邏輯設計實驗 Lab9 結報

105060012 張育菘

1 VGA displaying functions.

1.1 Inputs of the VGA controller are clk, reset, en and outputs of the VGA controller are hsync, vsync, vga_red[3:0], vga_green[3:0], vga_blue[3:0].

1.2 At the beginning or when reset (button) is pressed, the VGA display shows the image (e.g. amumu.jpg). The VGA image stay still until en (button) is pressed.

1.3 Pressing odd times en button to start/resume scrolling. Pressing even times en button to pause scrolling. Counter for en press is reset to zero when reset is pressed.

Design Specification

Input：clk,rst,en,;

Output：led,hsync,vsync,vgaRed[3:0],vgaGreen[3:0],vgaBlue[3:0];

Inout：PS2_DATA,PS2_CLK;

*W19 控制 rst　*T17 控制 state

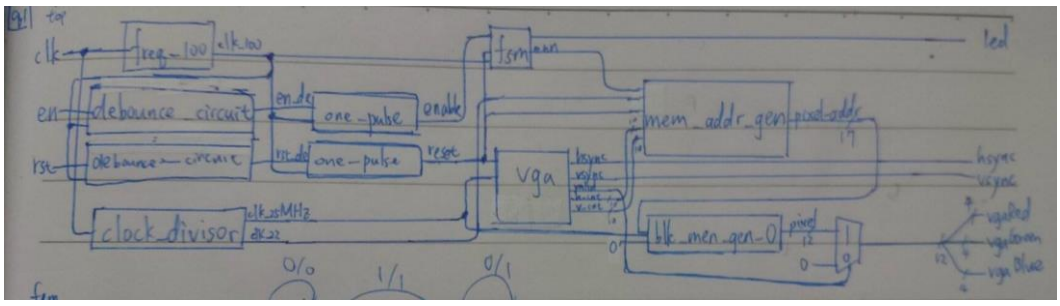
Design Implementation

Logic function :

1. top：

此 top module，專門拿來呼叫其他小 moudule 的。

還有給定 vgaRed, vgaGreen, vgaBlue 的值{vgaRed, vgaGreen, vgaBlue} = (valid==1'b1) ? pixel:12'h0;

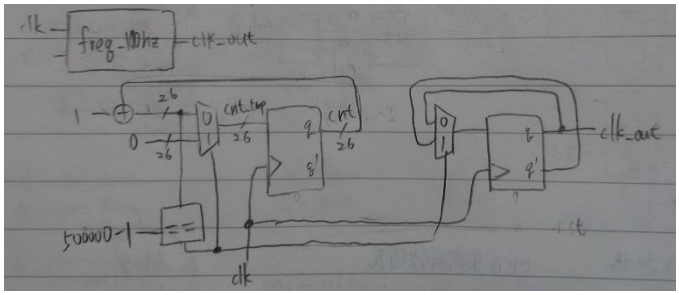


2. freq_100hz：

此為除頻的 module，輸出 100hz 的 clk_out 當作 debounce_circuit、one_pulse 以及 fsm 的 clk。

- 這裡不能接 top 的 input rst，因為那裡的 rst 是希望 count_up 重新開始數。

當 rst=0 時，freq_100hz 輸出的 clk 持續為 0→freq_100hz 不做事。

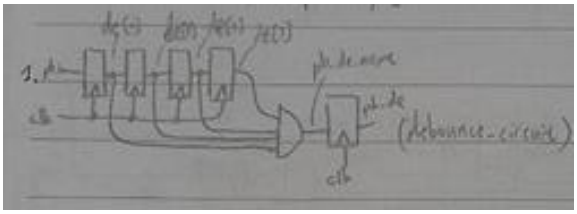


3. debounce_circuit：

此為**除按鈕雜訊**的 module，原理是，當 debounce_circuit 收到連續 4 個 1 時才產生一個 0→1 的訊號，這樣可以避免收到前段起伏不定的雜訊，達到除雜訊的效果。

且要接 100hz 的 clk，因為若接此的 clk 頻率太快就無法達到除雜訊的效果，太慢可能會超過按鈕維持穩定值的時間，因此我選擇 100hz 作為 debounce_circuit 的 clk。

- 這裡不能接 top 的 input rst，因為那裡的 rst 是希望 count_up 重新開始數。當 rst=0 時，freq_100hz 輸出的 clk_100 持續為 0→ debounce_circuit 不做事。

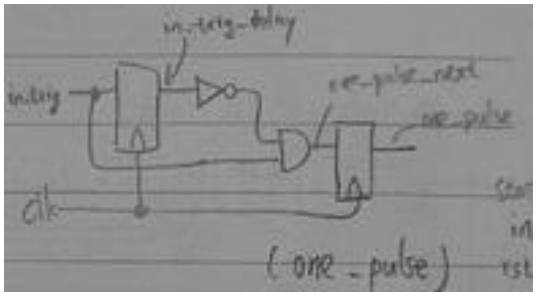


4. one_pulse：

輸入訊號經過 debounce_circuit 處理後，其週期可能會超過 1 個 clk 的時間，因此需要此 module 使訊號的週期為 1 個 clk 的時間。

這裡的 clk 我也是接與 debounce_circuit 同的 100hz。

- 這裡不能接 top 的 input rst，因為那裡的 rst 是希望 count_up 重新開始數。當 rst=0 時，freq_100hz 輸出的 clk_100 持續為 0→ one_pulse 不做事。



