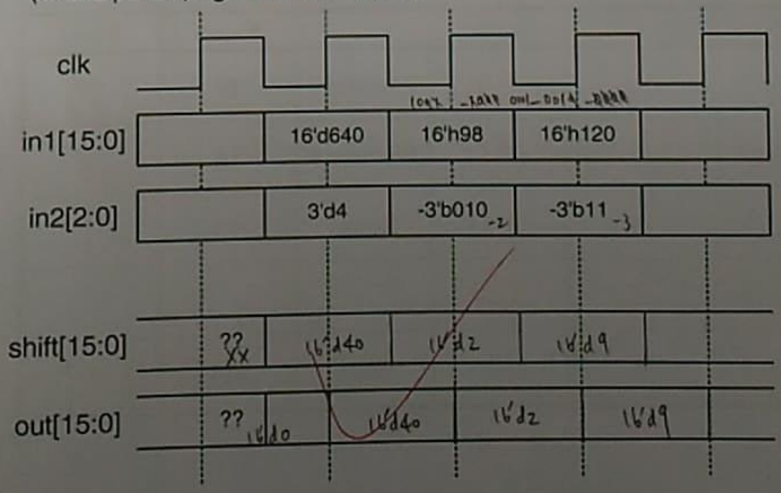
1. < Code Debugging and Simulation > (10pts)

A. **(5pts)** Identify syntax error, correct, and explain: 0.5pts for each. Identify inappropriate code (or semantics error), correct, and explain: 0.5pts for each.

```
module 2%shifter (out,clk, rst, in1, in2);  ② 少了分號
          ① 數字不能當成開頭
input clk, rst;
input [15:0] in1;
input [2:0] in2;
       reg
output [15:0] out;  ③ out 放在 always 內，宣告成 reg

wire
~~reg~~[15:0] shift;  ④ shift 用 assign → 宣告成 wire
assign shift = in1 >> in2;
/* variable shifter */  ⑤ 註解少 */

                          ⑥ 用 or
always @(posedge clk and posedge rst)  begin  ⑦ 少 begin
if(!rst) out <= 16'd0;
else  out <= shift;
end          ⑨ sequential block → non-blocking

endmodule  ⑩ 少 endmodule
```

⑧ 不用!
Active high reset

B. **(5pts)** Finish the waveform below based on the circuit in part A. Note that you should use the decimal number representation to answer (such as 16'd0). Use "xx" to indicate values that cannot be determined from the information given. (In this period, signal rst is always 0)



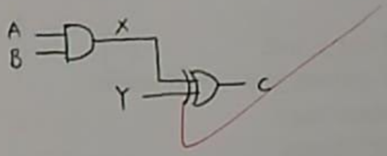
640/2^4
= 640/16
= 40

110 = 6
101 = 5

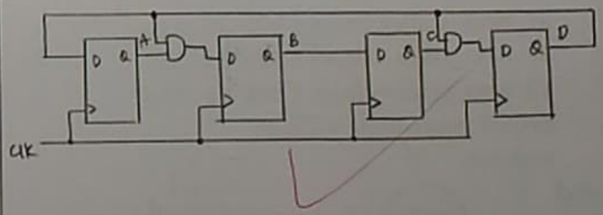
2. < Logic Synthesis + Blocking & Non-Blocking > (10pts)

In the following table, the left column show some pieces of Verilog RTL code. Please draw the corresponding circuits in the right column. You can use AND, OR, NAND, NOR, XOR, XNOR, NOT, MUX, D Flip-Flop, Latch in the circuit diagram.

