Computer-Aided VLSI System Design
Midterm Examination
2013.11.06

Name __________________________________

Student ID __________________________________

Instructions

This is a ***CLOSED*** book exam. The exam is to be completed in 90 minutes. If you need scratch paper, just use the blank parts of these pages; show all of your work on these pages. Before you start writing, please check if you have all 12 pages of the exam.

Score Board (to be filled by TAs)

	Points	Score		Points	Score
Problem 1	18		Problem 5	10	
Problem 2	15		Problem 6	10	
Problem 3	10		Problem 7	15	
Problem 4	10		Problem 8	12	
			Total	100	

1. <General Concept> (18%)

A. Briefly explain the following terms using a few sentences.

(a) (3pts) What is *Combinational Loop*?

(b) (3pts) What is *Boundary Optimization*?

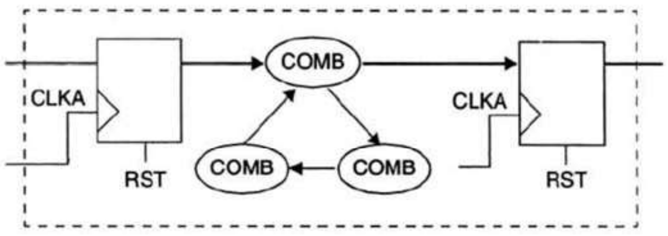
(c) (3pts) What is *Fault Coverage*?

(d) (3pts) Explain how to reduce timing on critical path by pipelining technique?

(a) (20131009_CVSD_L6_Synthesizable_Coding.pdf 第 37 頁)

An output of a combinational block feeds back to an input of the same block.

Bad: Combinational processes are looped

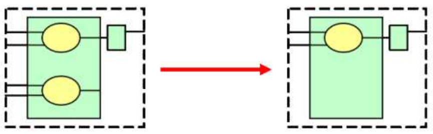


(b) (20131009_CVSD_L6_Synthesizable_Coding.pdf 第 9 頁)

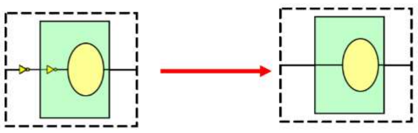
Optimize across boundaries

1. Remove logic driving unconnected outputs
2. Remove redundant inverters
3. Propagates constants to reduce logic

- Remove logic driving unconnected outputs



- Remove redundant inverters



- Propagates constants to reduce logic

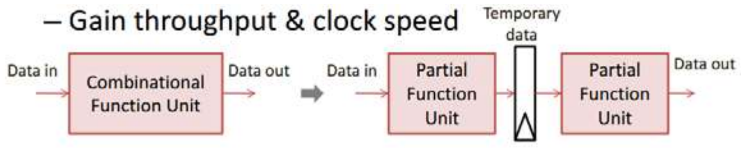
(c) (20131030_CVSD_LA_Testing.pdf 第 28 頁)

$$\text{Fault Coverage} = \frac{\text{number of detected faults}}{\text{total number of faults}}$$

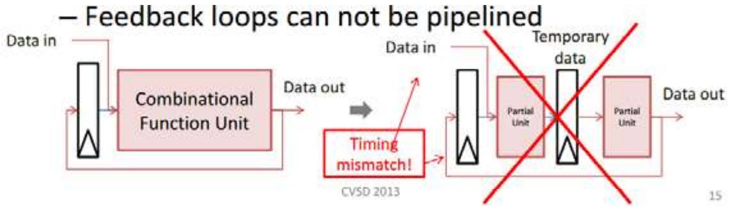
(d) (20131009_CVSD_L5_Design_Guideline.pdf 第 15 頁)

