Final Project report

學號: 110062106　　姓名: 黎秉諺

110062128　　姚東廷

組別: 29

A. 動機、設計概念

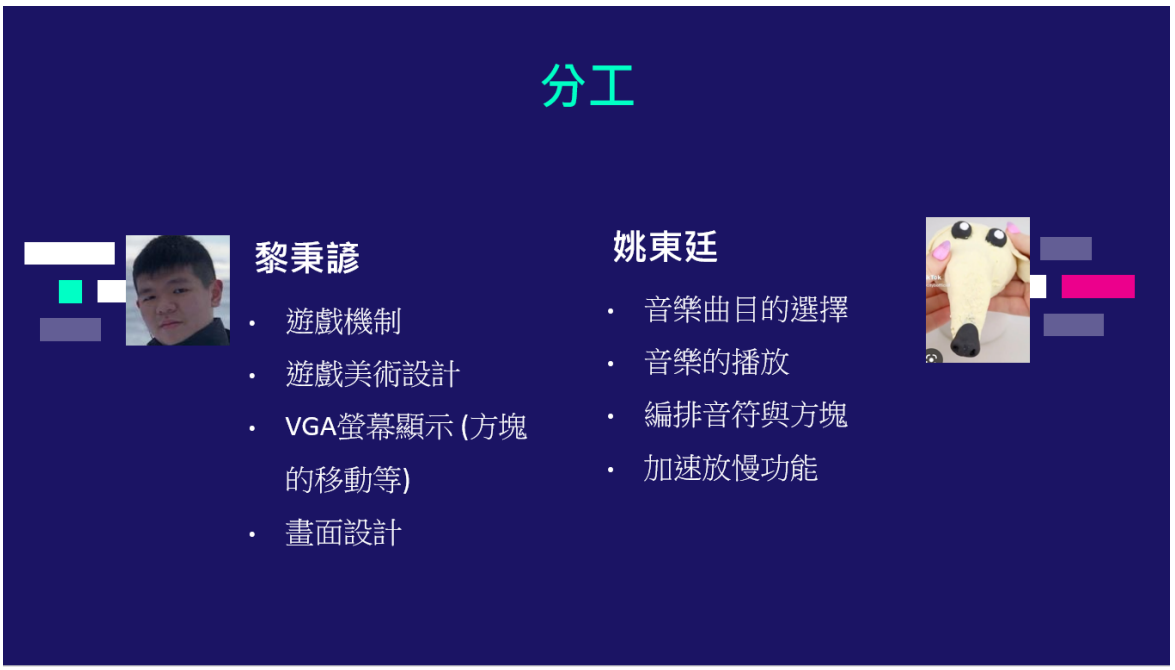
此次的final project參考知名手機音遊「別踩白塊兒」。 「別踩白塊兒」為一款曾風靡全球的手機音樂遊戲，遊戲方式很簡單，就是在黑塊落下時按下就會發出相對應的聲音。但我們不想要全然地做出簡化版的別踩白塊兒。在經過反覆思考後，我們決定要做出一款既是音樂遊戲又是音樂播放器的project，並追求簡約的質感。歌曲方面，我們選用了最近電影院很火紅的經典動、漫畫「灌籃高手」中其中一個知名的片尾曲「直到世界盡頭」。而考慮到不同人對音樂遊戲的熟悉度不同，我們提供了三種速度 (難易度) 給玩家選擇。

跟一般遊戲不同，此project會在每一輪結束後自動進到下一輪，並會播放玩家上一輪有打到的黑塊對應到的音符，而畫面只會顯示上一輪miss掉的黑塊。如此一來，玩家可以通過一次又一次地修飾，最後拼湊出完整悅耳的音樂，因此獲得成就感以及驚喜感。

畫面的部分，會分成五道，而如果在黑塊與白線重疊時按下對應的鍵，被消除的黑塊的那一道的基準線就會變為藍色，否則就會顯示紅色。7-segment display方面則是顯示剩下音符的數量。