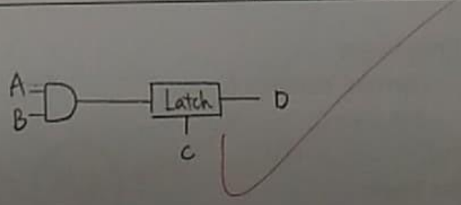
(a) Verilog Code (2pts)	Circuit Diagram

always @(*) begin
X = A&B;
Y = X^C;
end
	

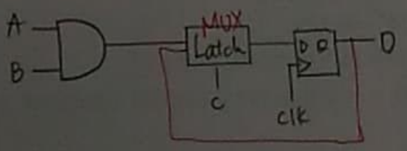
(b) Verilog Code (2pts)	Circuit Diagram

always @(posedge clk)
begin
A <= D;
B <= A ^ D;
C <= B;
D <= C ^ D;
end
	

(c) Verilog Code (3pts)	Circuit Diagram

always@(A or B or C)
begin
if (C)
D = A & B;
end
	

(d) Verilog Code (3pts)	Circuit Diagram

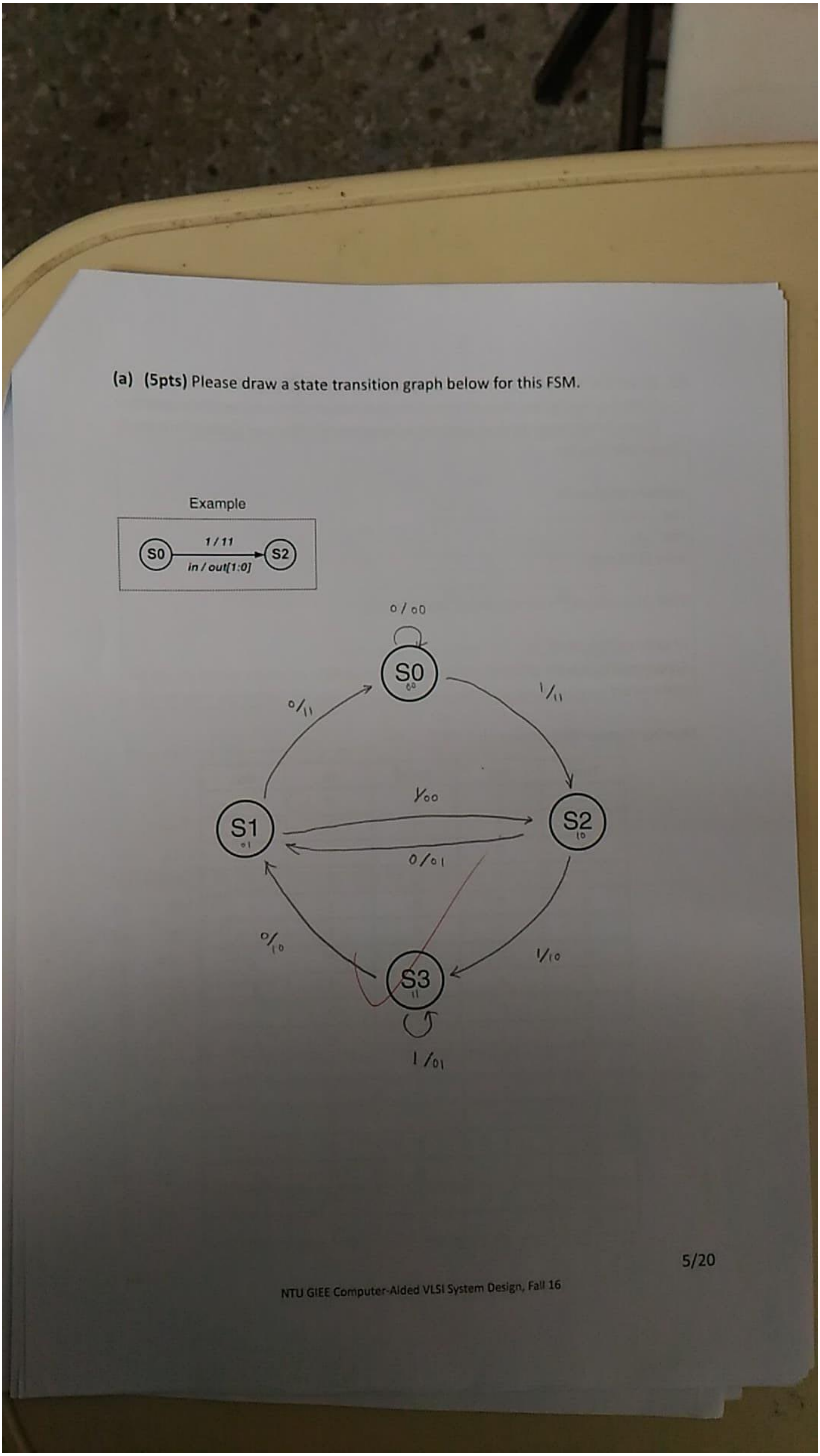
always@(posedge clk)
begin
if (C)
D <= A & B;
end
	

