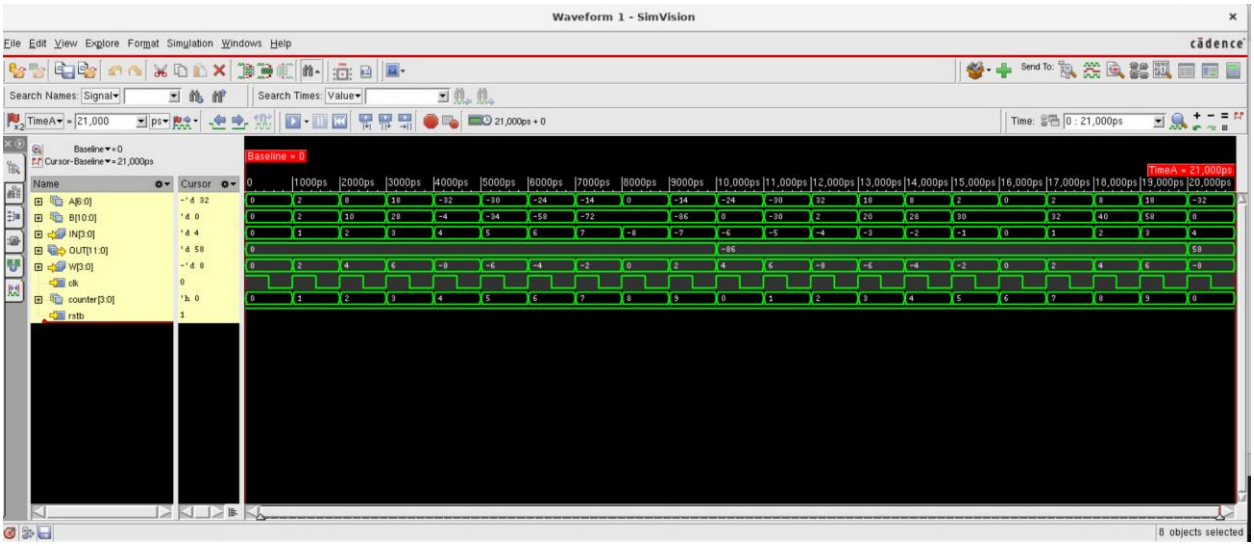
Problem 1:



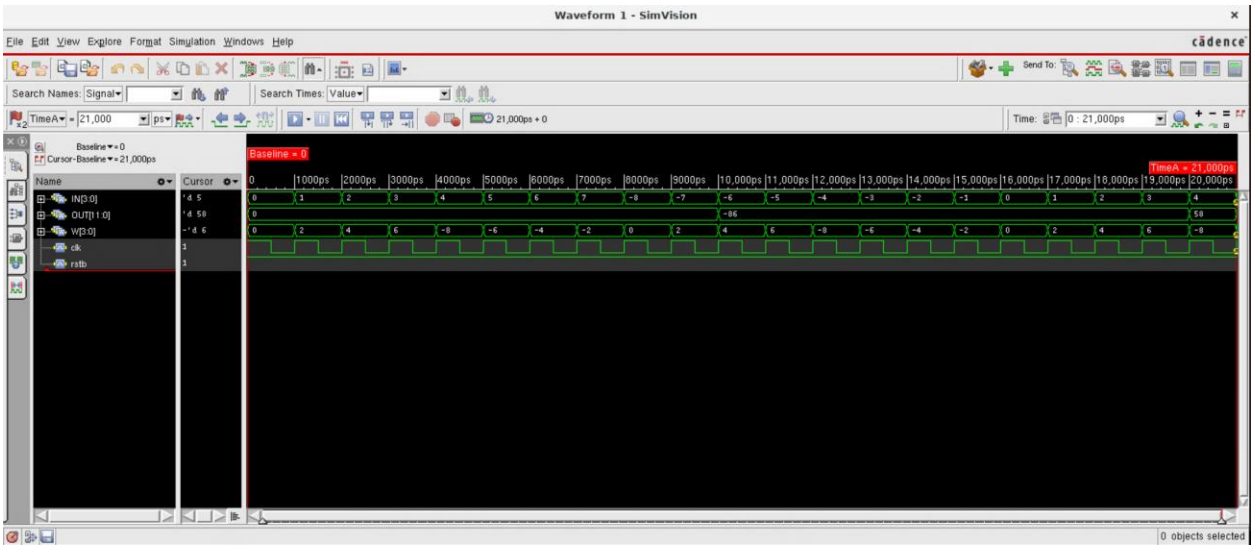
Area: 520.786	Cell Counts: 139	Timing Slack: 12

