|  |  |
| --- | --- |
| **Final Project** | |
| 學號: 109062174、108032053 | 姓名: 謝承恩、陳凱揚 |

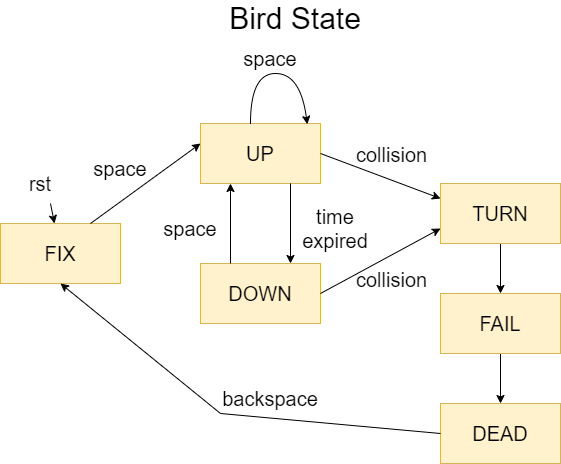
1. 設計概念：Flappy bird

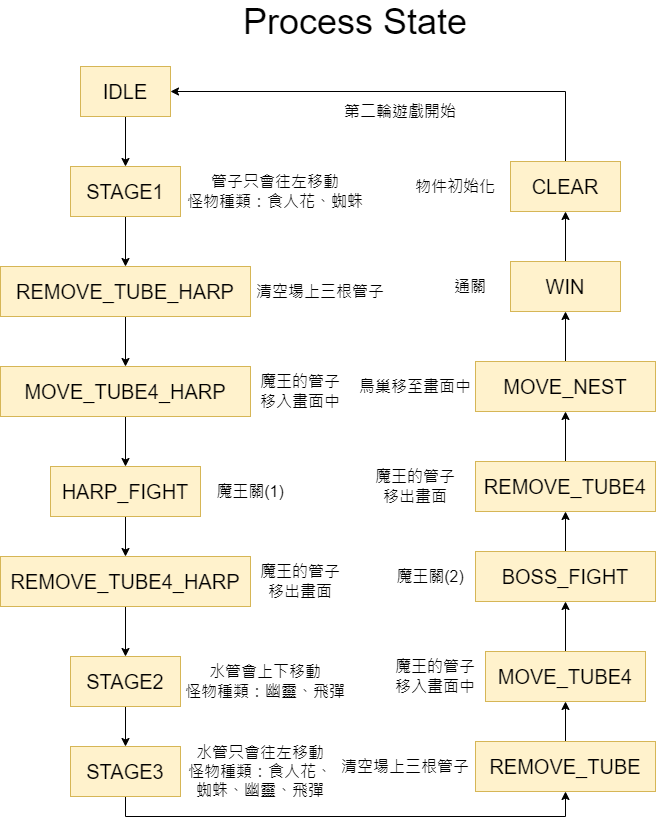


使用者透過空白鍵控制小鳥的高度，讓小鳥不斷通過水管，途中閃避各種怪物的攻擊，最後回到自己的鳥巢，若中途小鳥死掉或成功通關可以按backspace開啟下一輪遊戲。

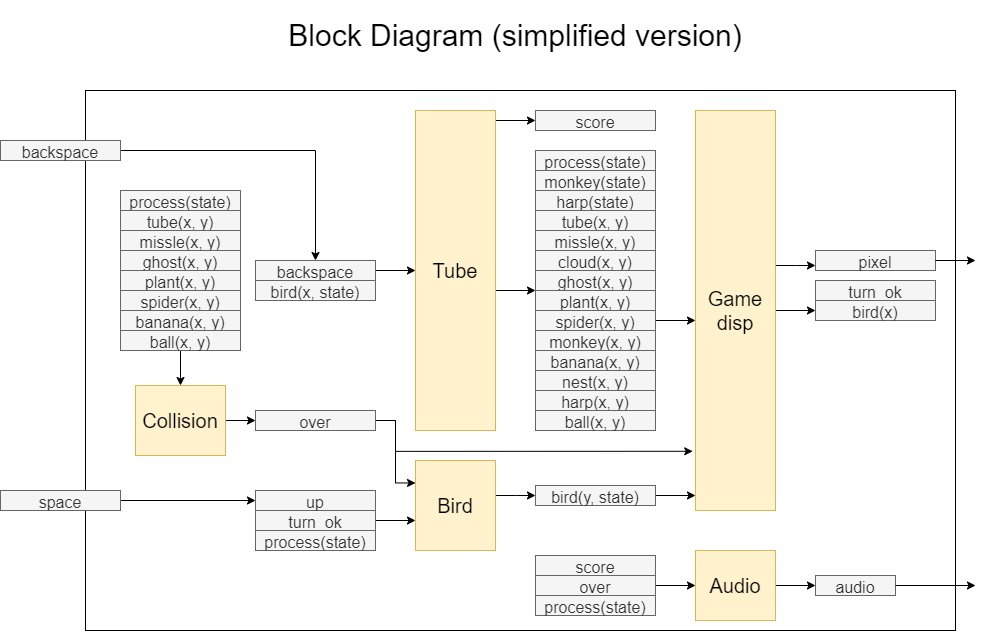
1. 架構細節與方塊圖：

FSM：





Block\_diagram：



1. Bird.v

這個模組用來控制鳥的 y 座標，鳥的飛行狀態分為三種：UP, DOWN, FIX

分別對應到鳥在上升、下降及高度不變，上升的效果，我們的做法就是用一個counter (move\_up\_cnt) 來計時，每當數到一個常數時就將鳥的y座標遞減1(往上移)，下降的原理同上，但由於鳥在上升時我們想呈現短暫加速的效果，因此UP\_SEC會比DOWN\_SEC還小，另外STATE\_SEC是用來控制鳥往上升的時間。