-1

3. < **Finite State Machine and Simulation** > **(10pts)**

Given below is a Finite-State-Machine (FSM).

```
module FSM   (clk, rst, in, out_r);
parameter S0 = 2'b00, S1 = 2'b01, S2 = 2'b10, S3 = 2'b11;

input   clk, rst, in;
output [1:0] out_r;

reg [1:0] out_r, out;
reg [1:0] current_state, next_state;
// Next State Logic
always@(*) begin
    case(current_state)
    S0: next_state = (in == 1'b0)? S0 : S2;
    S1: next_state = (in == 1'b0)? S0 : S2;
    S2: next_state = (in == 1'b0)? S1 : S3;
    S3: next_state = (in == 1'b0)? S1 : S3;
    default: next_state = 2'b00;
    endcase
end
// Current State Memory & Output Register
always@(posedge clk or posedge rst)
begin
    if(rst) begin
        current_state <= 0;
        out_r <= 0;
    end
    else begin
        current_state <= next_state;
        out_r <= out;
    end
end
// Output Logic
always@(*) begin
    out[1] = in ^ current_state[0];
    out[0] = in ^ current_state[0] ^ current_state[1];
end

endmodule
```



(a) **(5pts)** Please draw a state transition graph below for this FSM.

Example

S0
1/11
in / out[1:0]
S2
0/00
S0
00
0/11
1/11
1/00
S1
01
S2
10
0/01
0/10
1/10
S3
11
1/01

(b) **(5pts)** We have put this module in our testbench as a Design-Under-Test (DUT). After the simulation, the command window has printed response from monitor. Please finish the output results below based on this FSM and given information.

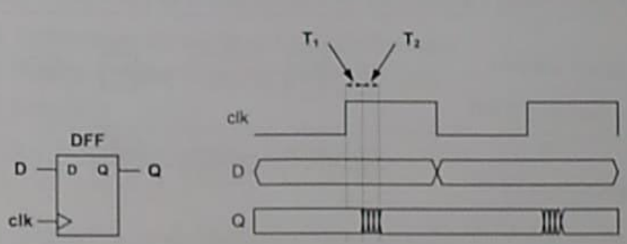
```
`timescale 1ns/1ns

module testbench;
reg   clk, rst;
reg   in;
wire [1:0] out;

FSM DUT(.clk(clk), .rst(rst), .in(in), .out_r(out));

// APPLY STIMULUS
$monitor("%t %b %b %b %b", $time, clk, rst, in, out);
endmodule
```

Monitor Output Response:

Time	clk	reset	in	out	cs	ns
0	0	0	0	XX		
✓ 1	1	1	0	00	S0	
2	0	0	1	00	S0	S2
✓ 3	1	0	1	1 1	S2	S3
4	0	0	1	1 1	S2	S3
✓ 5	1	0	1	1 0	S3	S3
6	0	0	1	1 0	S3	S3
✓ 7	1	0	1	0 1	S3	S3
8	0	0	0	0 1	S3	S1
✓ 9	1	0	0	1 0	S1	S0
10	0	0	1	1 0	S1	S2
✓ 11	1	0	1	0 0	S2	S3
12	0	0	0	0 0	S2	S1
✓ 13	1	0	0	0 1	S1	S0
14	0	0	0	0 1	S1	S0
✓ 15	1	0	0	1 1	S0	S0
16	0	0	0	1 1	S0	S0

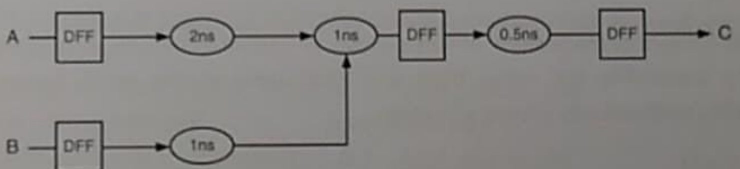
4. < Important Timing Parameters > (15pts)

Suppose that the timing characteristics of the flip-flops in the circuit are the same. Their timing diagrams and parameters can be described as follows:



T_1=0.2ns T_2= 0.3ns $t = \frac{1}{250\times10^6} = \frac{4}{4\times250\times10^6} = 4$ ns

The circuit below operates at the clock frequency of **250MHz.** Suppose that the rise, fall, and turn-off delays for each combinational element are the same.



