Final Project Report

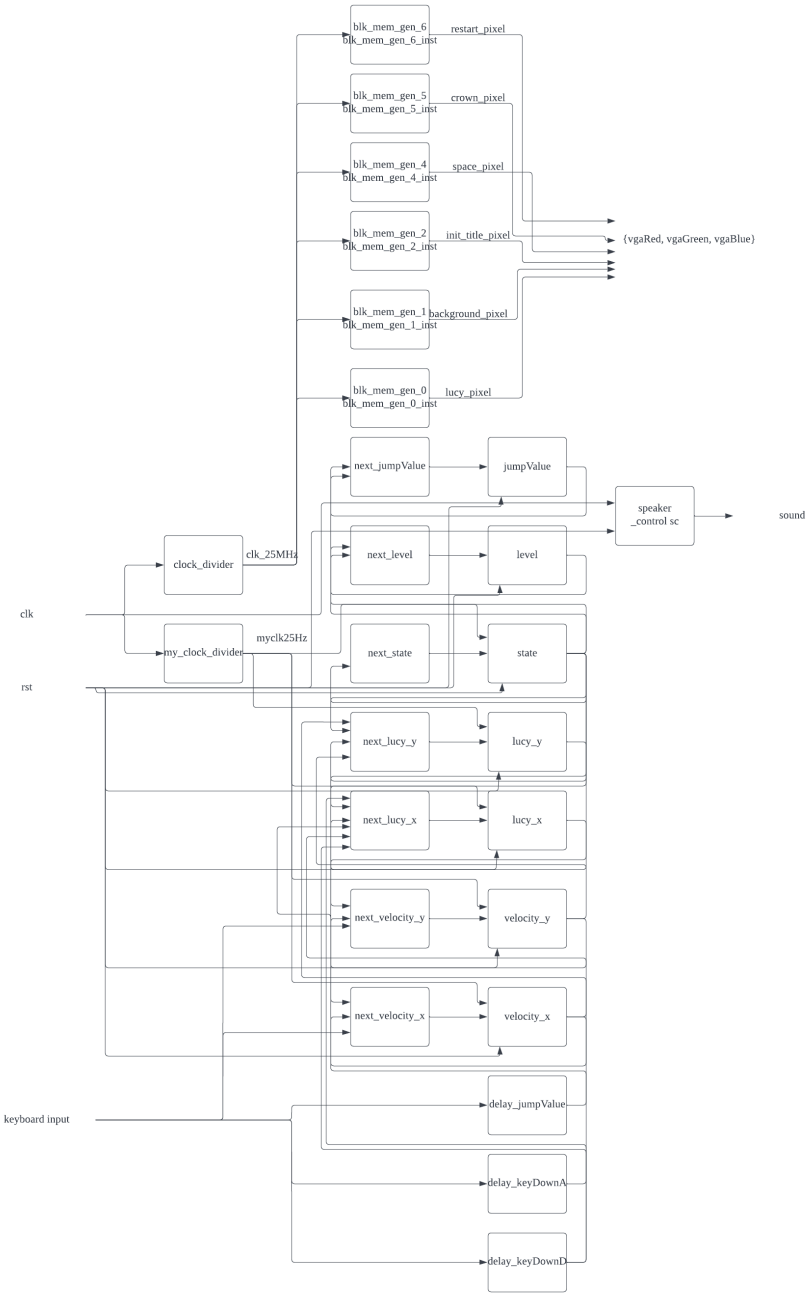
學號: 110062334 姓名: 周峻平

學號: 110062131 姓名: 馬毓昇

A. 設計概念

利用 FPGA 板還原 Jump King，模擬其遊戲機制。

B. Block Diagram



C. 架構細節

遊戲 FSM

整個遊戲我們是用 FSM 來架構的，總共有 3 個 state：INIT、GAME、END。按下空白鍵後就會進入 GAME state，可以開始進行遊戲。當角色的圖案覆蓋在皇冠上時，就代表贏了，會進入 END state，這時只要按下 R 鍵就可以誠心開始遊戲。

角色移動

給角色設定 x 方向速度、y 方向速度、x 方向位置、y 方向位置。
依據：加速度 x 時間=速度變化 與 速度 x 時間=位置變化 來更新角色位置。
左移：按下左鍵，給角色一個向左（-x）的速度
右移：按下右鍵，給角色一個向右（+x）的速度
跳躍：按下跳躍建會累積"junpValue"（register counter），鬆開跳躍鍵時會依據 jumpValue 來決定要給角色多少向上（-y）的速度
落下：當在空中時，角色會總是受到一個向下（+y）的加速度，依據上述公式就會達到烙下的效果。落地後，角色變不再受此加速度影響
此外，為了限制角色在空中不能再次跳躍或左右移動、蓄力時也不能左右移動，所以有多判斷 isGrounded 與 key_down[“跳躍鍵”]來判斷 1.~4.可行與否

