105061254 林士平 邏輯設計實驗報告 Lab9

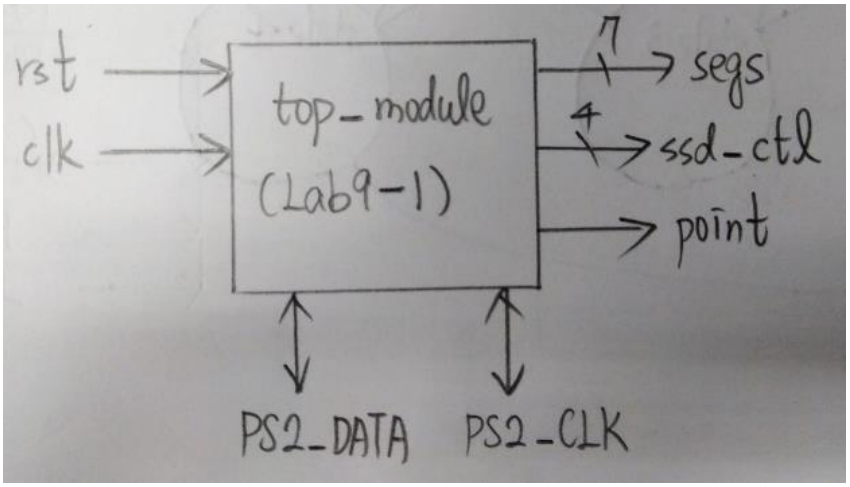
1.

(1) Design specification :

A. Inputs and outputs(表一)：

Inputs	rst, clk
Outputs	[6:0]segs, [3:0]ssd_ctl, point
Inouts	PS2_DATA, PS2_CLK
↑表一：Inputs and outputs of 1	