(a) (3pts) Write the timing inequality for setup time and hold time.

(i) Setup time

$T_{clk\text{-}Q} = 0.5$ ns

$$\begin{cases} 0.5+2+1+T_{setup} < 4 \\ 0.5+1+1+T_{setup} < 4 \\ 0.5+0.5+T_{setup} < 4 \end{cases}$$

$\Rightarrow T_{setup} < 0.5$ ns

(ii) hold time

$T_{clk\text{-}Q} = 0.2$

$$\begin{cases} 0.2+2+1 > T_{hold} \\ 0.2+1+1 > T_{hold} \\ 0.2+0.5 > T_{hold} \end{cases}$$

$\Rightarrow 0.7$ (ns) $> T_{hold}$

(b) (2pts) If T_{setup} (Setup Time) = 1ns T_{hold} (Hold Time) = 1ns

Are there setup time and hold time violations in this circuit? Use the timing inequalities in (a) for setup/hold time at the clock frequency to check them.

$T_{setup} = 1$ ns ($T_{setup} < 0.5$ (ns)) ⇒ setup time violation

$T_{hold} = 1$ ns ($T_{hold} < 0.7$ (ns)) ⇒ hold time violation

(c) (5pts) Followed by part (b), if there are setup/hold time violations in this circuit, how to perform "retiming" to solve these issues? Suppose that all of combinational elements cannot be further separated. Please draw your circuit and write the timing inequalities of the modified circuits after retiming.

(i) Setup

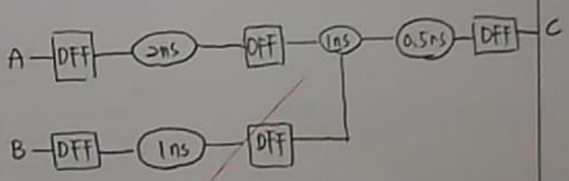
$0.5 + T_{compute} + 1 \le 4$ (ns)

$\Rightarrow T_{compute}$ (long) ≤ 2.5 (ns)

(ii) hold

$0.2 + T_{compute}$ (short) ≥ 1 (ns)

$\Rightarrow T_{compute}$ (short) ≥ 0.8 (ns)



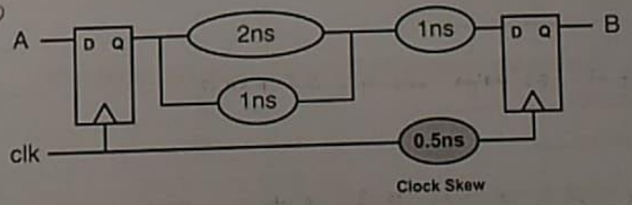
$0.5 + 2 + T_{setup} < 4 \Rightarrow T_{setup} < 1.5$ (ns)

$0.2 + 1 > T_{hold} \Rightarrow T_{hold} < 1.2$ (ns)

$\Rightarrow$ no timing violation

(d) (5pts) If there is clock skew in this circuit, as shown in bellow. Please write the timing inequality for setup time and hold time at the clock frequency of **200MHz without** any timing violation.

$\frac{1}{200 \times 10^6} = \frac{5}{5 \times 200 \times 10^6} = 5$ (ns)



-1

(i) $0.5 + 3 + T_{setup} < 5 + 0.5 \Rightarrow T_{setup} < 2$ (ns)

(ii) $0.2 + 1 + 1 > T_{hold} + 0.5 \Rightarrow T_{hold} < 2.2 - 0.5 = 1.7$ (ns)

30

5. < Synthesis Issues > (30pts)

A. < Important files related to Design Compiler >

Please explain the meaning of the following terminologies and where to use them:

(a) (2pts) Technology library (e.g: slow.db/fast.db)

(b) (2pts) Standard Delay File (e.g: CHIP_syn.sdf)

(c) (2pts) tsmc13.v

(a) 記錄在不同操作環境下的 cell 以及 timing 資訊.（ss. ff. corner 等）

(提供給dc做合成用)

(b) .sdf 檔記錄了合成完後電路的時間資訊 (delay 等.)

