Hamad Khawaja

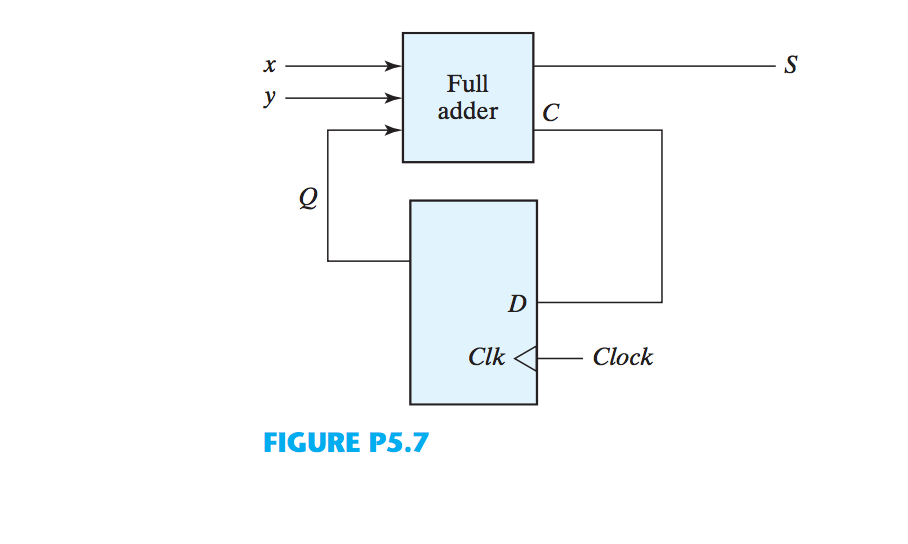
CS-385

Digital design Review

Extra Credit

**Consider the sequential circuit implementing a serial adder built of a 1-bit full adder and a D flip-flop**

1. Create the state table and the state diagram of the circuit.

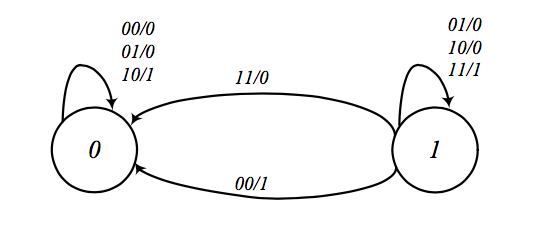
****

**STATE TABLE :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Q | x | y | Q | S |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

*S=x* ⊕ *y* ⊕ *Q*

*Q*(*t + 1*) *= xy + xQ + yQ*



**Full adder & D flip-flop**

module D\_flip\_flop(D,CLK,Q);

input D,CLK;

output Q;

wire CLK1, Y;

not not1 (CLK1,CLK);

D\_latch D1(D,CLK, Y),

D2(Y,CLK1,Q);

endmodule

module D\_latch(D,C,Q);

input D,C;

output Q;

wire x,y,D1,Q1;

nand nand1 (x,D, C),

nand2 (y,D1,C),

nand3 (Q,x,Q1),

nand4 (Q1,y,Q);

not not1 (D1,D);

endmodule

module fulladder(Sum, Cout, A, B, Cin);

input A, B, Cin;

output Cout, Sum;

wire xorab, cinxor1, andab;

xor xor1(xorab, A, B);

xor xor2(Sum, xorab, Cin);

and and1(cinxor1, xorab, Cin);

and and2(andab, A, B);

or orl(Cout, andab, cinxor1);

endmodule

**Serial Adder**

module serialadder(S,A,B,CLK);

input A,B,CLK;

output S;

wire Cin,Cout;

D\_flip\_flop dff1(Cout, CLK, Cin);

fulladder fa1(S, Cout, A, B, Cin);

endmodule

**TEST CASE 1-bit Serial Adder Using D Flip Flop**

module test;

reg A, B, CLK, reset;

wire RST;

output S;

serialadder a1(S,A,B,CLK);

initial

begin

$display("A B S");

A=0; B=0;

#1 CLK = 1;

#1 CLK = 0;

A = 1; B = 1;

#1 CLK = 1;

#1 CLK = 0;

A = 0; B = 0;

#1 CLK = 1;

#1 CLK = 0;

A = 0; B = 1;

#1 CLK = 1;

#1 CLK = 0;

A = 1; B = 0;

#1 CLK = 1;

#1 CLK = 0;

