National Cheng Kung University

Department of Electrical Engineering

Introduction to VLSI CAD (Spring 2024)

Lab Session 5

FSM&Synthesis of Sequential Logic

Name	Student ID	
鄭喆嚴	E14096724	
Practical Sections	**Points**	**Marks**
Lab in class	15	
Prob A	20	
Prob B	10	
Prob C	15	
Report	35	
File hierarchy, naming…etc.	5	
Notes:		

Due Date: 14:59, April 4, 2024 @ moodle

Deliverables

1) All Verilog codes including testbenches for each problem should be uploaded.
   NOTE: Please **DO NOT** include source code in the paper report!
2) All homework requirements should follow the naming rule in this file hierarchy or you will not get the full credit.
   NOTE: Please **DO NOT** upload waveforms!
3) Important! TA will use the command in Appendix A to check your design under SoC Lab environment, if your code can not be recompiled by TA successfully using the commands, you will not get the full credit.
4) If you upload a dead body, which we cannot even compile, you will get **NO** credit!
5) All Verilog file before synthesizing should get at least 95% Superlint Coverage.
6) Lab5_Student_ID.tar (English alphabet of Student_ID should be **capital**.)

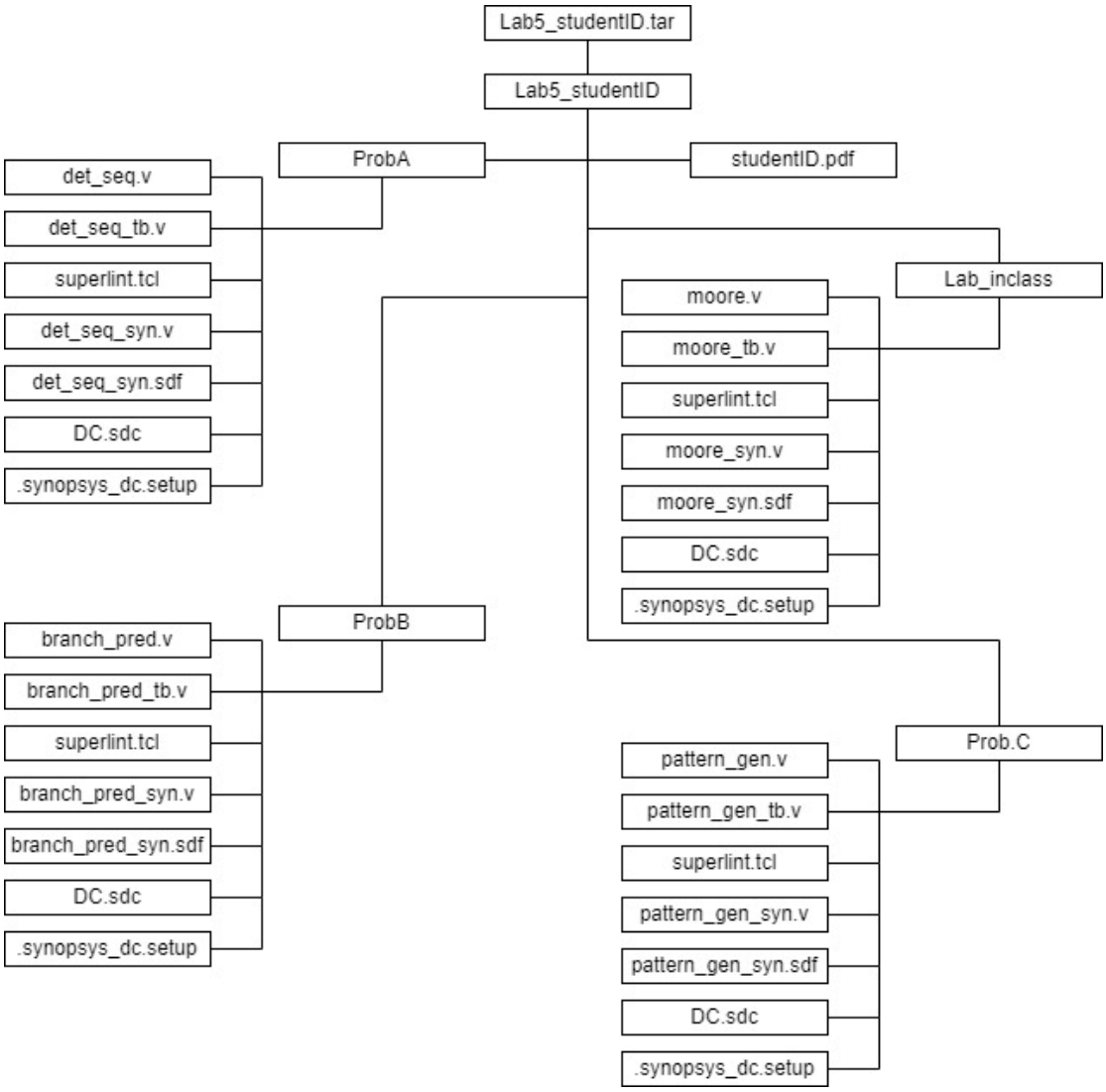
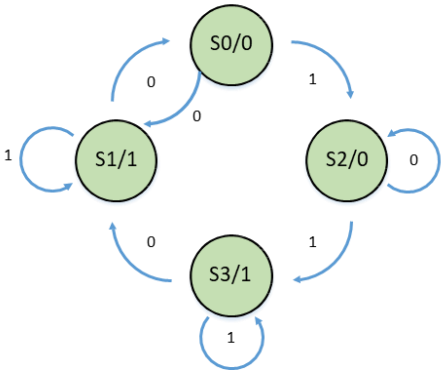