(提供給ncverilog做模擬用)

(c) tsmc13.v 為 library 檔. 提供合成完後的 verilog netlist 內部 cell 的資訊.

('behavior model) (〃)

B. < Synopsys Design Constraints File(SDC) >

Please explain the meaning of the following command and why we use them in Design Compiler:

(a) (2pts) *set_dont_touch_network [get_clocks clk]*
set_ideal_network [get_ports clk]

(b) (2pts) *set_clock_uncertainty 0.1 [get_clocks clk]*

(a) 在 clk 這條路徑上不做任何優化（等到 APR 再做），所以也不需考量這條路徑的 loading. 故有 set_ideal_network

(b) 在長 clk tree 時可能 clk 到每個 register 的時間會有些微差距. uncertainty 為此可能時間. 此指令提供計算 slack 時. 用較差情況計算 skew

C. (3pts) < STA & Post-sim >

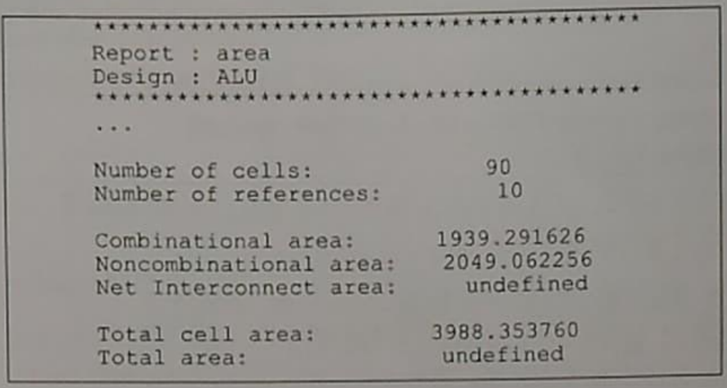
If we specify the clock to be 5.0ns during synthesis, the timing report shows that the constraints has been **met**. However, the gate-level simulation passed at 4.0ns with one set of test data. Is this possible? Why or why not?

有可能. critical path 的時間小於(≤) 5 ns. 但其他路徑可能不到 5(ns).

此組測資可能未經過 critical path, 所以 4 (ns) 仍有機會 pass

D. < Area Report >

The following figure is the area report after synthesis.



```
*********************************************
Report : area
Design : ALU
*********************************************
...

Number of cells:                  90
Number of references:             10

Combinational area:          1939.291626
Noncombinational area:       2049.062256
Net Interconnect area:         undefined

Total cell area:             3988.353760
Total area:                    undefined
```

(a) **(2pts)** It sometimes shows "undefined" in total area. Please explain it and describe how to fix it.

(b) Followed by part(a),

I. **(2pts)** In cell-based IC design flow, we will focus on total cell area instead of total area, please explain why.

II. **(4pts)** The total cell area will be underestimated in this situation. Please briefly explain why.

(c) **(3pts)** With the same RTL code, if we reduce the clock cycle, what part in the report will increase? Please briefly explain why.

(a) undefined：因為合成時未提供 wire load model 資訊，故無法得知 net area

⇒ set wire load model 即可解決

(b)

(I) 因為合成時，每個 cell 的實際位置並未確定，所以實際晶片上的繞線長度不確定，故不考慮 net 的 area

(II) 因為未設定 wire load model，合成時會以為線的 delay = 0，因此可以計算的時間上升，tool 會用小一點的 cell 來合成

(c) Combinational area 會上升，因為計算時間變少，需要用面積較大的 cell 來運算增加速度

E. < External IP issue >

If there's a memory module in DUT, and we generate several files from Memory Generator. Such as **rom_1024x4_t13_slow_syn.db, rom_1024x4_t13.v.**

(a) **(2pts)** Please explain what's the purpose of these files and what will happen if we don't have these files.

與 standard cell 類似，db 檔為 technology file，而 .v 檔則為 verilog 檔提供模擬時使用。若缺少 db 則不知道不同 operating-condition 下的資訊。若少了 .v 檔，gate-level simulation 時會無法模擬

nice!

