Final Project 14組 黃士軒(組長)，翁偉銘(副組長)

1. 設計概要:

我們這組是做打地鼠，以下附上遊戲規則和功能介紹。

基本遊戲規則:

* LCD會在**30秒**內顯示盡量的顯示地鼠，30秒後**GAME OVER**。
* 打中地鼠**加分**，打錯**扣分**，超時**不扣分**。
* 分數**最低為0分**不會倒扣，其中分數會顯示在14段顯示器。
* 可以藉由調整底下的**push button調整遊戲賠率**。
* 上面的dip也可以調整地鼠出現的**時間間格**，調控遊戲難易度。
* LED則會進行**倒數**，燈滿遊戲即結束。
* 在你打錯地鼠的同時會發出**低音**提示，打對則是**高音**提示。

功能介紹:

LCD:

* LCD分成左右(64x64)**兩部分**。
* 左邊固定動畫的地鼠**每0.5秒**換一次圖(ROM)，最後一張圖**停2秒**。
* 右邊切割成**16格的RAM**，地鼠會根據設定時間改變出現的格子。
* 一直不打地鼠的話，地鼠會以一個**固定的複雜規律**出現在16格中。

Keypad:

* Keypad總共16個按鍵，恰巧**對應到LCD右半邊的16格切割RAM**。
* 程式內有設置一個**亂數產生器**，只有當輸入的**key=key\_random**或**超時**才會換地鼠位置。
* Key\_random亂數在等於key的時候會**鎖住0.1秒**，確定你不是矇到那個按鍵後才加分。

Push button:

* 用到了**4個push button**，1個是**rst\_n**，讓遊戲**重新開始**的功能

(類似start鍵)。

* 另外3個pb(pb1，pb2，pb3)是調整得分**賠率**用的。
* 按下pb1時，fsm\_pb1=1進入**對得2分，錯扣2分**的賠率，再按一次則恢復成**得1扣1**。
* 按下pb2時，fsm\_pb2=1進入**對得3分，錯扣4分**的賠率，再按一次則恢復成**得1扣1**。
* 按下pb3時，fsm\_pb3=1進入**對得5分，錯扣7分**的賠率，再按一次則恢復成**得1扣1**。
* 其中1個fsm\_pb變成1時，其他的fsm\_pb會**reset成0**。

DIP:

* DIP改變地鼠速度，設置**4種遊戲難度**(用到2個DIP)。
* 地鼠改變頻率有**0.5hz**(DIP1=**0**，DIP2=**0**)。
* 地鼠改變頻率有**1hz**(DIP1=**1**，DIP2=**0**)。
* 地鼠改變頻率有**2hz**(DIP1=**0**，DIP2=**1**)。
* 地鼠改變頻率有**4hz**(DIP1=**1**，DIP2=**1**)。

14-segment:

* 14-segment的功能是為了顯示**累積的分數**。
* **score**會自動加減目前賠率的所對應的分數，並將結果輸出至14-segment。

LED:

* LED的功能是為了顯示累積的時間。
* 一開始LED只會亮一個燈，之後每經過**2秒**，就會多亮一個燈，**30秒**後，燈會全亮，此時遊戲結束。

Sound:

* 我們會根據按鍵的**按對按錯**來發出**2個**不同的聲音。
* 此外，聲音只會停留大約**0.1秒**左右，就消失了。

1. 所用到的module 和block diagram介紹:

畫個blk diagram 和所有module的基本介紹好了(點到即可)

1. 設計亮點:

我們遇到最大的挑戰之一包含了要將LCD左右分成兩個區塊分別進行。左邊要跑ROM，右邊要跑RAM。除了要設計圖片之外，讓lcd跑的流暢也是花了不少功夫。以下附上部分ROM的lcd\_ctl和RAM的lcd\_ctl並概略介紹如何設計出來的。

**lcd\_ctl\_ram:**

……

Display: begin

if(flag == 1'b1 && x\_cnt == 3'd0 && y\_cnt == 6'd0)

begin

state\_next = IDLE;

**finish\_ram\_tmp = 1'd1;//out**

end

end

IDLE:begin

begin

if(**start\_ram==1**) //input

begin

state\_next = Display;

**finish\_ram\_tmp = 1'd0;**

end

else

begin

state\_next = IDLE;

**finish\_ram\_tmp = finish\_ram**;

……

我們在lcd\_ctl\_ram裡面加入兩個參數:input start\_ram和 output finish\_ram。

當state準備要從display進入到idle時，會將finish\_ram變成1。

當state準備從idle進入display時(start\_ram==1)，會將finish\_ram變成1。

現在再看向lcd\_ctl\_rom:

……

`LCD\_IDLE:

begin

{lcd\_di\_next,lcd\_rw\_next,lcd\_data\_next} = `LCD\_DISPLAY\_IDLE;

if (idle\_counter >= 16'd10000 && **start\_rom==1**)

begin

**finish\_rom\_tmp = 1'd0;**

state\_next = `REQUEST\_DATA;

idle\_counter\_next = 16'd0;

counter\_y\_next = 7'd0;

counter\_page\_next = 4'd0;

end

else

begin

state\_next = `LCD\_IDLE;

idle\_counter\_next = idle\_counter + 16'b1;

