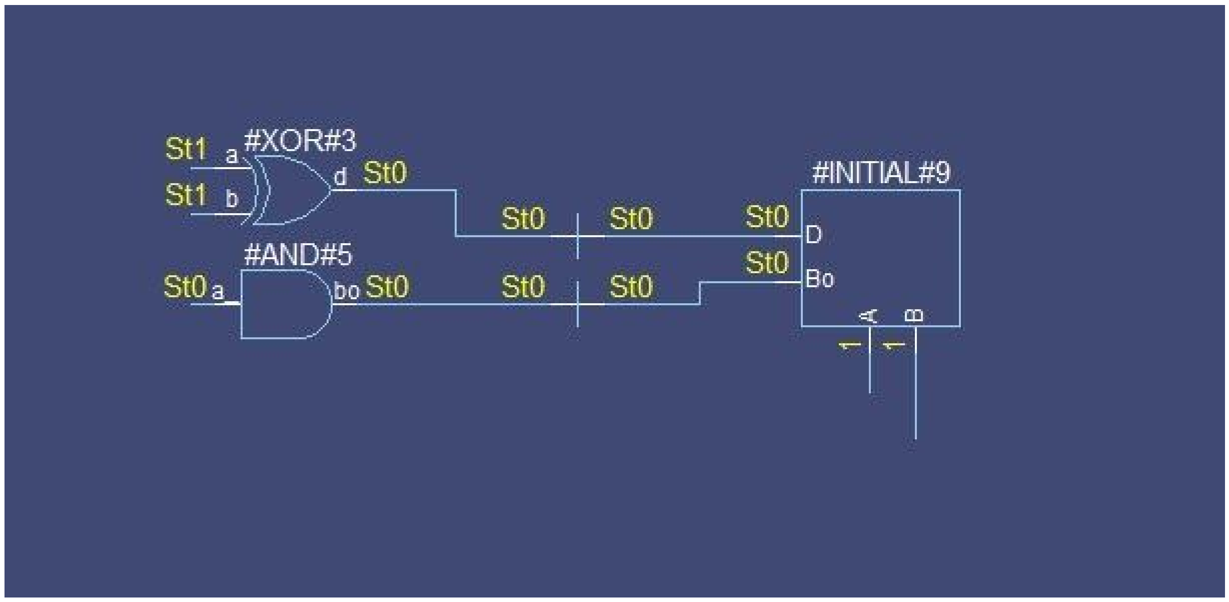
```
endmodule




module stimHSub();

wire D,Bo;

reg A,B;


halfSubractor hs(D,Bo,A,B);


initial

begin


$display("A     B       D       B");

A=0;

B=0;

#1 $display("%b     %b      %b      %b",A,B,D,Bo);
```

```
A=0;

B=1;

#1 $display("%b     %b      %b      %b",A,B,D,Bo);



A=1;

B=0;

#1 $display("%b     %b      %b      %b",A,B,D,Bo);



A=1;

B=1;

#1 $display("%b     %b      %b      %b",A,B,D,Bo);



end

endmodule
```





```
/sim ash.stimHSub
# vsim ash.stimHSub
# Loading ash.stimHSub
# Loading ash.halfSubractor
un
#A   B    D    B
#0   0    0    1
#0   1    1    1
#1   0    1    0
#1   1    0    0

/SIM 33>
```



Task:

Write a Verilog code for Adder and Subtractor using Gate Level modeling.

Code:

```
module full_adder(S, Cout, A, B, Cin);

   output S;

   output Cout;

   input  A;

   input  B;

   input  Cin;
   wire   w1;

   wire   w2;

   wire   w3;

   wire   w4;



   xor(w1, A, B);

   xor(S, Cin, w1), (w2, A, B),(w3, A, Cin),(w4, B, Cin);

   or(Cout, w2, w3, w4);

endmodule




module adderSubtractor(S, C, V, A, B, Op);

   output [3:0] S;

   output    C;

   output    V;

   input [3:0]      A;

   input [3:0]      B;
   input     Op;


   wire      C0;

   wire      C1;

   wire      C2;

   wire      C3;


   wire      B0;

   wire      B1;

   wire      B2;

   wire      B3;

   xor(B0, B[0], Op);

   xor(B1, B[1], Op);

   xor(B2, B[2], Op);

   xor(B3, B[3], Op);

   xor(C, C3, Op);

   xor(V, C3, C2);

   full_adder fa0(S[0], C0, A[0], B0, Op);    // Least significant bit.
```

```
   full_adder fa1(S[1], C1, A[1], B1, C0);

   full_adder fa2(S[2], C2, A[2], B2, C1);

   full_adder fa3(S[3], C3, A[3], B3, C2);    // Most significant bit.

endmodule
```