(b) **(2pts)** Please modify Design Compiler setting file .synopsys_dc.setup as shown below. **(JUST NEED TO POINT OUT WHERE TO MODIFY)**

```
set search_path "Your_path/CBDK_TSMC013_Arm/CIC/SynopsysDC $search_path "
set target_library "slow.db fast.db" rom_1024x4_t13_slow_syn.db
set link_library    " * $target_library dw_foundation.db"
set symbol library "tsmc13.sdb generic.sdb"
set synthetic_library "dw_foundation.sldb"
...
```

(c) **(2pts)** Should we synthesis **rom_1024x4_t13.v** with DUT.v ? Please explain why.

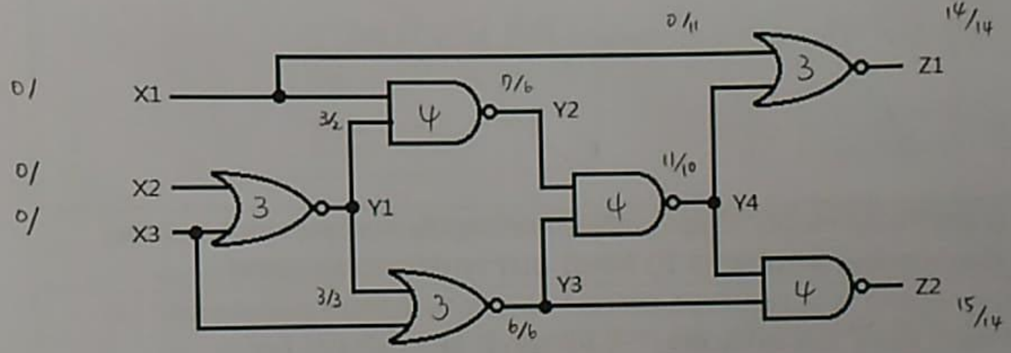
不用，.v 檔內部為用來提供行為模擬時的資訊，與合成時無關。

memory layout 檔已有

10

6. **< Timing Analysis > (10pts)**

Calculate the arrival time, required time, and slack at each gate output, and find a critical path from primary input to primary output. Assume the delays of NAND gates and NOR gates are **4ns** and **3ns**, respectively. The arrival time at primary inputs is **0ns** and the required time at primary outputs is **14ns**.



X1: Arrival 0 ; Required 2 ; Slack 2

X2: Arrival 0 ; Required -1 ; Slack -1

X3: Arrival 0 ; Required -1 ; Slack -1

Y1: Arrival 3 ; Required 2 ; Slack -1

Y2: Arrival 7 ; Required 6 ; Slack -1

Y3: Arrival 6 ; Required 6 ; Slack 0

Y4: Arrival 11 ; Required 10 ; Slack -1

Z1: Arrival 14 ; Required 14 ; Slack 0

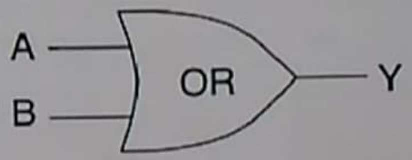
Z2: Arrival 15 ; Required 14 ; Slack -1

Critical path: X2, Y1, Y2, Y4, Z2, slack=-1

7. < Design for Testability > (15pts)

(a) **(5pts)** Given one OR gate for your reference below. Answer the following questions.

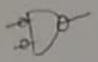
15



Complete the single stuck-at-fault (SSF) table of two-input OR gate below for the output value with SSF. By *signal/logic_value* we mean single stuck-at-*logic_value* fault on *signal*, e.g. *A*/0 means stuck-at-0 fault on signal *A*. **Please also mark the output value at Y differing from fault free one with "*" character (such as "1*").**

Input		Fault-free Output	Output Value on Y with SSF					
A	B	Y	A/0	A/1	B/0	B/1	Y/0	Y/1
0	0	0	0	1*	0	1*	0	1*
0	1	1	1	1	0*	1	0*	1
1	0	1	0*	1	1	1	0*	1
1	1	1	1	1	1	1	0*	1