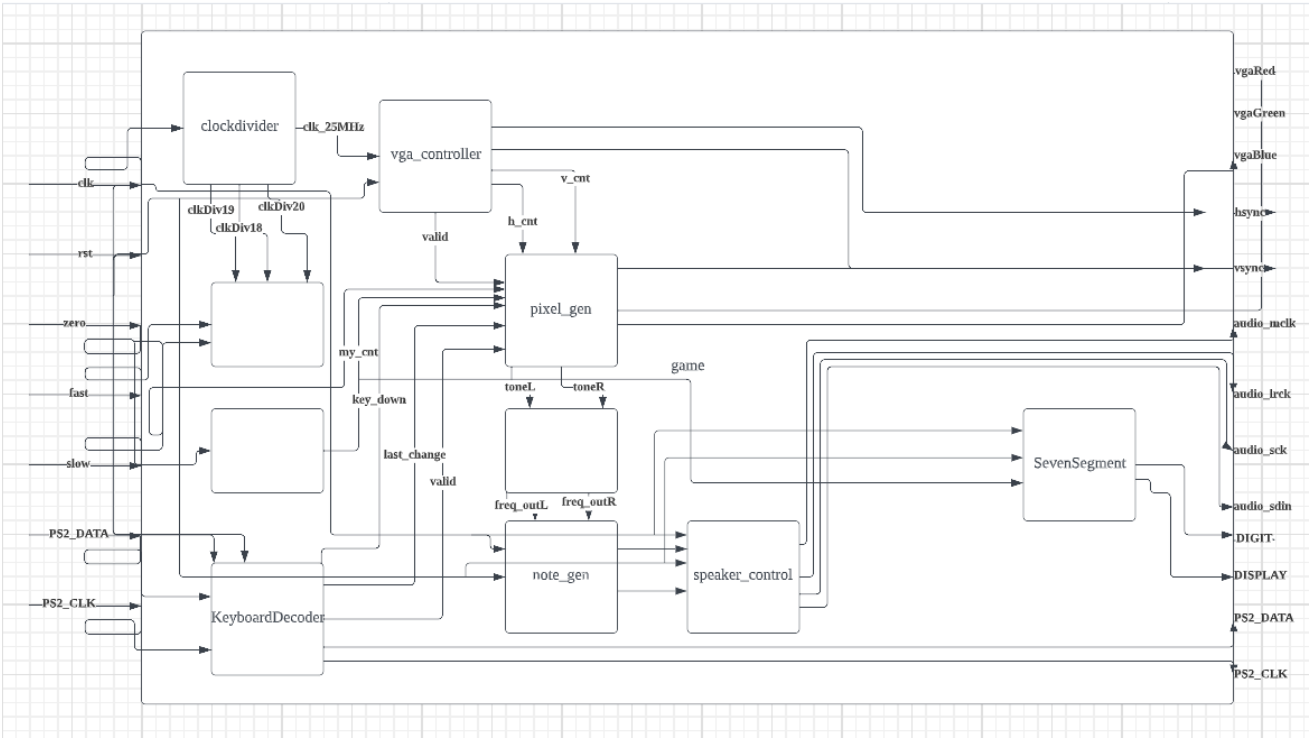
B. 分工

組長黎秉諺主要負責VGA以及遊戲設計，姚東廷則主要負責音樂的部分



C. Project Implementation

1. Block Diagram



2. Code Explanation

demo1 (top):

```
reg myclk;

clock_divider #(.n(2)) clock_2(.clk(clk), .clk_div(clk_25MHz));
clock_divider #(.n(19)) clock_19(.clk(clk), .clk_div(clkDiv19));
clock_divider #(.n(18)) clock_18(.clk(clk), .clk_div(clkDiv18));
clock_divider #(.n(20)) clock_20(.clk(clk), .clk_div(clkDiv20));

always@(*)begin
    if(fast && !slow) myclk = clkDiv18;
    else if(slow && !fast) myclk = clkDiv20;
    else if(fast && slow) myclk = clkDiv19;
    else myclk=clkDiv19;
end
```