Fig.1 File hierarchy for Homework submission

Lab in class: Design a circuit "Moore machine"

1) **Design a Moore machine circuit that can be synthesized. The following is Moore machine module's specification. (Do NOT add or delete I/O ports, but you can change their behavior.)**

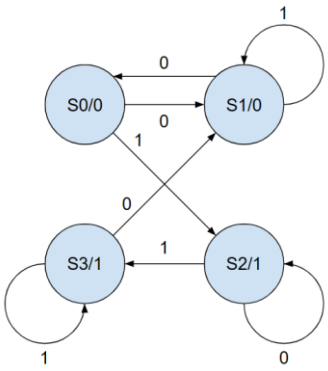


Current State	Next State		qout
	din=0	din=1	
S0=00	S1	S2	0
S1=01	S0	S1	0
S2=10	S2	S3	1
S3=11	S1	S3	1

Signal	Type	Bits	Description
clk	input	1	Clock pin.
rst	input	1	Reset pin. Reset all of the flip flops to zeros.
din	input	1	Control signal for fsm.
qout	output	1	1:when current state==S1 or current state==S3 0: when current state==S0 or current state==S2

2) **Please describe your FSM in detail**

Explanation about your FSM



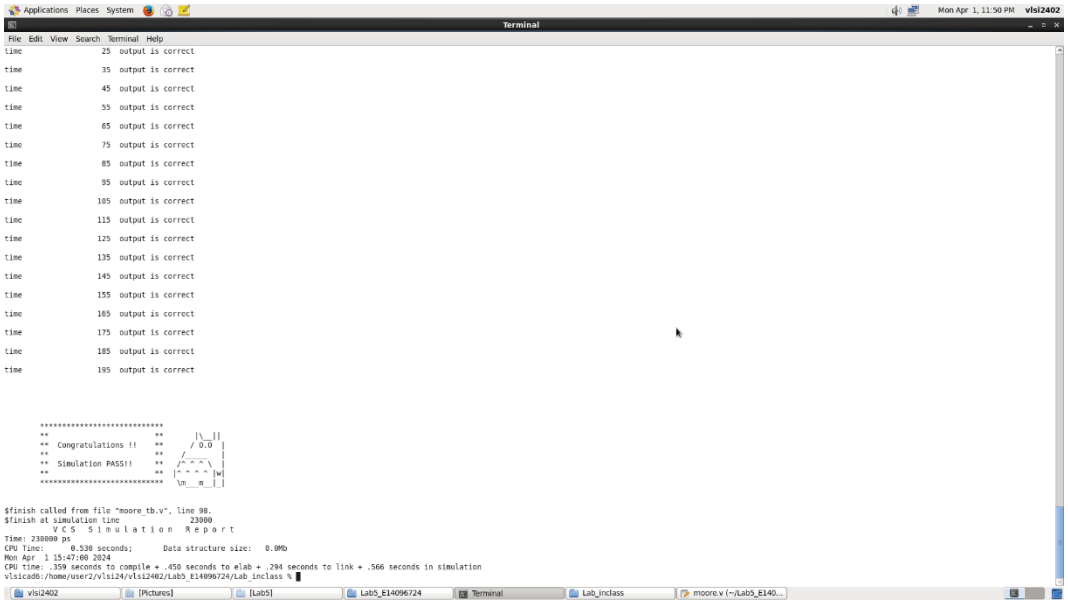
state	next_state		output
	din = 0	din = 1	
S0	S1	S2	q = 0
S1	S0	S1	q = 0
S2	S2	S3	q = 1
S3	S1	S3	q = 1

3) **After synthesizing your design, you may have some information about the circuit. Please fill in the following form.**

Timing (slack)	Area (total cell area)	Power (total)
0.32	4.147200	6.7439e-03mW

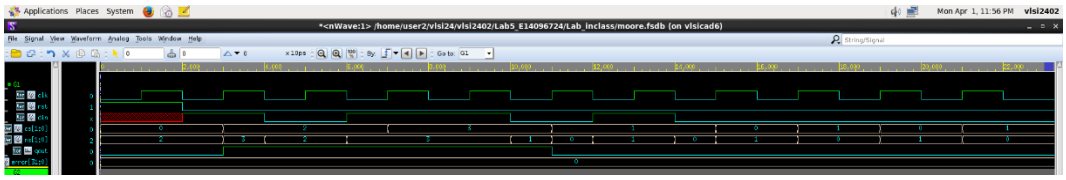
4) Please attach your design waveforms.