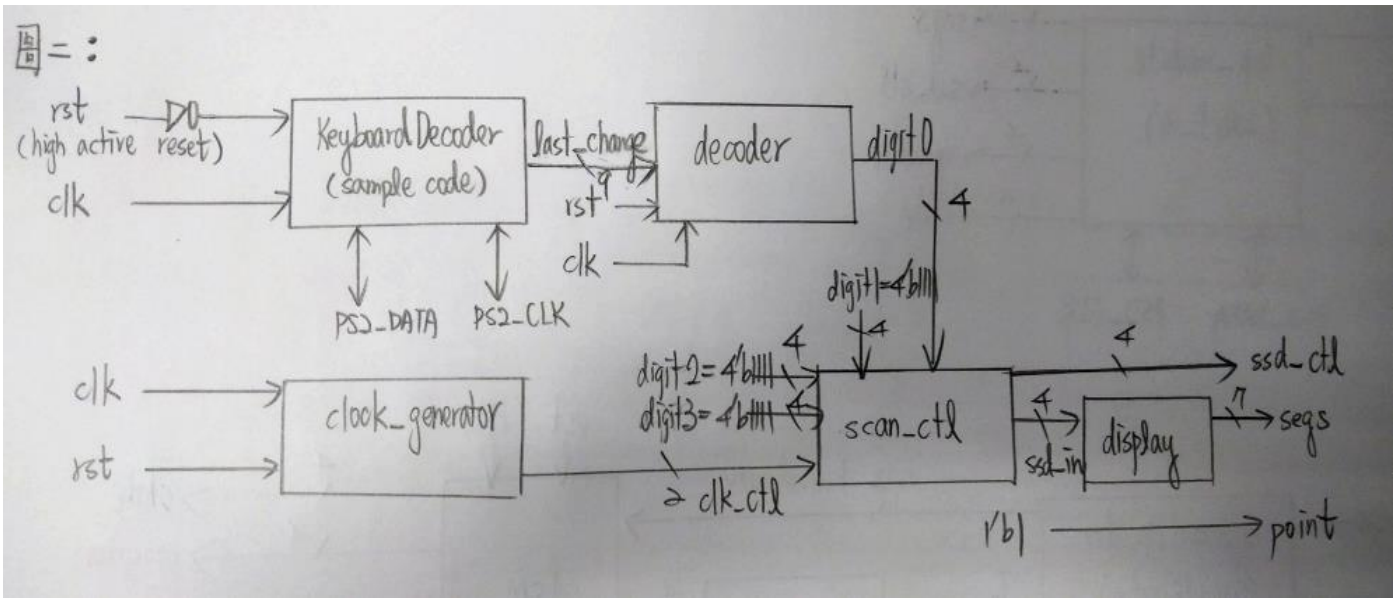
B. Block diagram(function table)(圖一)：



↑圖一：The block diagram of 1

(2) Design implementation :

A. Logic diagram(function table)(圖二)：



↑圖二：logic diagram of 1

B. I/O pin assignment(表二)：

I/O	clk	rst	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]	segs[6]
LOC	W5	V17	W4	V4	U4	U2	W7
I/O	segs[5]	segs[4]	segs[3]	segs[2]	segs[1]	segs[0]	point
LOC	W6	U8	V8	U5	V5	U7	V7
I/O	PS2_CLK	PS2_DATA					
LOC	C17	B17					

↑表二：I/O pin assignment of 1

C.功能與做法說明：

本題內容為當按下鍵盤上 0/1/2/3/4/5/6/7/8/9 按鍵時(我用右邊九宮格的數字鍵)，以及鍵盤上的 a/s/m 按鍵時，七段顯示器會顯示相對應的數字及符號，當按下 enter 鍵時，七段顯示器會清空。

本題最重要的模組為 decoder，輸入為來自 KeyboardDecoder 模組的 last_change。last_change 代表最後一次按下的按鍵，由 last_change 來判斷現在七段顯示器應該顯示的數值：當 last_change = 0 ~ 9 按鍵時，七段顯示器顯示 0~9；當 last_change = a/s/m 按鍵時，七段顯示器分別顯示 A/-/X(my own defined A/S/M pattern)；當按下 enter 鍵後，七段顯示器會清空；當按下上述按鍵以外的按鍵時，七段顯示器不會改變。

最後將 decoder decode 出來的值送到顯示模組顯示。

2

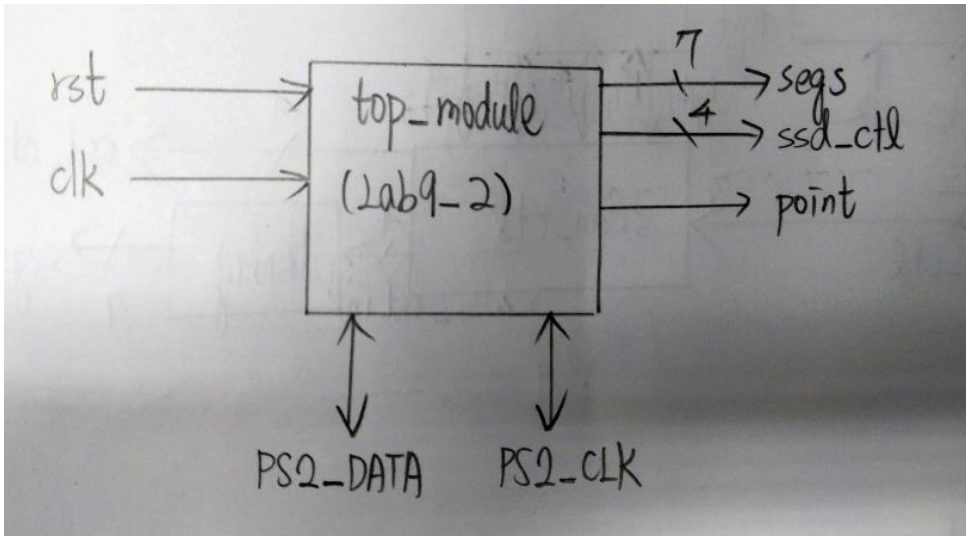
(1) Design specification :

