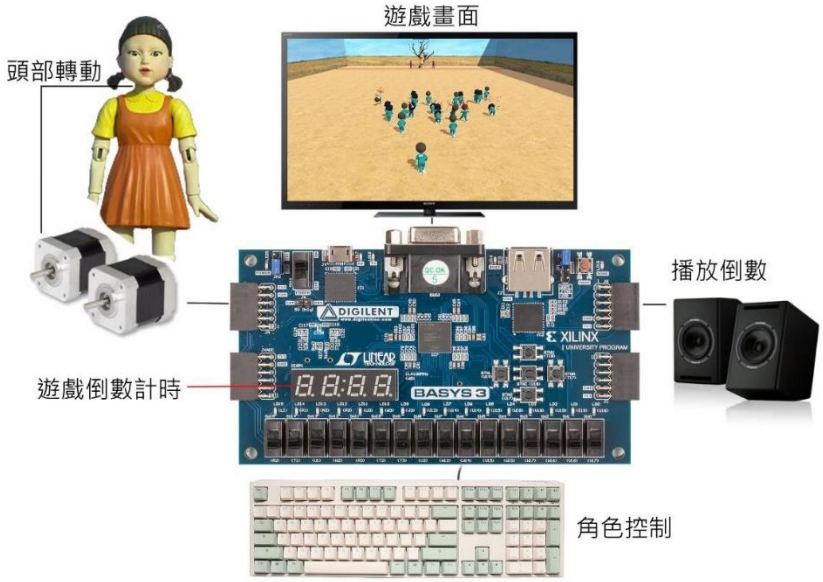
Logic Design Laboratory Final Project Group24 report

109062202 陳禹辰 109062203 林祐禾

Project title：123 木頭人です

設計概念：

我們製作的是一款 123 木頭人遊戲，靈感是來自於去年很紅的電視劇魷魚遊戲。遊戲一次可以有四名玩家同時遊玩，玩家的目的是在時限一分鐘向前移動抵達終點線，遊戲中有一個自動控制的木頭人當鬼，玩家需要在木頭人背對玩家時才能向前移動，若是在木頭人倒數完回頭看向玩家時移動該玩家就會出局。為了增加遊戲刺激性，當一名玩家通過終點時遊戲就結束，且該玩家成為遊戲的贏家，而若四名玩家在時限內都沒有抵達終點則全員出局。另外我們手工製作一個真實的木頭人，且木頭人有各種倒數的模式，以增加真實性及遊戲性。



I/O device:

鍵盤：玩家操作遊戲中的四位角色

螢幕：顯示遊戲畫面

揚聲器：播放木偶人倒數的音效以及遊戲結束時的音樂

七段顯示器：遊戲倒數計時

Push-Button：遊戲開始及 reset

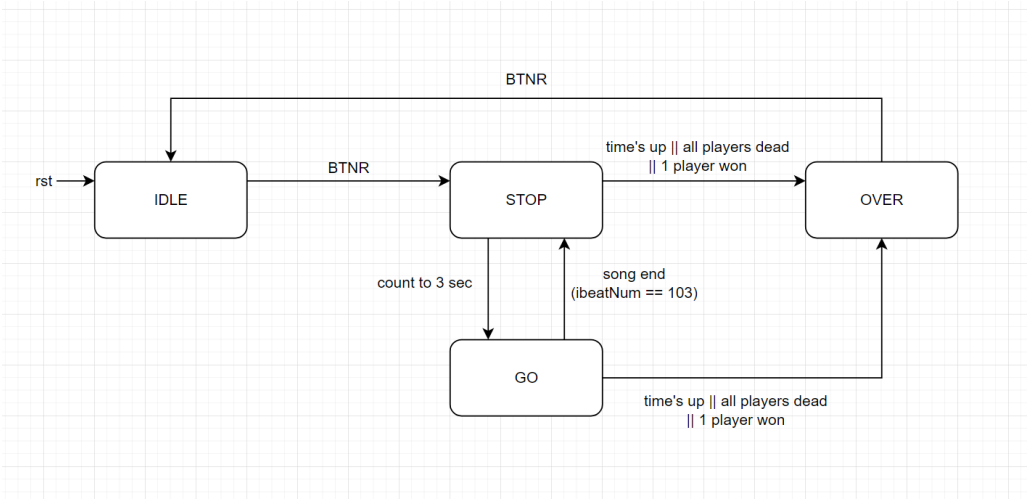
馬達：控制實體木偶人頭部的轉動

LED：顯示現在木偶人轉頭的模式（測試用）

Switch：可以直接改變木偶人轉頭的模式(測試用)