Your simulation result on the terminal.

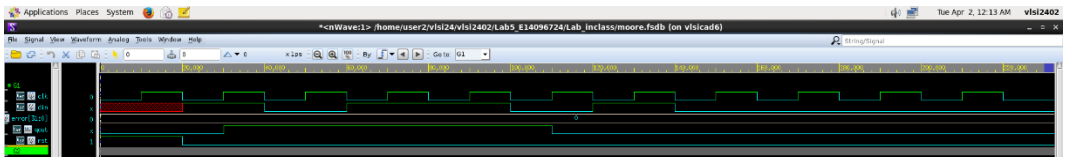


Your waveform :

RTL:



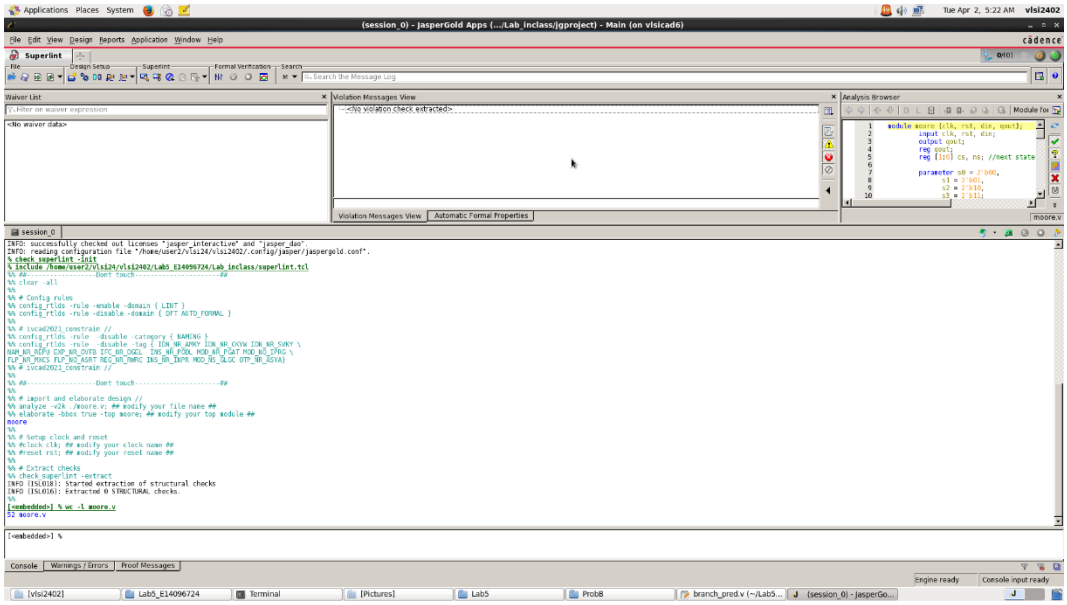
Synthesis:



Explanation of your waveform :

波形圖中 error 一直都為 0，代表全部的測資皆正確，波行中還可以看到這是一個 moore machine，state 隨著輸出改變，而 output 只隨著 state 改變，可以看到 state 和 output 的結果皆與 state diagram 和 state table 相符。

Superlint Coverage



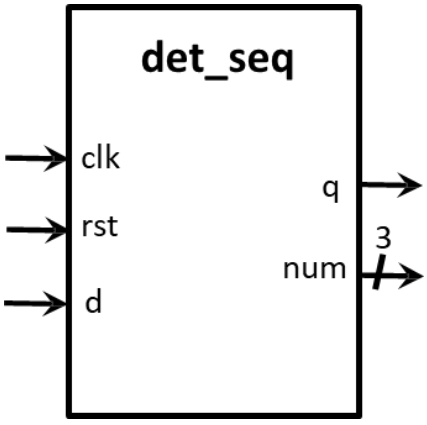
Coverage :

moore.v : (1 - 0/52) * 100% = 100%

ProbA: Design a circuit "detecting pattern 101011"

---

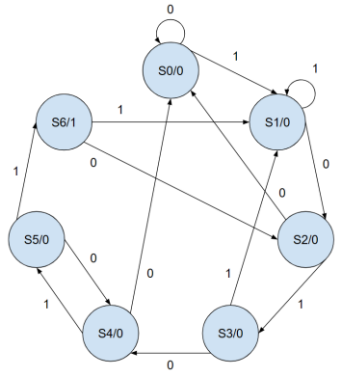
1) **Design a pattern seq-detecting circuit that can be synthesized with moore machine. The following is det_seq module's specification. (Do NOT add or delete I/O ports, but you can change their behavior.)**



Signal	Type	Bits	Description
clk	input	1	clock
rst	input	1	reset, active high
d	input	1	pattern bit
q	output	1	When detect pattern 101011, q pulls to high at next clock posedge. Otherwise, q is low.
num	output	3	Count the number of pattern 101011

