國立彰化師範大學 資訊工程學系 硬體描述語言 期中考

2024/11/21

1. Answer the following questions: [**24%**: (3*8)%]

   (1) What is the “programmable” property of an FPGA chip?

   (2) What is the major purpose for performing “synthesis” for a Verilog-based design?

   (3) In the “inertial delay” model, what kind of input pulse can affect the output of a gate?

   (4) When the delay assumption is made for primitive gates in a design, what is the relation between this delay and the delay of the physical gates after synthesis?

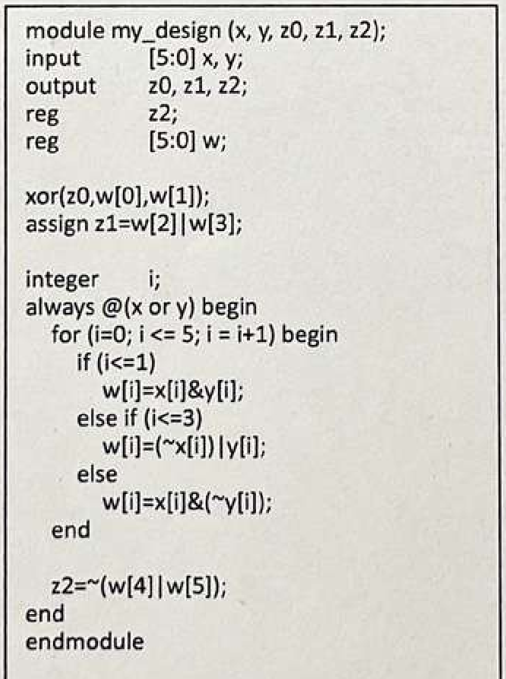
   (5) How to design an adder module that can be used as an 8-bit, 16-bit, or 32-bit adder, etc?

   (6) To show the text messages during simulation, what is the difference between “$monitor()” and “$display()”?

   (7) What is the “edge-triggered” property of a D-flipflop?

   (8) In the C/C++ programming, the final product is an executable file, what is the final product of a Verilog-based design?

2. For the Verilog module in Fig. 1, draw its gate-level implementation. [**12%**]

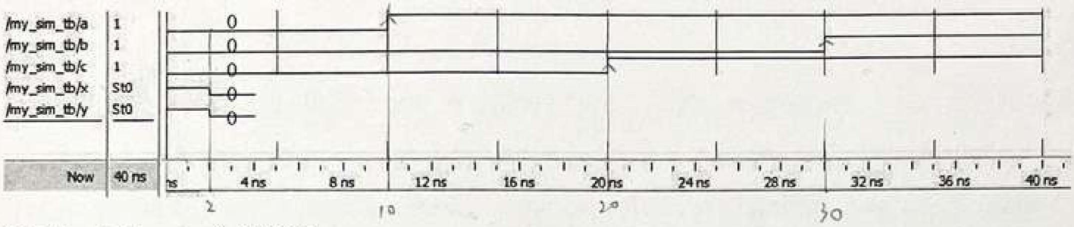


```
module my_design (x, y, z0, z1, z2);
input       [5:0] x, y;
output      z0, z1, z2;
reg         z2;
reg         [5:0] w;

xor(z0,w[0],w[1]);
assign z1=w[2]|w[3];

integer     i;
always @(x or y) begin
  for (i=0; i <= 5; i = i+1) begin
    if (i<=1)
      w[i]=x[i]&y[i];
    else if (i<=3)
      w[i]=(~x[i])|y[i];
    else
      w[i]=x[i]&(~y[i]);
  end

  z2=~(w[4]|w[5]);
end
endmodule
```

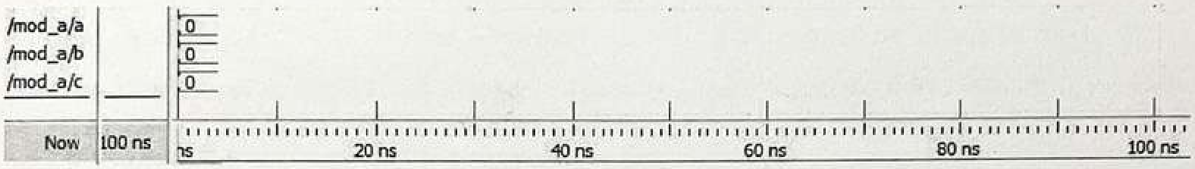
Fig. 1

3. Sketch the simulation waveforms of the modules in (a) Fig. 2, (b) Fig.3, (c) Fig. 4, (d) Fig. 5, and (e) Fig. 6, and (f) Fig. 7. (請標示出信號變化之時間點)