5. fsm：

此 module 做為 finite state machine，以接出 state(在 display 內)決定要顯示 lap 抑或是顯示正在數的狀態。

這裡的 clk 我是接 100hz，因為 clk 在 0→1 時讀 one_pulse(其 clk 為 100hz)處

理過的值，因此 top 的 clk 不能太快。

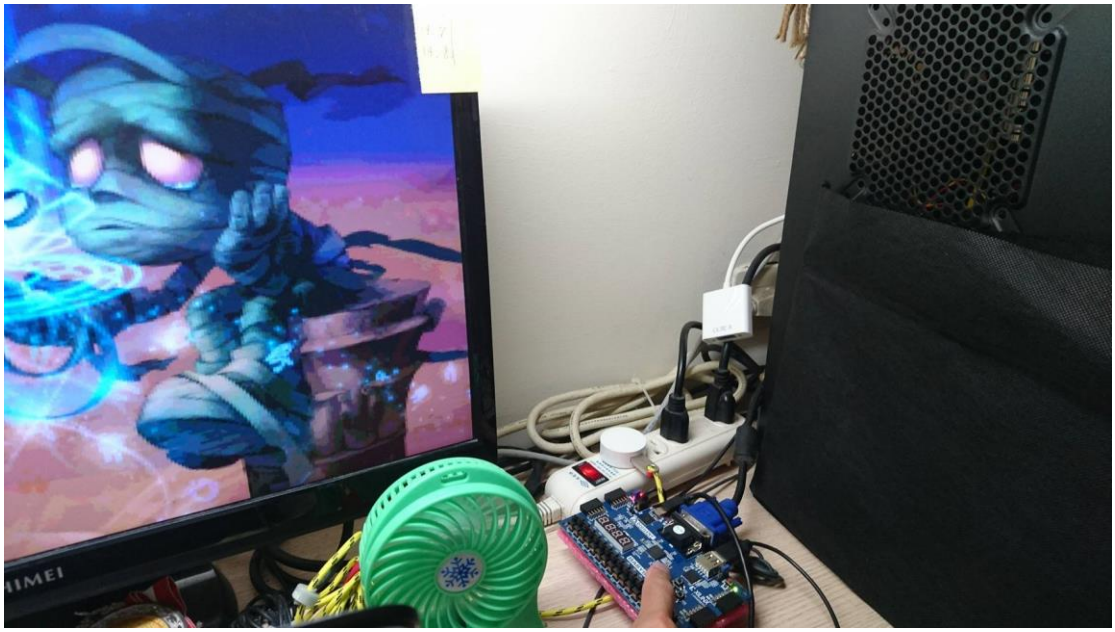
6. Clock_divisor：
   此 module 是用來產生 clk，分別產生 clk22(約 24hz)以及 clk_25MHz。
7. vga_controller：
   此 module 是用來更新畫面的，產生的 h_cnt 和 v_cnt 可以用來讀圖片。
8. mem_addr_gen：
   此 module 是用來讀圖片檔的，由於要產生往下滑的感覺，因此我設計 position_next = position + 1;讓 y 軸每經一個 clk 就讀不一樣的位置

Result

1. 圖二：當按下 T17 時圖片暫停往上移。
2. 圖三，當按下”W19”(rst)時，回歸至上移前的畫面。

圖一

圖二

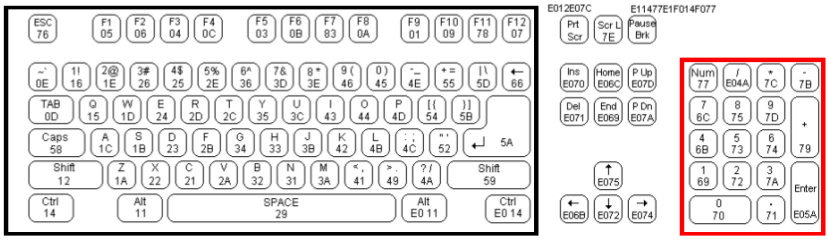


Discussion

1. adder
   當{in1+in2}>=4'd16 時會有 overflow 的情形產生，我是利用"規律性"發現當有 overflow 情形產生時，需加 4’d6 並可以解決此問題。
2. 設計 key_down[last_change] && key_valid 去讀值，因為 key_valid 是一當按下或放下才會產生的訊號，因此拿他與 key_down[last_change]做交集可判斷 last_change 是否有被按下去，最後至 mux 轉換成 BCD(Binary-Coded Decimal)的形式。
3. 設計 mem_addr_gen 時，position_next = position + 1;讓 y 軸每經一個 clk 就讀不一樣的位置

2 Calculator display.

2.1 Combine the key board controller and VGA displaying controller to design a calculator with 2-digit addition/subtraction/multiplication. The display function should be the same as usual calculator or APP in the smartphone.



state[1]	state[0]	clk	rst
P1	N3	W5	V17

Design Specification

Input：clk,rst;

Output：hsync,vsync,vgaRed[3:0],vgaGreen[3:0],vgaBlue[3:0],state[1:0];

Inout：PS2_DATA,PS2_CLK;

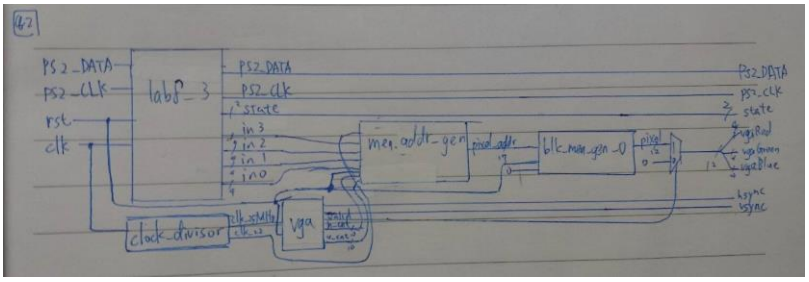
*V17 控制 rst，當 rst==0 時會產生 rst 的訊號

Design Implementation

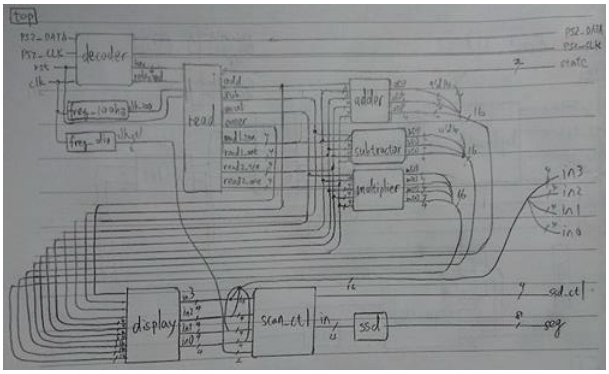
Logic function :