A. Inputs and outputs(表三)：

Inputs	rst, clk
Outputs	[6:0]segs, [3:0]ssd_ctl, point
Inouts	PS2_DATA, PS2_CLK

↑表三：Inputs and outputs of 2

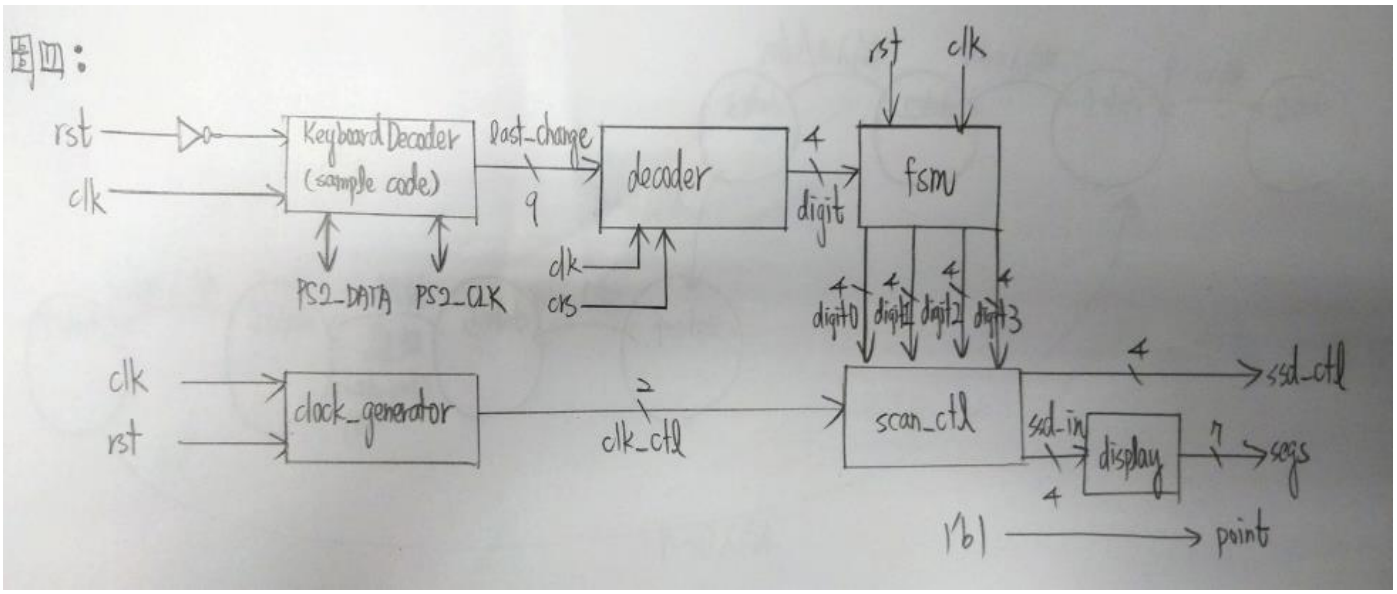
B. Block diagram(function table)(圖三)：



↑圖三：The block diagram of 2

(2) Design implementation :