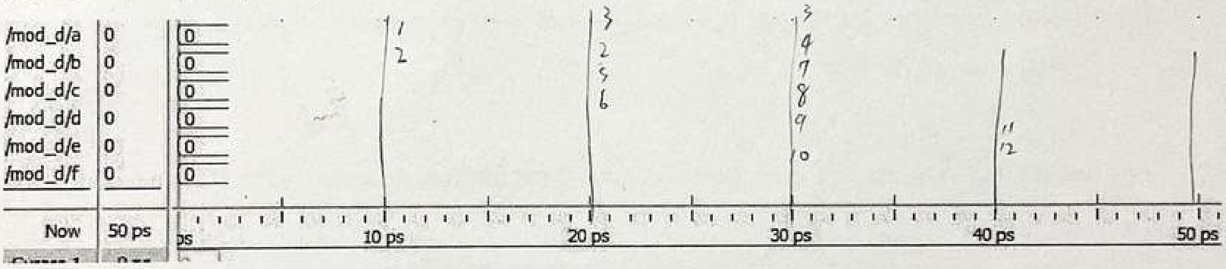
(a) Fig. 2 (my_sim) **[8%]**



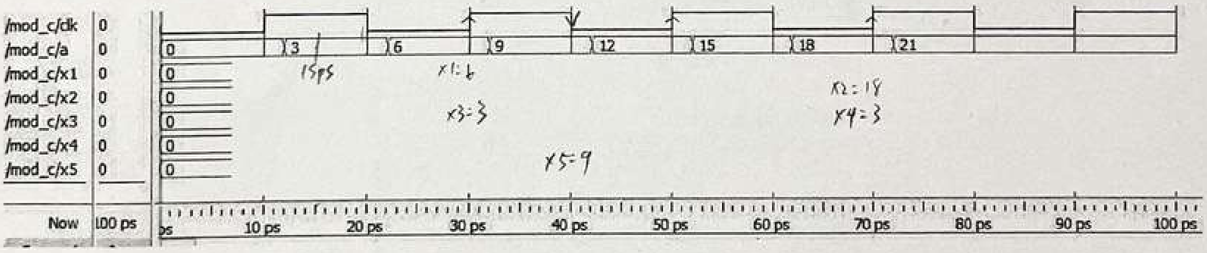
(b) Fig. 3 (mod_a) **[10%]**



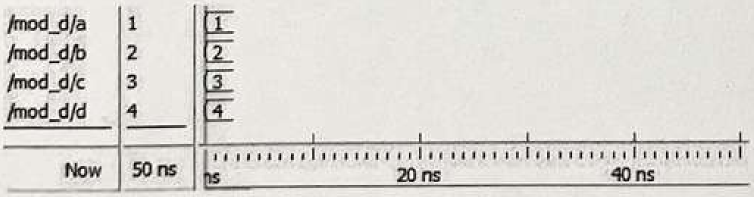
(c) Fig. 4 (mod_b) **[12%]**



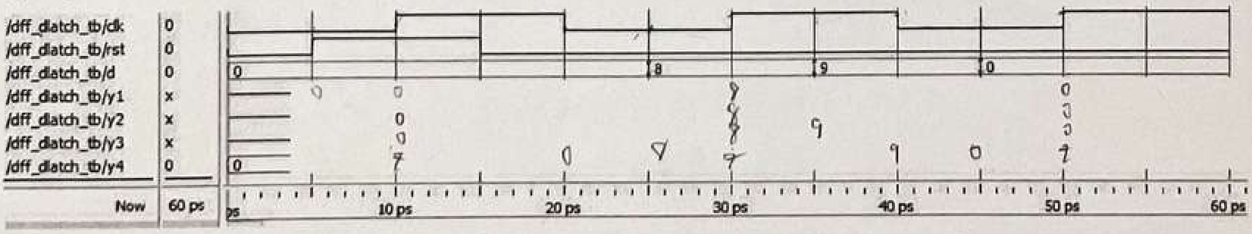
(d) Fig. 5 (mod_c) **[10%]**



(e) Fig. 6 (mod_d) **[12%]**



(f) Fig. 7 (dff) **[12%]**



```
module my_sim (a, b, c, x, y);
input      a, b, c;
output   x, y;

          0   1   1
xor #2 (x, a, b);
          0    0  1
and #2 (y, x, c);
endmodule
```

Fig. 2

```
module mod_a;
reg [3:0] a, b, c;

initial begin
        a = 0; b = 0; c = 0;
end

initial begin
        fork
          10    #10 a = 1;
          20    #20 b = 2;
        join

        fork
          30    #10 b = 3;
          40    #20 c = 4;
        join

        fork
                a = 5;

                fork
                  50  #10 a = 6;
                  50  #10 b = 7;
                  60  #20 c = 8;
                join

             90 #30 c = 9;
        join

end

initial
        #100 $finish;
endmodule
```

Fig. 3

```
module mod_b;
reg [3:0] a, b, c, d, e, f;

initial begin
        a=0; b=0; c=0;
        d=0; e=0; f=0;
end

initial begin
        #10
        a <= 1;
        b <= 2;
        a = #10 3;
        b = #10 4;
end

initial begin
        #20
        c = 5;
        d = 6;
        c <= #10 7;
        d <= #10 8;
end

initial begin
        #30
        e <= 9;
        f <= 10;
        e = 11;
        f = 12;
end

initial
        #50 $finish;
endmodule
```

Fig. 4

```
module mod_c;
reg clk;
reg [4:0] a, x1, x2, x3, x4, x5;

initial begin
        clk=0; a=0;