1. top：

此 top module，專門拿來呼叫其他小 moudule 的。

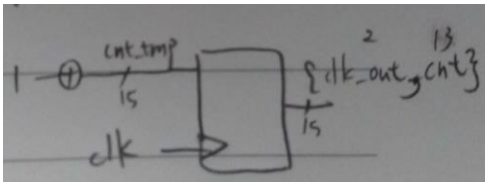


2. lab8_3：



3. freq_div：

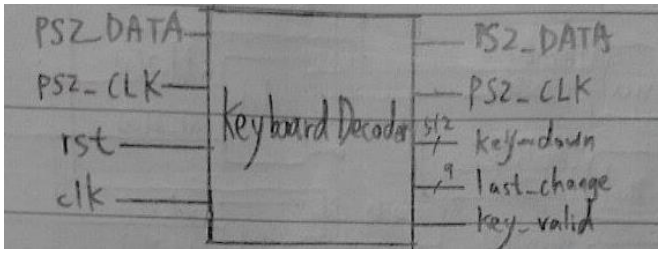
   此為除頻的 module，輸出的 clk_ctl 當作 scan_ctl 的 clk，其頻率極高。



4. KeyboardDecoder：

   這個 module 主要的目的是讀取鍵盤的值。

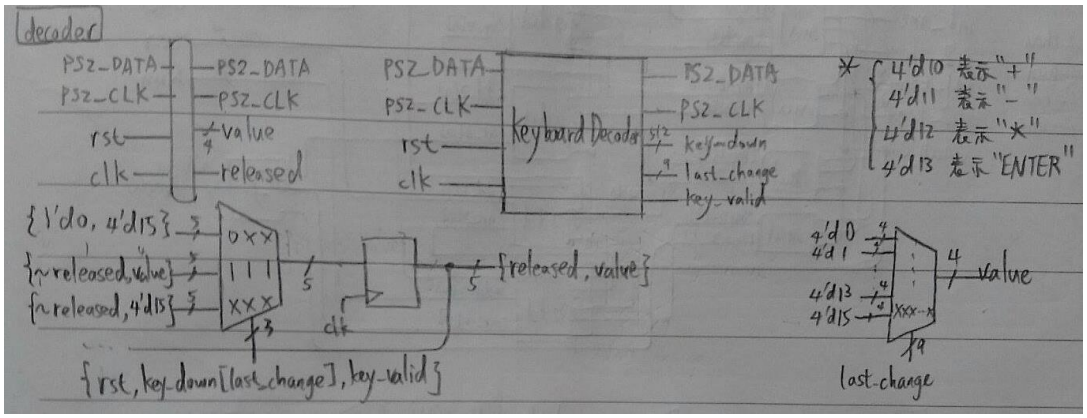
   - key_down 是表示那一個鍵有沒有被按下去；last_change 是表示按鍵按下去的值；key_valid 在按鍵被按下去或放開的當下方會產生一個突波(理論上)。



5. decoder：

   此 module 是用來讀取從 key_board 輸入值，並轉成 BCD 的形式。

   - 取值的方法與第一題同，都是用 key_down[last_change] && key_valid 的方式判斷 last_change 為何值的被按下去。
   - 把 KeyboradDecoder 包在這個 module 的目的在於，使 KeyboradDecoder 影響範圍只限在這個 module 內，不會因產生不規則的訊號而破壞結果。
   - 此輸出的 released 是在 read module 內讀值用。
   - 4'd10 表示"+"；4'd11 表示"-"；4'd12 表示"*"；4'd10 表示"ENTER"

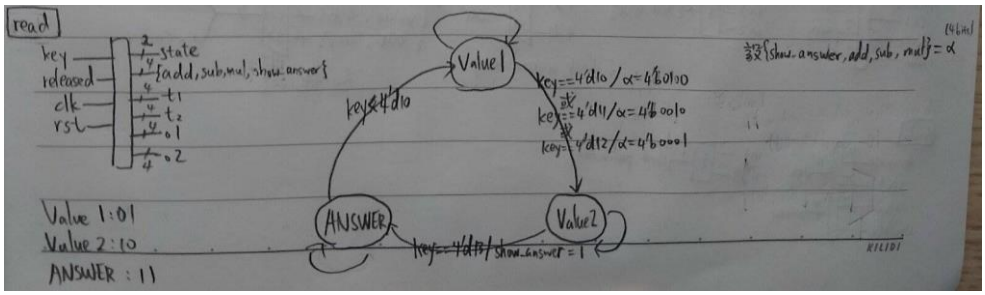


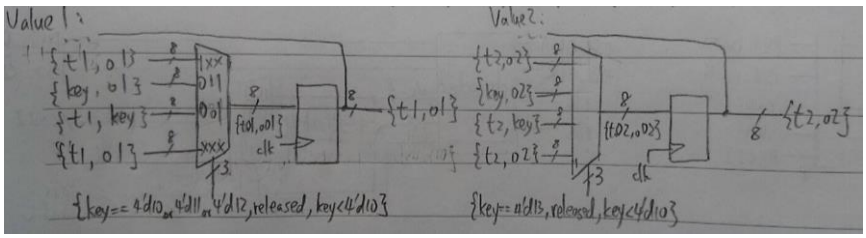
6. read：

   此 module 是用來讀取兩個數的值。

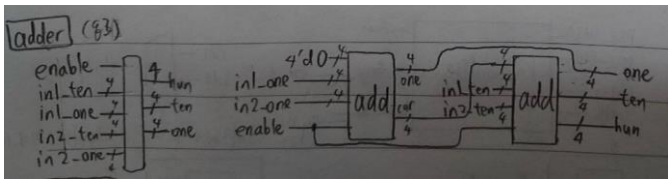
   - 特別在此設計切換每一個 state 的條件都不同，目的在於能避免彼此間互相影響，以及避免 key_valid 產生多的突波使 state 跳得太快。

- 從`Value1 →`Value2 是當讀取到"+","-"or"*"的鍵盤訊號才會切換
  從`Value2→`ANSWER 是當讀取到"ENTER"的鍵盤訊號才會切換
  從`ANSWER →`Value1 是當讀取到”數字”的鍵盤訊號才會切換
- state==`Value1 表示讀取第一個數；state==`Value2 表示讀取第二個數
  state==`ANSWER 表示顯示運算結果。