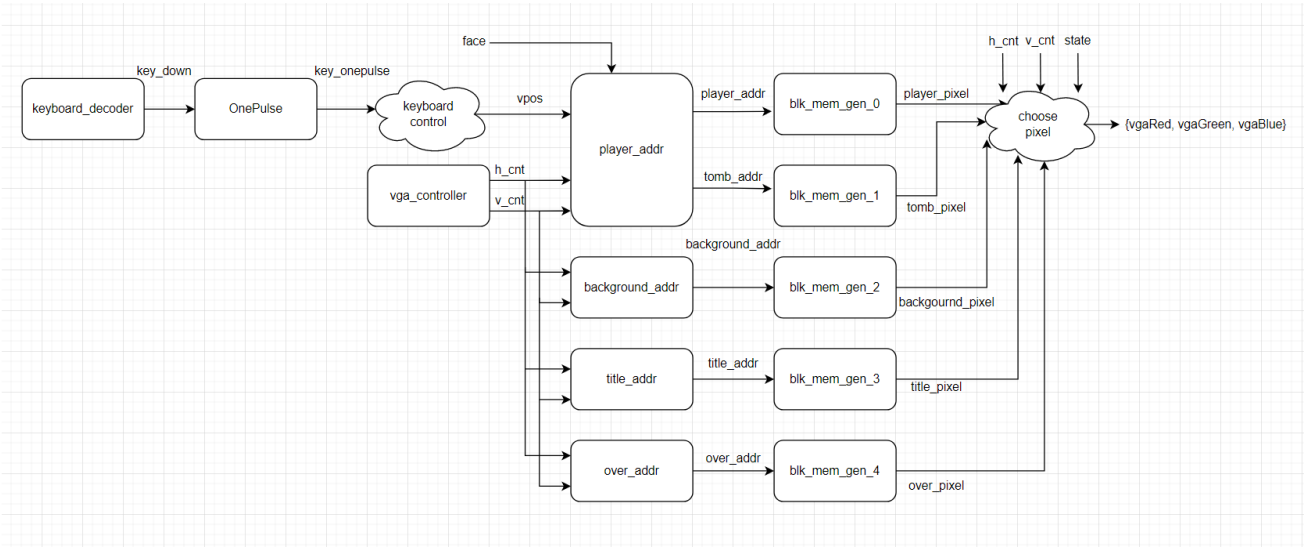
架構細節及方塊圖：

State:



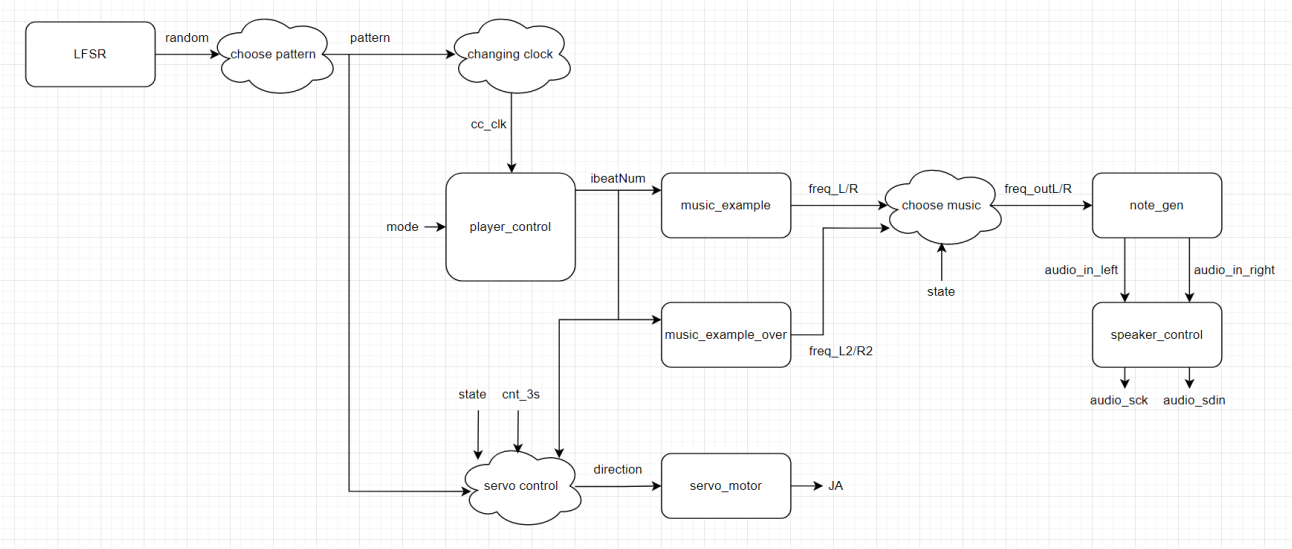
我們總共設計了 4 個 state，IDLE state 為遊戲主頁面，STOP state 及 GO state 時進行遊戲，其中 STOP state 代表木頭人面對玩家，此時若玩家移動玩家就會出局，GO state 則代表木頭人背對玩家，此時玩家可以安全移動，OVER state 則是遊戲結束的頁面。在 reset 時會先進到 IDLE state，在 IDLE state 時按下 BTNR 就會開始遊戲並先進入 STOP state，STOP state 時會啟動一個 counter，數到三秒後進到 GO state，GO state 時木頭人會進行倒數，倒數結束後(從 ibeatNum 判斷)會進到 STOP state，遊玩時會不斷在這兩個 state 中來回切換，直到滿足遊戲結束的條件時就會進到 OVER state，OVER state 同樣由 BTNR 回到 IDLE。

Keyboard and VGA:



VGA 顯示的架構為使用 vga_controller 所輸出的 h_cnt 及 v_cnt 計算出我們所使用的五張圖片對應該 h_cnt 與 v_cnt 要輸出的 RGB 值，再從 state 以及位置來判斷要使用哪一張圖片的 RGB 值並將其輸出。其中因為玩家的腳色會移動，在計算玩家圖片的要輸出的 addr 時還要加入玩家的位置(vpos)進行計算。而 vpos 則是由 keyboard_decoder 的 key_down 來控制，而且我們希望讓玩家按一次按鍵只會移動一步以增加遊戲性，因此我們也將 key_down 進行了 onepulse 的處理。

Speaker and Servo:



首先喇叭會輸出兩首音樂，一首是木頭人倒數的音樂，另一首則是遊戲結束的音樂，兩首音樂的切換使用與之前 lab 中一樣的方式實作就行了。再來喇叭及馬達控制都是屬於木頭人自動控制的範圍，大致架構為先使用 LFSR 得到一個隨機的 pattern，再對喇叭及馬達進行對應的控制。

實作完成度/難易度說明/分工：

實作完成度：

我們認為最後實作的結果與當初所規劃的相比完成度有 90%，因為原本預計的基本功能如遊戲本體、畫面、音效、木頭人控制等都有成功實作出來，不過還是有少數未達成或是可以再加強的地方。

沒有達成或可以再加強的部分：

1. 我們原本希望在木頭人面對玩家時，如果有玩家移動死掉了，可以讓木頭人看向死掉的玩家，但這個功能我們最後沒有做出來，因為若是要讓木頭人轉頭之後再轉向死掉的玩家，那麼所需要的轉動角度少要 270 度，這樣超過了我們所購買的伺服馬達的最大轉動角度了。我們有想出一個解決辦法，就是使用兩個伺服馬達，由下面的來控制頭的轉向、上面的控制頭轉向死亡的玩