        x1=0; x2=0; x3=0; x4=0; x5=0;
end

initial begin
        #12 a=3; #10 a=6; #10 a=9;
        #10 a=12; #10 a=15; #10 a=18;
        #10 a=21;
end

always
        #10 clk=~clk;

initial
        #15 @(posedge clk) x1=a;
initial
        #15 repeat(3)
                @(posedge clk) x2=a;
initial
        #15 x3 = @(posedge clk) a;
initial
        #15 x4 = repeat(3) @(posedge clk) a;
initial
        #15 @(posedge clk) @(negedge clk) x5=a;

initial
        #100 $finish;
endmodule
```

Fig. 5

```
module mod_d;   a → b → c → d
reg [3:0] a, b, c, d; ↑
                      5
initial begin
        a=1; b=2; c=3; d=4;
end

initial begin
        #10
        d <= c; c <= b; b <= a; a <= 5;
end

initial begin
        #20
        d = c; c = b; b = a; a = 6;
end

initial begin
        #30
        a <= 7; b <= a; c <= b; d <= c;
end

initial begin
        #40
        a = 8; b = a; c = b; d = c;
end

initial
        #50 $finish;
endmodule
```

Fig. 6

```
module dff_dlatch(clk, rst, d, y1, y2, y3, y4);
input       clk, rst;
input       [3:0] d;
output      [3:0] y1, y2, y3, y4;
reg         [3:0] y1, y2, y3, y4;

always @(posedge clk or posedge rst)
        y1 = (rst==1) ? 0 : d;

always @(posedge clk)
        y2 = (rst==1) ? 0 : d;

always @(clk or d)
        if (clk==1) y3=d; else y3=y3;

always @(clk or d)
        if (clk==0) y4=d; else y4=4'hz;
endmodule
```

Fig. 7

學年度第　　　　學期

國立彰化師範大學　　考試答案卷

評　　分	評　閱　者
80	

學系 資工　班別 116　科目

學號 S1254040　姓名 羅章弘

+12

1. (1) 電路行為可用程式改變

+1 (2) 將程式合成電路

(3) 輸入訊號的持續時間大於 inertial delay

+1 (4) 模擬電路無 delay 或自己加上固定 delay, 實際電路 physical getes 的 delay 會根據晶片大小、溫度等因素浮動。

(5)

