EE577A Lab1 Report

Huayu Fu
9/5/2017
4745159848

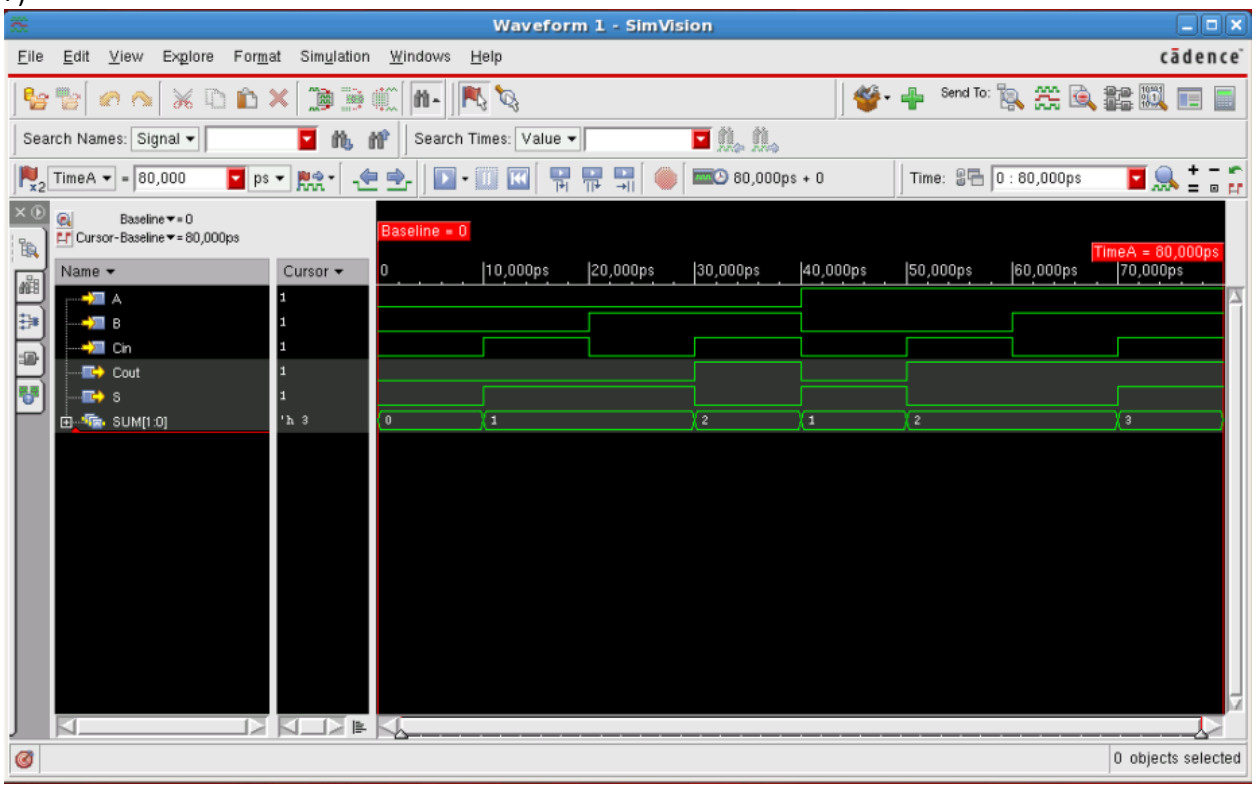
1.

C) “make simg” do the simulation and open GUI. “make sim” just do the simulation and do not open GUI.

D) If do not make clean, it will submit huge temporary files. And the grader do not want to see this.

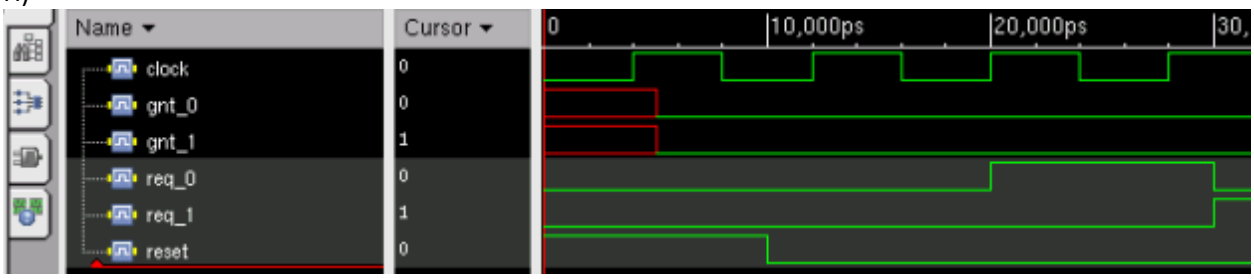
E) No, you cannot perform the simulation. The name in tcl file also need to be changed to perform simulation.

