Lab9.1 When Press 0/1/2/3/4/5/6/7/8/9/a/s/m show them in the seven-segment display. When a new number is pressed, the previous number is refreshed and over written. When press enter the seven segment turns off.

1.Design specification:

功能: 鍵盤數字 / 字母顯示器

輸入: PS2_DATA (鍵盤訊號輸入)

PS2_CLK (鍵盤訊號輸入)

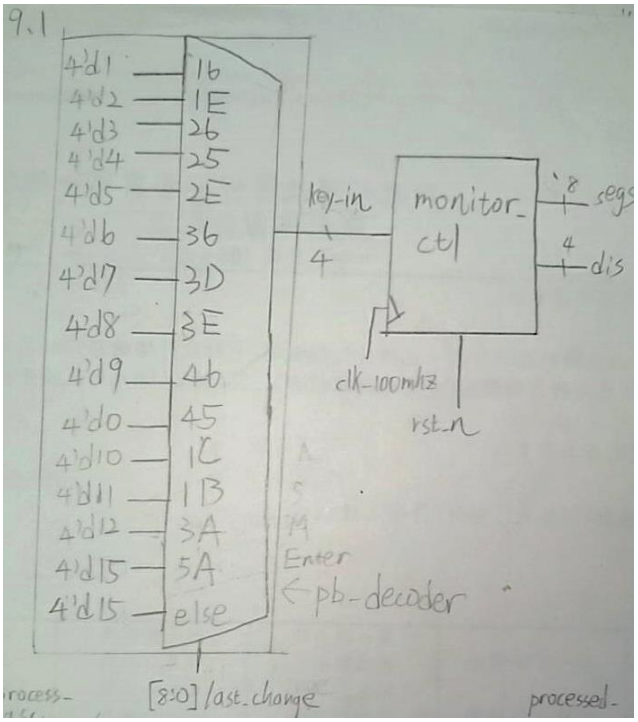
clk_100mhz (石英震盪器輸入)

rst_n (重置全部模組)

輸出: [3:0]dis (控制每個七段顯示器亮暗)

[7:0]segs (控制七段顯示器中每條燈管亮暗)

2.Design implementation:



由於在 keyboard_decoder 中的 last_change 已經幫我們處理並讀取出最後一個被改變的鍵位位置，所以我們只需要將其位置解碼，轉移成鍵盤上的符號，再用 monitor_ctl 處理顯示出即可。

在設計中，用來解碼的是 pb_decoder，pb_decoder 本質是一個 decoder，如果 last_change 輸入題目規定的符號按鈕位置，那麼 decoder 會輸入其對應的數字。如輸入的 last_change(按鍵盤位置 0)為 9'h45，那麼 decoder 會輸出零到顯示模組。順便一提，按 a 的按鍵時會輸出 10，s 輸出 11，m 輸出 12，enter 及其他數字都是十五。在此模組中，a 被定義為十，顯示模組讀到十時會顯示 a，其他以此類推，enter 則是燈管都不亮。

3. I/O pin assignment

I/O 變數	對應腳位	補充說明
輸入		
PS2_DATA	B17	鍵盤訊號輸入
PS2_CLK	C17	鍵盤訊號輸入
clk_100mhz	W5	石英震盪器輸入
rst_n	V16	重置全部模組
輸出		
seg[7]	W7	控制七段顯示器腳位
seg[6]	W6	
seg[5]	U8	
seg[4]	V8	
seg[3]	U5	
seg[2]	V5	
seg[1]	U7	
seg[0]	V7	
dis[3]	W4	控制每個七段顯示器開關
dis[2]	V4	
dis[1]	U4	
dis[0]	U2	

Lab9.2 Implement a single digit decimal adder using the key board as the input and display the results on the 14-segment display (The first two digit are the addend / augend, and the last two digits are the sum).