一張含有 文字 的圖片

自動產生的描述

另外我們有做防止鳥超出螢幕上下邊界的設計

一張含有 文字 的圖片

自動產生的描述 一張含有 文字 的圖片

自動產生的描述

UP state的程式碼(DOWN state 原理相同)：

一張含有 文字 的圖片

自動產生的描述

除了上面三種狀態，鳥還有三種狀態用來呈現死亡的動畫，分別是TURN、FALL、DEAD，TURN state僅是用來告知圖片應該轉向，FALL state就是讓鳥不斷落下直到直面，DEAD state不需要對鳥的y座標做處理，只需等待使用者按下backspace鍵後，遊戲回復到初始畫面，將鳥的y座標設定成螢幕中間(240 = 480 / 2) 即可

一張含有 文字 的圖片

自動產生的描述

接下來說明鳥的FSM：

FIX state：

當鳥在FIX state 的時候(遊戲初始化面)，按下空白鍵就會進入UP state

一張含有 文字 的圖片

自動產生的描述

UP state:

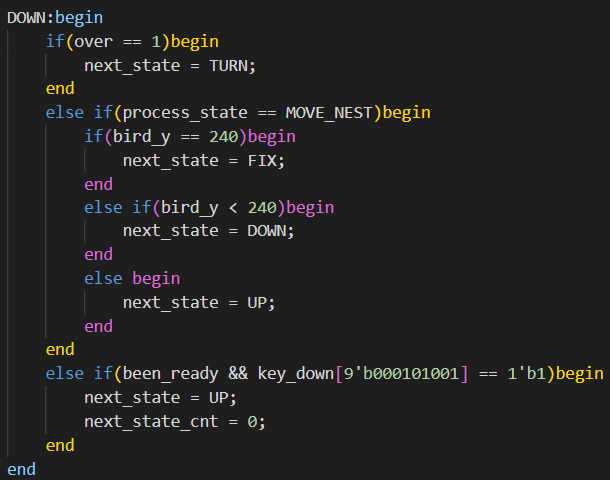
這個狀態持續一小段時間後就會自動進入DOWN state，因為我們的設計是鳥會自動落下，如果在UP state按下空白鍵就會更新counter的值，但如果遊戲結束(鳥撞到東西)，就會進入TURN state，或者如果遊戲通關的話，鳥要飛回鳥巢，因此鳥巢的位置是在螢幕正中間，此時就控制鳥的y座標直到240的位置。

一張含有 文字 的圖片

自動產生的描述

DOWN state:

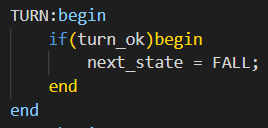
若玩家沒有按下空白，則一直維持在DOWN state，其他部分和UP state 一樣，如果遊戲結束，就會進入TURN state，或者如果遊戲通關的話，鳥要飛回鳥巢，此時就控制鳥的y座標直到240的位置



TURN state

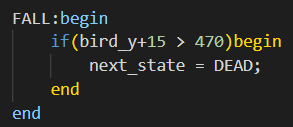
進到TURN state 後，Gamp\_disp module會將小鳥的圖案轉成斜45度，當小鳥的圖案

轉好後，會輸出一個 turn\_ok 為1的訊號，小鳥的狀態就會進到FALL state



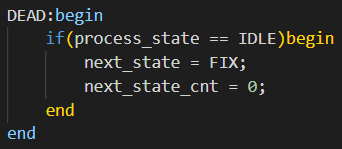
FALL state

讓鳥不斷下降，直到掉落地面( y = 470)，則進入FALL state



DEAD state

如果遊戲回到初始化面的話(IDLE)，則鳥會回到FIX state(鳥保持在 y = 240 的地方，不會隨時間往下掉)

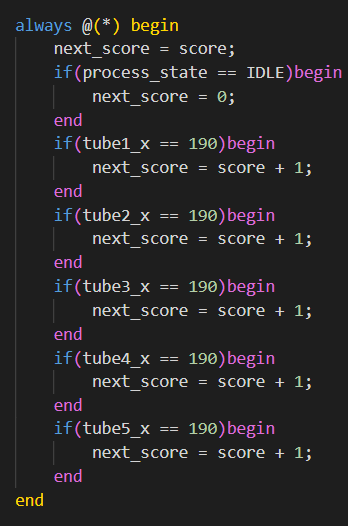


1. Tube.v

這個module有三個功能，第一是用來控制所有物件(除了小鳥)的座標，並將這些座標傳送到Game\_disp module，因為我們最一開始生成的物件是水管，因此取名為Tube.v ，但後來直接將其他物件也加在這個module，第二個功能是紀錄分數，因為我們是用管子的x 座標判斷加分的，因此在這個 module 紀錄分數非常適合，第三是紀錄所有關卡的流程狀態(process\_ state)。

首先解釋加分的機制：

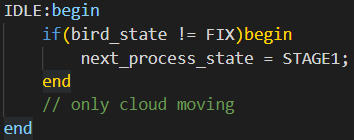
這部份很容易實現，只要每當有管子的x座標等於190，就將分數加1，因為鳥的x座標是固定不變的(除了IDLE state和 MOVE state)，因此水管移動到 x = 190時(鳥的後面)，就代表通過一根水管。