✓

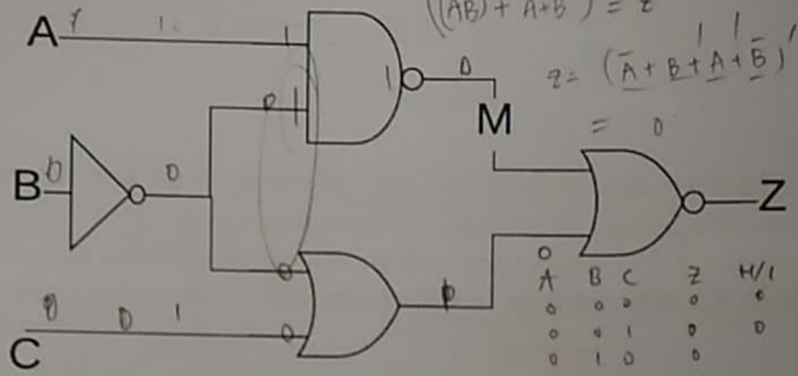


(b) **(10pts)** Given the circuit below, please generate one test pattern to detect the faults given as below. You may use D-Algorithm to generate the pattern. Please write down your pattern in the form of {abc}, e.g. {01X}, where X is don't-care bit.

$((A\bar{B})' + A+B')' = Z$

$Z = (\bar{A} + B + A + \bar{B})'$

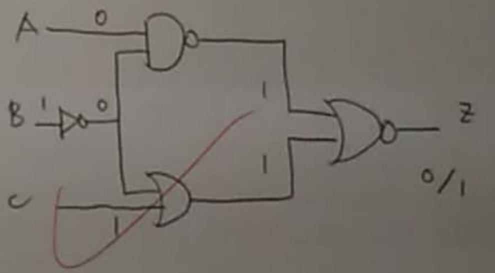
$= 0$



A	B	C	Z	M/1
0	0	0	0	0
0	0	1	0	0
0	1	0	0	
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	0	
1	1	1	0	

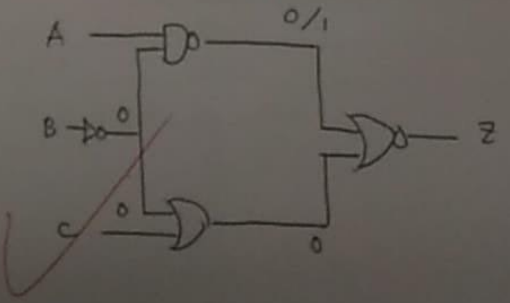
1. **(5pts)** Z s-a-1

{ABC} = {011}



2. **(5pts)** M s-a-1

No pattern can detect M SA1.



Bonus!! < STA & Post-sim inconsistent issue > (15pts)

The following is the RTL code from testbench (tb) and Design Under Test (DUT) module. The RTL simulation with those files already passed. However, if we use the SDC (Synopsys Design Constraints) file shown below, the timing report shows that the constraints is **met**, but the post-sim will fail.

