Lab9 report

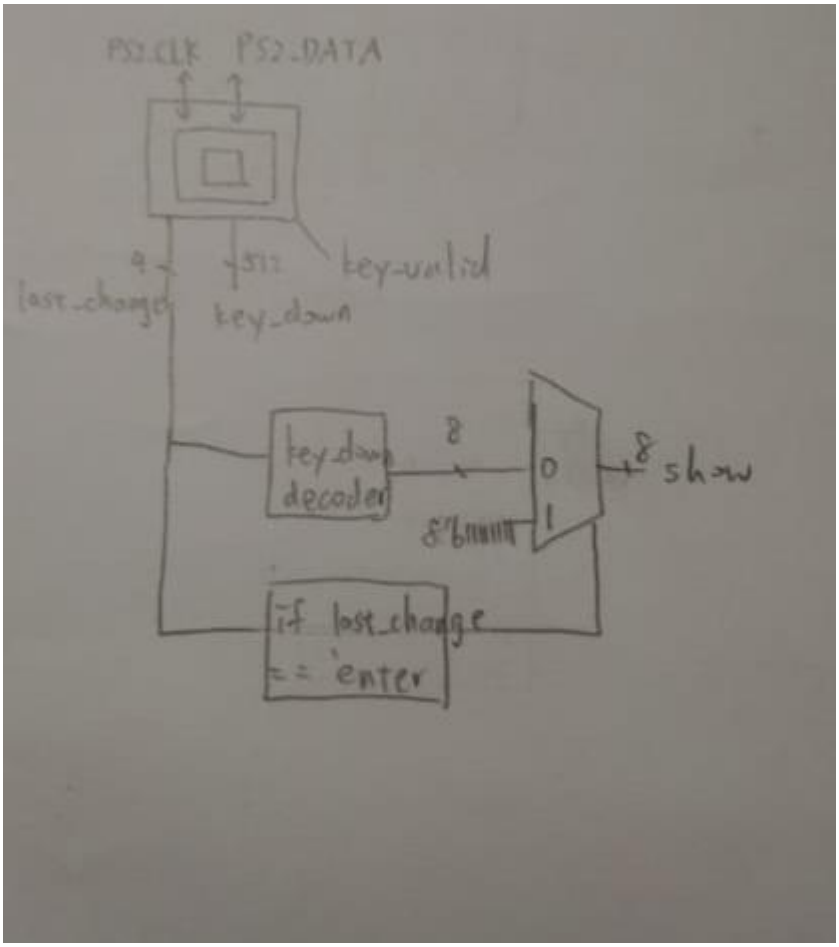
107061218 謝霖泳

1.

- Design Specification

  Inout: PS2_CLK, PS2_DATA

  Output: [3:0] ssd_ctrl, [7:0] show



- Design Implementation

  將鍵盤範例程式碼的 last_change 拉出來進入 last_change_decoder 中，會得到 8-bit 的 seven segment decode 結果。再判斷剛剛按下的鍵是否為 enter 鍵，如果是，就將螢幕清空，將螢幕清空的方法就是讓 4 個七段顯示器的燈都暗掉，也就是 ssd_ctrl = 4'b1111，如果不是，就顯示現在 decode 出來的結果。

  IO pin assignment:

ssd_ctrl[3]	ssd_ctrl[2]	ssd_ctrl[1]	ssd_ctrl[0]
W4	V4	U4	U2

dec[7]	dec[6]	dec[5]	dec[4]	dec[3]	dec[2]	dec[1]	dec[0]
W7	W6	U8	V8	U5	V5	U7	V7

clk	rst	PS2_CLK	PS2_DATA
W5	V17	C17	B17

- Discussion

  因為老師有給範例程式碼的關係，這一題很順利做出來了，按了相對應的要求按鍵就會顯示相對應的符號在七段顯示器上。

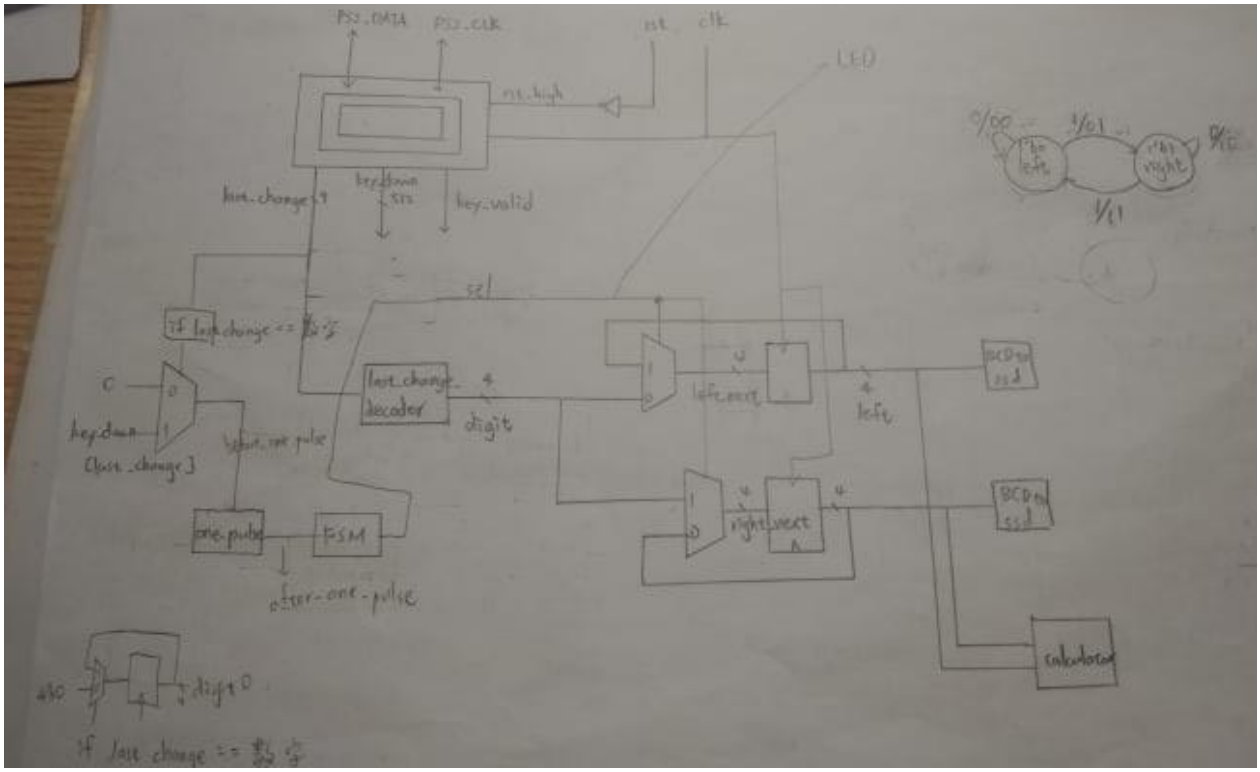
2.

- Design Specification

  Inout: PS2_CLK, PS2_DATA

  Input: clk, rst

  Output: [7:0] show, [3:0] ssd_ctrl



- Design Implementation

  將 last_change 拉出來，因為我們要做出兩個數字相加的結果，因此我們要輸入數字。首先，要判斷數字是否輸入成功，若輸入成功，則將 FSM 跳到下一個數字的 state。但是我們心裡其實知道，last change 會一直停留在最後動過的那個按鍵，即便你放開也不會消失，因此，我想到一個方法

可以讓 FSM 只跳一個 state，那就是 one pulse。先判斷現在的 last change 是否為數字鍵，如果是，那就將 key_down[last_change]這一位傳入 one pulse 的 module 中，出來的 one pulse 訊號再去控制 FSM，這樣就能讓 FSM 只跳動一個 state，當我們輸入一個數字的時候。

我的 FSM 分成兩個 state，分別名為 left 和 right。意思就是輸入左樹的 state 跟輸入右數的 state。

再來，要進行兩數的運算之前，必須先有站存器將兩數值儲存起來，因此，就有了 block diagram 中間偏右的兩組 D-FF。其運作機制為：在 FSM 的訊號為左為輸入成功時，左位的暫存器就會更新為我輸入的值，右位則保持不變。反之，FSM 的訊號為右為輸入成功時，右位的暫存器就會更新為我輸入的值，左位暫存器則保持不變。