1.Design specification:

功能: 一位元加法器

輸入: PS2_DATA (鍵盤訊號輸入)

PS2_CLK (鍵盤訊號輸入)

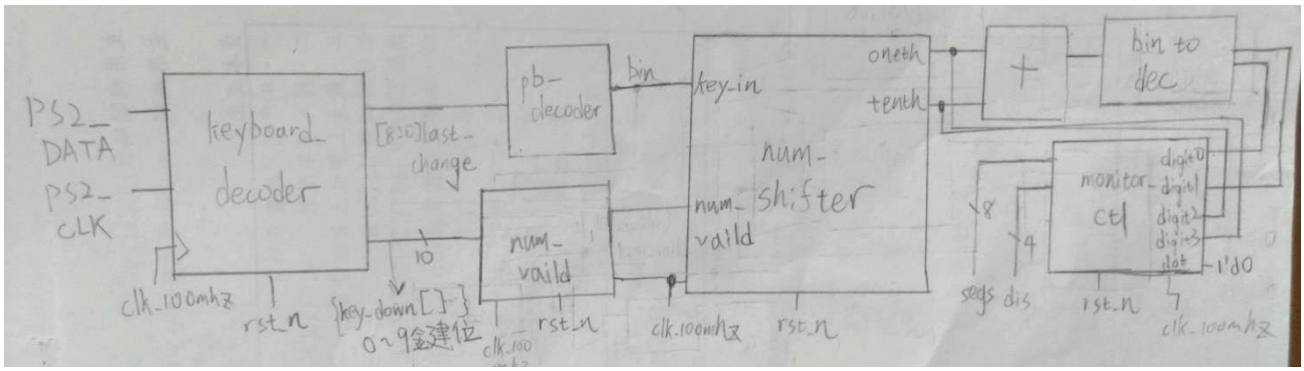
clk_100mhz (石英震盪器輸入)

rst_n (重置全部模組)

輸出: [3:0]dis (控制每個七段顯示器亮暗)

[7:0]segs (控制七段顯示器中每條燈管亮暗)

2.Design implementation



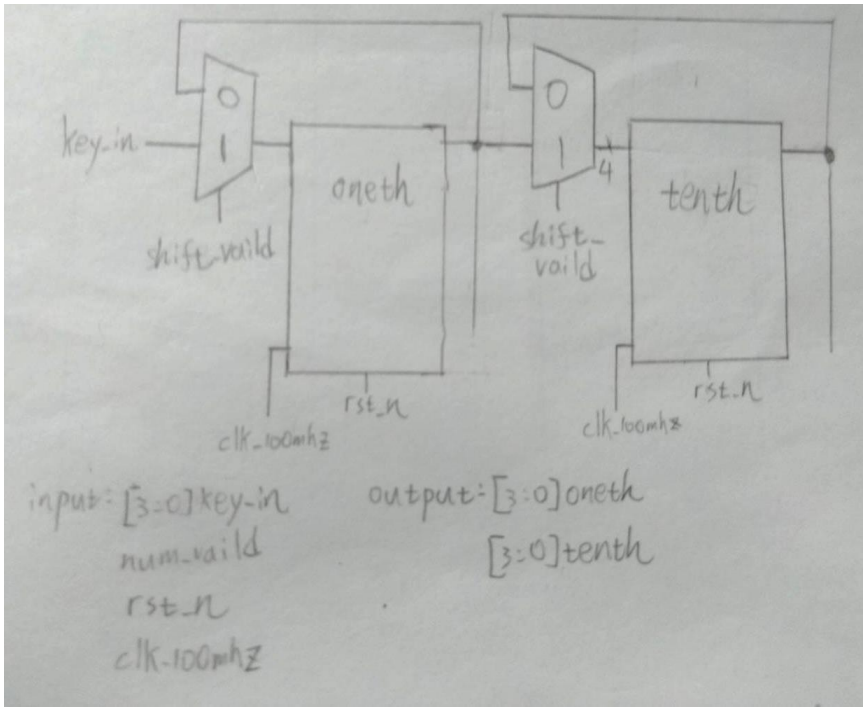
在這題，我們要將輸入的最後兩個數字做總和，首先將鍵入的按鈕位置轉成二進位數字，這由上題pb_decoder來負責，之後儲存鍵入的最後兩個數字，這兩個數字用adder相加後，後續由bin_to_dec轉成BCD，最後經monitor_ctl處理，由左至右分別顯示最後一個輸入的數、最後第二個輸入的數、兩個數之和。處理過程不複雜，但問題是要如何儲存最後兩個數字呢?