使用clock_divider來實作快(clkDiv18)、正常(clkDiv19)、慢等速度(clkDiv20)，每個速度相差2倍。快與慢分別對應到SW[1]與SW[2]，若兩個同時==1或==0就用正常速度。

```
always@(posedge myclk)begin
    if(zero||my_cnt==0) my_cnt<=14'b10100101100100;
    else begin
        my_cnt<=my_cnt2;
    end
end
```

my_cnt為紀錄我們的圖顯示到哪裡的counter (圖片會因為my_cnt不斷-1，也就是往下捲)，而若==0就代表到底了，那my_cnt就會重新設置並重新往下捲。

```
pixel_gen pixel_gen_inst(
    .zero(zero),
    .key_down(key_down),
    .last_change(last_change),
    .my_cnt(my_cnt),
    .v_cnt(v_cnt),
    .h_cnt(h_cnt),
    .valid(valid),
    .vgaRed(vgaRed),
    .vgaGreen(vgaGreen),
    .vgaBlue(vgaBlue),
     .toneL(freqL),
     .toneR(freqR)
);
```

因為VGA會因為鍵盤按下熱鍵而顯示不同東西，且vga所顯示的東西也會影響到要

播出甚麼音，所以要將key_down與last_change、freqL與freqR放入pixel_gen中。

pixel_gen (top):

```
reg [300:0] used1=301'd0;
reg [300:0] used2=301'd0;
reg [300:0] used3=301'd0;
reg [300:0] used4=301'd0;
reg [300:0] used5=301'd0;
```

used1~5分別為在不同道上的某一黑塊有無被消除。

```
my_music[0]=5;my_music[1]=5;
my_music[2]=5;my_music[3]=5;
my_music[4]=5;my_music[5]=5;
my_music[6]=5;my_music[7]=5;
my_music[8]=5;my_music[9]=5;
my_music[10]=5;my_music[11]=5;
my_music[12]=5;my_music[13]=5;
my_music[14]=5;my_music[15]=5;
my_music[16]=4;my_music[17]=3;
my_music[18]=1;my_music[19]=5;
my_music[20]=4;my_music[21]=5;
my_music[22]=1;my_music[23]=2;
my_music[24]=3;
my_music[25]=5;my_music[26]=5;
my_music[27]=5;my_music[28]=3;
my_music[29]=4;my_music[30]=5;
my_music[31]=1;my_music[32]=0;
my_music[33]=5;my_music[34]=0;
my_music[35]=5;my_music[36]=1;
```

.....

my_music[]的index為這是第幾個黑塊(音符)，共有153個(0~152)，而value則是他應該在第幾道，若==5代表不會出現 (空拍)。

```
if(used1[(my_cnt+420)/60]==1||used2[(my_cnt+420)/60]==1||used3[(my_cnt+420)/60]==1||used4[(my_cnt+420)/60]==1||used5[(my_cnt+420)/60]==1)begin
case((my_cnt+420)/60)
152:begin
toneR = `hf;
toneL = `hf;
end
151:begin
toneR = `hf;
toneL = `hf;
end
150:begin
toneR = `hf;
toneL = `hf;
end
149:begin
toneR = `hf;
toneL = `hf;
end
148:begin
toneR = `hf;
toneL = `hf;
end
147:begin
toneR = `hf;
toneL = `hf;
```
.....

確認used後就會將頻率存入toneR與toneL。

```
if(!valid)
        {vgaRed, vgaGreen, vgaBlue} = 12'h0;
 else if(h_cnt<3) {vgaRed, vgaGreen, vgaBlue} = 12'h099;
 else if(h_cnt>630) {vgaRed, vgaGreen, vgaBlue} = 12'h099;
```