接著解釋關卡流程的部分：

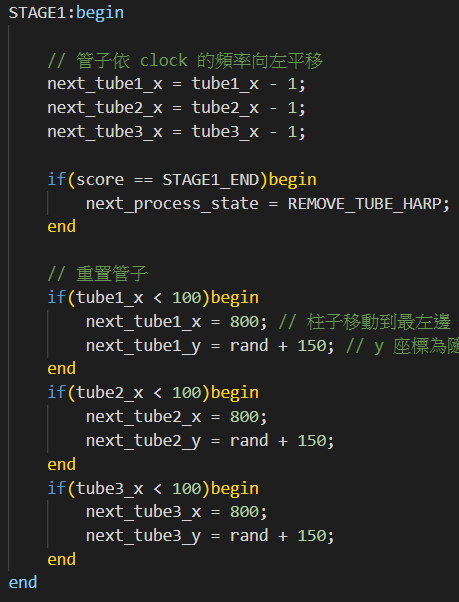
IDLE state：

一開始會在IDLE state(顯示初始畫面)，直到玩家按下第一次空白鍵後，才會進入stage1(管子開始移動)，我們一開始並沒有設計這個 state，因此reset之後遊戲就會直接開始，但玩家可能還沒準備好，顯然這樣的設計不合理。



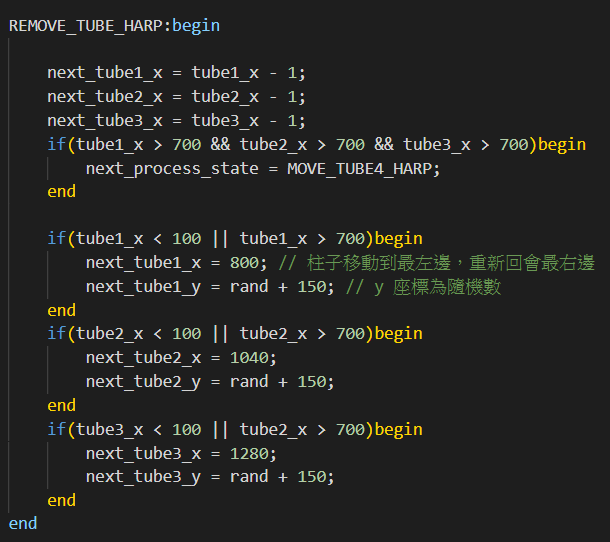
STAGE1 state:

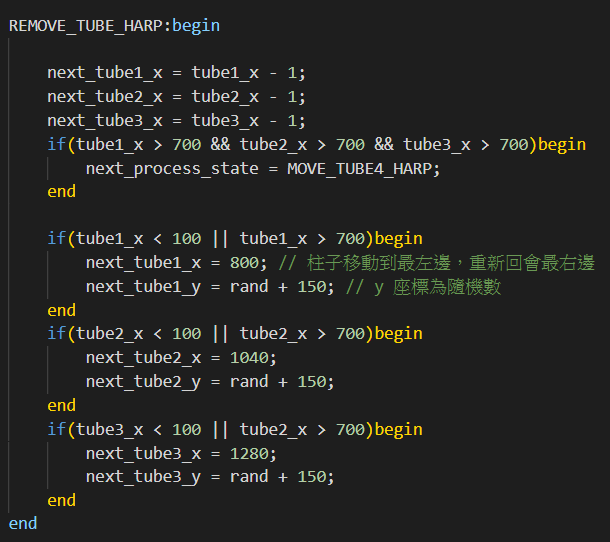
這個狀態水管會開始往左移動，因此不斷讓三個水管的x座標遞減，如果水管的x座標小於100時，我們就重置水管的x座標到800，並且將高度設為一個隨機數，因此可以達到水管不斷出現的效果，當達到某個分數時(自行設定)，就會進入REMOVE\_TUBE\_HARP，準備打第一個魔王。



REMOVE\_TUBE\_HARP：

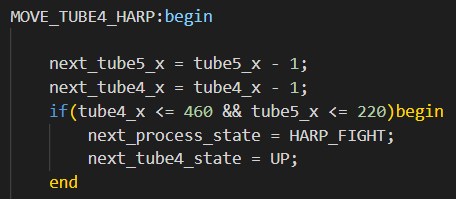
這個階段是魔王關的第一個前置作業，將場上的水管全部清空，因此還是讓水管的 x座標不斷遞減，但是當水管的x座標達到100時，就要將水管的x座標固定在800, 1040, 1280(螢幕外面)，等所有水管移出螢幕後，就會進入MOVE\_TUBE4\_HARP





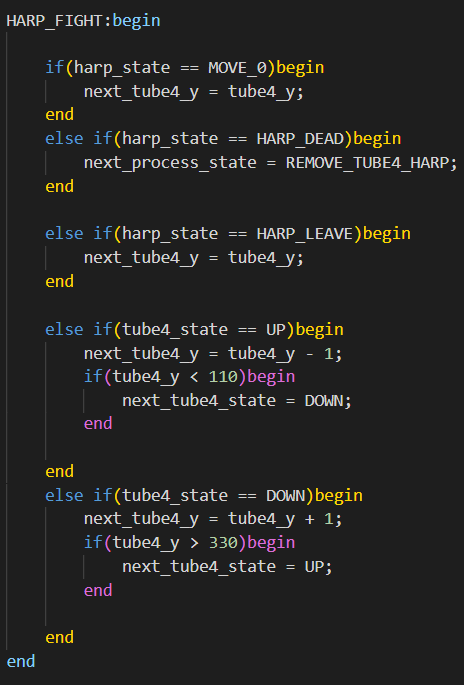
MOVE\_TUBE4\_HARP：

這個狀態是魔王關的第二個前置作業，將玩家的水管(會限制鳥的上下高度)，以及魔王的水管(金色水管)移動到x = 220, x = 460的位置，等兩根水管就定位後，就會進入HARP\_FIGHT，正式開始打魔王



