EECS 2070 02 Digital Design Labs 2019

Lab 7

學號：107062107　姓名：王領崧

1.前言

此次 lab 利用 FPGA 板來外接螢幕，搭配板子上的按鈕，實作出各種螢幕特效，譬如左右移動、黑幕的擴張以及分成四個小螢幕，各自上下左右移動。

2.實作過程

lab07_1

lab07_1 需要使螢幕中的圖片不停地左右移動循環，配合 en 和 dir 按鈕來達到停止與變換方向的功能。SampleCode 示範的功能是使圖片不停的往上移動循環，使用了每次提供 pixel_addr 時，都加上累加的 position*螢幕的寬度(240)，造成每次取圖片中的 pixel 時，都會依序往正下方取，達到圖片往上的效果。而我們要實作的是圖片的左右移動，相當於每次計算 pixel_addr 時，都應該往右取(左移)或是往左取(右移)，因此只需要加或減 position(counter 模式)，並%320，避免算到最右邊時會取到下一排的第一位，就可以順利完成了。

```
assign pixel_addr = (((h_cnt>>1)+ position) % 320 + 320*(v_cnt>>1))% 76800;  //640*480 --> 320*240

always @ (posedge clk or posedge rst) begin
   if(rst) begin
       position <= 0;
   end
   else begin
       if (en) begin
           if (dir == 0) begin
               if (position < 319) position <= position + 1;
               else position <= 0;
           end
           else begin
               if (position > 0) position <= position - 1;
               else position <= 319;
           end
       end
       else begin
           position <= position;
       end
   end
```

lab07_2

lab07_2，分成 split 和 shift 兩種功能，也都是以 SampleCode 下去做修改，主要是更改 position 與 valid 的部分，以下會對兩種功能分別敘述，並說明改了各自改了什麼地方。

Shift 功能是黑幕漸漸向左延伸，再進行下移，圖片漸漸由上面顯現，配合黑幕漸漸變少，因此我分成兩個 state(shift_left, shift_down)下去實作。

不管是 shift_left 還是 shift_down 中，圖片本身是沒有進行移動的，在變化的只有黑幕的大小與移動方向，因此在 shift_left 中，pixel_addr 的地方不需要做更改，但是在 valid 的部分，為控制黑幕的部分，shift_left 中黑幕是從螢幕的右邊開始，慢慢延伸到左邊，因此對於 pixel_cnt(寬的部分)來說，範圍應該要漸漸的縮小，因此利用將 position 進行累加，再將範圍設定為 319-position，就可以達到 valid 為 1 的部分越來越小，並且黑幕的部分從右邊延伸，越來越多。

在 shift_down 的部分，與 shift_left 不同的是方向以及黑幕是由多到少，因此 pixel_addr 一樣不需做任何的更動，在 line_cnt(高的部分)來說，範圍應該要漸漸變大至整張圖片的寬度，因此一樣使用了 position 進行累加，再將範圍設定為 position，這樣範圍會因 position 的累加變的更大，讓黑幕從上方漸漸消失，圖片也因此慢慢顯現出來。

```
default begin
    pixel_addr = ((h_cnt>>1)+320*(v_cnt>>1)) % 76800;
end
```

```
`shift_left : begin
    if (position < 319) begin
        next_position = position + 1;
        next_state = `shift_left;
    end
    else begin
        next_state = `shift_down;
    end
end
```

```
`shift_down : begin
    if (position2 < 239) begin
        next_position2 = position2 + 1;
        next_state = `shift_down;
    end
    else begin
        next_state = `init;
    end
end
```

```
`shift_left : begin
    valid = ((pixel_cnt < HD) && (line_cnt < VD) && (pixel_cnt >> 1 < 319-pos));
end
`shift_down : begin
    valid = ((pixel_cnt < HD) && (line_cnt < VD) && (line_cnt >> 1 < pos));
end
```