沒valid就顯示黑色，邊框為tiffany綠

```
else if(h_cnt < 128&& h_cnt>0)begin //基準線判斷
    if(v_cnt<=421&&v_cnt>=419)begin
    if(key_down[last_change]==1&&last_change==9'b0_0010_0011&&used1[(my_cnt+v_cnt)/60]==1) {vgaRed, vgaGreen, vgaBlue} = 12'h00f;
    else if(key_down[last_change]==1&&last_change==9'b0_0010_0011) {vgaRed, vgaGreen, vgaBlue} = 12'hf00;
    else {vgaRed, vgaGreen, vgaBlue} = 12'hfff;
    end
```

horizontal的0~128為第一道的範圍，vertical的419~421則是基準線的位置，若按下的鍵為"D"且有黑塊曾經出現，那一道的基準線就會變藍色，否則按下時將會是紅色，若沒按下"D"則基準線會是白色。

```
else if(v_cnt<3||v_cnt>476) {vgaRed, vgaGreen, vgaBlue} = 12'h099;
```

位上邊框與下邊框畫上tiffany綠

```
else if(((my_cnt+v_cnt)/60)<153)begin
    if(my_music[(my_cnt+v_cnt)/60]==0)begin
        if(((my_cnt+v_cnt)/60)==(my_cnt+420)/60)begin
            if(key_down[last_change]==1&&last_change==9'b0_0010_0011&&(used1[(my_cnt+v_cnt)/60]==0)) begin
                used1[(my_cnt+v_cnt)/60]=1;
                end
```

若目前黑塊在最後一個黑塊(包含最後一個黑塊#152)之前((my_cnt+v_cnt)/60)<153)，且黑塊在最左邊那道(my_music[(my_cnt+v_cnt)/60]==0)，且黑塊在基準線上((my_cnt+v_cnt)/60)==(my_cnt+420)/60)，且有按下"D"，且那個

黑塊不在used的話就把黑塊加到used (通過將黑塊位置傳入index來實現)。

```
else if(key_down[last_change]==1&&last_change==9'b0_0010_0011) {vgaRed, vgaGreen, vgaBlue} = 12'hf0a;
```

按下"D"時會最左邊那道會變為比較深的桃紅(按鍵回饋)。

```
else {vgaRed, vgaGreen, vgaBlue} = 12'hf0f;
```

若甚麼都沒按就預設顏色為桃紅(背景色)。

(剩下4道也是同樣邏輯依此類推)

SevenSegment:

```
assign nums=(my_cnt+420)/60-7;
```

nums用來計算剩下多少音符，並在SevenSegment的module中用16進位顯示。

```
always @ (*) begin
    case (display_num)
        0 : DISPLAY = 7'b1000000;   //0000
        1 : DISPLAY = 7'b1111001;   //0001
        2 : DISPLAY = 7'b0100100;   //0010
        3 : DISPLAY = 7'b0110000;   //0011
        4 : DISPLAY = 7'b0011001;   //0100
        5 : DISPLAY = 7'b0010010;   //0101
        6 : DISPLAY = 7'b0000010;   //0110
        7 : DISPLAY = 7'b1111000;   //0111
        8 : DISPLAY = 7'b0000000;   //1000
        9 : DISPLAY = 7'b0010000;   //1001
        10: DISPLAY = 7'b0100000;
        11: DISPLAY = 7'b0000011;
        12: DISPLAY = 7'b0100111;
        13: DISPLAY = 7'b0100001;
        14: DISPLAY = 7'b0000110;
        15: DISPLAY = 7'b0001110;
        default : DISPLAY = 7'b1000000;
    endcase
end
```

D. Problem Encountered

在實作音樂中需要去學習許多樂理知識，例如要對拍子有概念才能知道要如何去編排黑塊哪時候應該顯示，哪時候不該 (拉長拍)。

螢幕的部分是在寫的過程中，常常發生找不出原因的意外。例如改了某些數字明

明可以generate bitstream，螢幕卻突然收不到訊號保持全黑。以及一開始弄的時候方塊的消除難以實行，與音樂的配合也是想了很久才有完善的配套措施，這次project遇到的問題應該是最多的，畢竟碰了很多之前沒碰過的，但最終花了時間還是成功一一克服了!

E. 實作完成度

由於我們在途中有因為實作的可行性更換過proposal，所以根據第2個proposal都有實際做到我們要的功能。

F. 笑話

一加一等於二

麵包超人加一等於多少

…

……

……

麵包超大!!