在這裡，我使用了num_vaild及num_shifter來處理這部分，num_shifter的輸入方式和計算機類似，數字會從左邊一直輸入，而最右邊的數字則會被拋棄。num_valid則是給予num_shifter該在什麼時候輸入bin，而不是讓shifter每個clk_100mhz的正緣就輸bin進去。之後會一一說明這些模組的原理。

Bin to dec:

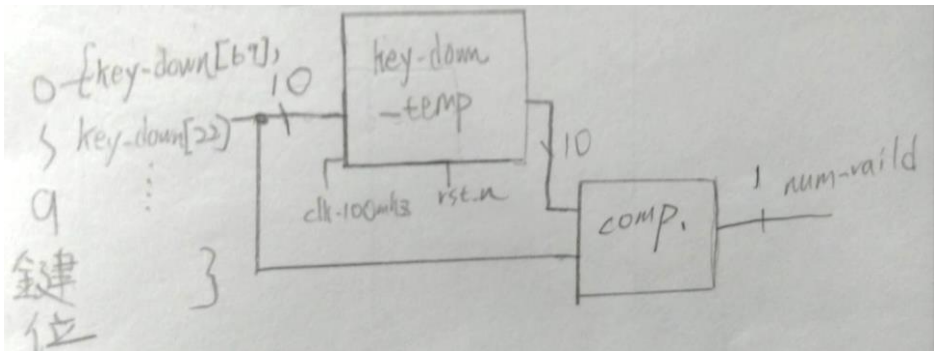
轉換的邏輯很簡單，千位數就是原二進位數除一千的商，百位數是原二進位數除一千的餘數再除一百的商，十進位是原二進位數除一百的餘數再除十的商，個位數是原二進位數除十的餘數。

num_shifter:



首先設置4bit的flip-flop oneth及tenth，用來儲存我們要的數字並持續輸出。在每個4bit flip-flop前設置多工器，控制在我們要的時候oneth輸入key_in(之前pb_decoder的結果)，tenth則輸入前面oneth的數字，而平常時flip-flop裡的數字都不動，如圖所畫。而shift_valid變數是用來控制上述的多工器。那麼我們要什麼時段輸入，當然是在我們按下數字鍵的那一剎那。原本我是使用keyboard_decoder的last_change來做為shift_vaild，然而這使我在放開按鍵的時候會重複輸入該數字，所以後來改採letter_vaild來判斷何時該「放行」數字。

letter_vaild:



letter_valid參考之前的pb_debounce做法，接收10bit的key_down[位置](數字按鍵零到九的位置)，10-bit正反器會儲存上一正緣所在的狀態，而這些數值會和現在狀態做比較，如果現在數值比上一正緣大，就代表有按新數字，反之則無，藉以濾除放開按鍵的情況。

3. I/O pin assignment

I/O 變數	對應腳位	補充說明
輸入		
PS2_DATA	B17	鍵盤訊號輸入
PS2_CLK	C17	鍵盤訊號輸入
clk_100mhz	W5	石英震盪器輸入
rst_n	V16	重置全部模組
輸出		
seg[7]	W7	控制七段顯示器腳位
seg[6]	W6	
seg[5]	U8	
seg[4]	V8	
seg[3]	U5	
seg[2]	V5	
seg[1]	U7	
seg[0]	V7	
dis[3]	W4	控制每個七段顯示器開關
dis[2]	V4	
dis[1]	U4	
dis[0]	U2	

Lab9.3 Implement a two-digit decimal adder/subtractor/multiplier using the right-hand-side keyboard. You don't need to show all inputs and outputs at the same time in the 7-segment display.

1.Design specification:

功能: 二位元計算機，包含加法、減法、乘法

使用方式:扳起最右邊 DIP 後，先透過小鍵盤輸入一個二位數，在按下運算符號，輸入第二個二位數，最後按大鍵盤的 Enter，輸出結果。

輸入: PS2_DATA (鍵盤訊號輸入)

PS2_CLK (鍵盤訊號輸入)