1. Gain throughput & clock speed
2. Feedback loops can't be pipelined

– Gain throughput & clock speed

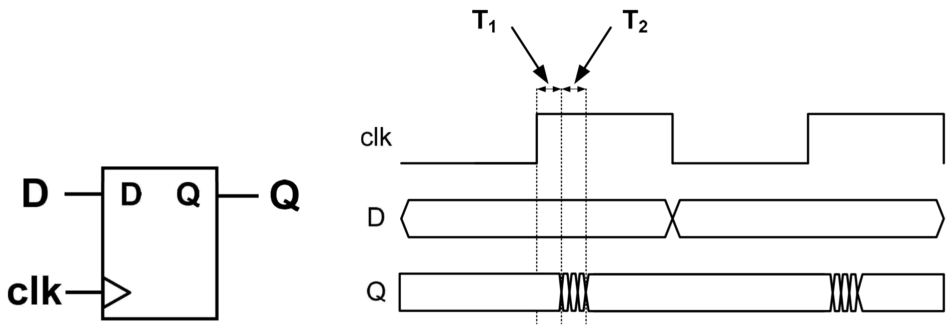


– Feedback loops can not be pipelined



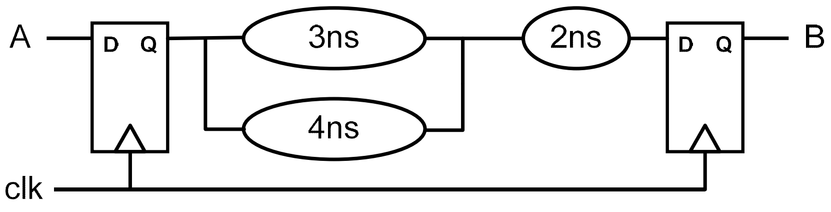
B. Setup time and hold time:

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



T_1=0.5ns T_2= 0.5ns

If the circuit in the below operates at the clock frequency of **125MHz without** any timing violation:



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

(b) (3pts) Write the timing inequality for hold time.

(20130925_CVSD_L2_Behavior_Modeling.pdf 第 19~22 頁)

(a) $T_{setup} < T_{clock} - T_{cq} - T_{logic}$

T_{clock} = 1000/125 = 8.0ns T_{logic} = 4.0ns+2.0ns = 6.0ns $T_{cq} = T_1 + T_2$= 1.0ns

$\rightarrow T_{setup} <$ 8.0ns – 1.0ns – 6.0ns = 1.0ns

(b) $T_{cq,cd} + T_{logic,cd} > T_{hold}$

$T_{logic,cd}$ = 3.0ns + 2.0ns = 5.0ns $T_{cq,cd} = T_1$ = 0.5ns

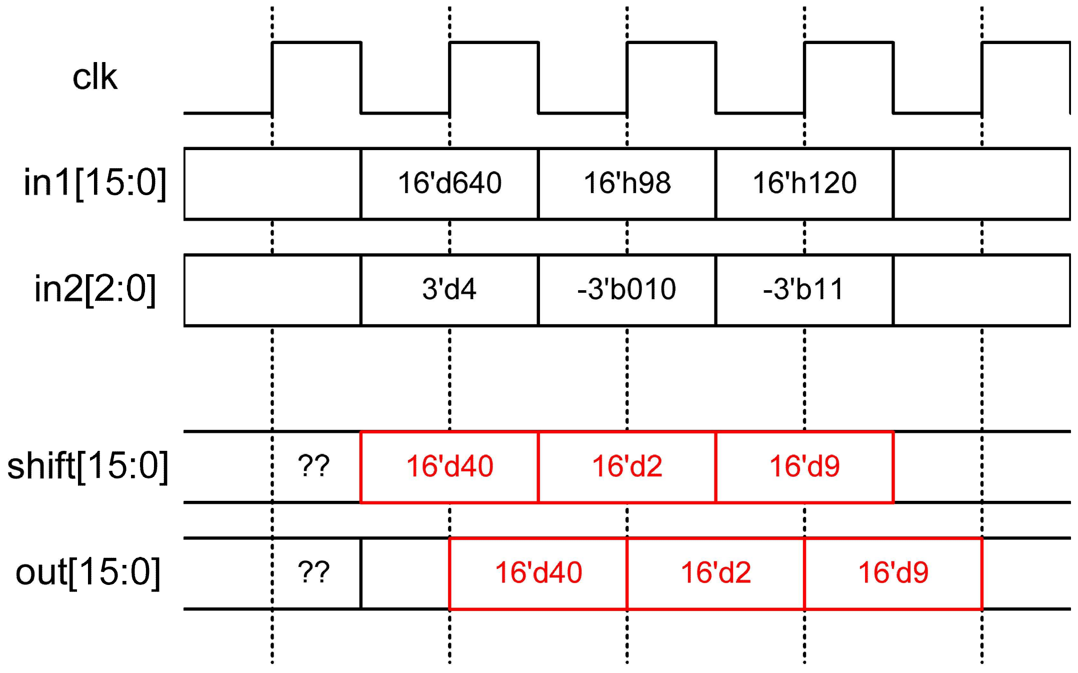
$\rightarrow$ 0.5ns + 5.0ns = 5.5ns $> T_{hold}$