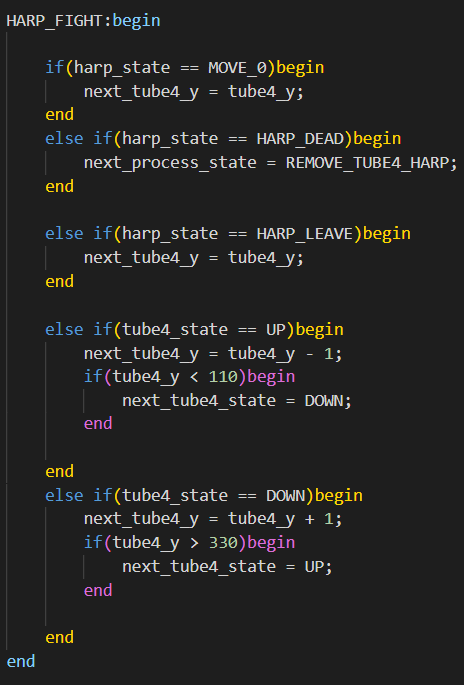
HARP\_FIGHT:

此時魔王的水管會不斷上下移動(魔王跟著移動)，因此tube4有UP, DOWN兩個state



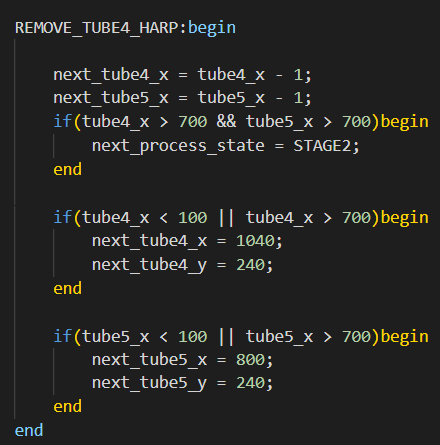
但是水管不能一開始就馬上移動，要先等魔王就定位，因此魔王在MOVE\_0 state的時候，水管的位置固定，此外當魔王要離場時，水管也不能動，因此當魔王在

HARP\_LEAVE state的時候，水管位置也固定，等到魔王完全離開(HARP\_DEAD)，就可以進入REMOVE\_TUBE4\_HARP state。



REMOVE\_TUBE4\_HARP:

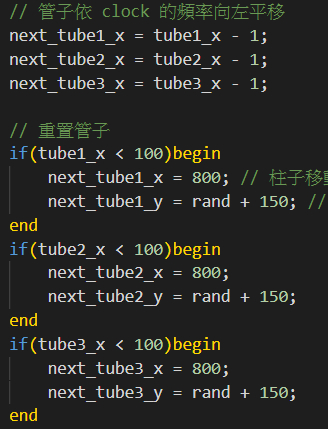
這個狀態用來清場，將場面上的兩根水管(玩家的, 魔王的)移動到螢幕外，之後進入 STAGE2 state



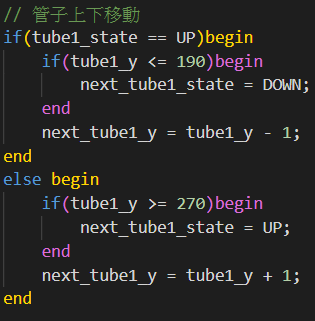
STAGE2:

這個state水管不只要往左移動，還要上下移動

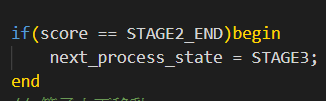
往左移動的部分和剛才STAGE2一樣，這部份已經解釋過



上下移動的部分就和剛才魔王的水管上下移動一樣，每個水管都有一個state，根據UP或DOWN來遞減或遞增y座標，

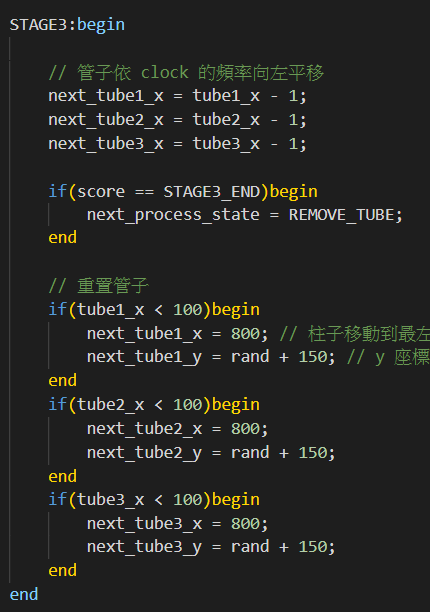


當達到一定分數後，進入STAGE3

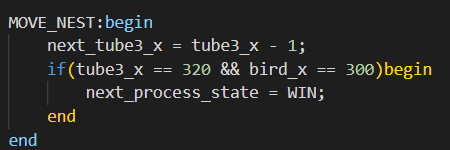


STAGE3:

這個階段做的事情和STAGE1一模一樣，水管只會往左移動



接下來從REMOVE\_TUBE state到 REMOVE\_TUBE4 state是第二個魔王關，這部份得邏輯和剛才第一隻魔王的部分一模一樣，因此就不再重複說明一次了，直接說明MOVE\_NEST state：此時遊戲準備通關，將帶有鳥巢的水管移至 x = 320的位置，等鳥也就定位後(x = 300)就會進入WIN state，



WIN state:

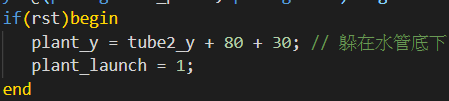
僅用作通知其他module(聲音模組撥放通關音效)，水管本身不需要在移動了。

在上面的任一個狀態中，如果小鳥死掉了，玩家按下backspace後，就會回到IDLE state，或者玩家通關後(WIN state) 按下backspace也會回到IDLE state