F)



G) Close the SimVision, make clean and make simg again. An alternative way is to choose Simulation -> Reinvoke Simulator in any SimVision window and choose Yes, recompile and then re-simulate.

H)



I) The simulation is not started when start the gui. Type the run in console to start simulation.

2.

a)

Number	#of bits	Base	Interpreted value
4'b11	4	Binary	11
8'b10101101	8	Binary	10101101
3'd6	3	Decimal	110
7'bx110	7	Binary	x110
6'o33	6	Octal	11011
'h6f	32	Hexadecimal	1101111
5'b10011	5	Binary	10011

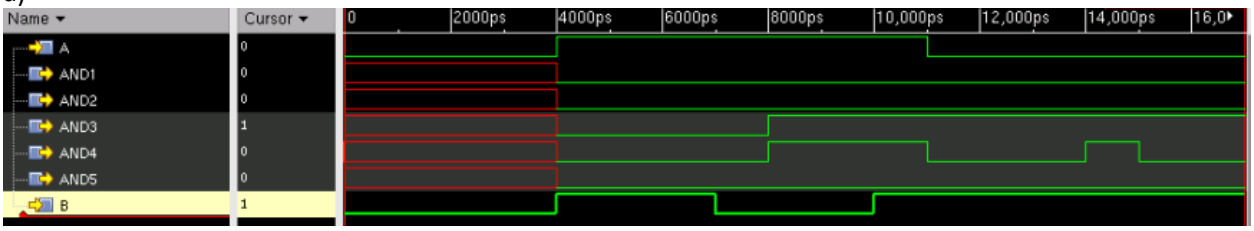
b) (a) clk is flipping at the maximum speed. (b) clk flipped every #2 time.

c) (a) 1 (b) 0 (c) 1 (d) 1101 (e) 0 (f) 1

d) (a) 1x11 (b) x (c) x (d) x

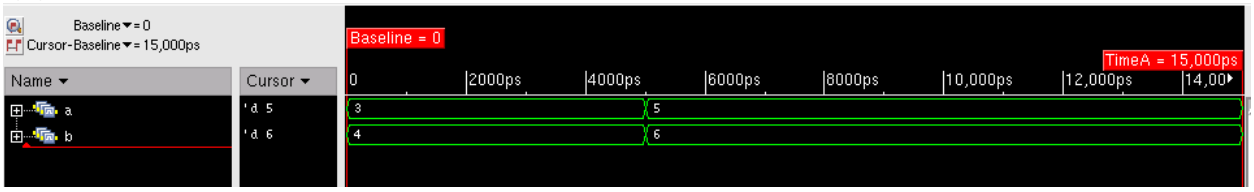
3.

a)

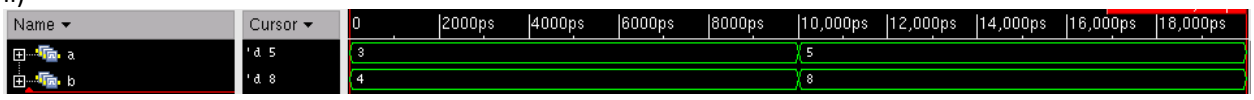


b) all of them will result in following response. Except the first one define out 2 times will create compiler error, and the second one missing the define of wire out, will also cause error.

c) i)

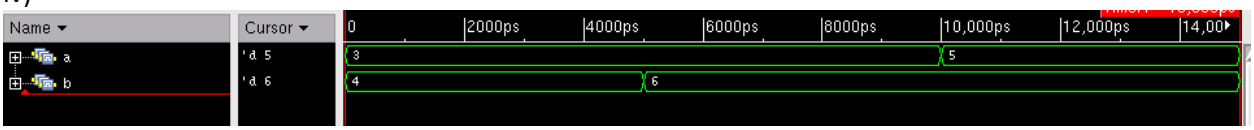


ii)

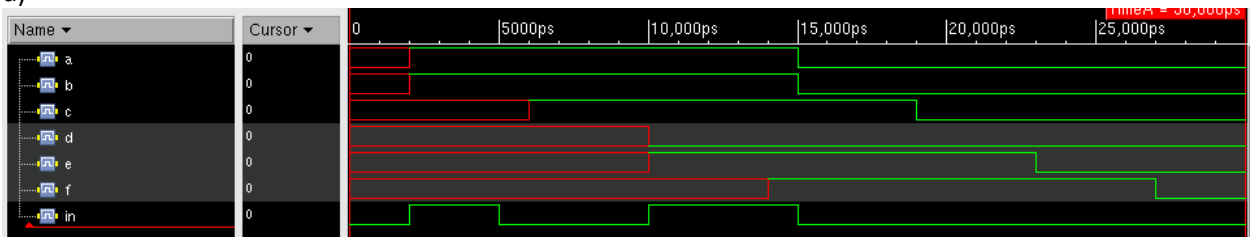


iii)

iv)



d)



e) shift din into a from left to right. Shift in once at posedge clk. Design A and C would work. Design B will always changing a[7] with din value at posedge clk. And other bits of a is the value of a[0]. Design D will change all bits of a to din value at posedge clk.