碰撞

根據角色碰撞箱及所有障礙物碰撞箱去判斷，當角色的下一次位置與任一障礙物重疊時，會將角色下一次位置控制在剛好貼齊障礙物的位置同時，將角色目前的速度方向設相反（ex:在空中要撞進右邊牆壁的話，下一次會顯示成貼在右邊牆壁上，並且把速度改為向左）。

地圖轉換

一張地圖的 y_size 是 240，三張是 720，角色的合法 y 位置便是 0~719。判斷角色位置目前位於哪張地圖內（0~239、240~479、480~719），以 offset 的方式來決定目前 VGA 應該要輸出哪一張地圖的外觀。

音樂

音樂就上網抓譜然後用 py 轉成 verilog code 以 counter determine 的邏輯去播音樂。

角色動畫及螢幕顯示

在處理角色移動動畫部分，我們有一張上面有角色所有動作的圖，這些圖是用原版的圖去做些微改變的。原版遊戲的角色圖更多，而且像素大小是 32*32，而我們為了讓這些圖可以用在 FPGA 板上，必須把它改成 24*16，還要把圖的細節用小畫家作處理，否則會變得很模糊。

至於要用角色動作圖的哪個區塊時，我是用一個 reg search 去尋找，在我按下相對應的指令或角色處於某種狀態時，這個 search 就會改變。另外，當按下左右移動鍵時，我有用一個

counter 去數，當它數到某些數字時就會更新 search 的值，這樣就可以有左右移動動畫的效果。接著用 searh 的值來得到我們要取角色圖的 row 和 col，例如當 search 為 7 時，我們要的圖就是第一行第二列的那個。在生成角色的 pixel_addr 時就跟用 lab7 一樣，利用 row 和 col 算出來。其他的圖片，包刮背景、皇冠、提示字樣等等，就只是單純的讀圖來得到 pixel_addr。

讀完圖片檔並生成他們的 pixel_addr 後，我們要判斷該讓螢幕上的哪個部分顯示什麼。我用了一個 reg print_what 來判斷，當螢幕的 h_cnt 和 v_cnt 數到要顯示的圖的範圍時就給 print_what 一個值，這個值會決定要賦予哪張圖的 pixel 給{vgaRed, vgaGreen, vgaBlue}。提示字樣閃爍的部分也是在這邊作的，用一個 counter 去數，數到某些數字時就讓 print_what == PRINT_TITLE，否則就讓 print_what == PRINT_BACKGROUND。

在決定好螢幕上哪個部份該顯示哪張圖後，我們要將被圖片的背景去背。像角色圖的話我們是給他一個判定，如果角色的 pixel 為 12'hFFF(白色)時，就讓{vgaRed, vgaGreen, vgaBlue} = background_pixel；提示字樣的部分則是當他們的 pixel 為 12'h000 時讓{vgaRed, vgaGreen, vgaBlue} = background_pixel，其他顏色就顯示他們自己的 pixel。