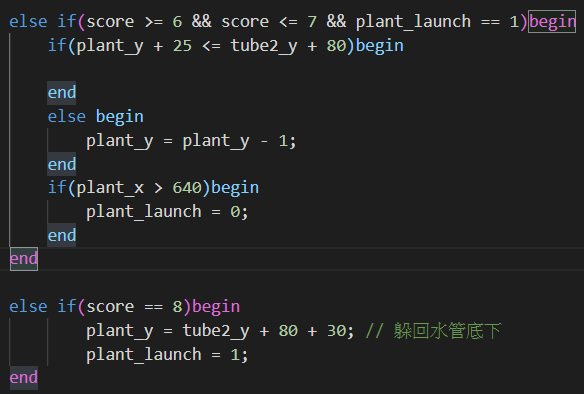
接下來解釋怪物是如何釋放的，我們僅以植物為例作說明：

植物的x座標是綁定其中一根水管，因為植物是從水管中生起，因此x座標和水管的x座標必須相同，y座標的部分，一開始先讓植物隱藏在水管下，並將ready(plant\_launch) 設為1。





當分數在某個區間時(自行設定或隨機決定)且plant\_launch為1時，將植物開始往上移動，植物露出水管，若植物已經隨水管移動到螢幕外(x > 640)，便將plant\_launch設為0，直到分數遞增1後，就將植物移動回水管下面，並將plant\_launch設為1



其餘怪物的原理都差不多，只是差在飛彈是水平移動，植物和蜘蛛是上下移動，幽靈則是垂直加水平移動而已，且飛彈, 幽靈和水管的x座標無關。

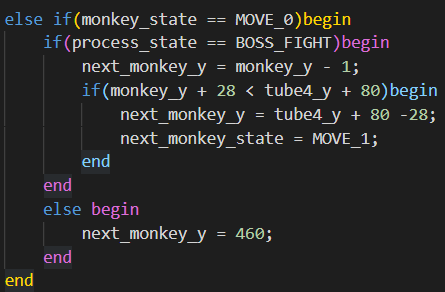
魔王的部分(以monkey為例)：

魔王會有自己的state，用來顯示不同的動作，猴子的動作總共分為10張圖片

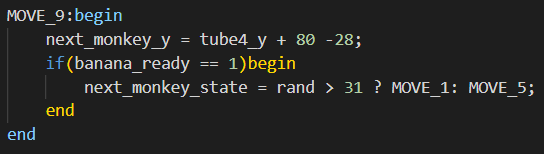
猴子的x 座標和金色水管相同



魔王一開始也是躲在水管下，當process\_state進到BOSS\_FIGHT後，就會從管子中浮出來，並進入到MOVE\_1 state(魔王會開始動作)

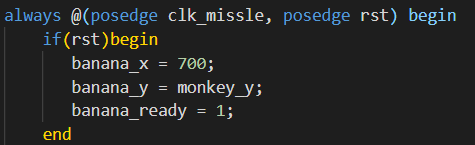


接下來就是不斷依序切換猴子的狀態(圖片)，但為了增加樂趣，猴子可以選擇吃香蕉或丟香蕉，選擇的方式就是根據隨機數的值做決定

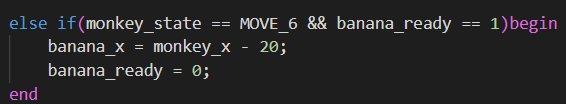
’

接著說明猴子丟香蕉是如何實現的

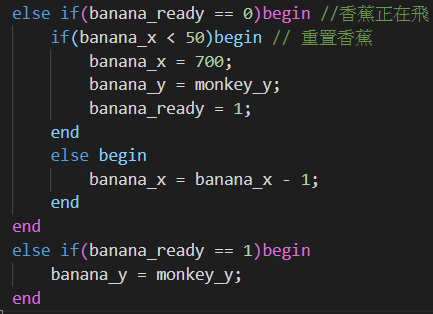
一開始香蕉的x座標是在螢幕外(x = 700)，y座標跟猴子的y座標相同，



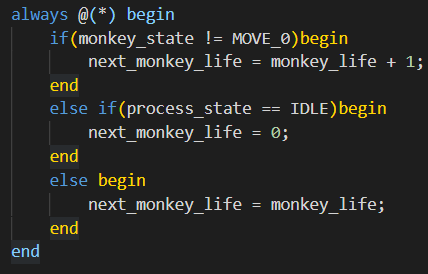
當猴子做出丟香蕉的動作時，香蕉會直接出現在猴子左邊20 pixel的位置



接著香蕉便開始水平移動，此時香蕉的y座標就固定了，x座標不斷遞減，直到飛出邊界



魔王有一個生命值，當魔王關開始後就會開始計時，當到達設定的數字後就會自動死亡



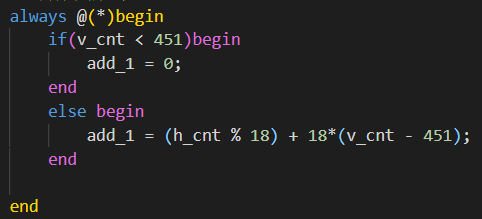
1. Game\_disp.v

這個module用來顯示所有的物件，但由於程式碼實在太過龐大，因此我僅挑選比較特別的部分做說明。

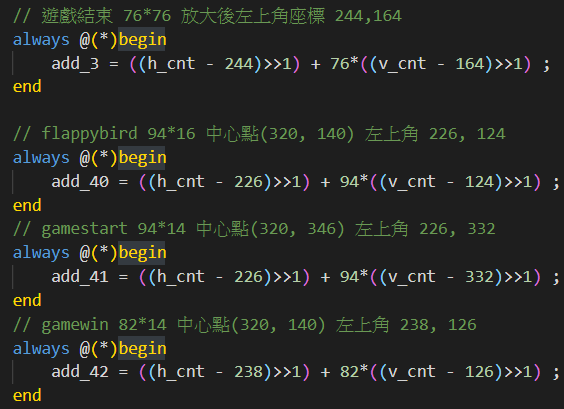
1. 圖案若可以重複畫，就不要儲存

我們的背景圖片只有18\*30 pixel，比小鳥還小，因為背景圖片大多地方都重複，

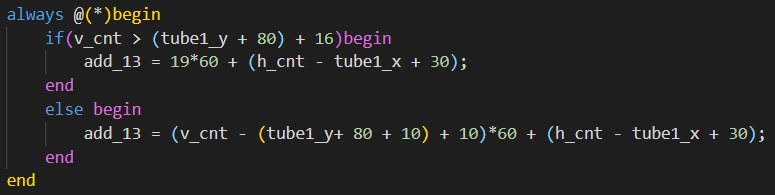
因此不需要儲存整張圖片

’