2. <Verilog HDL –Code Debugging> (15%)

A. (10pts) Identify syntax errors, correct them and explain: 1pt for each. Identify inappropriate code (or semantics errors), correct them and explain: 1pt for each.

```
module 2%shifter (out,clk, rst, in1, in2);   (1.0pts)  分號結尾
          (1.0pts)  命名開頭必須為大小寫字母或底線(_)
input clk, rst;
input [15:0] in1;
input [2:0] in2;

output [15:0] out; reg [15:0] out;   (1.0pts) Register Output

~~reg[15:0] shift;~~  wire [15:0] shift;   (1.0pts) Continuous Assignment 的對象必須為 Wire
assign shift = in1 >> in2;
/* variable shifter*/ (1.0pts)  少一個星號必須補上

always @(posedge clk or posedge rst) begin (1.0pts)  少一個 begin
~~if(!rst) out   =   16'd0~~;   if(rst) out   <=   16'd0; (2.0pts) (1) High Level 重置  (2) Non-blocking
~~else   out   =   shift;~~     else   out   <=   shift;   (1.0pts) (1) Non-blocking
end

endmodule (1.0pts) (1) endmodule 沒有 s
```

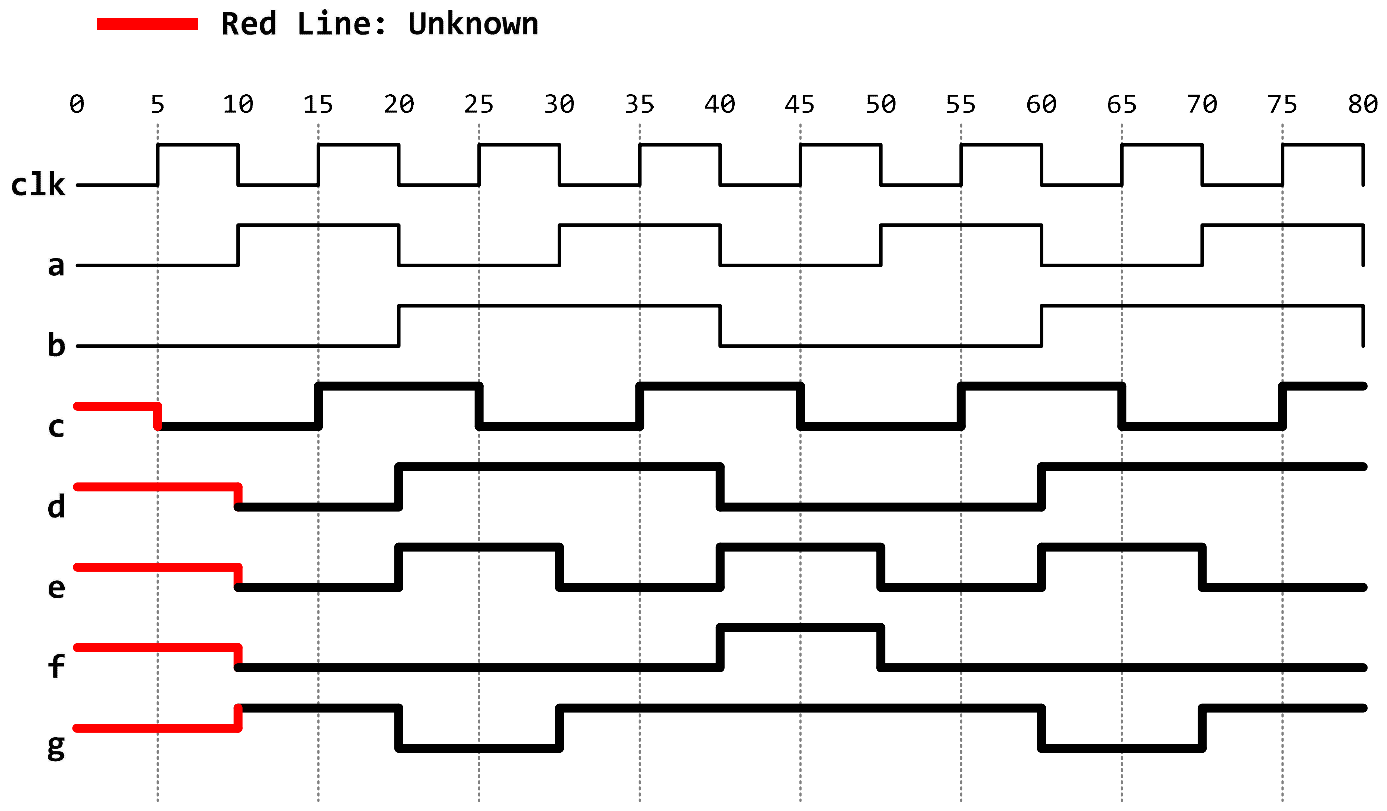
B. (5pts) Finish the waveform below based on the circuit in part A. Note that you should use the decimal number representation to answer (such as 16'd0).Use "xx" to indicate values that cannot be determined from the information given.