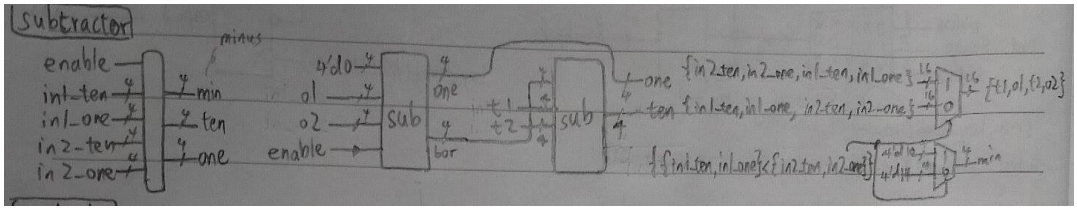




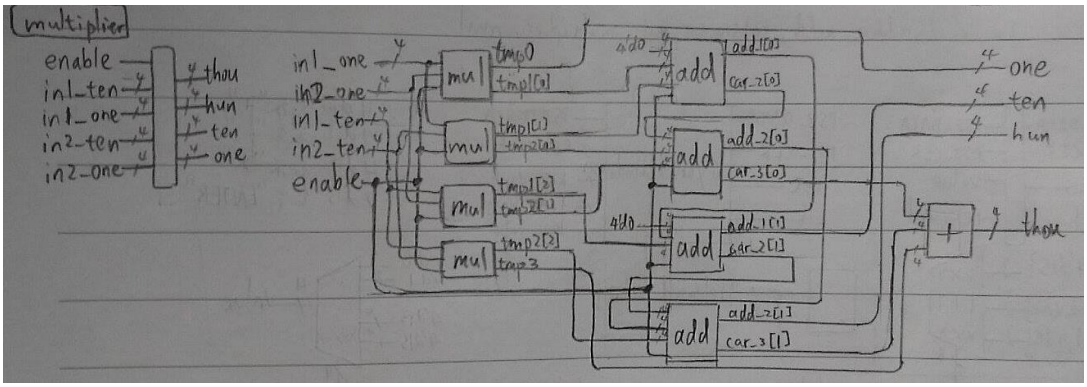
7. adder：
   此 module 是用來做二位數的加法，內含 add(做一位數的加法)。
   - enable 是從 read module 接出來的 add 訊號。



8. subtractor：
   此 module 是用來做二位數的減法，內含 sub(做一位數的減法)。
   - min 是當"(較小的數) - (較大的數)"產生的"負號"，在七段顯示器上顯示。
   - enable 是從 read module 接出來 sub 訊號。



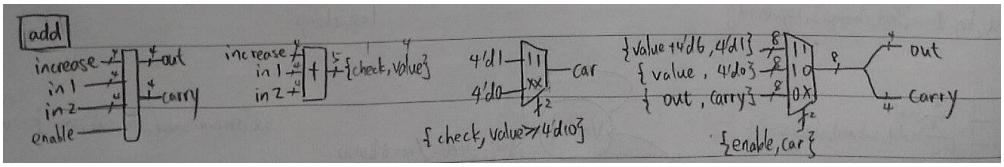
9. multiplier：
   此 module 是用來做二位數的乘法，內含 add(做一位數的加法)以及 mul(做一位數的乘法)。
   - 由於二位數的乘法是 4 個一位數乘法產生的值，依照位數一一做相加，因此才會包 add 以及 mul module。
   - enable 是從 read module 接出來的 mul 訊號。



10. add：

此 module 是用來做一位數的加法。

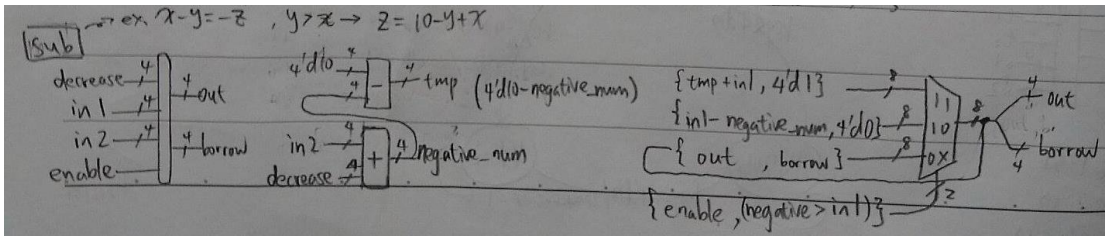
- 使用 enable 作為 mux 的判斷值，可以使在 enable==1 的當下才會運算，其餘時間則維持當下的值。
- 當 increase + in1 + in2 超過 4'd15 時會有 overflow 的產生，因此多接一個 check 訊號出來判斷是否有 overflow 的產生，若有 overflow 的產生，便使運算出來的值再加上 4'd6(經驗使然)使結果產生應有的值。



11. sub：

此 module 是用來做一位數的減法。

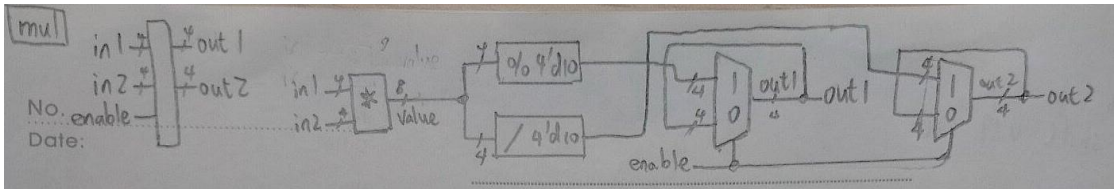
- 使用 enable 作為 mux 的判斷值，可以使在 enable==1 的當下才會運算，其餘時間則維持當下的值。
- 想法來源：ex. X-Y=-Z, Y>X → Z = 10 – Y + X