家，只是因為我們只有購買一顆伺服馬達以及我們製作木頭人的方式，我們沒有去完成這個項目。

2. 一個可加強的地方是音樂隨機的模式可以有更多種，我們在實作時只有作出六種音樂的隨機模式，但是其實可以再增加他的音樂變化，這樣遊玩起來就會有更多變的感覺，只是因為時間問題，且我們嘗試過幾種後，覺得有些改變會影響到整個音樂撥放出來的感覺，導致聽起來已經不像是倒數音樂的樣式了，所以就先以我們試出來的六種來做為我們音樂隨機的模式。
3. 我們原本也有想要實作讓角色在前進時圖片會縮小以營造遊戲的 3D 感，但是因為在實作時我們把這個項目當作最後如果還有時間才要額外處理的功能，迫於時程壓力，就沒有去實作出來。
4. 原本設計時，有想過讓玩家可以左右移動去阻擋其他的玩家，以此來增加玩家之間的互動性與競爭性，但同樣的我們將此做為最後還有時間才要額外處理的功能，礙於時程壓力，就沒有實作出來。

難易度說明：

我們給這項作品的難易度有大概 90 分，大致上覺得困難的點如下

1. 我們使用了許多的 I/O device，因此在規劃整體架構以及實作時都要不斷去思考各個裝置或模組之間要如何進行互動，否則在實作時會很容易亂掉。而且因為我們製作的是一個遊戲，而且遊戲的內容是與時機點相關的，因此需要確保畫面、音效、馬達等裝置的運作是協調的，才不會破壞遊戲體驗。
2. 因為使用了許多的 I/O device，程式碼也是相當的龐大，因此在 debug 或是新增東西時都相當不容易。
3. 因為在 coding style 上我們想要盡量統一，所以在實作時一定要兩人都在場，這樣才能了解 code 到底是如何運作得，因此就要想辦法排出雙方都能的時間，還要顧慮到考試以及作業要準備，我們算是很早就開始進行這項期末 project，以避免將所有要做的東西都壓在最後幾天。
4. 而在實作時，其實遇到困難的頻率是蠻高的，因為若是之前學過的東西，就不會遇到什麼太大的問題，但是因為很多功能之前試沒有實作過，所以只要遇到，通常就要花點時間去查找資料或進行幾遍嘗試，累積下來就會花費蠻多的時間的，所以雖然早早就開始進行，但其實也是壓到 demo 前幾天才完成這份 project。

分工：

圖形部分、背景圖片製作與玩家角色圖片去背：林祐禾

音效部分、音樂 code 生成：陳禹辰

其他(遊戲本體、角色控制、Report……等)：共同完成。

測試完整度：

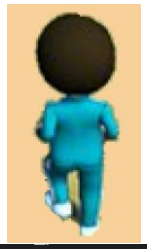
在測試完整度方面，我們在每完成一個項目的實作就會去測試該項目的完整度，並且與之前的 module 之間的配合是否有出問題，例如做完木偶人的轉頭音樂後就去測試，馬達轉動木偶人與木偶人倒數音樂之間是否有平衡，沒有出現音樂撥完木偶人還沒轉完頭等情況出現。我們也使用 led 及 switch 來確認特定功能或是訊號數值的正確性。

最後遊戲完成時，我們也找來其他人來實際遊玩這場遊戲，觀察在實際遊玩時是否有出現 bug，或給出意見來看我們的遊戲平衡性是否合理，或是太簡單或太難。而在測試最後的成品時我們都沒有遇到過 bug 或是預料外的狀況。