D. 實作完成度

跟我們原本的目標一樣，有成功還原遊戲，但有些地方由於某些原因我們尚未完成。

1. 地圖只有三層，且沒有背景動畫：

   我們的 BRAM 已經用了 95%，沒辦法再做出更多遊戲畫面了。

2. 沒有特殊地形：

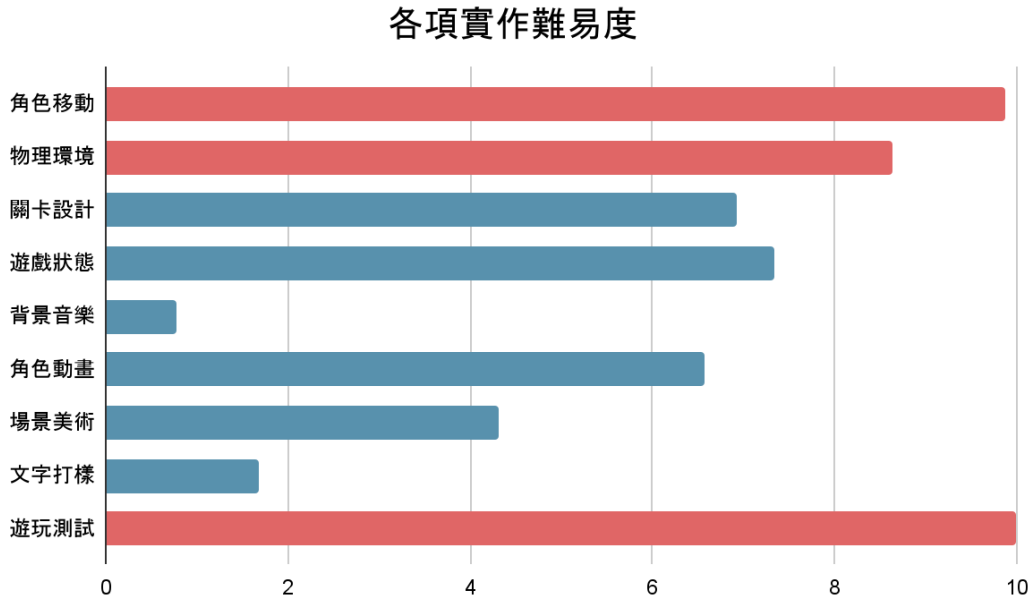
   這方面是因為時做難度太高，沒辦法做出斜坡的效果。

3. 只有背景音樂，沒有音效：

   這是因為聲音處力技術受限，沒辦法做出像是跳躍、摔落的音效。

E. 難易度說明

角色移動的部分需要考慮 jump buffer，而物理環境的彈性碰撞也比較複雜，因此難度較高。遊玩測試難度也很高，我們一開始花了一小才通關。



F. 分工

馬毓昇：角色移動機制、遊戲物理環境、背景音樂還原、關卡場地設計

周峻平：角色移動動畫、標籤字樣打印、關卡場景美術、遊戲轉換場景

G. 困難與解決方法

在遊戲設計的領域中，角色控制有個很重要的概念是 “Jump Buffer” ，讓角色在快接近地面時就能接收跳躍指令，並在落地的那瞬間起跳。用意就是為了避免玩家在還沒有完全落地前的那一小段時間就按下空白鍵，被程式忽略而不起跳了。

在我們的專案中，角色左跳或右跳是根據玩家在起跳瞬間（放開跳躍鍵）有沒有按下方向鍵來決定。而當玩家想讓角色小幅度的向左或右跳時，玩家會同時按下並放開跳躍及方向鍵。如果沒有引入 “Jump Buffer” 的設計，程式往往會認為玩家是很快速地先放開方向鍵才放開跳躍鍵，這樣只會讓角色橫向移動一小步再原地向上起跳，並非是玩家預想中的小幅度左跳或右跳。那當我們引入 “Jump Buffer” 的設計時，就不會出現這樣的問題了，讓玩家在操作上不再像以往那麼彆扭。

至於實作 “Jump Buffer” 的方式，就是利用到了 delay register，來延續鍵盤方向鍵的輸入訊號，在放開跳躍鍵決定起跳方向時，不只考慮到當下的方向鍵輸入，還同時考慮到此前所延續的訊號。這樣的小細節便能讓玩家的操作變得更加地順暢。

由於我們的螢幕長寬比為 4:3，所以如果讓我們的角色圖為正方形的 16*16，角色會有一個被壓扁的感覺。這是因為螢幕上每格由 h_cnt 和 v_cnt 組成的小方塊的比例都是 4:3。要解決這個問題就只要把我們角色圖的長寬比變為 3:4 就好，這樣看起來就會是正方形。

我們為了讓角色可以看起來更清楚，我們在生成角色的 pixel_addr 時並不是把 h_cnt 和 v_cnt 除以二再作運算，而是用原本的 h_cnt 和 v_cnt，並把角色的 x 座標和 y 座標乘以二再作運算。可以由下方部分 code 來了解。

角色 pixel_addr：

```
always @*
begin
    lucy_pixel_addr = ((h_cnt) - (lucy_x << 1) - (MAP_INIT_X << 1) + 18 * col + 72 * ((v_cnt) + (offset_y << 1) - (lucy_y << 1) + 24 * row));
end
```

地圖 pixel_addr：

```
always @*
begin
    background_pixel_addr = ((h_cnt >> 1) - MAP_INIT_X + 160 * ((v_cnt >> 1) + offset_y));
end
```

H. 心得討論

馬毓昇：

期末專題是邏輯設計課程的重要組成部分，它讓我們能夠將所學知識應用到實際項目中。我與周峻平組成小組，選擇一個 VGA 項目設計並實現。通過這個專題，我們學習到了如何在限制的時間和資源內完成一個項目，並且學習到了如何協調和溝通團隊成員。我們還學習到了如何解決實際問題和如何改進我們的設計。我們的專題是一個 VGA 遊戲，我們使用 VGA 實現了顯示圖形和動畫，並且使用鍵盤作為遊戲控制器。這個專題結合了我們在邏輯設計課程中學習到的知識，尤其是 Verilog 的撰寫。總而言之，期末專題是一個非常有意義的經驗，它讓我們能夠更好地應用所學知識，並且增強了我們的團隊合作能力。

周峻平：

在做這個期末專題時，我覺得整個過程都很有趣並且不會有排斥的感覺。透過這個學期學習到的 VGA 顯示器和音效模組來還原一個遊戲，讓我感覺非常充實，在完成的時候相當有成就感。這也是我們第一次從零開始自己做的 project，對我們來說意義非凡。另外，透過這次的 project，我感覺到了團體 project 和個人 project 的不一樣。個人 project 只需要一直自己做就好，全部的想法都由自己實現；而團體 project 我們必須要融合彼此的想法，討論和分工非常重要，團體合作更是完成 project 的關鍵。