A. Logic diagram(function table)(圖四)：



↑圖四：logic diagram of 2

B. I/O pin assignment(表四)：

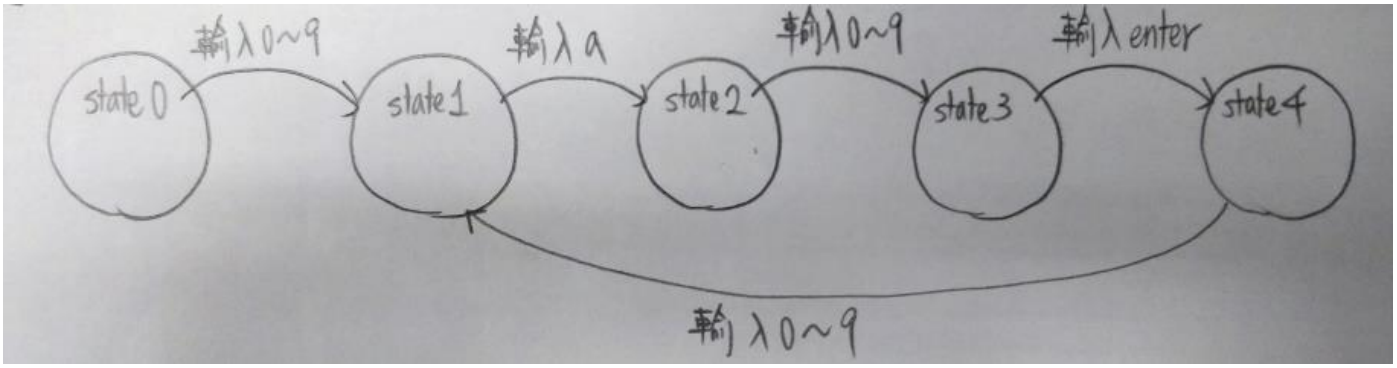
I/O	clk	rst	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]	segs[6]
LOC	W5	V17	W4	V4	U4	U2	W7
I/O	segs[5]	segs[4]	segs[3]	segs[2]	segs[1]	segs[0]	point
LOC	W6	U8	V8	U5	V5	U7	V7
I/O	PS2_CLK	PS2_DATA					
LOC	C17	B17					

↑表四：I/O pin assignment of 2

C.功能與做法說明：

本題為製作一個 single digit 的 decimal adder，使用方法如下：當輸入第一個數字後，七段顯示器會顯示第一個數字在最左側，接著按 a(表示+)，便可輸入第二個數字；當輸入第二個數字後，七段顯示器會顯示第二個數字在左邊數來第二個，接著按 enter，便會顯示結果在七段顯示器右側兩位。如果想要重新開始只要再輸入數字即可。

本題最重要的模組為 decoder 和 fsm，decoder 和第一題相同，將由 KeyboardDecoder 模組得到的 last_change 訊號做 decode，得到相對應的 output，將其輸入 fsm 作為控制訊號。fsm 模組為實現計算機的功能，總共有 5 個 state，state transition diagram 如下頁圖五所示，在 state0 七段顯示器不顯示；在 state1 七段顯示器顯示輸入的第一個數字；在 state2 七段顯示器維持顯示輸入的第一個數字；在 state3 七段顯示器顯示輸入的第二個數字與第一個數字；最後在 state4 七段顯示器顯示輸入的第一個數字和第二個數字，以及計算結果。



↑圖五：state transition diagram of 2

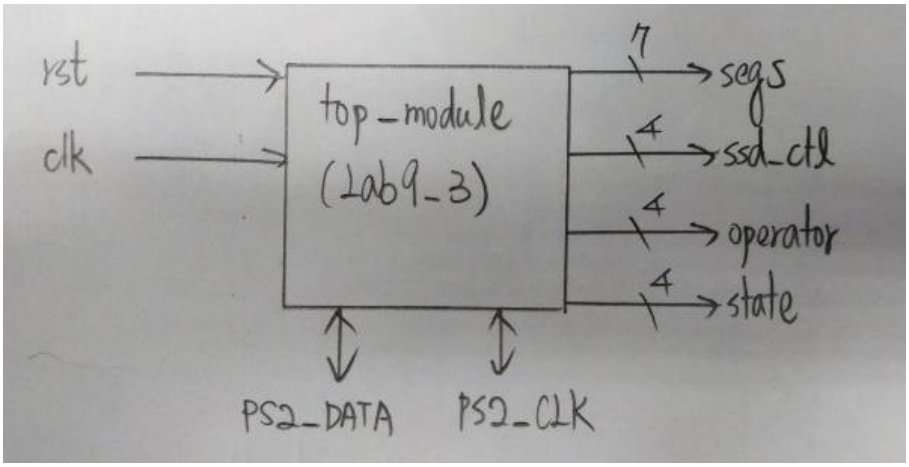
3

(1) Design specification :