2) Please describe your FSM in detail

Explanation about your FSM



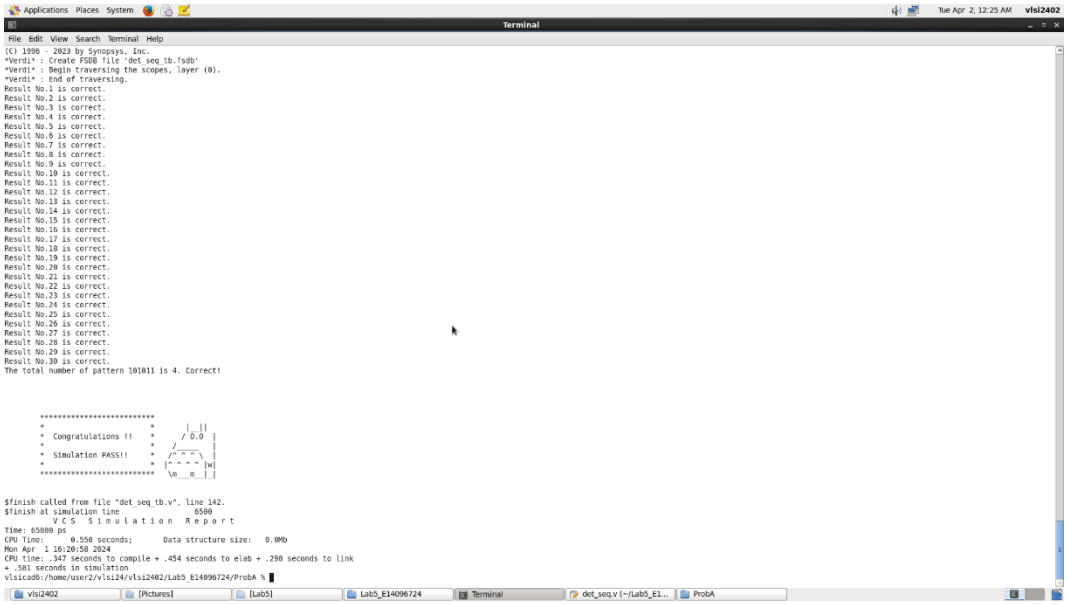
state	next_state		output	state information
	d = 0	d = 1		
S0	S0	S1	q = 0	initial
S1	S2	S1	q = 0	receive 1
S2	S0	S3	q = 0	receive 10
S3	S4	S1	q = 0	receive 101
S4	S0	S5	q = 0	receive 1010
S5	S4	S6	q = 0	receive 10101
S6	S2	S1	q = 1	receive 101011

3) After synthesizing your design, you may have some information about the circuit. Please fill in the following form.

Timing (slack)	Area (total cell area)	Power (total)
0.30	**12.286080**	**1.4213e-02mW**

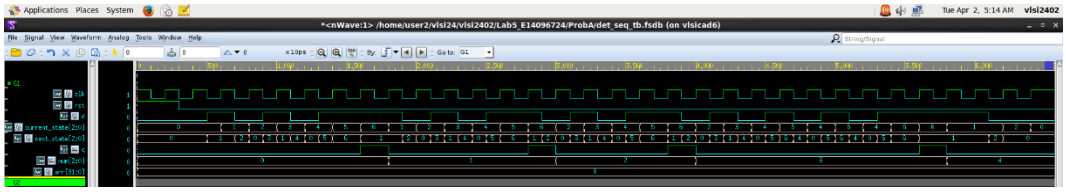
4) Please attach your design waveforms.

Your simulation result on the terminal.

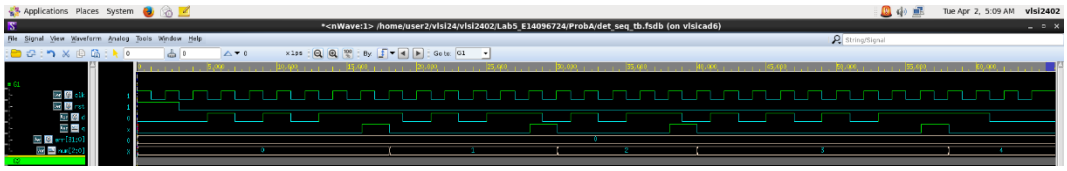


Your waveform :

RTL:



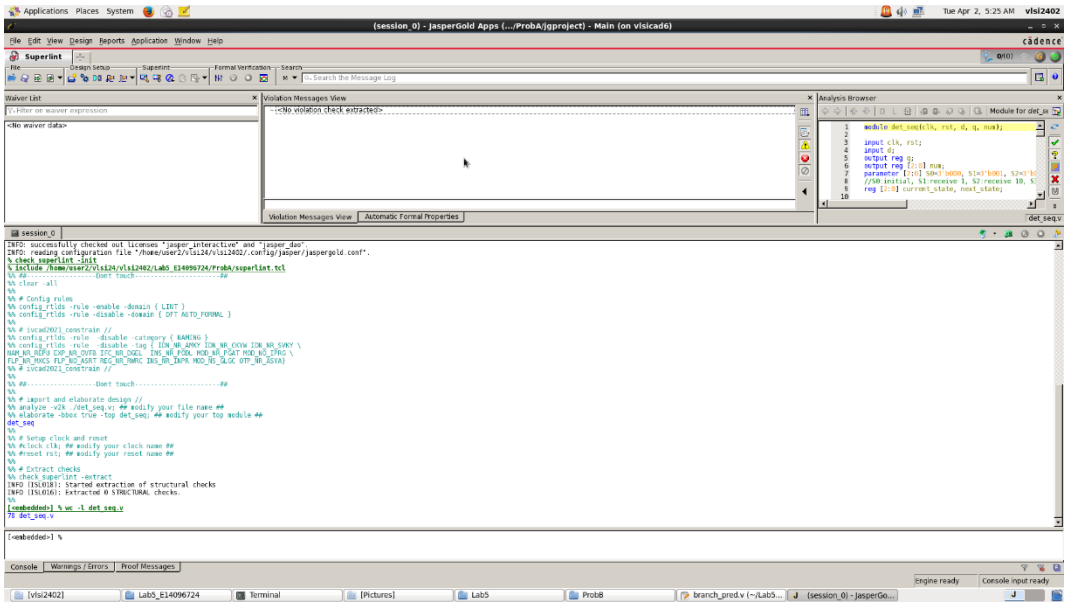
Synthesis



Explanation of your waveform :

波形圖中 error 一直都為 0，代表全部的測資皆正確，波行中還可以看到這是一個 moore machine，state 隨著輸出改變，而 output 只隨著 state 改變，可以看到 state 和 output 的結果皆與 state diagram 和 state table 相符。

Superlint Coverage

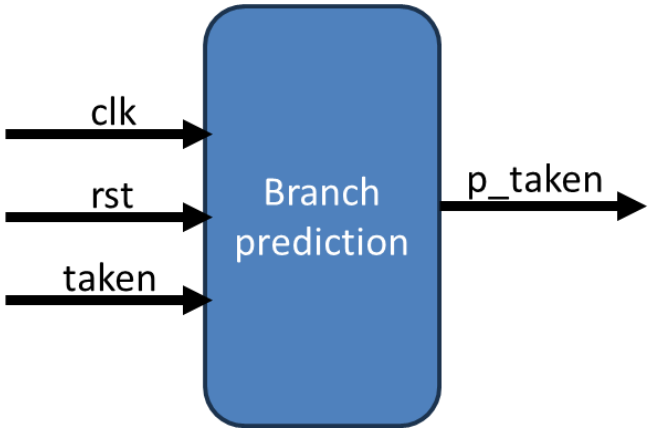


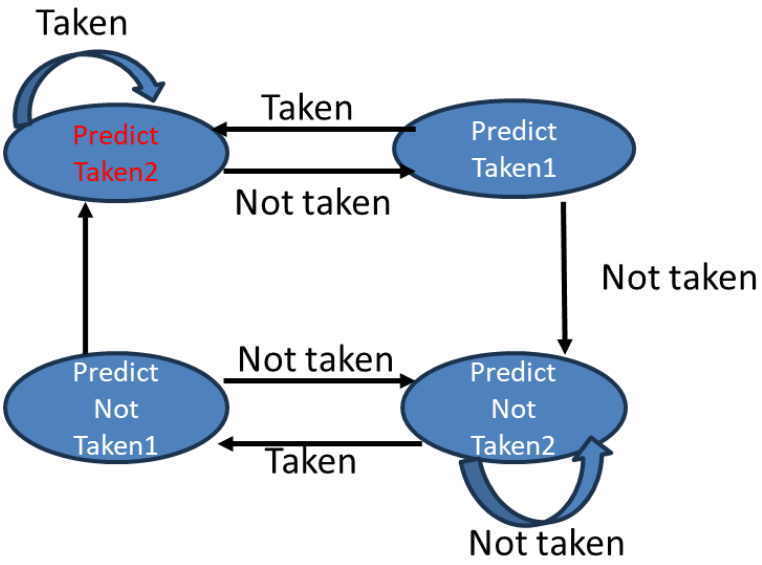
Coverage :

Det_seq.v : (1 - 0/70) * 100% = 100%

ProbB: Design a 2-bit branch prediction

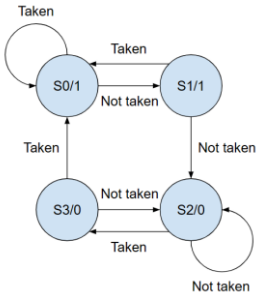
1) **Design a 2-bit branch prediction with moore machine. The following is 2-bit branch prediction module's specification. (Do NOT add or delete any I/O ports, but you can change their behavior.)**





2) Please describe your FSM in detail.