12. mul：

此 module 是用來做一位數的乘法。

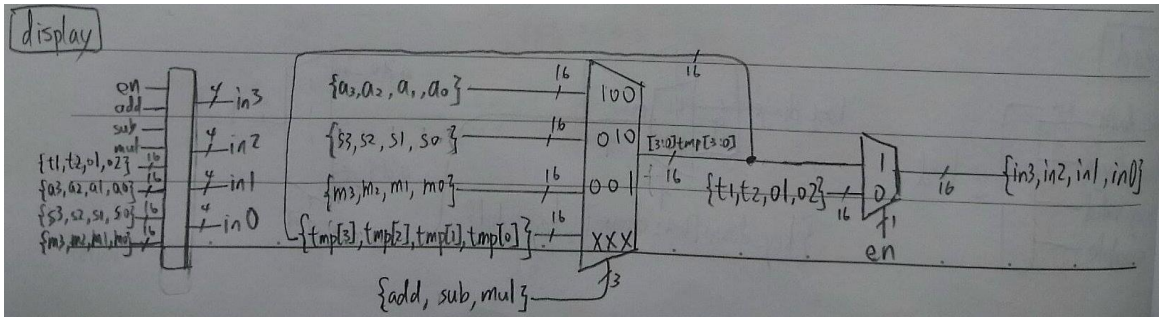
- 使用 enable 作為 mux 的判斷值，可以使在 enable==1 的當下才會運算，其餘時間則維持當下的值。



13. display：

此 module 是用來決定 men_addr_gen 的 input :in3, in2,in1,in0。

- en 是從 read module 接出來的 enter 訊號，目的是判斷要顯示的數是計算前抑或是計算後的值。
- 這裡接{add,sub,mul}作為 mux 訊號用意在判斷要顯示經個何種運算的結果。



14. Clock_divisor：

此 module 是用來產生 clk，分別產生 clk22(約 24hz)以及 clk_25MHz。

15. vga_controller：

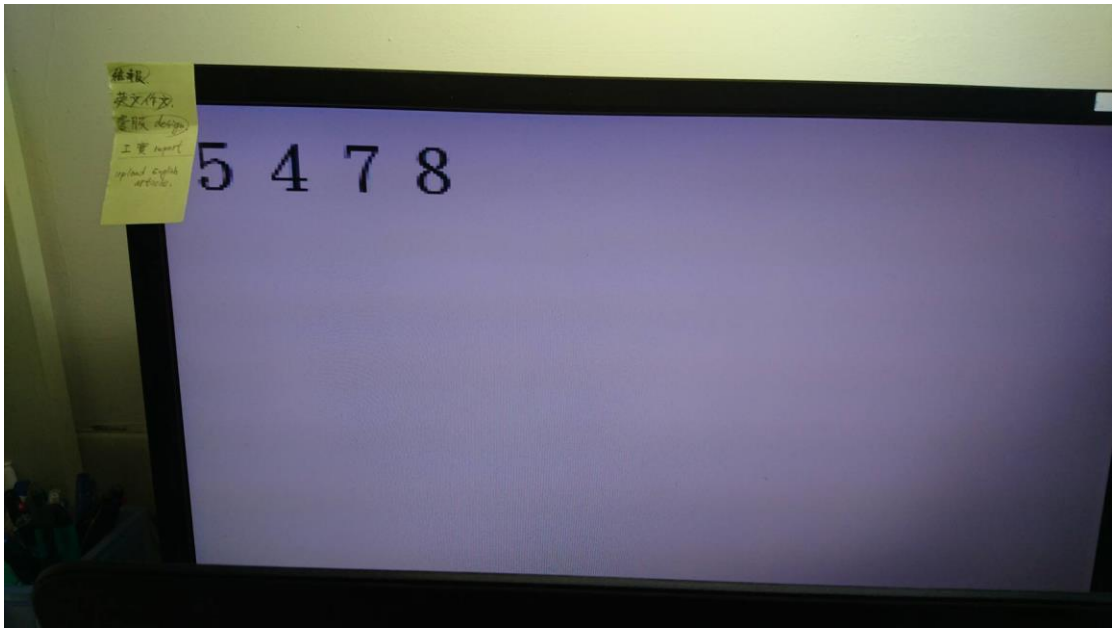
此 module 是用來更新畫面的，產生的 h_cnt 和 v_cnt 可以用來讀圖片 。

16. mem_addr_gen：

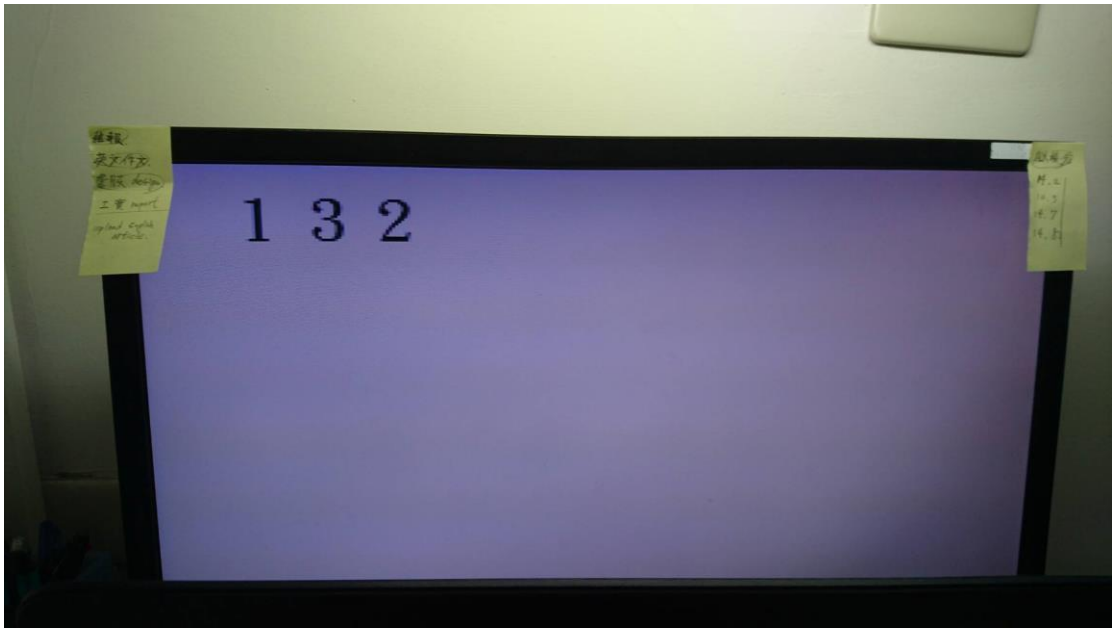
此 module 是用來讀圖片檔的。因此有從 display 接出 in3, in2,in1,in0 訊號，作為在某個位置要讀哪一個圖片的依據。