Split 功能為將照片分成上下左右四部分，依序進行上移、右移、左移、下移的動作，因為同時有左右移與上下移，因此在 position 部分分為 position 與 position2 分別作為寬與高的 counter，而且因為移動的部分都只有整張圖片的一半，所以分別只要累加到 159 和 119，在判斷是上下左右何種移動，來決定是要加減寬、高何種 position。

pixel_addr 的部分，根據跑到的 h_cnt 和 v_cnt 的位置，來判斷是四塊圖片中的哪一塊，在進行相對應的移動，以左上塊的圖片為例，h_cnt 範圍會落在 159 以內，且 v_cnt 會落在 119 內，所要進行的圖片上移動作，就是在加上 240*position2 即可，其他塊也是依照相同的概念下去實作。

Valid 黑幕的部分，也需要分成四塊圖片下去做判斷，以左上那塊為例，黑幕要隨著圖片上移而擴增黑幕的範圍，也就是說 valid 的範圍要越來越小，因此我們只需將圖片寬度的一半，扣除 position，就可以讓範圍越來越小，黑幕自然會越來越多。

```
`SPLIT : begin
    valid = ((pixel_cnt < HD) && (line_cnt < VD));
    if(pixel_cnt>>1 < 159 && line_cnt>>1 < 119) // left_up dir = up
        valid = ((pixel_cnt < HD) && (line_cnt < VD) && (line_cnt>>1 < 119-pos));
    else if(pixel_cnt>>1 > 159 && line_cnt>>1 < 119) // right_up dir = right
        valid = ((pixel_cnt < HD) && (line_cnt < VD) && (pixel_cnt>>1 > 160+pos));
    else if(pixel_cnt>>1 < 159 && line_cnt>>1 > 119) //left_down dir = left
        valid = ((pixel_cnt < HD) && (line_cnt < VD) && (pixel_cnt>>1 < 159-pos));
    else // right_down dir = down
        valid = ((pixel_cnt < HD) && (line_cnt < VD) && (line_cnt>>1 > 120+pos));
end
```

```
`SPLIT : begin
    if(h_cnt>>1 < 159 && v_cnt>>1 < 119) // left_up dir = up
        pixel_addr = ((h_cnt>>1) + 320*(v_cnt>>1) + 320*position2) % 76800;
    else if(h_cnt>>1 > 159 && v_cnt>>1 < 119)  // right_up dir = right
        pixel_addr = ((h_cnt>>1) + 320*(v_cnt>>1) - position) % 76800;
    else if(h_cnt>>1 < 159 && v_cnt>>1 > 119) //left_down dir = left
        pixel_addr = ((h_cnt>>1) + 320*(v_cnt>>1) + position) % 76800;
    else // right_down dir = down
        pixel_addr =  ((h_cnt>>1) + 320*(v_cnt>>1) - 320*position2) % 76800;
end
```

因為 pixel_addr 和 valid 屬於兩個不同的 module 中，所以我有額外接兩條 output，分別為 pos 和 state 出去至 VGA controller，讓 valid 可以有對應的值去做判斷。pos 的部分，會依照當時的範圍，來決定 output 代表寬的 position 還是代表高的 position2。

3.學習到的東西與困難

這次在理解 SampleCode 中，花費了蠻多的時間，因為有不同 module 的相互連接，還有圖片處理縮小的部分，在與同學互相討論請教後，才對於程式碼更加了解。lab07_1 中的圖片，一開始發現圖片在經過一段時間的移動後，都會由下移的狀況發生，但是自己想了很久還是 de 不出 bug，後來是在請教同學後，才了解到在加到最右邊的邊界時，會進位到下一排去，因此要進行取餘數的動作，我也才發現自己對於 code 的掌握度還不夠好，又在仔細研究了一番。

lab07_2 中，比較麻煩的是在調整邊界的部分，有時候差個 1 就會導致圖片分裂時有些微的誤差，出現不正確的結果，但往往稍微改個程式碼，就要再花上 3-4 分鐘在等待燒板子，因此能夠耐心的等待結果，我覺得在此次 lab 中，也是十分重要的能力。