困難與解決方法：

1. 圖片的重疊與去背：

   我們的作品中的主畫面圖形、角色圖片及遊戲結束畫面的文字都是使用與背景不同的圖檔，而這些圖案的形狀都是不規則的，並不是一個長方形，因此我們遇到的困難是需要想辦法讓圖案之外的部分顯示背景的畫面。我們想到的解決方式是在原本的圖檔中將我們不需要顯示的部分塗上一個特定的顏色，例如在我們在人物圖片的背景填上一個土黃色，這個顏色在轉為 coe 檔後數值會是 FD9，而後面在輸出人物的圖片時時若碰到 FD9 就改為輸出背景的圖片。要注意的是這個顏色要選擇在要顯示的圖案中不存在的，才不會移除到圖案的部分。



```
if(in1)
    if(life1 == 1)
        {vgaRed, vgaGreen, vgaBlue} = player_pixel == 12'hFD9 ? background_pixel:player_pixel;
    else
        {vgaRed, vgaGreen, vgaBlue} = tomb_pixel == 12'hFD9 ? background_pixel:tomb_pixel;
```

2. 如何讓角色移動及讓移動時有走動的感覺：

   需要讓角色可以前進的話就必須記錄每個角色所在的位置，並根據所在的位置畫出一個範圍，然後移動角色所在位置就能讓範圍也跟著移動，就可以呈現出讓角色往前的感覺了。至於要如何讓角色有移動的感覺，我們所使用的角色圖片為單腳抬起的，我們發現若將圖片左右翻轉就會有移動的感覺，因此我們紀錄每個角色的方向，每次角色移動時就改變其方向，並依照方向在輸出圖片時將圖片進行翻轉。

```
if(h_cnt < hpos1 + 24 && h_cnt > hpos1 - 24 && v_cnt < vpos1 + 43 && v_cnt > vpos1 - 43)
    pixel_addr = face1 == 0 ? ((h_cnt - (hpos1 - 24)) + 49 * (v_cnt - (vpos1 - 43))) % 4263 :
    (((hpos1 - 24) - h_cnt) + 49 * (v_cnt - (vpos1 - 43))) % 4263;
else if(h_cnt < hpos2 + 24 && h_cnt > hpos2 - 24 && v_cnt < vpos2 + 43 && v_cnt > vpos2 - 43)
    pixel_addr = face2 == 0 ? ((h_cnt - (hpos2 - 24)) + 49 * (v_cnt - (vpos2 - 43))) % 4263:
    (((hpos2 - 24) - h_cnt) + 49 * (v_cnt - (vpos2 - 43))) % 4263;
else if(h_cnt < hpos3 + 24 && h_cnt > hpos3 - 24 && v_cnt < vpos3 + 43 && v_cnt > vpos3 - 43)
    pixel_addr = face3 == 0 ? ((h_cnt - (hpos3 - 24)) + 49 * (v_cnt - (vpos3 - 43))) % 4263:
    (((hpos3 - 24) - h_cnt) + 49 * (v_cnt - (vpos3 - 43))) % 4263;
else if(h_cnt < hpos4 + 24 && h_cnt > hpos4 - 24 && v_cnt < vpos4 + 43 && v_cnt > vpos4 - 43)
    pixel_addr = face4 == 0 ? ((h_cnt - (hpos4 - 24)) + 49 * (v_cnt - (vpos4 - 43))) % 4263:
    (((hpos4 - 24) - h_cnt) + 49 * (v_cnt - (vpos4 - 43))) % 4263;
else pixel_addr = 0;
```

3. 如何讓木頭人有隨機的模式：

   我們是運用上課教的 LFSR 來產生一個循環的 4bit 數字，然後在不同時機點來選取其產生的值，來當作我們音樂規律是要以哪種呈現。然後就是處理要如何能夠在遊戲進行時能夠在不同時機點選取 LFSR 所產生的值，我們是以玩家按下鍵盤內角色前進按鍵作為時機點，也就是每當玩家在木頭人倒數時按下前進鍵，就會從 LFSR 選取一個值來作為下一次木頭人倒數音樂的規律選擇，而若是沒有人按下鍵盤，那就會跟上一次的倒數規律一樣，不過這個狀況在正常遊戲進行下應該不太會發生，這是我們覺得比較能夠呈現出隨機性的辦法。

```
module LFSR (
input wire clk,
input wire rst,
output reg [3:0] random
);
    always @(posedge clk or posedge rst) begin
        if (rst == 1'b1)
            random[3:0] <= 4'b1000;
        else begin
            random[2:0] <= random[3:1];
            random[3] <= random[1] ^ random[0];
        end
    end
endmodule
```

```
wire [3:0] random;
LFSR r(clk, rst, random);
```

```
else if(state == GO && key_pressed == 1)
    pattern_next <= random;
```