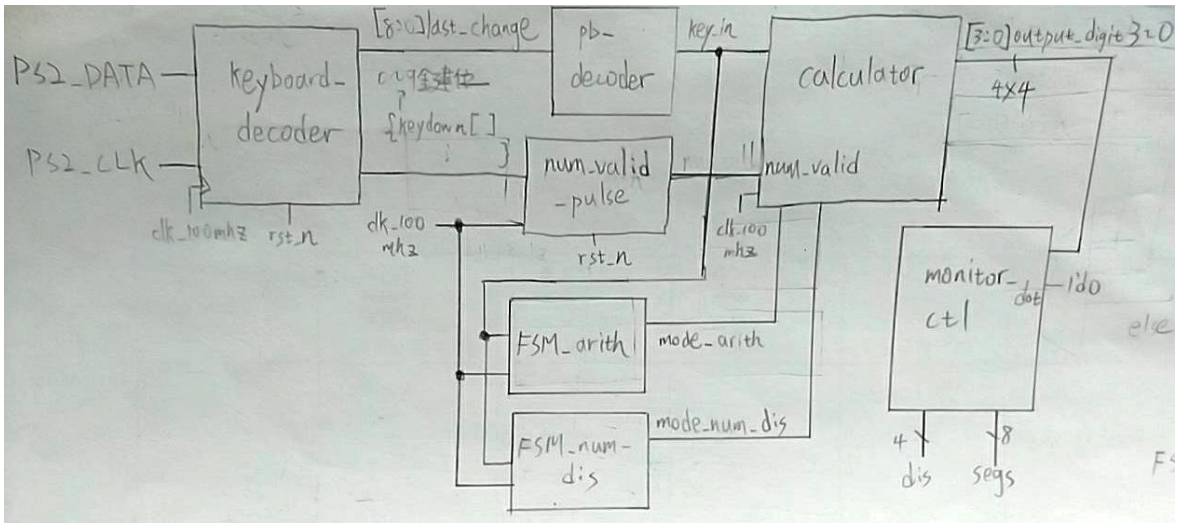
clk_100mhz (石英震盪器輸入)

rst_n (重置全部模組)

輸出: [3:0]dis (控制每個七段顯示器亮暗)

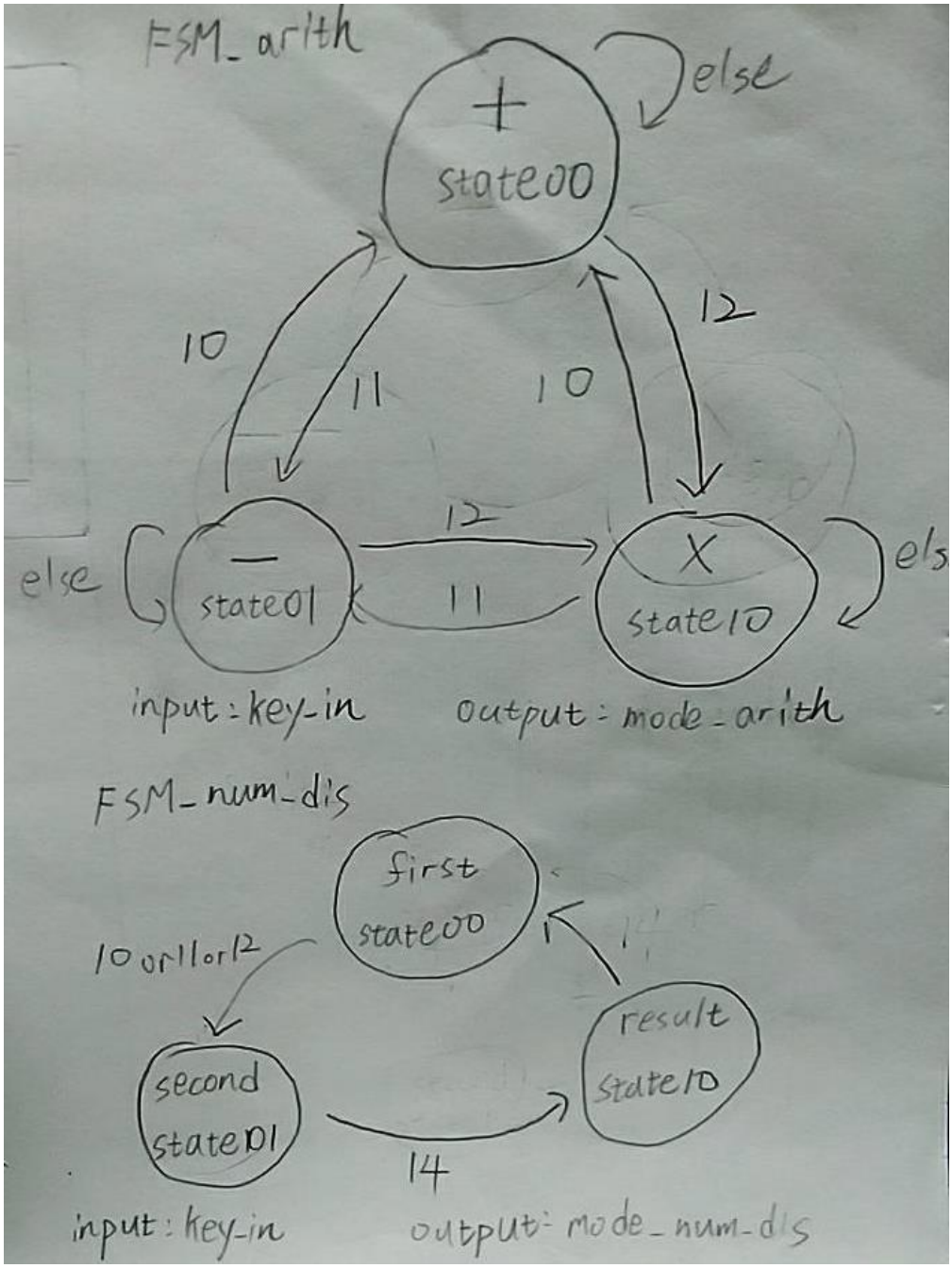
[7:0]segs (控制七段顯示器中每條燈管亮暗)