Report Area:

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 18.14-s037_1
  Generated on:           Dec 05 2024  08:09:37 am
  Module:                 mac
  Operating conditions:   typical
  Interconnect mode:      global
  Area mode:              physical library
====================================================================

Instance Module  Cell Count  Cell Area  Net Area   Total Area
--------------------------------------------------------------------
mac                    139     329.574   191.212       520.786
```

Report Timing:

```
====================================================================
  Generated by:           Genus(TM) Synthesis Solution 18.14-s037_1
  Generated on:           Dec 05 2024  08:09:00 am
  Module:                 mac
  Operating conditions:   typical
  Interconnect mode:      global
  Area mode:              physical library
====================================================================


Path 1: MET (12 ps) Setup Check with Pin A_reg[5]/CK->D
          Group: clk
     Startpoint: (F) W[2]
          Clock: (R) clk
       Endpoint: (R) A_reg[5]/D
          Clock: (R) clk

                     Capture       Launch
        Clock Edge:+    1000            0
        Drv Adjust:+       0            0
       Src Latency:+       0            0
       Net Latency:+       0 (I)        0 (I)
           Arrival:=    1000            0

             Setup:-      33
     Required Time:=     967
      Launch Clock:-       0
       Input Delay:-     500
         Data Path:-     455
             Slack:=      12

Exceptions/Constraints:
  input_delay             500             mac.sdc_line_3_5_1

#-------------------------------------------------------------------------------------------
# Timing Point   Flags    Arc    Edge   Cell     Fanout Load Trans Delay Arrival Instance
#                                                       (fF)  (ps)  (ps)   (ps)  Location
#-------------------------------------------------------------------------------------------
  W[2]           -        -      F     (arrival)      4  5.6     0     0     500    (-,-)
  g3341__6877/ZN -        A2->ZN F     AND2_X1        1  3.6     8    30     530    (-,-)
  g3318__5953/S  -        B->S   F     HA_X1          1  3.4    13    58     588    (-,-)
  g3282__1857/CO -        A->CO  F     HA_X1          1  1.1     5    29     617    (-,-)
  g3279__2703/ZN -        A1->ZN F     OR2_X1         1  3.9    12    48     666    (-,-)
  g3265__8757/CO -        A->CO  F     FA_X1          1  3.0    15    77     743    (-,-)
  g3261__2900/CO -        CI->CO F     FA_X1          1  3.0    15    72     815    (-,-)
  g3255__9682/S  -        CI->S  R     FA_X1          1  1.2    10   110     926    (-,-)
  g3254__1474/ZN -        A1->ZN R     AND2_X1        1  1.4     8    30     955    (-,-)
  A_reg[5]/D     <<<      -      R     DFFR_X1        1    -     -     0     955    (-,-)
#-------------------------------------------------------------------------------------------
```



RTL:

```
`timescale 1ns/1ps

module mac(
   input signed [3:0] IN,W, //should be signed
   input clk, rstb,
   output reg signed [11:0] OUT //should be signed
);
reg signed [6:0] A; //should be signed
reg signed [10:0] B; //should be signed
reg [3:0] counter;
always @(posedge clk or negedge rstb) begin
   if(~rstb) begin
      counter = 0;
      OUT = 0;
      B = 0;
      A = 0;
   end else if(counter == 9) begin
      counter = 0;
      OUT = B;
```

```
      B = 0;
      A = IN*W;
    end else if(counter < 9) begin
      counter = counter + 1;
      A = IN*W;
      B = A+B;
    end else begin
      counter = 0;
      OUT = 0;
      B = 0;
      A = 0;
    end
end
endmodule
```

Testbench:

```
`timescale 1ns/1ps
module mac_tb;
    reg signed [3:0] IN,W; //should be signed
    reg clk, rstb;
    wire signed [11:0] OUT; //should be signed
    mac mymac(IN,W,clk,rstb,OUT);
    always begin
      $dumpfile("mac.vcd");
      $dumpvars(0,mac_tb);
      IN = 0;
      W = 0;
      rstb = 1;
      clk = 1;
```

```
    repeat(21) begin
      #0.5
      clk = 0;

      #0.5
      clk = 1;
      IN = IN + 1;
      W = W + 2;
    end
    $finish;
  end
endmodule
```

SDC:

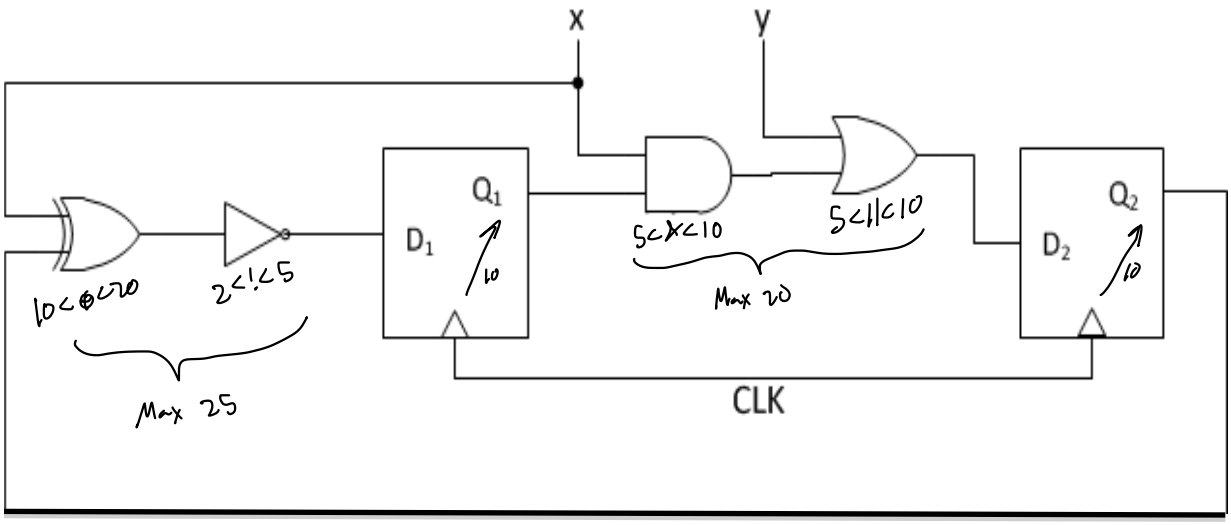
```
create_clock -name clk -period 1.0 -waveform { 0 0.5 } [get_ports clk]