A. Inputs and outputs(表五)：

Inputs	rst, clk
Outputs	[6:0]segs, [3:0]ssd_ctl, point, [3:0]operator, [3:0]state
Inouts	PS2_DATA, PS2_CLK
↑表五：Inputs and outputs of 3	

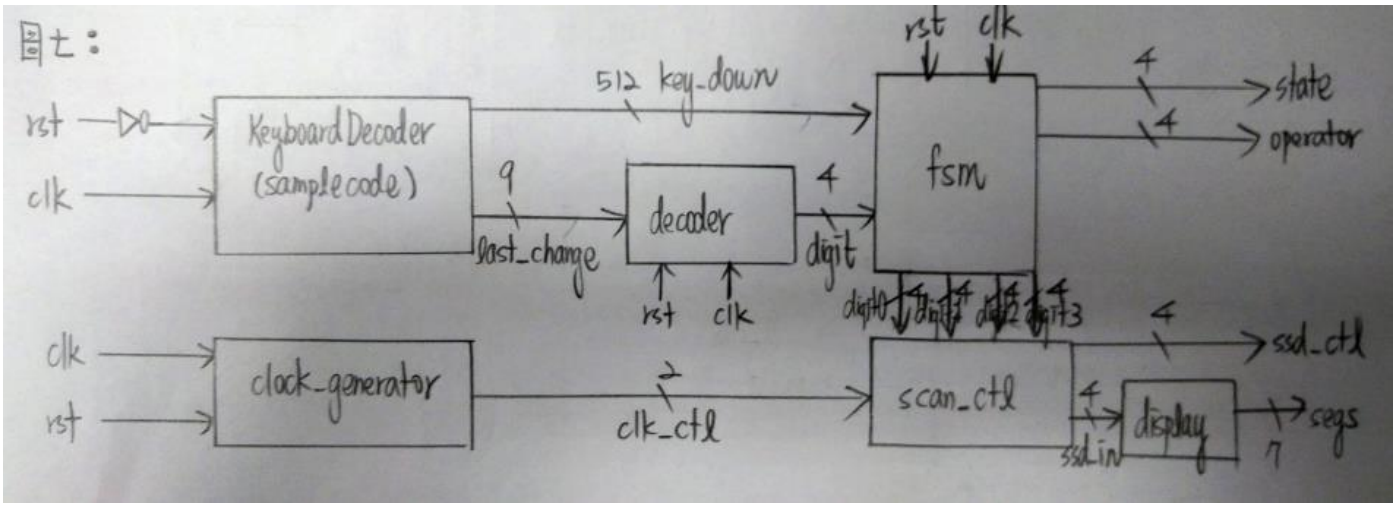
B. Block diagram(function table)(圖六)：



↑圖六：The block diagram of 3

(2) Design implementation :