3. <Verilog HDL –Simulation> (10%)

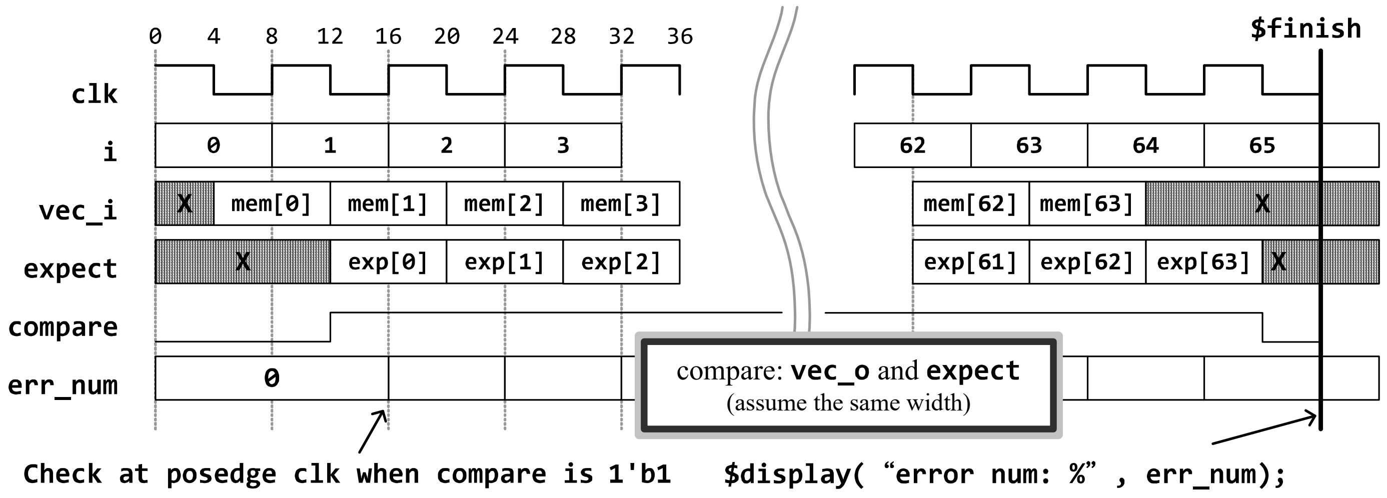
The bellow is the behavior code from a testbench. Draw the waveform before 80ns.

```
always@(posedge clk) begin
    c <= a;
    d <= #5 a^b;
    #5 e <= c;
    f <= @(negedge clk) a & d;
    @(negedge clk) g <= a|d;
end
```



4. <Verilog HDL –Testbench Writing> (15%)

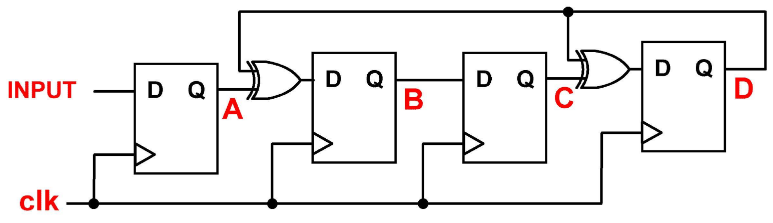
Complete the key initial/always blocks testbench to generate the following waveform and control the progress of simulation. Assume every necessary identifier has been declared, and the array of **mem** and **exp** is pre-loaded in another initial block at zero time. (Hint: Generate **clk** and **i** independently. Use different initial/always constructs for each of the rest signals: **vec_i**, **expect**, **compare**, and **err_num**. Remember to call system tasks at final).