另外像GAME START ,GAME WIN , GAME OVER等字樣我們也是用程式達到放大4倍的效果，不然記憶體真的會不夠用



水管也是指儲存一小截，其他往上下延伸的部分也是重複畫，但因為我們有水管上下移動的關卡，這個部分就會比較麻煩，但只要有水管的座標，就還是做得到。

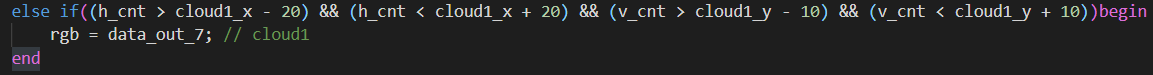


1. 處理物體重疊時的顯示問題：

比如當鳥和雲重疊、鳥和水管重疊、鳥和怪物重疊、鳥和鳥巢重疊、魔王和雲重疊…等，需要做特別處理，因為每張圖案都有自己的背景，若不做特別處理，圖案重疊時圖案的背景色就會顯示出來，因此針對有可能發生重疊的物件，我們都會做處理，方法簡單來說就是當兩種物件顯示的範圍重疊時，

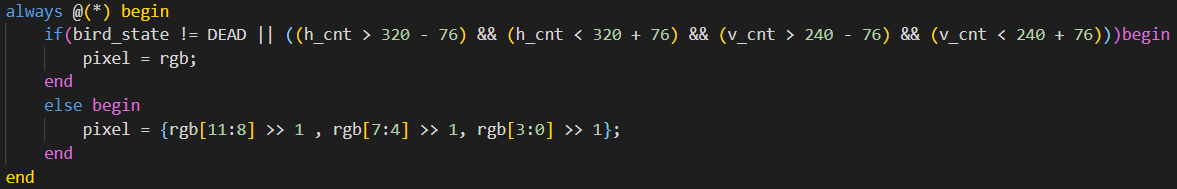
比如小鳥和雲，此時小鳥的背景要變成雲，而不是小鳥原本的背景(白色)，因此當小鳥的記憶體讀到的值為12’hfff (白色)時，就將輸出的pixel接到雲的記憶體，這樣就可以達到我們像要的效果，其他物件的重疊也都是利用這個邏輯





1. Gameover時背景要變暗：

當小鳥死掉後遊戲背景要變暗，並顯示game over的圖(但game over不會變暗)，我的作法是當遊戲結束時就將每個pixel的rgb值「分別」向右shift一個bit，這樣就可以達到變暗的效果，但game over圖片的區域仍然維持原本的顏色



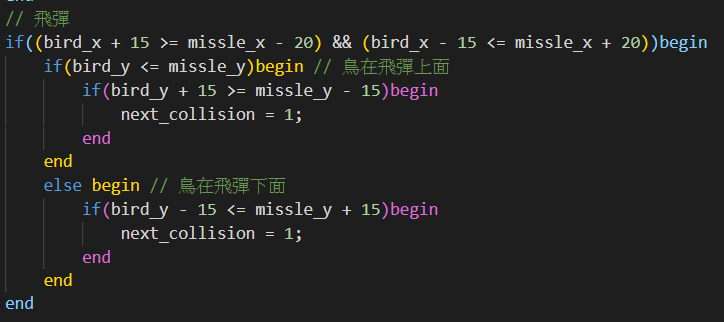
1. 動態的影像生成：

小鳥的翅膀會動、植物的嘴巴會動、幽靈的斗篷會動、魔王會有攻擊動作…等，前三者用一個1 bit counter就可以做到，比如counter == 1 就放第一張圖，counter == 0 就放第二張圖，然後不斷交錯，魔王的動畫就必須搭配魔王的狀態，看目前是哪個狀態就放哪張圖案

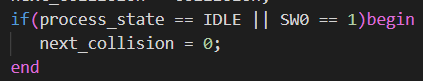
1. Collision.v

用來判斷發生碰撞的module，我們僅以飛彈做說明，因為判斷碰撞的邏輯每種怪物都是一樣的

當小鳥的x座標和飛彈的x座標重疊後()，要先判斷小鳥目前是在飛彈的上方還是下方，若小鳥在飛彈上方，此時若小鳥的底部超過飛彈的頂部，就判斷發生碰撞，若小鳥在飛彈下方，此時若小鳥的頂部超過飛彈的底部，就判斷發生碰撞。