A. Logic diagram(function table)(圖七)：



↑圖七：logic diagram of 3

B. I/O pin assignment(表六)：

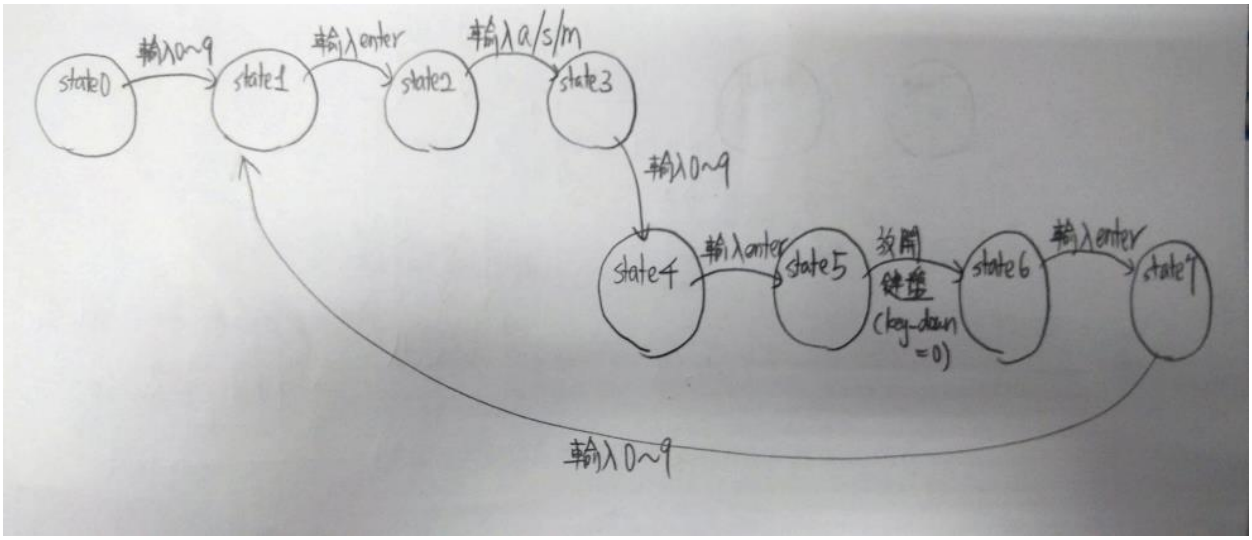
I/O	clk	rst	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]	segs[6]
LOC	W5	V17	W4	V4	U4	U2	W7
I/O	segs[5]	segs[4]	segs[3]	segs[2]	segs[1]	segs[0]	point
LOC	W6	U8	V8	U5	V5	U7	V7
I/O	PS2_CLK	PS2_DATA	operator[3]	operator[2]	operator[1]	operator[0]	state[3]
LOC	C17	B17	L1	P1	N3	P3	V19
I/O	state[2]	state[1]	state[0]				
LOC	U19	E19	U16				

↑表六：I/O pin assignment of 3

C.功能與做法說明：

本題為製作一個支援加/減/乘功能的兩位數計算機。使用說明如下：當輸入第一個數字的十位後，七段顯示器會顯示第一個數字的十位在最左側，接著按 enter，便可輸入第一個數字的個位；當輸入第一個數字的個位後，七段顯示器會顯示第一個數字的個位在左邊數來第二個，接著按下想要做的運算(加 -> a；減 -> s；乘 -> m)，便可輸入第二個數字的十位；當輸入第二個數字的十位後，七段顯示器會顯示第二數字的十位在最左邊數來第三個，接著按 enter，便可輸入第二個數字的個位；當輸入第二個數字的個位後，七段顯示器便會顯示第二個數字的個位在最左邊數來第四個。最後按下 enter，七段顯示器便會顯示最後的結果。

本題最重要的模組為 decoder 和 fsm，decoder 和第一題相同，將由 KeyboardDecoder 模組得到的 last_change 訊號做 decode，得到相對應的 output，將其輸入 fsm 作為控制訊號。fsm 模組為實現計算機的功能，總共有 8 個 state，state transition diagram 如下頁圖八所示，和第二題最不同的一點是為了要能夠區分出按兩次 enter 與按一次 enter，中間必須加一個 buffer state，來判斷是否有放開鍵盤(利用 key_down 做判斷)。