4. 如何製造木頭人不同模式的倒數音樂：

   我們所挑選的木頭人倒數音樂就是魷魚遊戲片中木頭人倒數的音樂，我們共有六種模式，分別是正常速度、正常速度中間加快、正常速度假動作、兩倍速度、兩倍速度中間加快及四倍速度，而我們取用 LFSR 輸出中的三個 bit 代表各模式，其中有兩個模式對應到兩種 bit 的組合，因此出現的機率較高。我們改變木頭人的倒數節奏的方式是在途中調整倒數音效的撥放速度，也就是讓 ibeatnum 在數到特定數字時時去更改音樂行進的 clk。而因為我們這段音樂只有十個音符，因此必須多嘗試幾次選擇要改變播放速度的點，

不然可能會出現整段音樂聽起來怪怪的情況。

```
always @* begin
    case(pattern[3:1])
        3'b000: c_clk = clk_22;
        3'b001: c_clk = clk_22;
        3'b100: c_clk = clk_21;
        3'b101: c_clk = clk_21;
        3'b010: c_clk = ibeatNum < 64 ? clk_22:clk_21;
        3'b110: c_clk = ibeatNum < 64 ? clk_21:clk_20;
        3'b011: c_clk = ibeatNum < 64 ? clk_20:clk_22;
        3'b111: c_clk = clk_20;
    endcase
end
```

5. 如何讓馬達轉動配合木頭人倒數的音效：

   馬達在旋轉 180 度時需要一定的時間，並不會瞬間到位，因此如果在木頭人倒數完時才讓馬達開始轉動會讓兩者間有一個延遲，影響遊戲的體驗。因此我們需要讓馬達在倒數完之前就提早開始轉動，而且因為我們的木頭人有六種倒數模式，而且各模式倒數的速度都不相同，對於每種模式我們都需要個別設計一個適當的轉頭時機。

```
else if(state == GO) begin
    if((pattern[3:1] == 3'b000 || pattern[3:1] == 3'b001) && ibeatNum == 87)
        direction <= 1;
    else if((pattern[3:1] == 3'b100 || pattern[3:1] == 3'b101) && ibeatNum == 72)
        direction <= 1;
    else if(pattern[3:1] == 3'b111 && ibeatNum == 48)
        direction <= 1;
    else if(pattern[3:1] == 3'b110 && ibeatNum == 48)
        direction <= 1;
    else if(pattern[3:1] == 3'b010 && ibeatNum == 72)
        direction <= 1;
    else if(pattern[3:1] == 3'b011)
        if(ibeatNum == 15)
            direction <= 1;
        else if(ibeatNum == 40)
            direction <= 0;
        else if(ibeatNum == 87)
            direction <= 1;
end
```

心得討論：

陳禹辰：

這次作這項 project 其實花費蠻多心力，因為 demo 時間卡在期末考周後沒幾天，而且還有其他科目的 final project 需要同時進行，我又報名了寒假的營隊，所以其實最後幾天在確認時，其實已經沒有什麼心力再去做功能上的新增或調整。但其實做這份作業也是蠻好玩的，看到了很多很有創意的作品，也讓我整合這學期所學到的東西，也算是讓這這門課程最後有了一個好的作品當作結束。

林祐禾:

我想這個 project 與之前的 lab 最大的不同就是要自己設定題目以及設計

要實作的功能，因為要製作的是自己所選擇且感興趣的題目，在實作的過程中就會比較有熱忱，也會想要將每個功能都做到最好，甚至是加入更多的功能，讓自己對成品感到更加滿意。另外我們是小組進行這個作業，最大的感悟就是兩個人合作時可以讓 debug 更加的順利，當一個人卡住時另一個人總是可以找到問題點，除此之外也學到了分組進行 project 時組員之間的分工以及交流。最後謝謝老師及助教們這個學期的教導!