Result (state==`Value1(2'b01)表示讀取第一個數；state==`Value2(2'b10)表示讀取第二個數 state==`ANSWER(2'b11)表示顯示運算結果。)

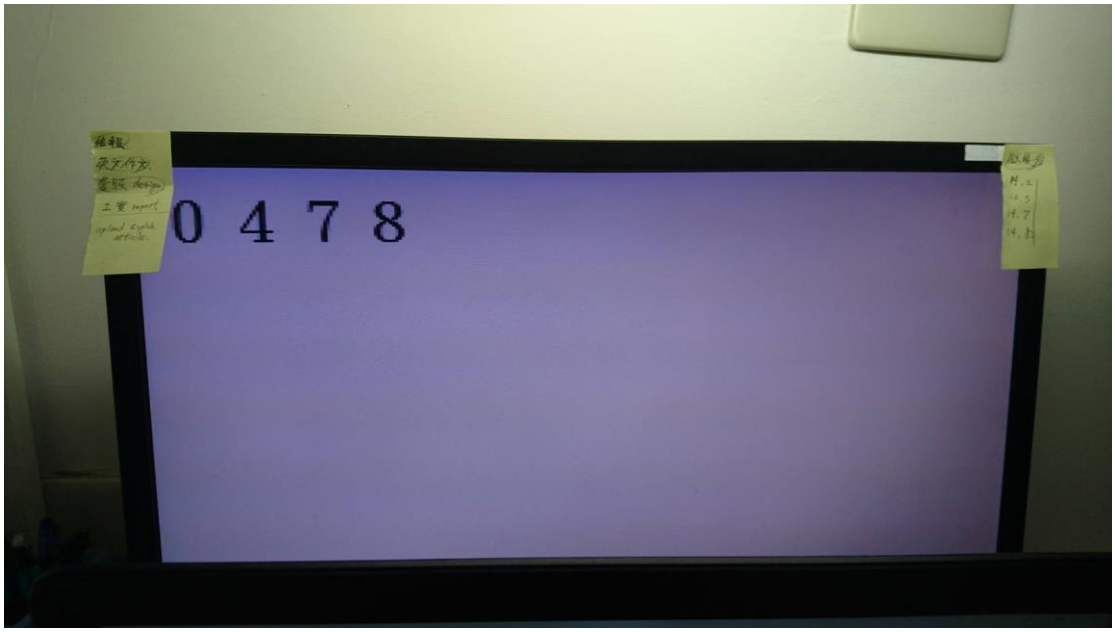
1. 圖一~圖四是在進行"19 + 59 = 78"的過程(加法)。
   I. 圖一二"54+78=132"。
   II. 圖三四"04*78=312"。
   III. 圖五六"05-78 = -73"。