↑圖八：state transition diagram of 3

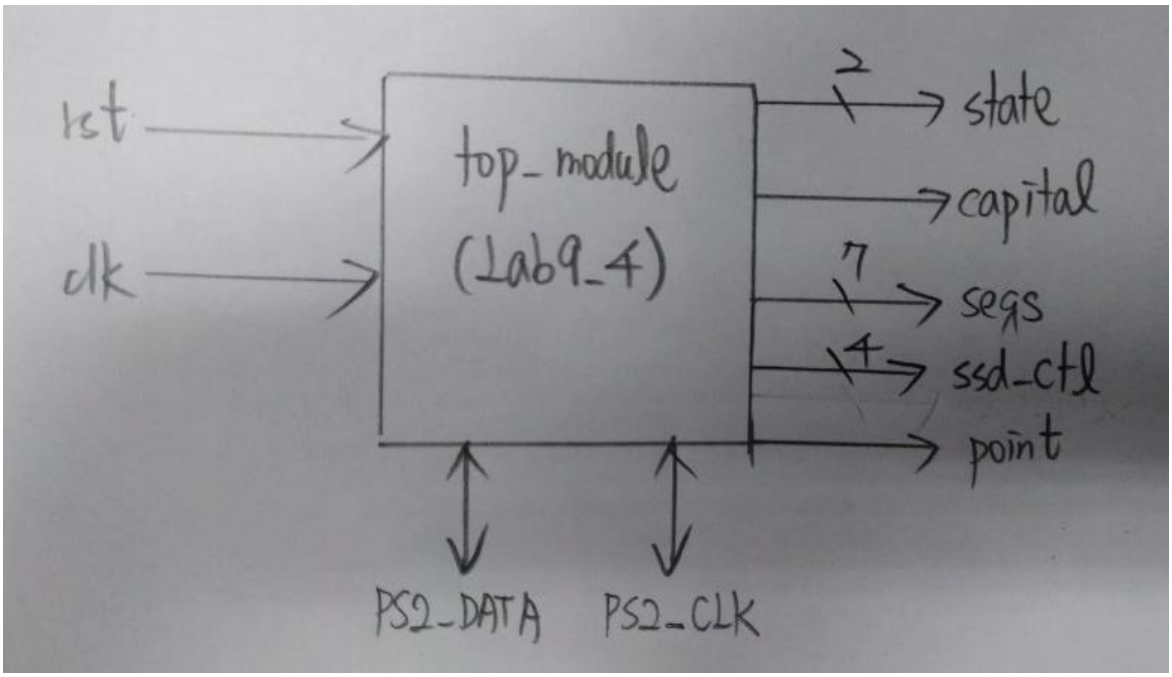
4.

(1) Design specification :

A. Inputs and outputs(表七)：

Inputs	rst, clk
Outputs	[6:0]segs, [3:0]ssd_ctl, point, [1:0]state, capital
Inouts	PS2_DATA, PS2_CLK
↑表七：Inputs and outputs of 4	