2.Design implementation



首先，鍵盤訊號會經由keyboard_decoder處理，輸出last_change和key_down訊號，pb_decoder會更進一步的把last_change訊號轉成4bit的數字，用來輸入calculator。Calculator模組主要功用是儲存輸入的數字，並且進行加、減、乘的運算，最後透過FSM，將我們想要的結果輸出到monitor_ctl，控制七段顯示器的燈管。FSM_arith是用來控制計算機的運算結果是加、減還是乘，FSM_num_dis則是選擇計算機要顯示第一個二位數、第二個二位數還是運算結果。之後會一一說明這些模組的原理。

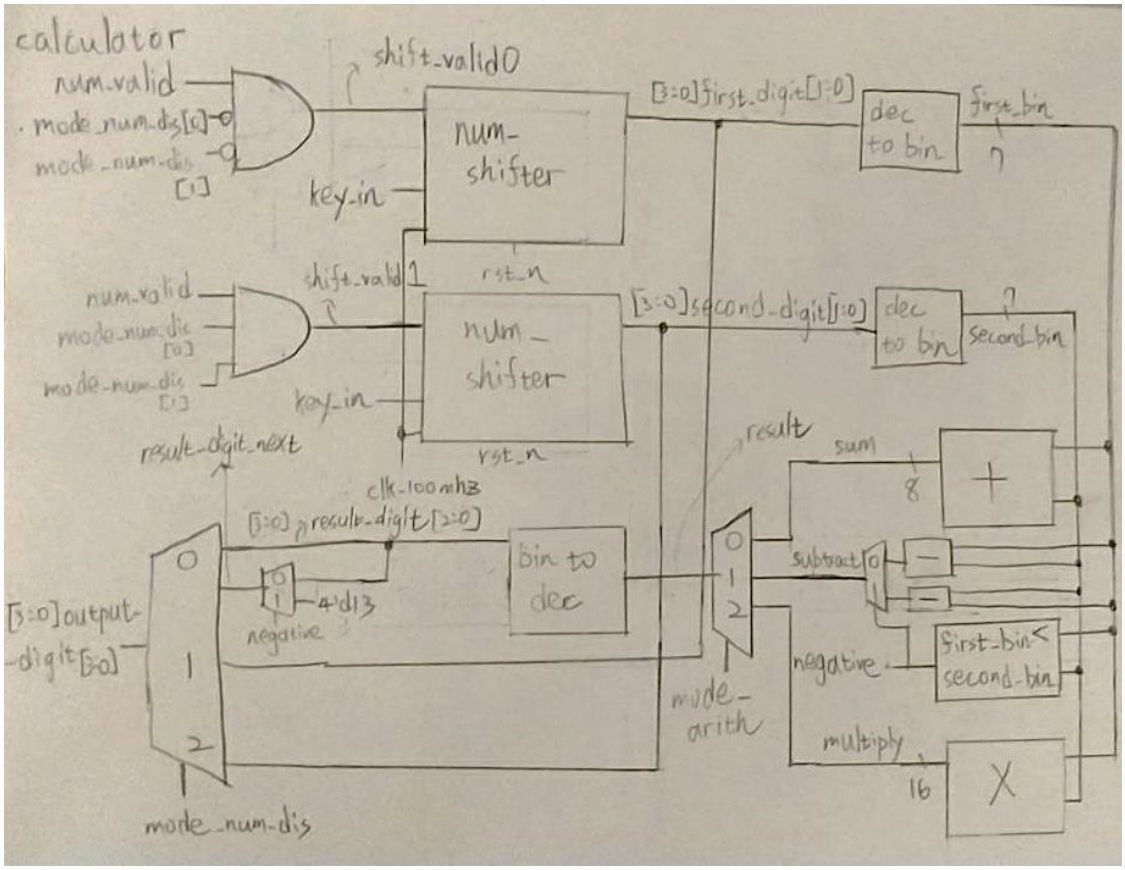
FSM_arith和FSM_num_dis:



FSM_arith的作用是選擇要被輸出的運算結果，因為兩個二位數數字在calculator模組中會同時產生加、減、乘的運算結果，我們需要用鍵盤選擇要哪一個。鍵盤輸入的運算符號會被pb_decoder轉換成4bit數字(10為加、11為減、12為乘)，按下加會使state變加，減則使state變減，乘則使state變乘。輸出mode_arith是上圖state後的數字，會被輸入到calculator。

FSM_num_dis的作用則是讓calculator選擇要輸出哪一組數字到顯示模組，state00是顯示第一組數(ex.減法的被減數)，state01是第二組數，state10是結果，當按下運算符號，state會跑到10，再按下enter時，state就跑到10。輸出mode_num_dis是上圖state後的數字，會被輸入到calculator。

Calculator:



Calculator模組是這個電路的核心，我們可以把這模組分為三部分，儲存輸入的數字，進行加、減、乘的運算並選擇運算的結果，控制我們想要輸出到顯示模組的數字。

首先，num_shifter的原理和上題一樣，只是因為第一個num_shifter是儲存第一個二位數(ex.減法的被減數)，第二個num_shifter則是第二個二位數，所以我們要個別控制兩個num_shifter的shift_valid(為一時輸入鍵盤按的數字)，以免同時被輸入進去。允許輸入的信號同時由FSM_num_dis和num_valid所決定，在FSM_num_dis為00時，第一個num_shifter允許輸入，FSM_num_dis為01時，則是第二個num_shifter允許輸入，FSM_num_dis為10時，兩個num_shifter都不允許輸入。

之後num_shifter輸出的數字，會被轉成二進位數字，然後各進行加、減、乘運算，其中注意減法，當減數大於被減數，會使negative為一，並使得兩個數字反過來相減，使結果不為負。這三個結果之後會被多工器選擇，控制其的是mode_arith。

有了運算結果後，我們就可以選擇要輸出顯示模組了，顯示模組一樣也由一個多工器選擇，由mode_num_dis控制，注意當negative為一時，會使得最後顯示器如果輸出減法結果時，第一位會顯示負號。

3. I/O pin assignment

I/O 變數	對應腳位	補充說明
輸入		
PS2_DATA	B17	鍵盤訊號輸入
PS2_CLK	C17	鍵盤訊號輸入
clk_100mhz	W5	石英震盪器輸入
rst_n	V16	重置全部模組
輸出		
seg[7]	W7	控制七段顯示器腳位
seg[6]	W6	
seg[5]	U8	
seg[4]	V8	
seg[3]	U5	
seg[2]	V5	
seg[1]	U7	
seg[0]	V7	
dis[3]	W4	控制每個七段顯示器開關
dis[2]	V4	
dis[1]	U4	
dis[0]	U2	

Lab9.4 Implement the “Caps” control in the keyboard. When you press A-Z and a-z in the keyboard, the ASCII code of the pressed key (letter) is shown on 7-bit LEDs.