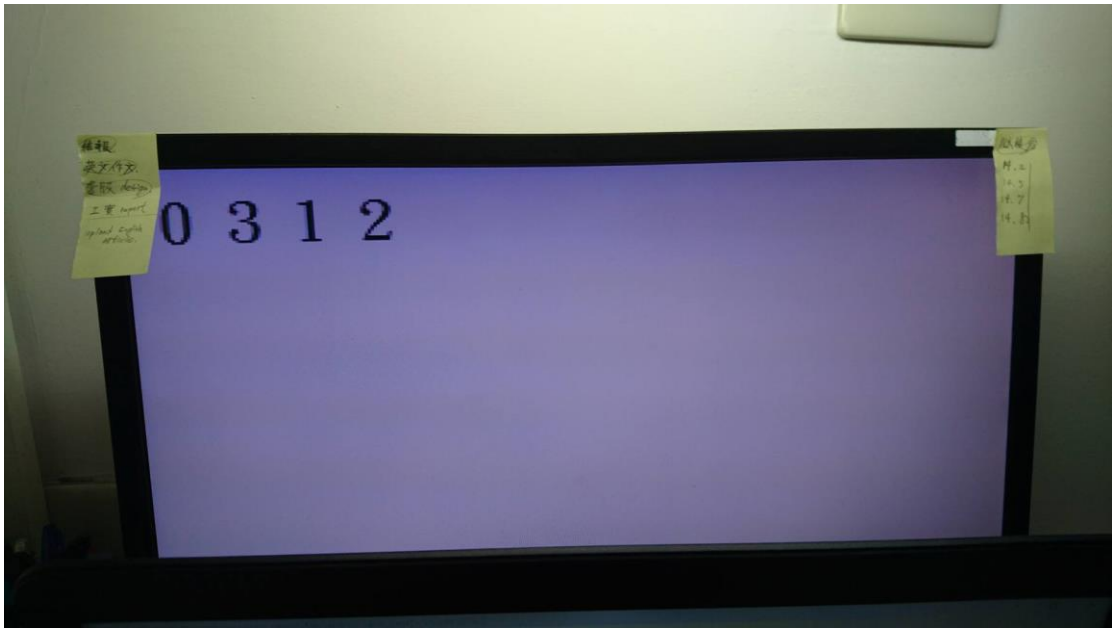
圖二



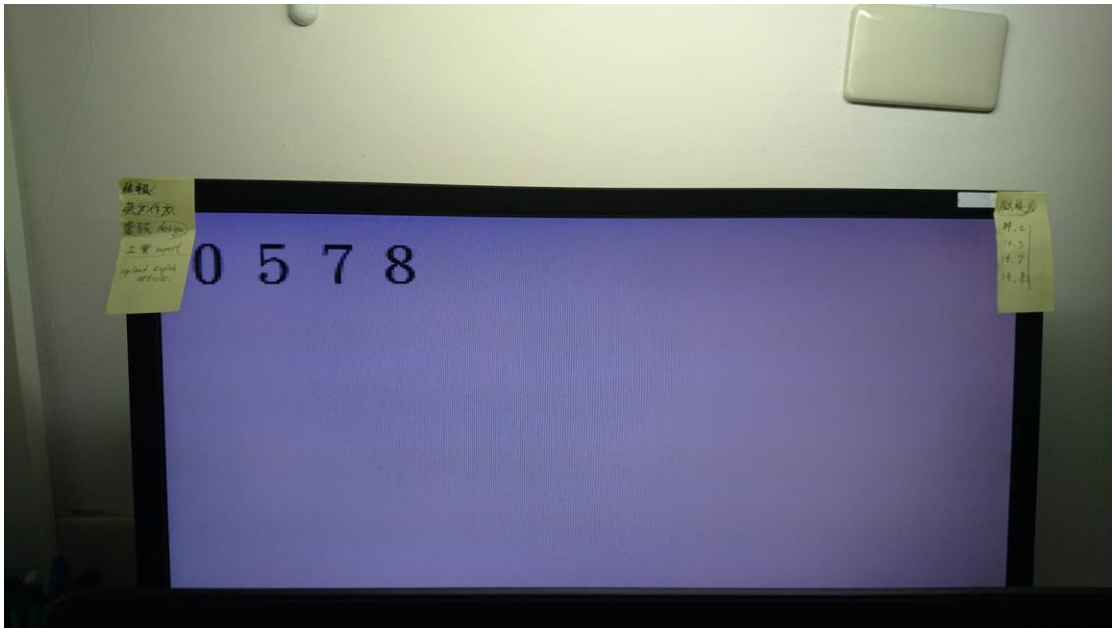
圖三



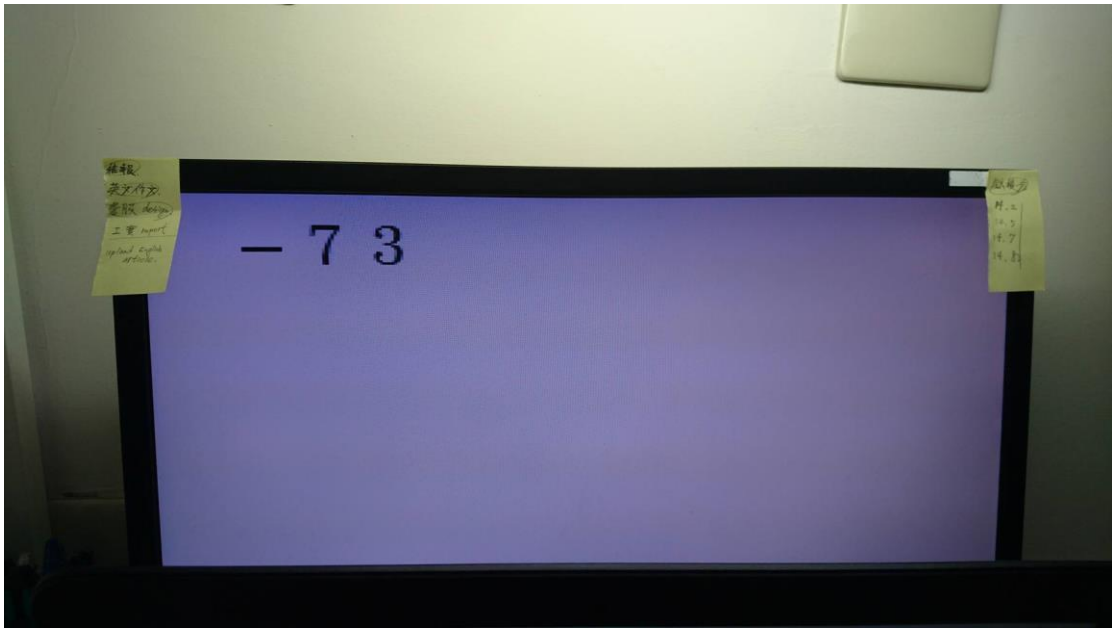
圖四



圖五



圖六



Discussion

1. decoder

   把 KeyboradDecoder 包在這個 module 的目的在於，使 KeyboradDecoder 影響範圍只限在這個 module 內，不會因產生不規則的訊號而破壞結果。

2. read

   特別在此設計切換每一個 state 的條件都不同，目的在於能避免彼此間互相影響，以及避免 key_valid 產生多的突波使 state 跳得太快。

3. add

   當 increase + in1 + in2 超過 4'd15 時會有 overflow 的產生，因此多接一個 check 訊號出來判斷是否有 overflow 的產生，若有 overflow 的產生，便使運算出來的值再加上 4'd6(經驗使然)使結果產生應有的值。

4. sub
   想法來源：ex. X-Y=-Z, Y>X → Z = 10 – Y + X。
5. mem_addr_gen
   做法是先給定個數字圖片左上角的 x 軸和 y 軸的值，pixel_addr 在應其需要讀取正確的值。

3 TETRIS element generator

3.1 Generate basic elements of TETRIC (as follows) randomly in the VGA monitor, and plot each of them in the center of the first row of the display, which is a 10 x 20 (WxH) square 2D playing space.

3.2 Each generated basic element moves down by the step of a square at the speed of 1Hz. Finally, they disappear below the playing space. When a basic element disappears, a new basic element is generated again and fall down again repeatedly.