Testbench (tb)

```
`timescale 1ns/10ps
`define CYCLE 2.0
`define SDFFILE "./DUT_syn.sdf"
`define End_CYCLE 100

`define PAT "./pattern.dat"
`define EXP "./golden.dat"

module tb;

parameter N_EXP = 10;
parameter N_PAT = N_EXP;

integer         i, exp_num, err;
reg             over;

reg    [3:0]   in_mem    [0:N_PAT-1];
reg    [3:0]   exp_mem [0:N_EXP-1];

reg             clk;
reg             rst;
reg             in_en;
reg    [1:0]   A, B;
wire            O_ready;
wire   [3:0]   O;
reg    [3:0]   O_exp;

DUT DUT( .clk(clk), .rst(rst), .in_en(in_en), .A(A), .B(B), .O_ready(O_ready), .O(O));

initial $sdf_annotate(`SDFFILE, DUT);

initial $readmemh(`PAT, in_mem);
initial $readmemh(`EXP, exp_mem);
initial begin
   clk              = 1'b0;
   rst              = 1'b0;
   in_en            = 1'b0;
   exp_num          = 0;
   err              = 0;
   over             = 0;
end

always #(`CYCLE/2) clk = ~clk;

// data input
initial begin
   @(negedge clk) rst = 1'b1;
   #(`CYCLE);        rst = 1'b0;

   @(negedge clk) i=0;
   while (i <= N_PAT) begin
      in_en = 1'b1;
      A        = in_mem[i][3:2];
      B        = in_mem[i][1:0];
      i = i +1;
      @(negedge clk);
   end
end

// result compare
always @(negedge clk) begin
   O_exp = exp_mem[exp_num];
   if(O_ready) begin
      if(O != O_exp) begin
         $display("ERROR at %5d:O %4h !=O_exp %4h " ,exp_num, O, O_exp);
         err = err + 1;
      end
      exp_num = exp_num + 1;
   end
   if(exp_num == N_EXP) over = 1;
end
```

```
initial begin
   #(`CYCLE * `End_CYCLE);
   $display("-----------------------------------------------------\n");
   $display("Error!!! Somethings' wrong with your code ...!\n");
   $display("-------------------------FAIL------------------------\n");
   $display("-----------------------------------------------------\n");
   $finish;
end

initial begin
   @(posedge over)
   if((over) && (exp_num!='d0)) begin
      $display("-----------------------------------------------------------\n");
      if (err == 0)    begin
         $display("All data have been generated successfully!\n");
         $display("-------------------------PASS------------------------------\n");
      end
      else begin
         $display("There are %d errors!\n", err);
         $display("-----------------------------------------------------\n");
      end
   end
   #(`CYCLE/2); $finish;
end

endmodule
```

DUT

```
module DUT( clk, rst, in_en, A, B, O_ready, O);
   input             clk;
   input             rst;
   input             in_en;
   input    [1:0]    A;
   input    [1:0]    B;
   output            O_ready;
   output [3:0]      O;

   reg           O_ready, O_ready_nxt;
   reg [3:0] O;
   reg [3:0] A_sqr, A_sqr_nxt;
   reg [3:0] B_sqr, B_sqr_nxt;
always@(*) begin
   A_sqr_nxt = A*A;
   B_sqr_nxt = B*B;

   if(in_en)
      O_ready_nxt = 1'b1;
   else
      O_ready_nxt = 1'b0;

      O = A_sqr + B_sqr;
end

always@(posedge clk or posedge rst) begin
   if(rst) begin
      A_sqr <= 2'd0;
      B_sqr <= 2'd0;
      O_ready <= 1'b0;
   end
   else begin
      A_sqr <= A_sqr_nxt;
      B_sqr <= B_sqr_nxt;
      O_ready <= O_ready_nxt;
   end
end
endmodule
```

Synopsys Design Constraints (SDC)

```
set cycle   2.0
create_clock -name clk -period $cycle [get_ports clk]
set_fix_hold                                    [get_clocks clk]
set_dont_touch_network                          [get_clocks clk]
set_ideal_network                               [get_ports clk]
set_clock_uncertainty                     0.1   [get_clocks clk]
set_clock_latency                         0.5   [get_clocks clk]

set_max_fanout 6 [all_inputs]

set_operating_conditions -min_library fast -min fast -max_library slow -max slow
set_drive          1        [all_inputs]
set_load           1        [all_outputs]
set_input_delay   0.1 -clock clk [remove_from_collection [all_inputs] [get_ports clk]]
set_output_delay 0.1 -clock clk [all_outputs]
set_wire_load_model -name tsmc13_wl10 -library slow
```

(a) **(5pts)** Please explain what message might show in terminal and why?

可能會有 setup time violation，因為 Input delay in sdc = 0.1 但 tb 在 negedge clk

+3 給值，運算時間可能會太長造成 setup time violation

Error at … /output delay

(b) **(5pts)** If we can modify the SDC file, please explain what's wrong in it and how to fix it.

input delay 設成 $ cycle/2 與 tb 相符合

+3. output delay " $ cycle/2 "

(c) **(5pts)** If we can **only** modify RTL code in DUT module, please describe what's wrong and how to fix it. **(YOU DON'T NEED TO WRITE MODIFIED CODE)**

若不想考慮 Input/output delay，則 RTL 寫的時候 input/output port 都

+5 擋一個 register，如此一來，皆有一個完整的 clk cycle 可以計算，不受

Input or output delay 影響 ✓