| EECS 2070 02 Digital Design Labs 2019  Lab 7 |
| --- |
| 學號：107000115 姓名：林珈卉 |

1. 實作過程
   1. lab07\_1
      1. 設計概念
         1. 需要讓圖片可以左右移動
         2. 可以使用 en 跟 dir 來控制是否移動以及移動的方向
         3. 圖片移動時，左右都會連接著同一張圖片一起移動
      2. 實作方法
         1. Top module 會呼叫 mem\_addr\_gen, clock\_divisor, blk\_mem\_gen\_0 及 vga\_controller
         2. mem\_addr\_gen 負責將每個 pixel 應該要放的顏色與圖片對應位置的編號傳給 top module
         3. 圖片大小為 320 \* 240，所以 pixel\_addr 會介於 0 和 76800（不含）之間
         4. 將 top module 的訊號輸入 mem\_addr\_gen 作為 input wire，以控制圖片移動的方式
         5. 宣告變數 reg [8:0] position，記錄圖片移動了幾個 pixel
         6. 設計一個 always block，以 posedge clk or posedge rst 去 trigger
            1. 若 rst==1，position = 0
            2. 使用 en 來判斷，若為 0 則 position = position，即不移動
            3. en 為 1 時，則判斷 dir

若 dir==0，圖片向右移動

若 position < 319，則 position = position +1，因為 position 的範圍是 0 至 319

否則 position = 0，回到圖片沒有位移的狀況

若 dir==1，圖片向左移動

若 position > 0，則 position = position - 1

否則 position = 0

* + - 1. pixel\_addr 則 assign 為 ((h\_cnt>>1) + position) + 320\*(v\_cnt>>1))
         1. 但須先判斷 ((h\_cnt>>1) + position 有沒有超過 319，若有就必須先減去 320
  1. lab07\_2
     1. 設計概念
        1. 設計出兩個模式，shift 及 split
        2. shift 模式可以讓圖片由右而左被黑屏，待全部消失後再由上往下出現
        3. split 模式讓圖片分成四塊，每一塊朝不同的方向向外移動，移動出去後的背景為黑底，直至全部都移動到屏幕外
     2. 實作方法
        1. 由於 shift 與 split 的訊號是用按鈕控制，兩個訊號皆須經過 debounce 及 one\_pulse
        2. 由於 shift 的模式會先由右到左消失，再從上到下出現，我宣告了一個變數 appearing 來判斷現在是消失中 (0) 還是出現中 (1)
        3. 設計一個 finite state machine
           1. state 分為三個，init、shifting 及 splitting
        4. 設計三個 always block
           1. 一個 always block 由 posedge clk or posedge rst trigger

rst 時，state = init，position = 0，appearing = 0

否則 state = next\_state，position = next\_position，next\_appearing = appearing

* + - * 1. 第二個 always block 則控制 next 的值

以 case 判斷 state

init state

判斷是否按下 shift 按鈕，若有則 next\_state = shifting，next\_position = 319（最右邊開始），next\_appearing = 0（先消失）

判斷是否按下 split 按鈕，若有則 next\_state = splitting，next\_position = 0

shifting state

判斷 appearing 的值

appearing 為 0，代表現在正在由右向左消失

position > 0 的話，next\_position = position - 1

否則 next\_appearing = 1，next\_position = 0

appearing 為 1，代表現在正在由上往下出現

position <319 的話，next\_position = position + 1

否則 next\_state 回到 init，next\_position = 0

splitting state

若 position < 159，next\_position = position + 1（位移量）

否則 next\_state 回到 init，next\_position = 0

* + - * 1. 宣告 output reg 變數 black，若 black == 1 則代表該 pixel 需顯示黑色
        2. top mdule 修改成 assign {vgaRed, vgaGreen, vgaBlue} = (valid==1'b1 && black==0) ? pixel:12'h0;
        3. 第三個 always block 則控制 output reg 訊號 pixel\_addr

init state

black = 0

pixel\_addr = (h\_cnt>>1) + 320\*(v\_cnt>>1)

shifting state

由右向左消失時，appearing == 0

若 h\_cnt>>1 > position，black = 1

否則 black = 0，pixel\_addr = (h\_cnt>>1) + 320\*(v\_cnt>>1)

由上往下出現時，appearing == 1

若 v\_cnt>>1 > position，black = 1

否則 black = 0，pixel\_addr = (h\_cnt>>1) + 320\*(v\_cnt>>1

* + - * 1. splitting state

根據 h\_cnt 及 v\_cnt 分為四塊判斷

如果在位移量 position 以內，black = 1

否則改變 pixel\_addr 的算式

左上為 (h\_cnt>>1) + 320\*((v\_cnt>>1) + position)

左下為 (h\_cnt>>1) + position + 320\*(v\_cnt>>1)

右上為 (h\_cnt>>1) + 320\*(v\_cnt>>1) - position

右下為 (h\_cnt>>1) + 320\*((v\_cnt>>1) - position)

1. 學到的東西與遇到的困難
   1. lab07\_1 的圖片會往上一個 pixel，且使用 %320 時會出問題，所以需要特別判斷當 ((h\_cnt>>1) + position) > 319 時要減去 320
   2. lab07\_2 在判斷 splitting state 左上的 pixel\_addr 時，會遇到 120-position overfloaw 的問題，必須要先判斷 position >= 120 時， black = 1
2. 想對老師或助教說的話
   1. Sample code 提供的範例圖片讓人看了很心寒（畢竟二退時間還沒過...），下一屆可不可以換一張圖片給學弟妹，才不會打擊太大ಥ\_ಥ