Explanation about your FSM



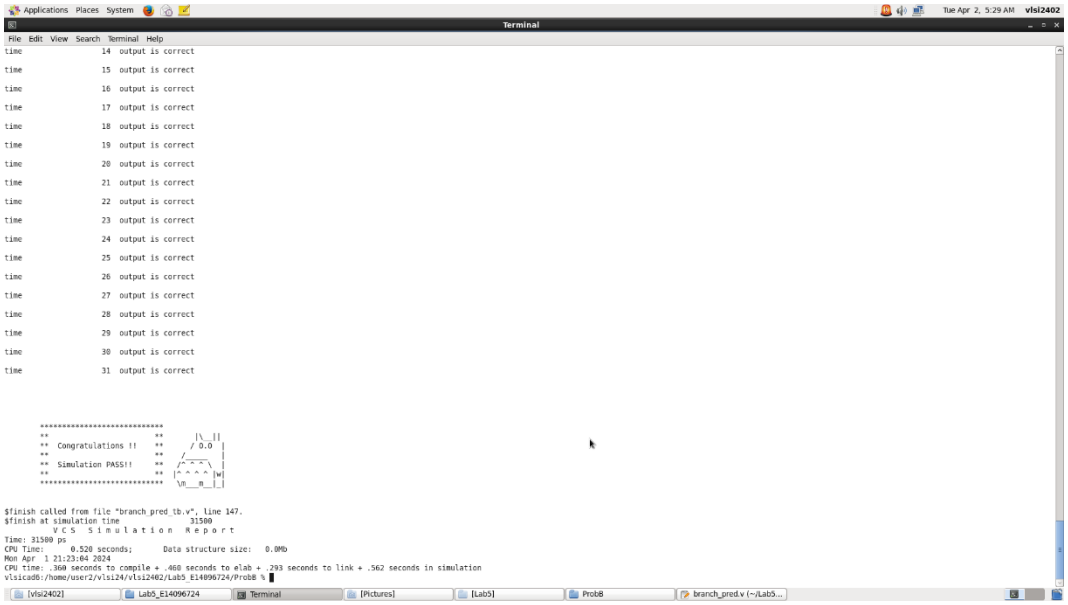
state	next_state		output	state information
	taken = 0	taken = 1		
S0	S1	S0	p_taken = 1	Predict Taken 2
S1	S2	S0	p_taken = 1	Predict Taken 1
S2	S2	S3	p_taken = 0	Predict Not Taken 2
S3	S2	S0	p_taken = 0	Predict Not Taken 1

3) After synthesizing your design, you may have some information about the circuit. Please fill in the following form.

Timing (slack)	Area (total cell area)	Power (total)
0.33	**3.784320**	**5.6621e-03mW**

4) Please attach your design waveforms.

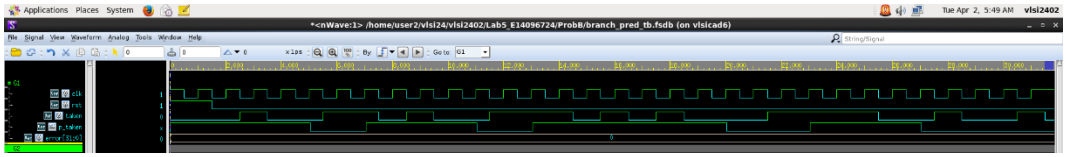
Your simulation result on the terminal.



Your waveform :

RTL:

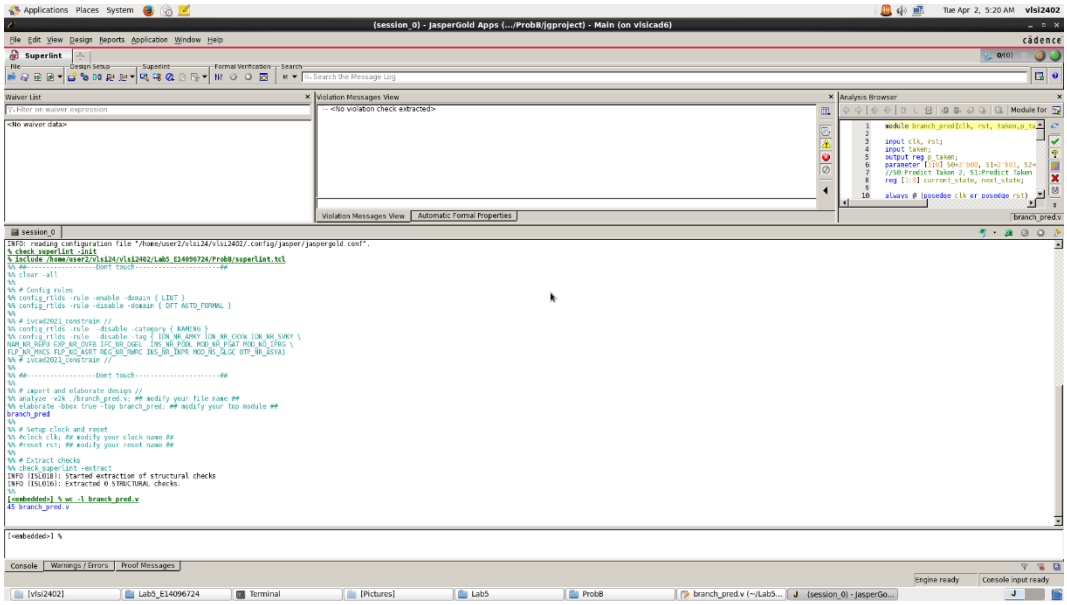
Synthesis



Explanation of your waveform :