A = 1; B = 1;

#1 CLK = 1;

#1 CLK = 0;

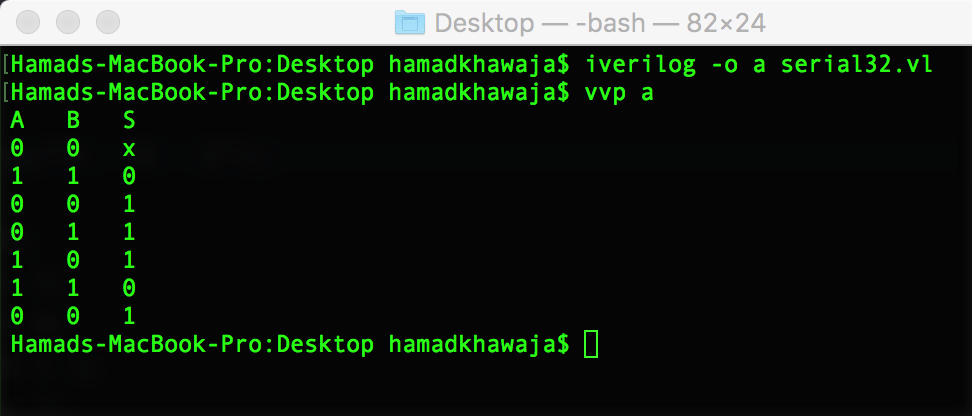
A = 0; B = 0;

end

initial

$monitor("%b %b %b ",A, B,S);

endmodule



**TEST CASE 32 Bit 2’s Complement Integer**

module test;

reg A, B, CLK, reset;

reg[31:0] X,Y,Z,i;

wire RST;

output S;

serialadder a1(S,A,B,CLK);

initial

begin

A=0; B=0;

#1 CLK = 1;

#1 CLK = 0;

X= 2147483647; Y= 2147483647;

$display("X= 2147483647 Y= 2147483647");

for(i=0;i<32;i=i+1)

begin

A=X[i];B=Y[i]; #1 CLK = 1;#1 CLK = 0; Z[i]=S;

end

$display("%b %b %b ",X,Y,Z);

$display("\n");

X= -2147483647;Y= 2147483647;

$display("X= -2147483647 Y= 2147483647");

for(i=0;i<32;i=i+1)

begin

A=X[i];B=Y[i]; #1 CLK = 1;#1 CLK = 0; Z[i]=S;

end

$display("%b %b %b ",X,Y,Z);

$display("\n");

X= 2147483647; Y= 1;

$display("X= 2147483647 Y= 1");

for(i=0;i<32;i=i+1)

begin

A=X[i];B=Y[i]; #1 CLK = 1;#1 CLK = 0; Z[i]=S;

end

$display("%b %b %b ",X,Y,Z);

$display("\n");

X= 2147483647; Y= 0;

$display("X= 2147483647 Y= 0");

for(i=0;i<32;i=i+1)

begin

A=X[i];B=Y[i]; #1 CLK = 1;#1 CLK = 0; Z[i]=S;

end

$display("%b %b %b ",X,Y,Z);

$display("\n");

X= 2147483647; Y= -1;

$display("X= 2147483647 Y= -1");

for(i=0;i<32;i=i+1)

begin

A=X[i];B=Y[i]; #1 CLK = 1;#1 CLK = 0; Z[i]=S;

end

$display("%b %b %b ",X,Y,Z);

$display("\n");

X= -2147483647; Y= -1;

$display("X=-2147483647 Y= -1");

for(i=0;i<32;i=i+1)

begin

A=X[i];B=Y[i]; #1 CLK = 1;#1 CLK = 0; Z[i]=S;

end

$display("%b %b %b ",X,Y,Z);

$display("\n");

X= -2147483647; Y= 1;

$display("X=-2147483647 Y= 1");

for(i=0;i<32;i=i+1)

begin

A=X[i];B=Y[i]; #1 CLK = 1;#1 CLK = 0; Z[i]=S;

end

$display("%b %b %b ",X,Y,Z);

$display("\n");

X= -2147483647; Y= 0;

$display("X=-2147483647 Y= 0");

for(i=0;i<32;i=i+1)

begin

A=X[i];B=Y[i]; #1 CLK = 1;#1 CLK = 0; Z[i]=S;

end

$display("%b %b %b ",X,Y,Z);

$display("\n");

end

endmodule