1.Design specification:

功能: 顯示最後按的字母 ASCll 數字，包含大小寫功能。

輸入: PS2_DATA (鍵盤訊號輸入)

PS2_CLK (鍵盤訊號輸入)

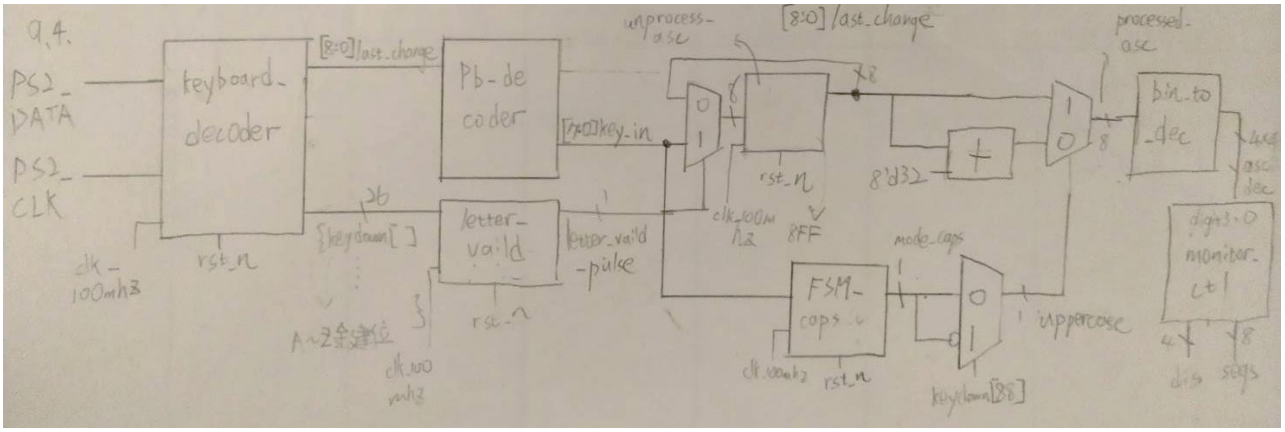
clk_100mhz (石英震盪器輸入)

rst_n (重置全部模組)

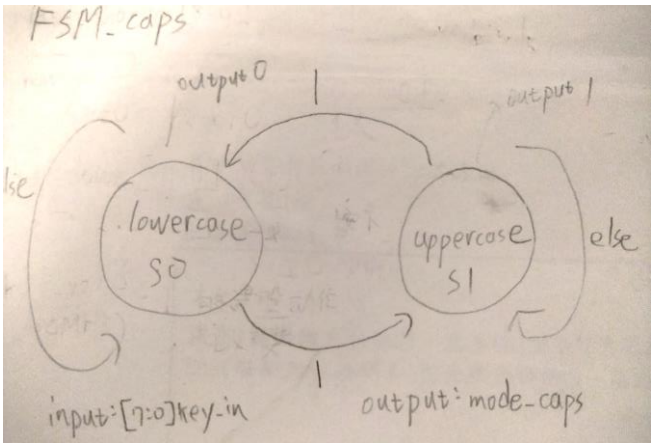
輸出: [3:0]dis (控制每個七段顯示器亮暗)

[7:0]segs (控制七段顯示器中每條燈管亮暗)

2.Design implementation



和前幾題一樣，pb_decoder 會處理進來的訊號，只是輸出的數字(key_in)，是他的小寫 ascii 數字，unprocessed_asc 專門來儲存每個輸進去的字母小寫 ascii 數字，而如何辨別什時要輸進去呢?藉由 letter_valid，方法和前幾題一樣。FSM_caps 專門讀取 CapsLock 信號，當按一下時 state 為一，再按一下時又變回零。如左圖，然後輸出 mode_caps 會被輸到處理 shift 的多工器，如果 shift 被壓下去時，輸出與 mode_caps 相反的訊號，沒壓時則為原樣。而這個訊號會決定 processed_asc 是否大寫(原 unprocessed_asc 加 32)或是小寫(保持原樣)，最後 processed_asc 會轉換成 BCD 後，被送到顯示模組去。



3. I/O pin assignment

I/O 變數	對應腳位	補充說明
輸入		
PS2_DATA	B17	鍵盤訊號輸入
PS2_CLK	C17	鍵盤訊號輸入
clk_100mhz	W5	石英震盪器輸入
rst_n	V16	重置全部模組
輸出		
seg[7]	W7	控制七段顯示器腳位
seg[6]	W6	
seg[5]	U8	
seg[4]	V8	
seg[3]	U5	
seg[2]	V5	
seg[1]	U7	
seg[0]	V7	
dis[3]	W4	控制每個七段顯示器開關
dis[2]	V4	
dis[1]	U4	
dis[0]	U2	
led	U16	右下角第一個燈

討論和心得:

這次的 lab 很需要動腦，尤其是構圖的階段花了我很多時間，應該要早點開始打的。