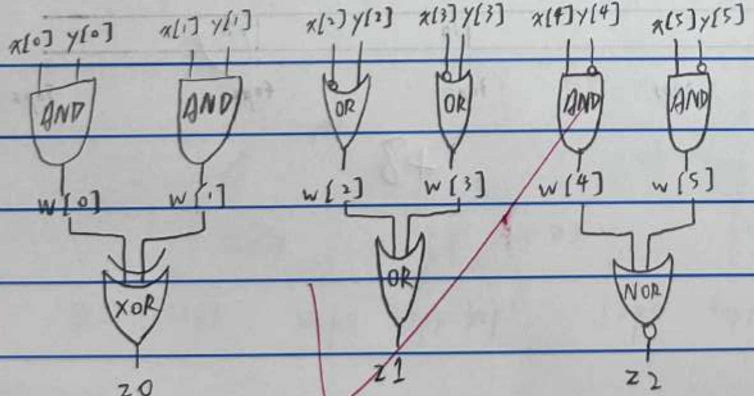
+1 (6) "$monitor()" 在所有動作之後才執行輸出, "<=" 之後 "$display()" 依由上到下的順序執行輸出

(7) D-flipflop：當 clk 做正緣/負緣改變的瞬間才可以儲存訊息。

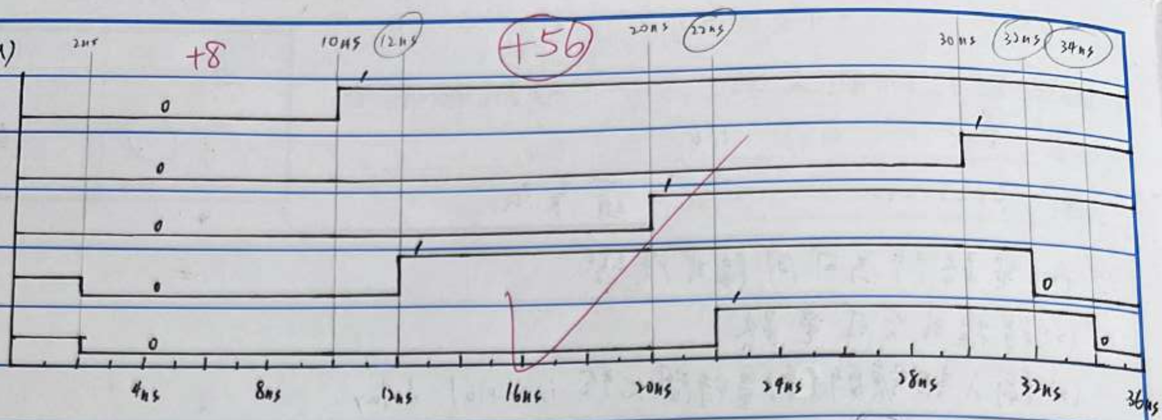
(8)

2.