a<=(a>>1)|{din,7'b0000000};

4)

a)
```
ncsim> run
0
Simulation stopped via $stop(1) at time 1 NS + 0
```
result is 0

b) they are the same logically but save lines.

c) compare case equality (including x and z) or compare logical equality.

d) 2'b01 & 2'b10=0 if(0) goes to false case

e)

Operator	**Symbols**
Unary, Multiply, Divide, Modulus	!, ~, *, /, %
Add, Subtract, Shift	+, - , <<, >>
Relation, Equality	<,>,<=,>=,==,!=,===,!==
Reduction	&, !&,^,^~,|,~|
Logic	&&, ||
Conditional	? :

f)

assign statement

```
module mux (d1,d2,sel,out);
  input d1,d2,sel;
  output out;
  wire out;
  assign out=(sel)?d1:d2;
endmodule // mux
```

if statement

```
module mux (d1,d2,sel,out);
  input d1,d2,sel;
  output out;
  reg out;
  always @ ( sel or d1 or d2 ) begin
    if(sel) out=d1;
    else out=d2;
  end
```

case statement

```
endmodule // mux
module mux (d1,d2,sel,out);
  input d1,d2,sel;
```

```
  output out;
  reg out;
  always @ ( sel or d1 or d2 ) begin
    case (sel)
      1'b0: out=d0;
      1'b1: out=d1;
    endcase
  end
endmodule // mux
```

g)

```
always @ (posedge clk)
begin
a <= b;
b <= a;
end
```

h)

1. always #10 clk = ~clk;
2. always

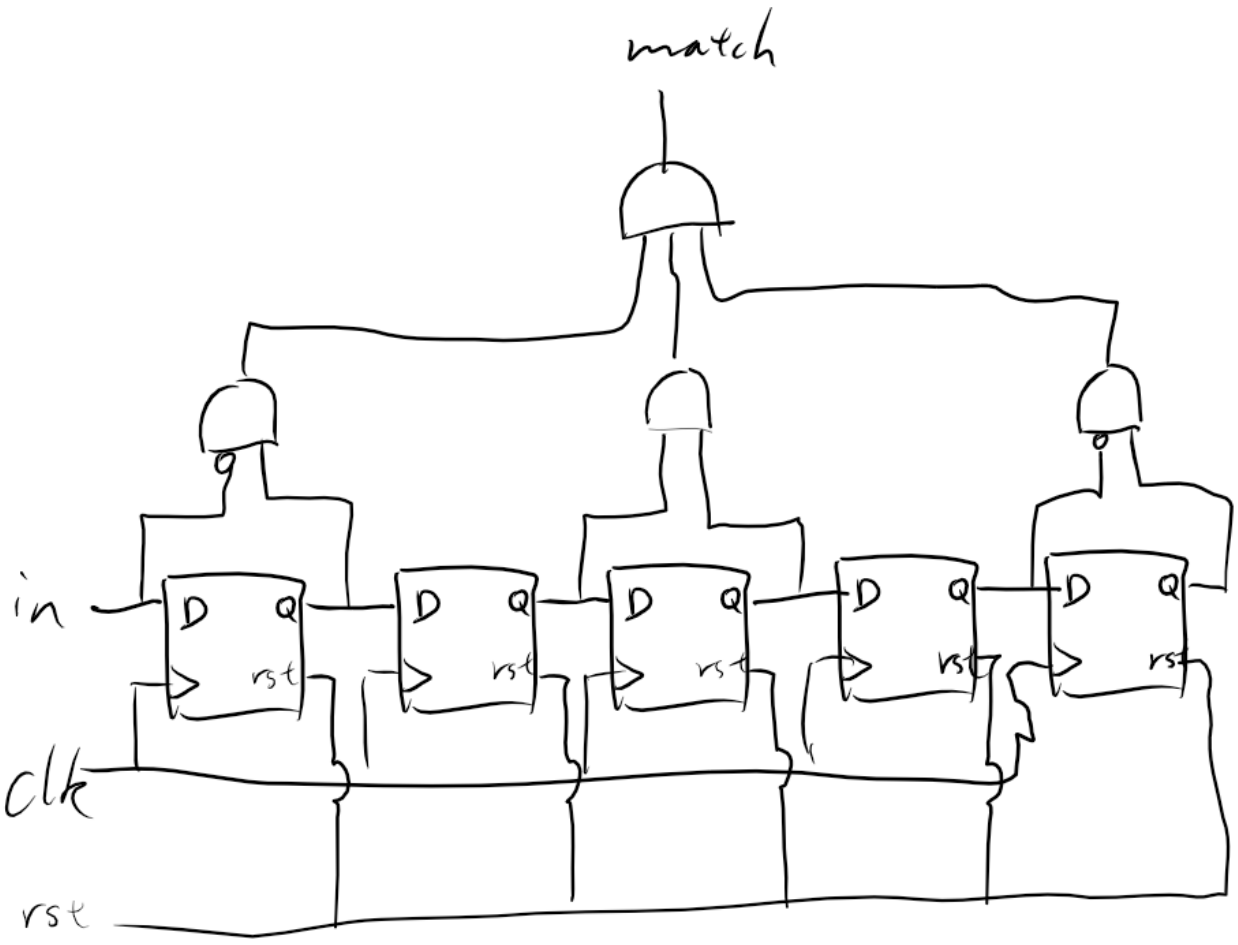
```
begin
clk = 0;
#10;
clk = 1;
#10;
end
```