Design Specification

Input：clk,rst;

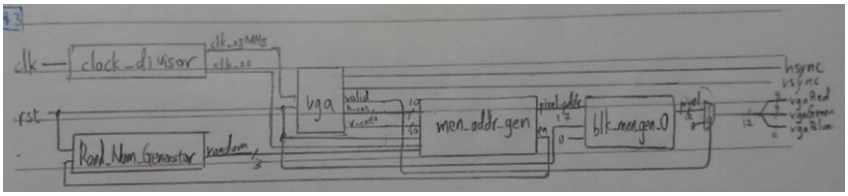
Output：hsync,vsync,vgaRed[3:0],vgaGreen[3:0],vgaBlue[3:0];

Inout：PS2_DATA,PS2_CLK; //V17 控制 rst，當 rst==0 時會產生 rst 的訊號

Design Implementation

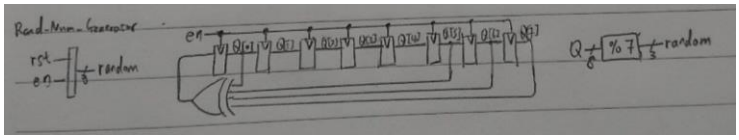
Logic function :

1. top：
   此 module，專門拿來呼叫其他小 moudule 的。

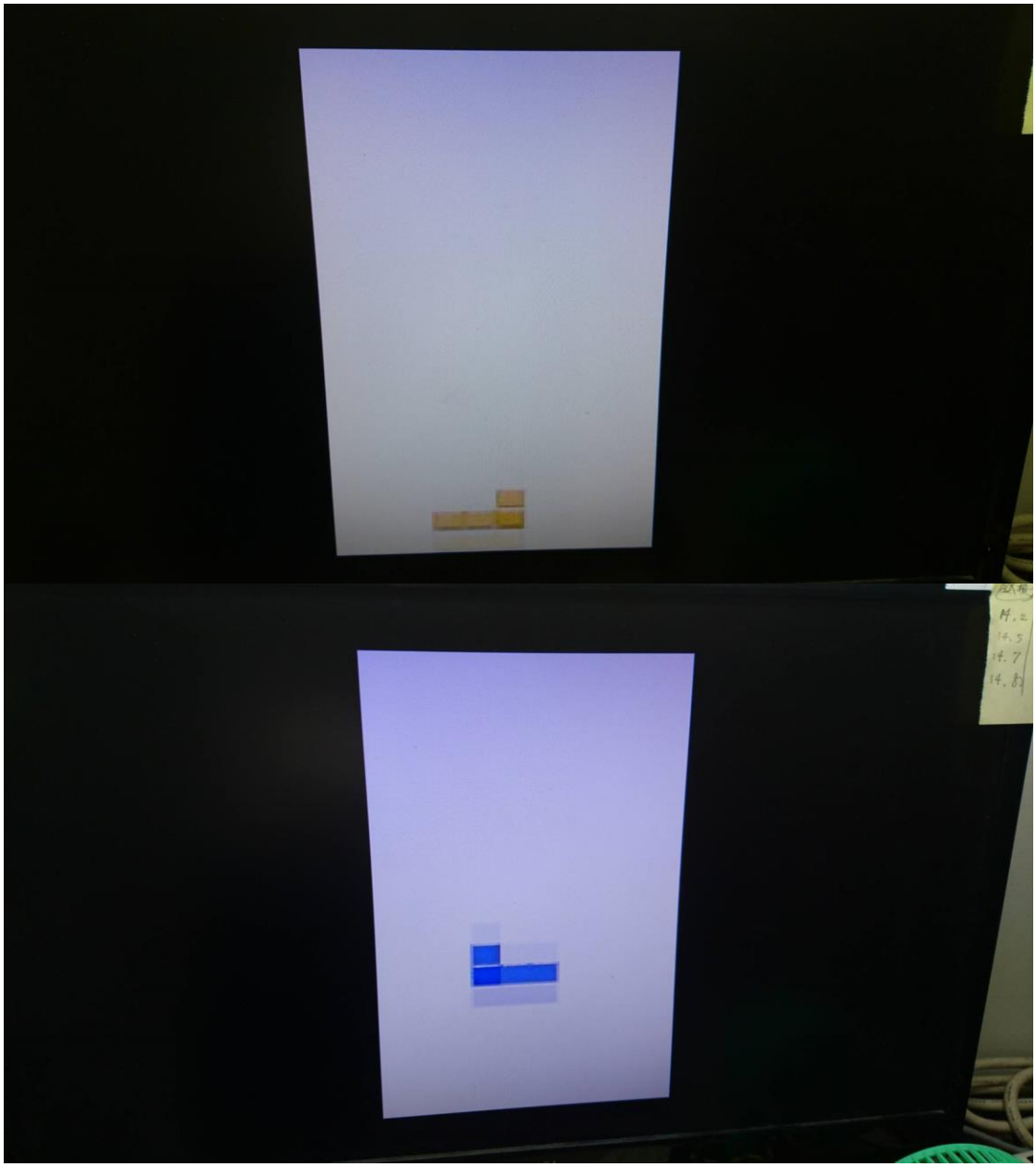


2. Clock_divisor：
   此 module 是用來產生 clk，分別產生 clk22(約 24hz)以及 clk_25MHz。
3. vga_controller：
   此 module 是用來更新畫面的，產生的 h_cnt 和 v_cnt 可以用來讀圖片 。
4. mem_addr_gen：
   此 module 是用來讀圖片檔的，並從 Rand_Num_Generator 接出 random 訊號，作為決定顯示哪個方塊的依據，且這裡讀圖的方法是一排方格一排方格一起顯示。
5. Rand_Num_Generator：

此 module 是用來產生亂數。



Result



Discussion

1. decoder
   這設計以 key_down[9'b0_0101_1000]作為給 caps 值的 DFF 的 clk。這樣可以使當按下"caps"鍵時就會進行存值。
2. 由於大寫與小寫字母的 ASCII code 都剛好相差 8'd32(小寫>大寫)，因此在做給 ASCII code 值時可以運用這個特性，不用把每個字母大小寫的 ASCII code 都解

碼。

3. 設計 men_gen_addr 時，從 Rand_Num_Generator 接出 random 訊號，作為決定顯示哪個方塊的依據，且這裡讀圖的方法是一排方格一排方格一起顯示。
4. 設計 men_gen_addr 時，還設定邊界條件，因此並不會掉落到最底才會產生新的方塊掉落。

Conclusion：

這次的 lab 做的真的頗挫折的，光是在顯示螢幕上就下了不少工夫，真的很感謝一路上幫助我的人。