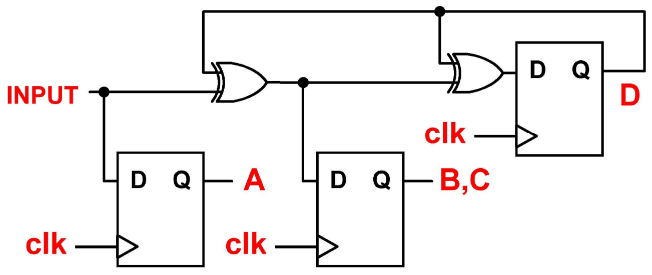
5. <Logic Synthesis + Blocking & Non-Blocking > (10%)

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 in the circuit diagram.

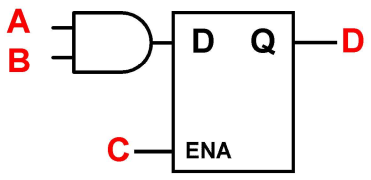
(a) Verilog Code (2pts)	Circuit Diagram

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

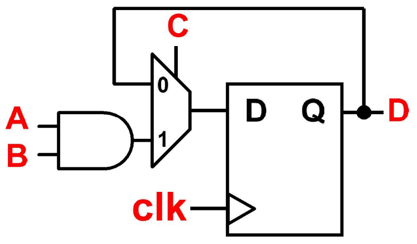
(b) Verilog Code (2pts)	**Circuit Diagram**

always @(posedge clk) begin
A = INPUT;
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
	

6. <Synthesis Issues> (10%)

(a) (6pts) The "multiple design instance" warning message results from using the same HDL description to represent more than one design instance. How would you handle it? Explain the property of each method. (Hint: There are **three** methods)

(20131023_CVSD_L9_SynthesisPart2b.pdf 第 35~40 頁)

Dont_touch: Hierarchy will be maintained.
During design optimization, the dont_touch block will not be re-optimized.
If dont_touch is placed on an unmapped design, the design will remain unmapped.
If you want to preserve the hierarchy & source sharing, use dont_touch.

Ungroup: Remove a single level of hierarchy
Does not preserve the hierarchy
Take more memory and compile time
If you want your design to have the BEST result, recommend to use ungroup, but it needs the most memory and compile time.

Uniquify: Create a unique design file for each instance
May select one cell or entire design hierarchy to be Uniquify
It is the easiest way to fix "multiple design instance", but needs much memory & compile time.

(b) (2pts) 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?

It is possible to pass at 4.0ns with one set of test data.

(c) (2pts) Use a simple example to explain how **retiming** improves the performance of a sequential circuit.

(20131016_CVSD_L7_SynthesisPart1.pdf 第 34~36 頁)

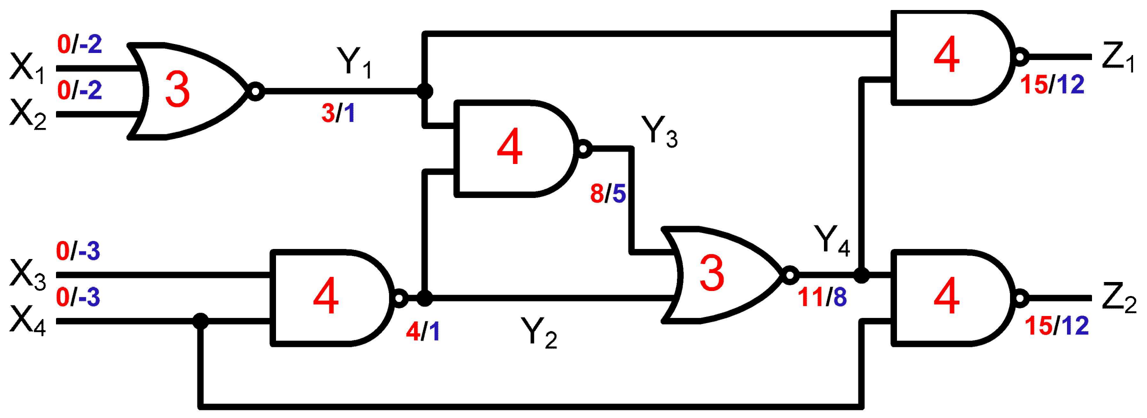
Retime registers from input to output of a gate or vice versa (does not affect the gate functionality) to have a larger combinational logic block.

—Relocating registers to explore optimality

—Cheap due to structural operation instead of functional operation

7. <Timing Analysis> (15%)

Calculate the arrival time, required time, and slack at each gate 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 12ns.