波形圖中 error 一直都為 0，代表全部的測資皆正確，波行中還可以看到這是一個 moore machine，state 隨著輸出改變，而 output 只隨著 state 改變，可以看到 state 和 output 的結果皆與 state diagram 和 state table 相符。

Superlint Coverage

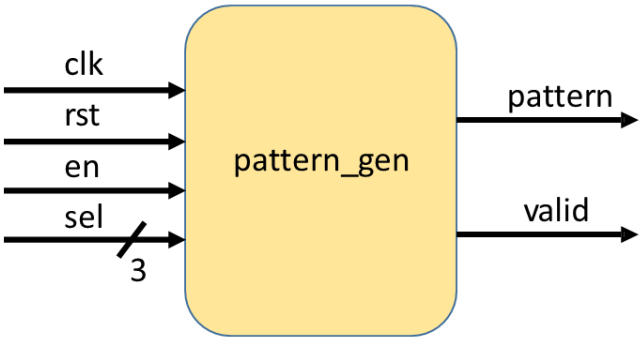


Coverage :

branch_pred.v : (1 - 0/45) * 100% = 100%

ProbC: Design a pattern generator

1) **Design a pattern generator which can create the following pattern and use mealy machine. The following is pattern generator specification.**

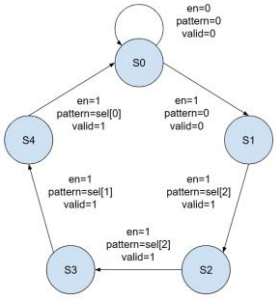


sel [2:0]	pattern
000	0000
001	0001
010	0010
011	0011
100	1100
101	1101
110	1110
111	1111

Signal	Bits	Type	Description
clk	1	input	clock
rst	1	input	reset, active high
en	1	input	When en is high, the system will start to make pattern. The pattern will be created once. If the host want to create the next pattern, it should pull down the en to 0,then restart en.
sel	3	input	According different sel signal to make different pattern
pattern	1	output	Pattern output
valid	1	output	When valid is 1, pattern's value is valid.

2) Please describe your FSM in detail.

Explanation about your FSM



state	next_state		output		state information
	en = 0	en = 1	pattern	valid	
S0	S0	S1	0	0	idle
S1	X	S2	sel[2]	1	pattern
S2	X	S3	sel[2]	1	pattern
S3	X	S4	sel[1]	1	pattern
S4	X	S0	sel[0]	1	pattern

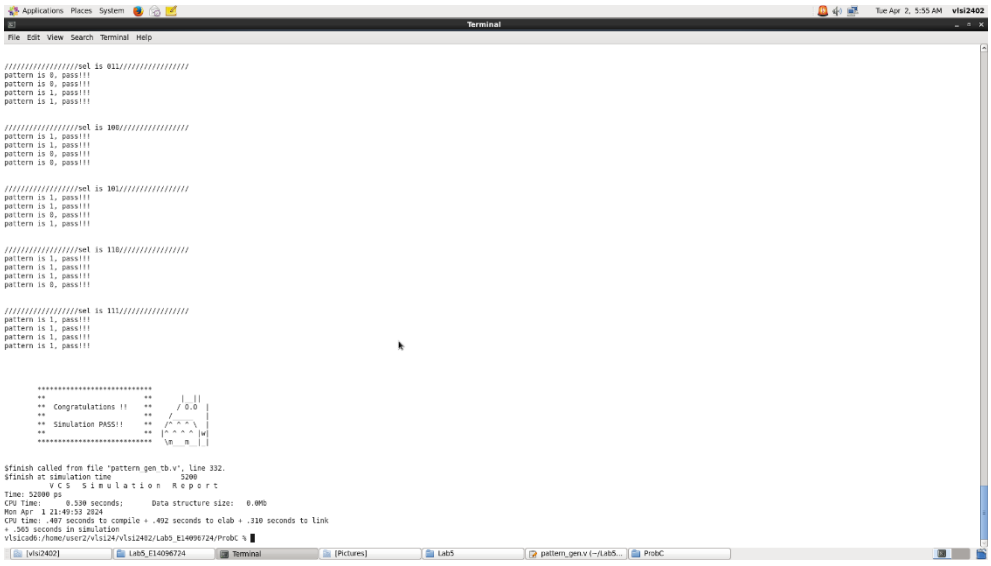
3) After synthesizing your design, you may have some information about the circuit. Please fill in the following form.

Timing (slack)	Area (total cell area)	Power (total)
0.91	**6.428160**	**9.2515e-03mW**

單純使用 Clock Specification 為 1ns，未使用 DC.sdc

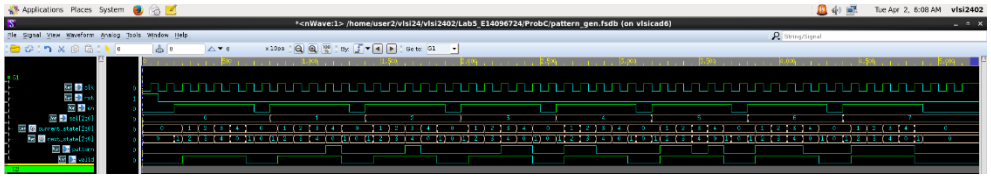
4) Please attach your design waveforms.