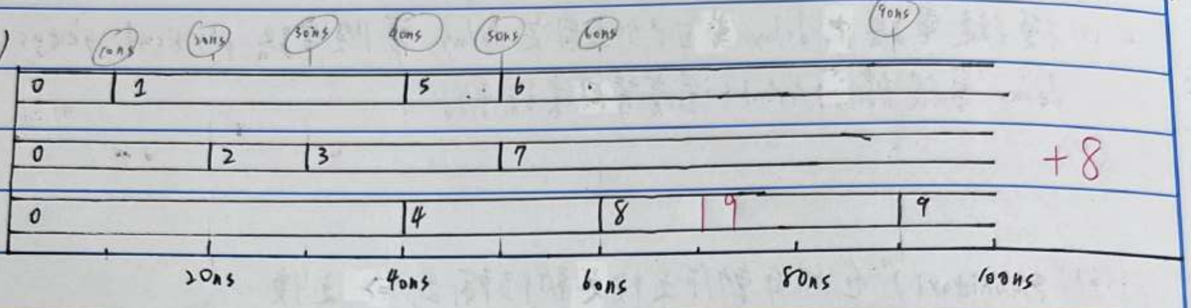
+12



3. (a) +8 +56

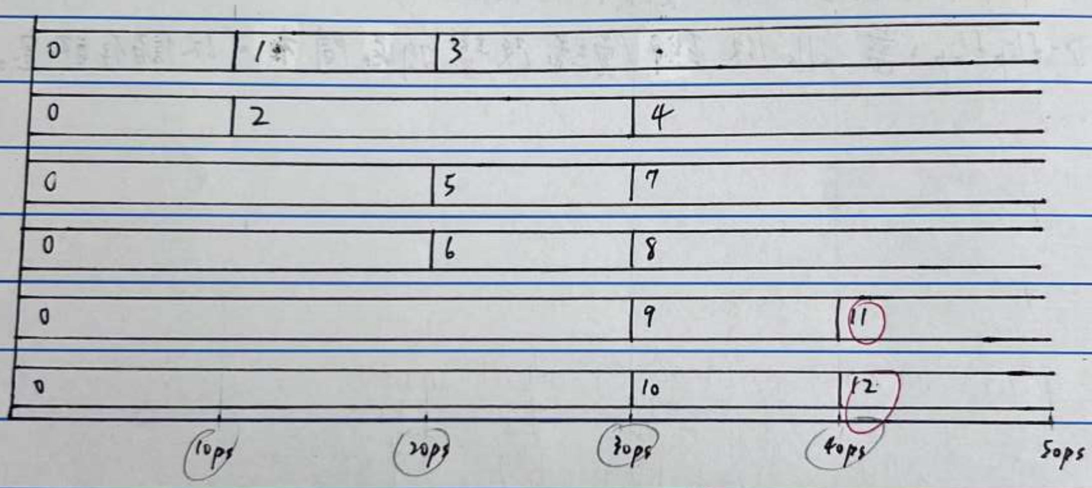


(b)



+8

(c)



+8

(d)

clk

a

✓x1

x2

✓x3

✓x4

✓x5

(e)

a

b

c

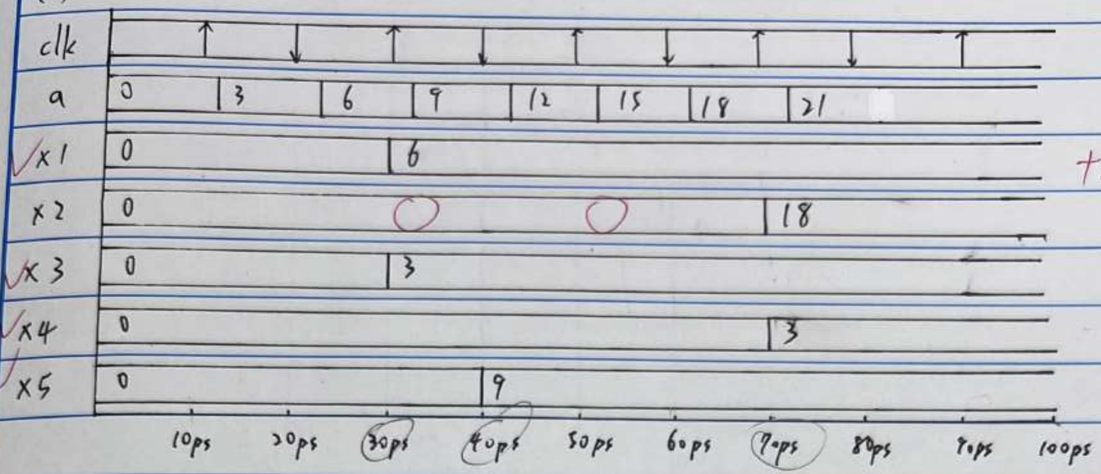
d

(f

34ns

36ns

(d)



+8

(e)

a	1	5	6	7	8
b	2	1	5	6	8
c	3	2	1	5	8
d	4	3	2	1	8

0 10ns 20ns 30ns 40ns 50ns

+12

(f)

+12

y1	x	0				8				0
y2	x		0			8				0
y3	x		0			8	9			0
y4	0		4'hz	0	8	4'hz		9	0	4'hz

5ps 10ps 20ps 25ps 30ps 35ps 40ps 45ps 50ps