set_input_delay -max 0.5 -clock clk [get_ports {IN W}]
set_input_delay -min -0.2 -clock clk [get_ports {IN W rstb}]
set_output_delay -max 0.5 -clock clk [get_ports {OUT}]
set_output_delay -min -0.2 -clock clk [get_ports {OUT}]
```

Problem 2. (10 Points) Sequential Circuits

Use the circuit diagram in Figure below with the following parameters:

- t_{pcq}= 10 ps
- 5 ps < $t_{p, AND}$ < 10 ps
- 5 ps < $t_{p, OR}$ < 10 ps
- 10 ps < $t_{p, XOR}$ < 20 ps
- 2 ps < $t_{p, INV}$ < 5 ps

Both flip-flops are identical rising-edge DFFs. Assume the values of x and y are always stable from one rising clock edge to the next rising clock edge.

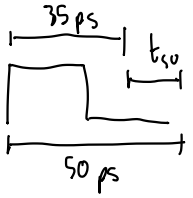


(a) What is the maximum setup time of the flip-flops in order to safely use a 50ps clock?

(b) What is the maximum hold time of the flip-flops in order to avoid hold violation?

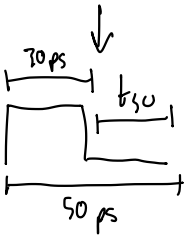
a) $D_1 \rightarrow D_2$ prop delay max = 30ps

$D_2 \rightarrow D_1$ prop delay max = 35ps



→ Max setup time is 15ps

b)



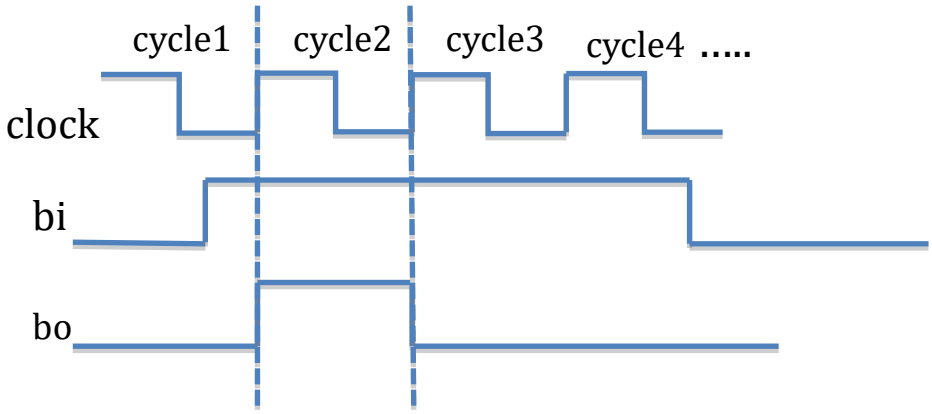
→ Max hold time to avoid hold violation is 30ps

Problem 3. (30 Points) FSM Design

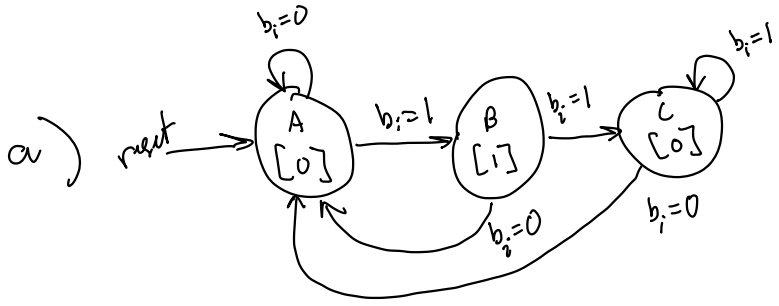
Design a Finite State Machine for a Button Press Synchronizer. This circuit should synchronize a button press to a clock signal, such that when the button is pressed, the output of this circuit is a signal that is '1' exactly for one clock cycle. Such a synchronized signal is useful to prevent a single button press that lasts multiple cycles from being interpreted as multiple button presses.