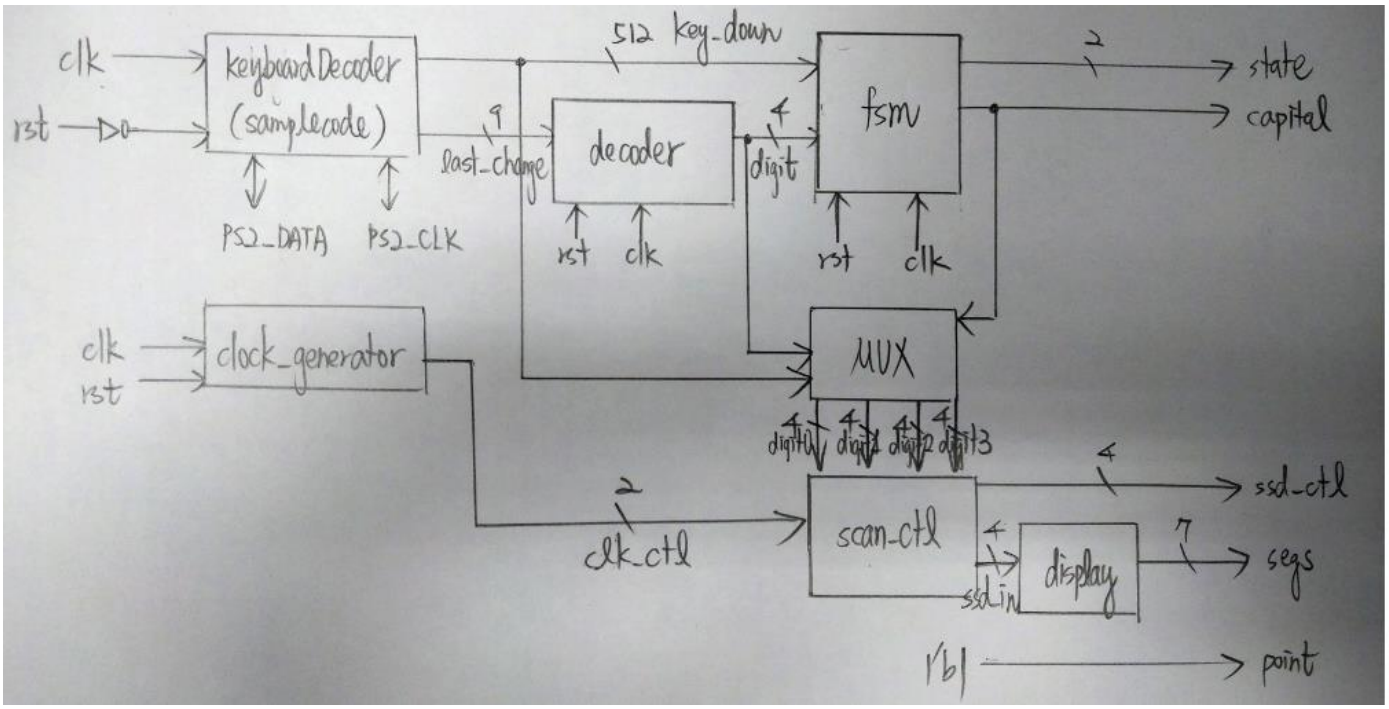
B. Block diagram(function table)(圖九)：



↑圖九：The block diagram of 4

(2) Design implementation :

A. Logic diagram(function table)(圖十)：



↑圖十：logic diagram of 4

B. I/O pin assignment(表八)：

I/O	clk	rst	ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]	segs[6]
LOC	W5	V17	W4	V4	U4	U2	W7
I/O	segs[5]	segs[4]	segs[3]	segs[2]	segs[1]	segs[0]	point
LOC	W6	U8	V8	U5	V5	U7	V7
I/O	PS2_CLK	PS2_DATA	state[1]	state[0]	capital		
LOC	C17	B17	E19	U16	L1		
↑表八：I/O pin assignment of 4							

C.功能與做法說明：

本題為實現複合按鍵功能：當 caps lock 沒有亮時(小寫)，為小寫模式，在此模式下輸入英文字母七段顯示器會顯示對應的小寫英文字母 ASCII code；如果同時按 shift +英文字母，七段顯示器則是會顯示對應的大寫英文字母 ASCII code。相反的，當 caps lock 亮時(大寫)，為大寫模式，在此模式下輸入英文字母七段顯示器會顯示對應的大寫英文字母 ASCII code；如果同時按 shift +英文字母，七段顯示器則是會顯示對應的小寫英文字母 ASCII code。

本題的重點在 fsm 和 MUX，decoder 和前面相同就不再重述。fsm 總共有四個：小寫 state、大寫 state 和兩個 buffer state，其中：小寫 state 的 capital = 1’b0、大寫 state 的 capital = 1’b1，buffer state 則是用來區分是按了 caps lock 兩次或一次(判斷是否有放開鍵盤)。MUX 則是輸入 capital、digit 和 key_down 來判斷是在 capital = 1’b0 或 capital = 1’b1 的情況下，以及是否有按 shift，由此來決定七段顯示器應該顯示的值。

5. Discussion

本次的 Lab 重點為學會如何使用由 sample code 得到的訊號(last_change, key_valid 和 key_down)，來達到我們想要的功能。最常使用的工具是 finite state machine。

這次 Lab 我遇到的困難是無法分辨同個按鍵是按兩下還是只按一下(因為兩種狀況下 last_change 是相同的)，後來才想到可以利用 key_down 來判斷是不是有放開鍵盤，以此來判斷是按了兩下還是一下。

6. Conclusion

本次的 Lab 我學會了另一個 FPGA 板的外接功能：利用 keyboard 輸入。為我的期末專題增加了一些可以用的元素。