k)

```
integer file;
string s;
file = $fopen("filename");
$fgets(s, n, file);
$fclose(file);
file = $fopen("filename");
$fdisplacy(file, s);
r = $fclose(file);
```

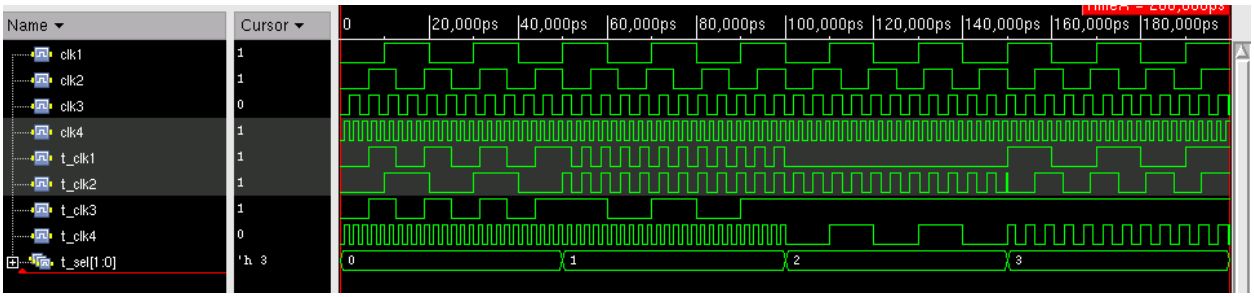
5） 2 extra credit: we will have 2 different kinds of structures. The difference is wither the NSL is in posedge clk always block or not. There is gated state signal and input signal for NSL if it is with in clocked always. block.

3 extra: have input connected to a series of registers. And then use combinational logic detect if the sequence match the desired one.



8)

9)



10) a)

```
module ten_a (clk,reset,in,out);
 input clk,reset,in;
 output out;
 wire clk,reset,in;
 reg out, delay;
 always @ ( clk ) begin
  if(reset) begin out=0; delay=0;end
  else begin
   if(out!=in)begin
    if(in)begin
     out=1;
    end
    else begin
     if(delay)begin out=1; delay=0 end
     else delay=1;
    end
   end
   else
    delay=0;
   end
  end
 end
endmodule // ten_a
```

b)

```
module ten_b (clk,reset,in1,in2,out);
  input clk,reset,in1,in2;
  output out;
  wire clk,reset,in1,in2;
  reg out, delay, rsame;
  reg[2:0] r1,r2;
  always @ ( clk ) begin
    if(reset) begin out=0; delay;r1=0;r2=0;end
    else begin
      r1<=r1<<1;
      r2<=r2<<1;
      r1[0]<=in1;
      r2[0]<=in2;
      if(r1=0'b010)begin
        if(r2[0]==1&&in2==0) out=1;
      end
      if(r2=0'b010)begin
        if(r1[0]==1&&in1==0) out=1;
      end

      if(r1[1:0]==2'b01&&r2[0]==1&&in2==0) delay<=1;
      if(r2[1:0]==2'b01&&r1[0]==1&&in1==0) delay<=1;

      if(r1[0]==r2[0]&&r1[0]==0&&in1==in2&&in1==1)rsame<=1;
      if(rsame==1&&r1[0]==r2[0]&&r1[0]==0)out=0;
      if(delay)begin out=1; delay=0; end
    end
  end
endmodule
```