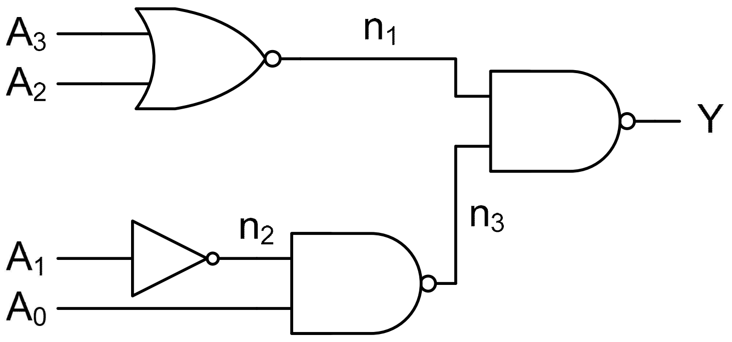
X1:	Arrival	0ns	; Required	-2ns	; Slack	-2ns
X2:	Arrival	0ns	; Required	-2ns	; Slack	-2ns
X3:	Arrival	0ns	; Required	-3ns	; Slack	-3ns
X4:	Arrival	0ns	; Required	-3ns	; Slack	-3ns
Y1:	Arrival	3ns	; Required	1ns	; Slack	-2ns
Y2:	Arrival	4ns	; Required	1ns	; Slack	-3ns
Y3:	Arrival	8ns	; Required	5ns	; Slack	-3ns
Y4:	Arrival	11ns	; Required	8ns	; Slack	-3ns
Z1:	Arrival	15ns	; Required	12ns	; Slack	-3ns
Z2:	Arrival	15ns	; Required	12ns	; Slack	-3ns

Please identify the critical paths (with slack less than 0).

$X_3,X_4 \rightarrow Y_2 \rightarrow Y_4 \rightarrow Z_1,Z_2$

8. <Design for Testability> (12%)

A. Given the circuit below, please answer the following questions.



(a) (2pts)How many stuck-at faults (SSF) are in the circuit?

(b) (4pts) Generate all possible test patterns for fault $n_1/0$.

(c) (4pts) Generate all possible test patterns for fault $n_2/0$.

(d) (2pts) What is the fault coverage for patterns generated for fault $n_3/1$?

(a) 8 x 2 = 16

(b) Activation: $n_1 = 1 \rightarrow A_3 = 0$ & $A_2 = 0$

Propagation: $n_3 = 1 \rightarrow (n_2,A_2) = (1,0)\ (0,1)\ (0,0) \rightarrow (A_1,A_2) = (0,0)$ or $(1,1)\ (1,0)$

Patterns (A_0,A_1,A_2,A_3) : (0,0,0,0) (1,1,0,0) (1,0,0,0)

(c) Activation: $n_2 = 1 \rightarrow A_1 = 0$ & $A_0 = 1$

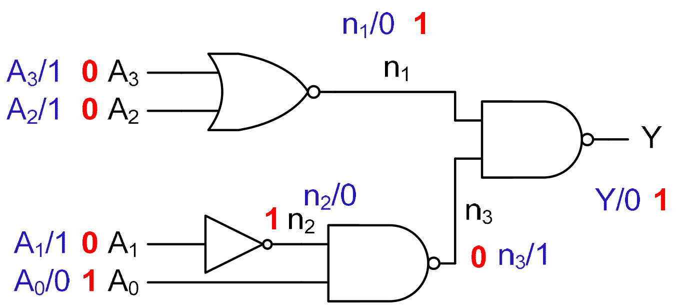
Propagation: $n_1 = 1 \rightarrow A_3 = 0$ & $A_2 = 0$

Patterns (A_0,A_1,A_2,A_3) : (1,0,0,0)

(d) Activation: $n_3 = 0 \rightarrow n_2 = 1$ & $A_0 = 1 \rightarrow A_1 = 0$ & $A_0 = 1$

Propagation: $n_1 = 1 \rightarrow A_3 = 0$ & $A_2 = 0$

Patterns (A_0,A_1,A_2,A_3) : (1,0,0,0)



n_3 = 0 →, faults cannot propagate through n_1, remove { n_1/0, A_2/1, A_3/1}

FC = 5/16 (detect 4 faults: {A_0/0, A_1/1, n_2/0, n_3/1, Y/0} out of 16 faults)