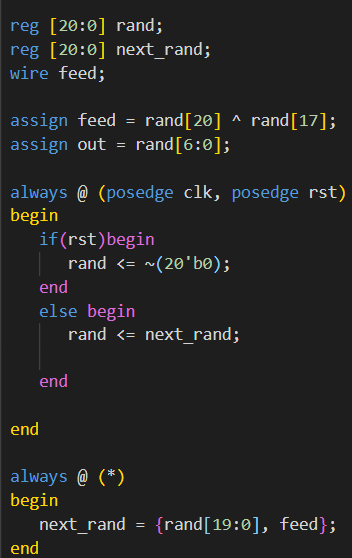


若關卡回到初始狀態(IDLE)，將collision reset成0，另外因為我們有做無碰撞模式，因此當SW0 拉起時，collision就會一直維持0



1. Random.v

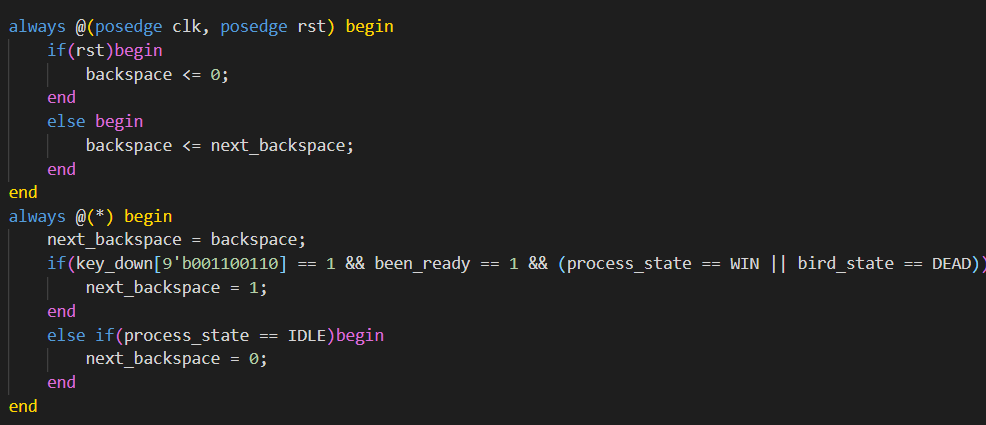
一個非常簡單的亂數產生器，feed為1個bit的值，由rand的其中兩個bit做XOR得到，接著就將feed不斷插入rand的第0個bit並將rand往左shift，並且輸出  
rand [6:0] 作為亂數



1. Top.v

Top module將所有訊號連接，並且處理backspace的輸入，當小鳥死掉或遊戲通關後

使用者按下backspace，此時backspace訊號拉成1，直到process\_state回到IDLE，backspace訊號拉回0

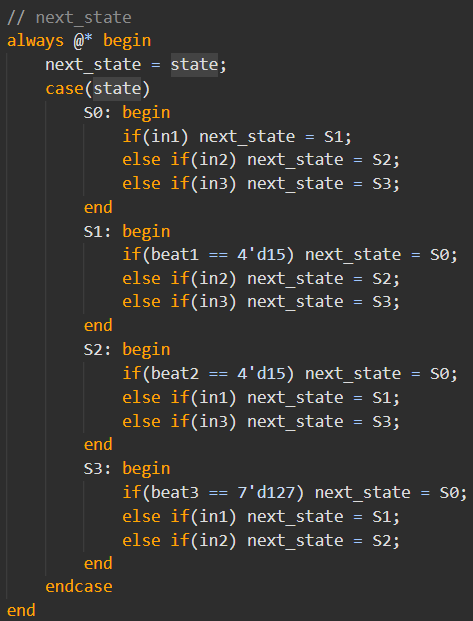
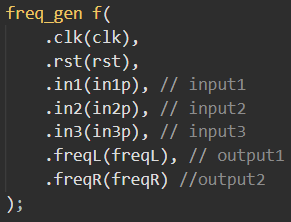


1. Audio.v

在這個module中，提供的功能相當簡單且實用，也就是在平時輸出背景音樂，並隨時接受撥放不同音效的需求，並在音效結束後，繼續撥放背景音樂。而在這個遊戲中，提供了三種音效，分別為得分、碰撞、通關。其架構與lab8大致相似，首先將所有input做debounce和onepulse，接著傳進自製的freq\_gen module中，其會傳回freqL和freqR，最後即可使用note\_gen和speaker\_control等module產出所需的input。

freq\_gen為主要控制聲音狀態的module，裡面有4種state，分別代表撥背景音樂及3個不同音效，由下圖可看出任何音效（S1、S2、S3）都可以打斷其他音效而撥出，並在音效撥完後就會回到S0，也就是不斷循環撥放背景音樂。此外，由於module的簡單架構，使我們可以簡單修改就能一直加入新的音效，也很容易與主結構整合起來，整體上相當方便。

而我們所選用的背景音樂為wii music，我認為他的節奏相當適合於這種電子的像素遊戲中，實際效果也搭配得不錯；得分聲為與原版遊戲相同的簡單兩個音所組成；通關聲為瑪莉歐遊戲的通關聲；最後的碰撞聲反而是較困難的部分，一開始想要模擬出碰撞後的爆炸聲，但發現只靠方波的聲音實在效果有限，後來改為使用一組由多個連續低音所組成的音效，希望製造出失敗的效果，意外的得到不錯的結果。



1. 實作完成度/難易度說明/分工
2. 實作完成度：我們的完成度遠超當初propasal的內容

proposal : 畫面只有鳥和水管，玩家利用空白鍵控制鳥的高度

實際完成：除了上述內容以外，新增以下功能：

* + 背景會有飄動的雲
  + 增加背景音樂、碰撞音效、通關音效
  + 新增遊戲字幕(開始、失敗、通關)
  + 水管上下移動的關卡
  + 增加 4 種怪物、2 種魔王，並且有攻擊的動畫
  + 為遊戲加上結尾(回到鳥巢)
  + 在通關(或game over)後按 backspace 能夠回到初始畫面