**finish\_rom\_tmp = finish\_rom;**

end

end

`READ\_DATA:

begin

if (counter\_y<7'd63)

begin

{lcd\_di\_next,lcd\_rw\_next,lcd\_data\_next} = {2'b1\_0,data};

counter\_y\_next = counter\_y + 1'b1;

end

else if (counter\_y==7'd63)

begin

{lcd\_di\_next,lcd\_rw\_next,lcd\_data\_next} = {2'b1\_0,data};

counter\_y\_next = counter\_y + 1'b1;

counter\_page\_next = counter\_page + 1'b1;

end

else if (counter\_y==7'd64)

begin

counter\_y\_next = 7'd0;

if (counter\_page==4'd8)

begin

if(image==4'd8)

begin

image\_next = 4'd8; // change to next image

state\_next = `PAUSE\_2;/////修改過

**finish\_rom\_tmp = 1'd1;**

end

else

begin

image\_next = image + 1'b1; // change to next image

state\_next = `LCD\_IDLE;

**finish\_rom\_tmp = 1'd1;**

end

end ///修改過

else

state\_next = `REQUEST\_DATA;

end

end

……

當state準備要從readdata進入到idle時，會將finish\_rom變成1。

當state準備從idle進入request\_data時(start\_rom==1)，會將finish\_rom變成1。

而這兩個還需要加上rst\_n時的條件:

…

always @(posedge clk or negedge rst\_n) begin

if (~rst\_n) begin

**finish\_ram = 1'd0;**

…

always @(posedge clk or negedge rst\_n)

if (~rst\_n)

begin

**finish\_rom<=1'd1;**

…

一個必須在rst\_n被按下時變成1，另個則要是0，如此一來才能互相切換。

最後是top module的部分:

……

lcd\_RAM\_1 D4(.clk(clk),.rst\_n(rst\_n),.LCD\_rst(LCD\_rst),.LCD\_cs(LCD\_cs\_1),.LCD\_rw(LCD\_rw\_1),.LCD\_di(LCD\_di\_1),.LCD\_data(LCD\_data\_1),.LCD\_en(LCD\_en\_1),.col\_n(col\_n),.row\_n(row\_n),.key\_random(key\_random),.key(key),.pressed(pressed),.cnt(cnt),.finish(finish),**.finish\_ram(finish\_ram),.start\_ram(finish\_rom),**.dip1(dip1),.dip2(dip2),.clk\_debounce(clk\_debounce));

…

lcd\_ROM D5(.clk(clk),.rst\_n(rst\_n),.data\_request(data\_request),.data\_ack(data\_ack),.lcd\_rst(LCD\_rst),.lcd\_cs(LCD\_cs\_2),.lcd\_rw(LCD\_rw\_2),.lcd\_di(LCD\_di\_2),.lcd\_d(LCD\_data\_2),.lcd\_e(LCD\_en\_2),**.start\_rom(finish\_ram),.finish\_rom(finish\_rom)** );

…

always@\*

begin

if(**finish\_rom==1 && finish\_ram ==0**)

begin

**LCD\_cs=LCD\_cs\_1;**

**LCD\_rw=LCD\_rw\_1;**

**LCD\_di=LCD\_di\_1;**

**LCD\_data=LCD\_data\_1;**

**LCD\_en=LCD\_en\_1;**

end

else if(**finish\_rom ==0 && finish\_ram ==1**)

begin

**LCD\_cs=LCD\_cs\_2;**

**LCD\_rw=LCD\_rw\_2;**

**LCD\_di=LCD\_di\_2;**

**LCD\_data=LCD\_data\_2;**

**LCD\_en=LCD\_en\_2;**

end

else

begin

**LCD\_cs=LCD\_cs\_1;**

**LCD\_rw=LCD\_rw\_1;**

**LCD\_di=LCD\_di\_1;**

**LCD\_data=LCD\_data\_1;**

**LCD\_en=LCD\_en\_1;**

end

end

利用對方的finish==1當作start開始的條件便能同時利用左右兩邊lcd跑rom和ram，並利用相同pin解決pin角不足的問題。

1. 遇到的問題和解決方法:

一開始跑程式的時候發現sound都發出一樣的提示音(成功和失敗都是一樣的低音提示)。仔細去探討之後發現，內部的亂數產生器產生的亂數(key\_random)在key==key\_random的時候就瞬間換值了。因此我們最後決定將key\_random**鎖住0.1秒**，及解決方法就是在加一個freq\_div module,而這個module的rst\_n為pressed，也就是按下keypad之後他會開始計時，並將output cnt丟入亂數產生器random module 內。

freq\_div1\_random R0\_0

(

.cnt(**cnt**),

.clk\_slow(clk\_slow),

.clk\_out(clk\_out),

.clk\_out2(clk\_out2),

.clk\_out3(clk\_out3),

.clk\_out4(clk\_out4),

.ftsd\_clk(ftsd\_clk),

.clk(clk),

.rst\_n( **pressed** )

);

**random** R1( .key\_random(key\_random),.clk\_4bit(clk\_4bit),.rst\_n(rst\_n),.clk(clk),.en(key==key\_random),.cnt(**cnt**),.en\_random(en\_random),.clk2hz(clk\_out4));//這行條件太重要了

random module內部的code:

…

always@\*

begin

if(**cnt>4000000** && en==1)

key\_random\_tmp = clk\_4bit\_new;

//一定要按超過0.1秒才換亂數

else if(en\_random==1)

key\_random\_tmp = clk\_4bit\_new;

else

key\_random\_tmp=key\_random;

end

…

另外還有遇到lcd不會寫的問題和latch，一概都涵蓋在**設計亮點**裡面了。

除此之外，應該沒有遇到其他太大的問題。

1. 討論:

(討論優化程式或是如何將學到的code運用到這份final project之類的，類似這學期心得吧)

1. 分工:

設計概念:黃士軒+翁偉銘

LCD, keypad, DIP:翁偉銘

Led, 14-segment, push button:黃士軒

sound:黃士軒+翁偉銘

此份報告: 黃士軒+翁偉銘

P.S.感謝助教一學期的指導和努力，真的是受益良多! 在此獻上最高的敬意XD!