The circuit's input will be a signal b_i and the output a signal b_o. When b_i becomes 1, representing the button being pressed, the system should set b_o to 1 for exactly one cycle. The system waits for b_i to return to 0 again, and then waits for b_i to become 1 again, which would represent the next pressing of the button.

A representative timing diagram of this behavior is shown below:



Part a. Derive the FSM diagram. Refer to the lecture slides and the reading material (FSM Tutorial.pdf) posted on Canvas if you need pointers to how to convert a verbal description into a state diagram.
Part b. Derive the transition table and assign binary codes of your choice to states.
Part c. From this table derive the next state and output logic expression and draw the resulting circuit diagram of the sequential circuit.
Part d. Implement the same FSM in Verilog using always statements and verify the behavior shown in the timing diagram above is matching your Verilog simulation.



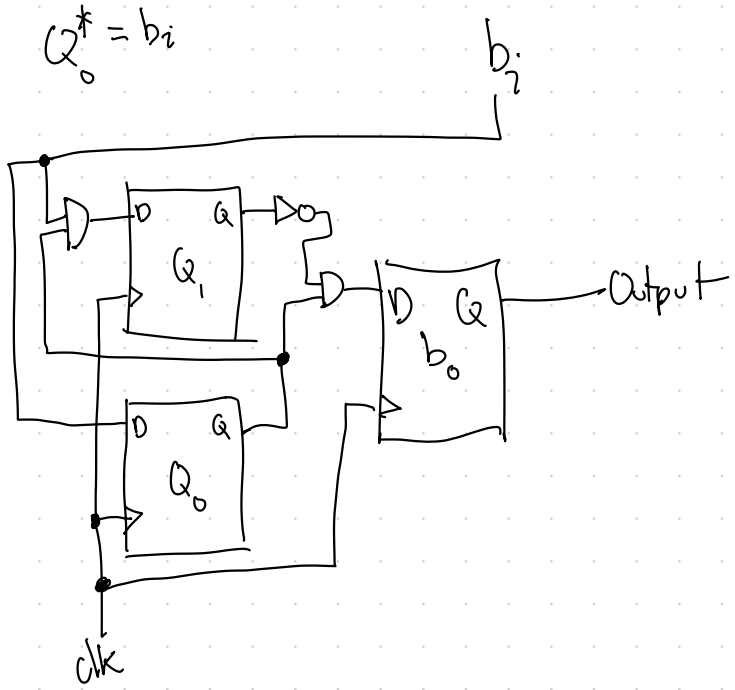
b)

$Q_1 Q_0$	$Q_1^* Q_0^*$ $b_i = 1$	$b_i = 0$
00	01	00
01	11	00
11	11	00

A: 00
B: 01
C: 11

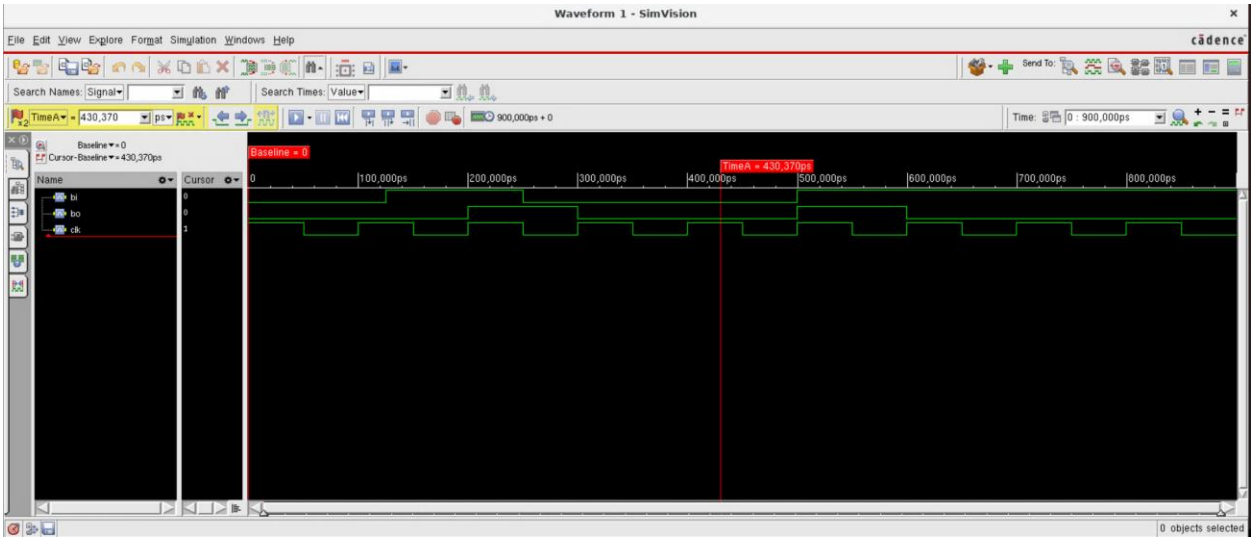
c) $Q_1^* = Q_0 b_i$ $\quad b_0 = \bar{Q}_1 Q_0$

$Q_0^* = b_i$



Problem 3)

Part d. Verification of FSM



FSM code:

```
`timescale 10ns/1ps
module fsm(
   input bi,clk,
   output reg bo
);
reg q0,q1;
always @(posedge clk) begin
   q1 = q0 & bi;
   q0 = bi;
   bo = ~q1 & q0;
end
endmodule
```