Your simulation result on the terminal.

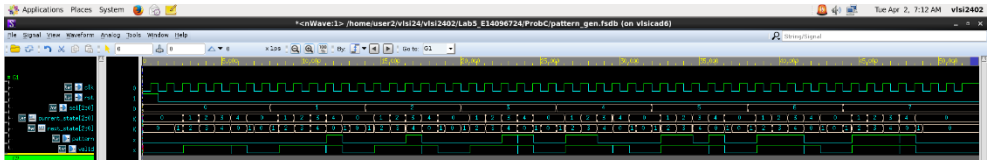


Your waveform :

RTL:



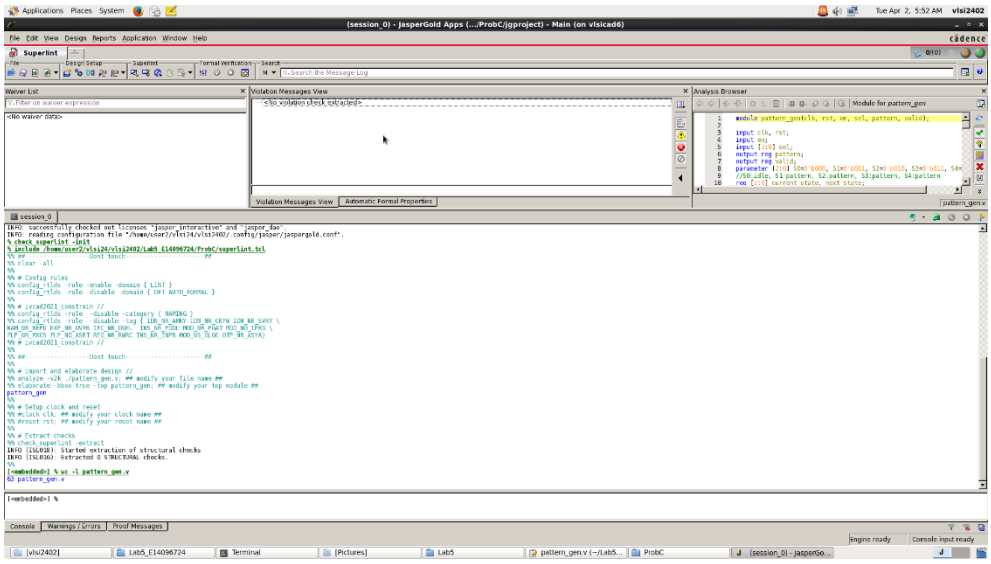
Synthesis



Explanation of your waveform :

波形圖中 error 一直都為 0，代表全部的測資皆正確，波行中還可以看到這是一個 mealy machine，state 隨著輸出改變，而 output 隨著 state 和 input 改變，可以看到 state 和 output 的結果皆與 state diagram 和 state table 相符。

Superlint Coverage



Coverage :

pattern_gen.v : (1 - 0/63) * 100% = 100%

5) **At last, please write the lessons learned from this lab session, or some suggestions for this lab session. Thank you.**

這次的實驗讓我學習到了如何實作一個 moore machine 和 mealy machine，moore machine 和 mealy machine 在我之前都有在邏輯系統課程中學過，在實作過後我更加了解了 moore machine 和 mealy machine 的輸入和輸出在波型上的差別，還有更加了解了 moore machine 和 mealy machine 在 verilog 中的細節，在用完 superlint 後把每個 warning 都看過並修正，這讓我學習了很多也更熟悉如何寫 verilog。

Appendix A : Commands we will use to check your homework

Problem		Command
Lab	Compile	% vcs -R moore.v -full64
	RTL-sim	% vcs -R moore_tb.v -debug_access+all -full64 +define+FSDB
	Gate-sim	% vcs -R moore_tb.v -debug_access+all -full64 +define+FSDB+syn

Problem		Command
ProbA	Compile	% vcs -R det_seq.v -full64
	RTL-sim	% vcs -R det_seq_tb.v -debug_access+all -full64 +define+FSDB
	Gate-sim	% vcs -R det_seq_tb.v -debug_access+all -full64 +define+FSDB+syn
ProbB	Compile	% vcs -R branch_pred.v -full64
	RTL-sim	% vcs -R branch_pred_tb.v -debug_access+all -full64 +define+FSDB
	Gate-sim	% vcs -R branch_pred_tb.v -debug_access+all -full64 +define+FSDB+syn
ProbC	Compile	% vcs -R pattern_gen.v -full64
	RTL-sim	% vcs -R pattern_gen_tb.v -debug_access+all -full64 +define+FSDB
	Gate-sim	% vcs -R pattern_gen_tb.v -debug_access+all -full64 +define+FSDB+syn