之後，將兩個數值傳入計算器，將兩數相加並將結果顯示到七段顯示器上面。

IO pin assignment:

clk	rst	PS2_CLK	PS2_DATA
W5	V17	C17	B17

show[7]	show[6]	show[5]	show[4]	show[3]	show[2]	show[1]	show[0]
W7	W6	U8	V8	U5	V5	U7	V7

ssd_ctrl[3]	ssd_ctrl[2]	ssd_ctrl[1]	ssd_ctrl[0]
W4	V4	U4	U2

- Discussion

  這一題所使用到的概念與上題相同，只是多了 FSM 來控制我現在輸入的為數為哪一位。並且多出了兩個暫存器來儲存我現在的左位數跟右位數。

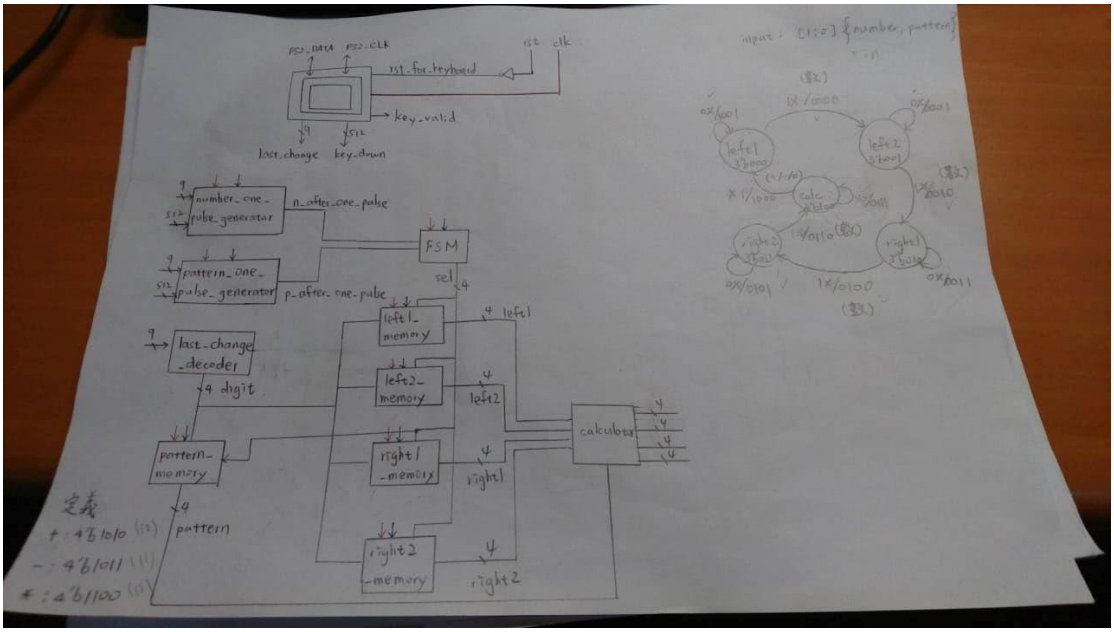
3.

- Design Specification

  Inout: PS2_CLK, PS2_DATA

  Input: clk, rst

  Output: [3:0] ssd_ctrl, [7:0] show, minus



- Design Implementation

這一題也會需要用到暫存器的結構，要把兩數的個位數和十位數都記下來，此外，與上一題的不同之處在於，還需要多一個暫存器將現在的運算元記下來。因此，在 block diagram 中可以看到 5 個 D-FF。

那麼，這 5 個暫存器的運作機制與上一題雷同，也就是以 FSM 的 output 來決定我現在要維持原本的值，抑或更新為我現在輸入新的值。再來，將這兩個二位數和運算元送入我的 calculator module 中，依照運算元將兩數作出相對應的運算。得到結果後將結果分離出千位、百位、十位以及個位，再將結果顯示到七段顯示器上便大功告成。

此外，減法的運算我有先判斷是大減小還是小減大，若有負號產生，則會讓 U15 這個 LED 燈亮起來，告訴我們現在的運算結果為負數。

IO pin assignment:

ssd_ctrl[3]	ssd_ctrl[2]	ssd_ctrl[1]	ssd_ctrl[0]
W4	V4	U4	U2

show[7]	show[6]	show[5]	show[4]	show[3]	show[2]	show[1]	show[0]
W7	W6	U8	V8	U5	V5	U7	V7

clk	rst	PS2_CLK	PS2_DATA	minus
W5	V17	C17	B17	U15

- Discussion

  這題的關鍵在於能不能將各個位數配合 FSM 的 output 儲存在 D-FF 中，以及做出相對應的運算結果之後能否成功地分離出個個位數。

4. conclusion

  這次的 lab 中遇到了幾個 bug，花了我不少的時間，就是我在要進 FSM 前會有經過一個 one pulse generator，而送進這個 one pulse generator 的訊號為 key_down[last_change]，也就是我現在按下去的那個按鍵。但是，我一開始把它打成 last_change[key_down]，然後竟然還編譯通過，連 bitstream 都過了，燒到板子上後發現有些結果一直不太對，找了非常非常久才發現這個粗心的錯誤。