1. 難易度說明：

我自己覺得我們的作品不簡單，但我想老師以及助教看完所有人的作品後，心中自有定論，因此這部份我就不多著墨，畢竟應該不會有人說自己的作品很簡單

1. 分工：

109062174 謝承恩

* VGA 顯示

108032053 陳凱揚

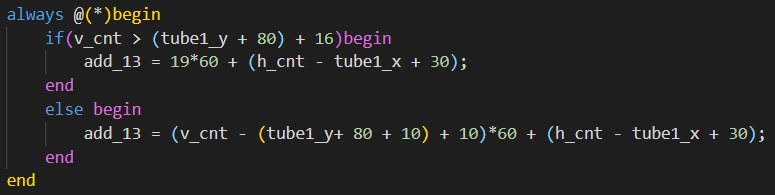
* 音效模組、圖片素材背景處理

4. 測試完整度：

以我們現有的設計來說，整體應該是沒有bug的，至少沒有會影響遊戲體驗的bug，我們在demo前一天不斷測試我們的程式，從晚上6點一直到demo當天的凌晨6點，我們嘗試了各種物體的碰撞是否正常(五根管子、植物、蜘蛛、幽靈、子彈、魔法球、香蕉)，然後嘗試了如果小鳥中途沒有死亡，順利通過第一輪，第二輪所有東西也都可以正常運作，第三輪也沒問題，以及小鳥如果在第一輪中途死掉後，重新開始一輪也沒有問題，接下來第二、三輪也沒有問題，以及開啟無敵模式跑一輪也沒有問題、開啟無敵模式在中途切換成碰撞模式也確實會發生碰撞，以及在各種關卡死掉後按backspace也都能順利回到初始畫面。

5. 困難與解決辦法

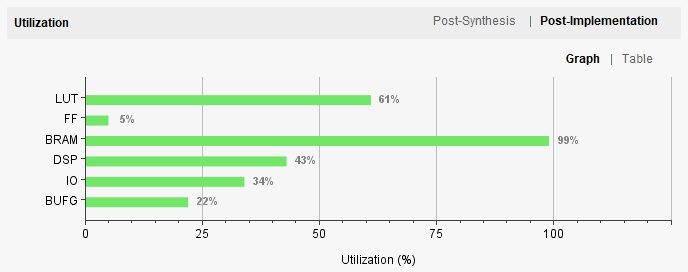
* 1. Vga顯示：我覺得這部份本身就是一個很大的挑戰，因為牽涉到很多物件同時在螢幕上移動，但我覺得這真的就是一步一步慢慢來，從一開始只有一張背景圖案，然後加上一隻鳥，再加上一根水管…，所有東西都是這樣慢慢加上去，到後來越來越熟悉後，要加新的東西就會比較快，
  2. 水管是隨機產生的，但是水管的圖片是固定大小，因此要讓程式動態決定水管顯示的區域，其實只要有水管的y座標就能夠做到，因為有水管的y座標就能夠知道水管開口的位置，等v\_cnt數到水管開口的位置時，就將原圖的水管開口印出，否則就一直重複的印水管壁



* 1. 物件重疊時要讓畫面看起來很流暢：

對於每種可能重疊的物件(不一定是碰撞)，都要做處理，否則重疊時會看起來圖片有破損的感覺，只要當物件發生重疊時，將上層圖像的背景替換成下層物件的圖像，而不是輸出上層圖像原本的背景，然後因為可能發生重疊的物件有蠻多種的，所以要一一去做處理，但邏輯都一樣，所以像顯示小鳥的部分就會多很多判斷式，目的就是為了處理小鳥和其他物件重疊的問題。

* 1. 記憶體空間不足：我們總共是用了51個BRAM，因為有小鳥、水管、怪物(4種)、雲朵(6朵)、魔王(2種)、遊戲字幕(4種) …，此外小鳥又分兩張飛行圖案、兩張死亡圖案，有些怪物要會動，因此也要不同圖案，然後猴子是花最多記憶體的，總共11張圖案，因為要讓牠呈現非常流暢的攻擊動畫，這些東西全部加起來要不少記憶體，但就是用剛才講的方法，可以重複畫的東西就不要儲存，最後我們成功將記憶體使用率控制在99%



* 1. 當遊戲結束後可以回到初始畫面，並初始化所有物件，開始第二輪遊戲：這部分我們真的花很多時間處理，但是我們堅持要將這個功能做出來，因為學期初老師就有說過，你的設計必須要能夠從最後一個狀態回到初始狀態，不能每次遊戲結束都要按reset才能重新開始，這樣是不合理的設計。其實這部分就是要仔細檢查每個module的每個物件、每個變數，在初始狀態應該要在什麼位置、是什麼值，不能有些東西reset了，有些卻還沒，但因為我們的物件真的太多了，過程中一直有一些物件沒有及時被reset，最後花了很多時間才讓所有東西都能夠正確的歸位。

6. 心得討論：

做完這個final project我覺得最大的收穫就是對vga的控制變得非常熟悉，加上我們的程式算是很龐大，每個東西都有自己的狀態(鳥的狀態、關卡的狀態、魔王的狀態… )，要怎麼讓整個關卡流程及每個物件按照我們的想法去進行，真的是一件不容易的事，尤其是遊戲結束後要將所有物件的位置、狀態回復，這部份我們處理了很久，我覺得很多時候問題本身並不是很困難，真正困難的是當程式碼到達4, 5千行的時候，很多東西都會互相影響，這個時候就很難找到真正引發問題的所在，有時候甚至要退回上一個版本重新再寫一遍。但是有了這種比較大型的程式開發經驗，我覺得自己的實力真的提升許多，我對待這個作品的態度是真的把它當作一個遊戲來開發，不僅僅是一個作業而已，我希望能夠做出讓人眼睛為之一亮的作品。

7. 想對老師或助教說的話

真的很感謝老師與助教這麼用心的指導我們，除了上課時間與demo時間外，在eeclass的討論區也一定會回覆我們的問題，整個學期下來收穫非常多，尤其最後做final project的部分，在學期初的時候我並不覺得自己能做出這樣的作品，最後我們突破了原先對自己的期待，看到成品的當下真的有些感動，而lab的部分，我覺得難易度適中，不至於太難，但還是要花一些時間寫，這門課除了讓我在這個領域打下扎實的基礎外，也讓我對數位設計產生興趣，未來我會想繼續修其他相關課程( DIC lab, VLSI system design…)

8. 笑話

一張含有 文字 的圖片

自動產生的描述