Pacmo

FINAL PROJECT REPORT

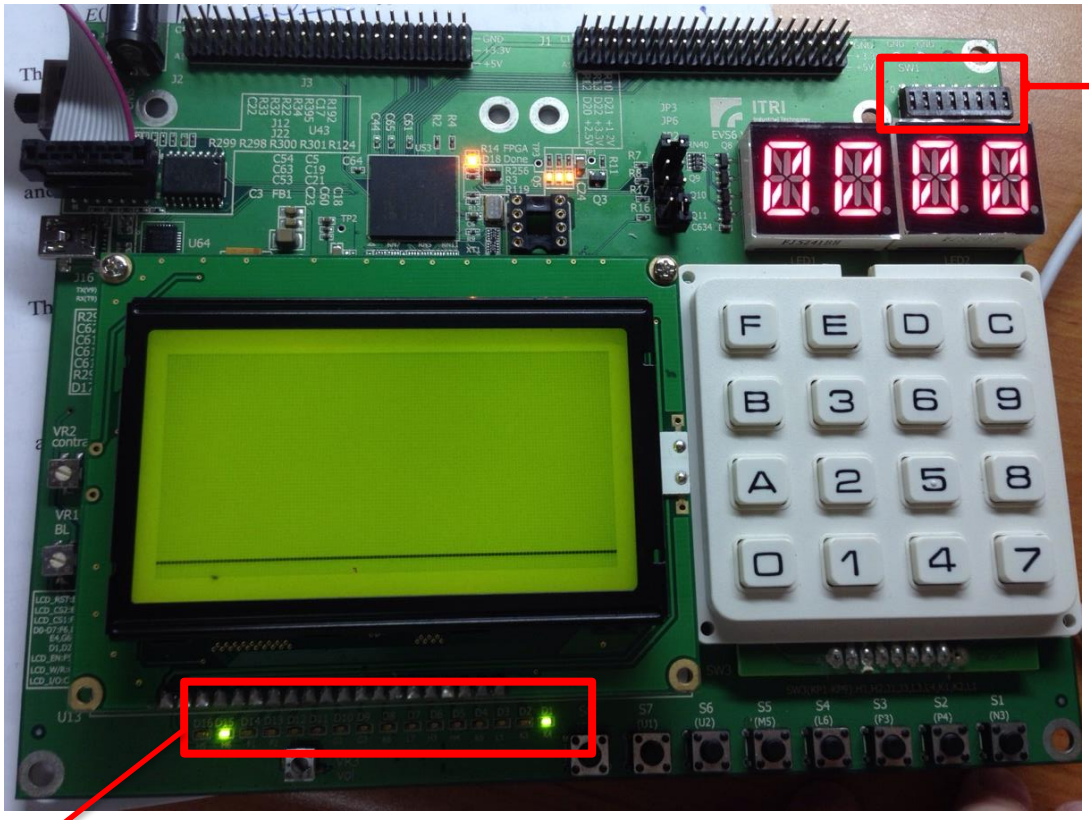
102062222 張心愉
102062305 李芹

Part I. Specification

1. Functionality –

In our final project, we have designed a **music tempo game**. The concept of the game is to press the button on the Keypad of the FPGA board and earn points when you successfully press the right button when objects drop down shown on the board. We will demonstrate and describe the details below.

Initially, when the game is programmed onto the board, there would be nothing but a line on the screen of the LCD.



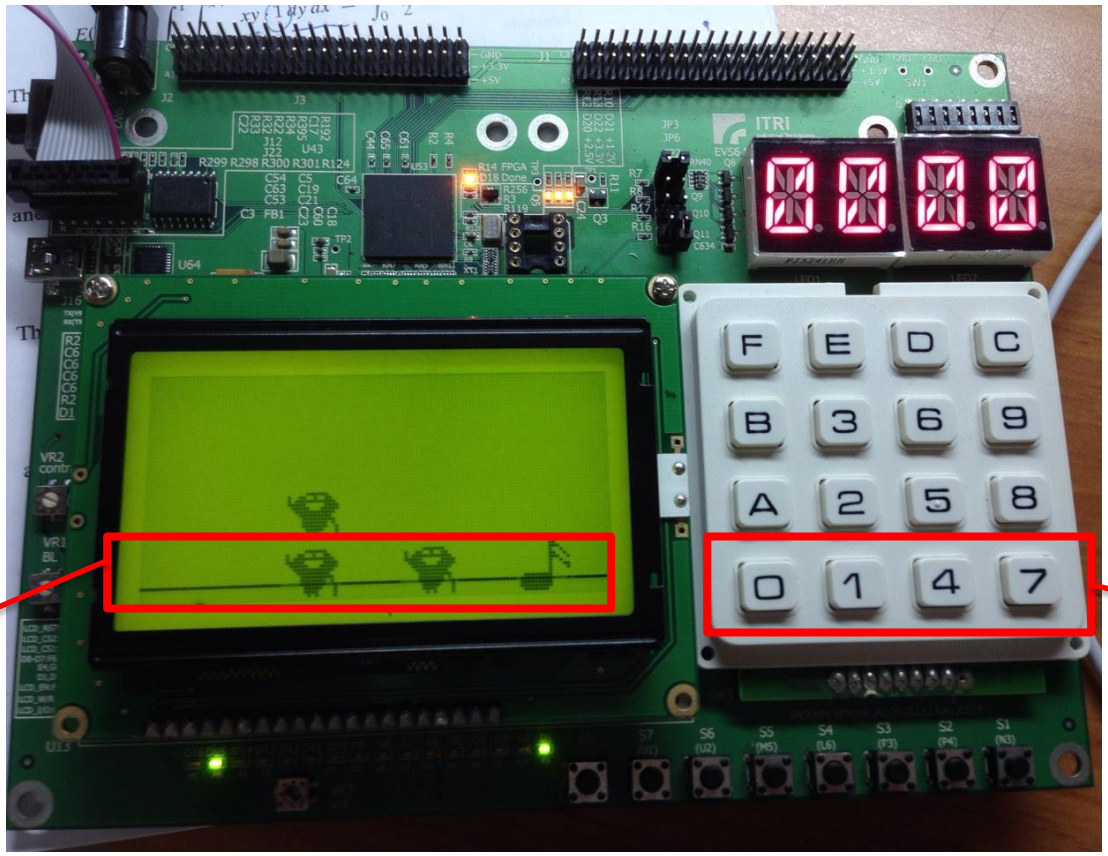
This is where the songs are chose and selected. (the two on the leftmost)

The rightmost PIN controls to **Start** or not.

LED lights up responding to the song you've chosen and whether the game has started or not.

The next step is to pick the song we want to play. For the final project, we have designed two songs for the players to pick. The PIN on the FPGA board is responsible for this task. When the input of the PIN is 1, then the song referred to that PIN is then selected and the LED on the bottom left of the board would lit up, reminding you which song you have chosen. After that, the next step is to be prepared and turn on the START PIN and wait for the game to start.

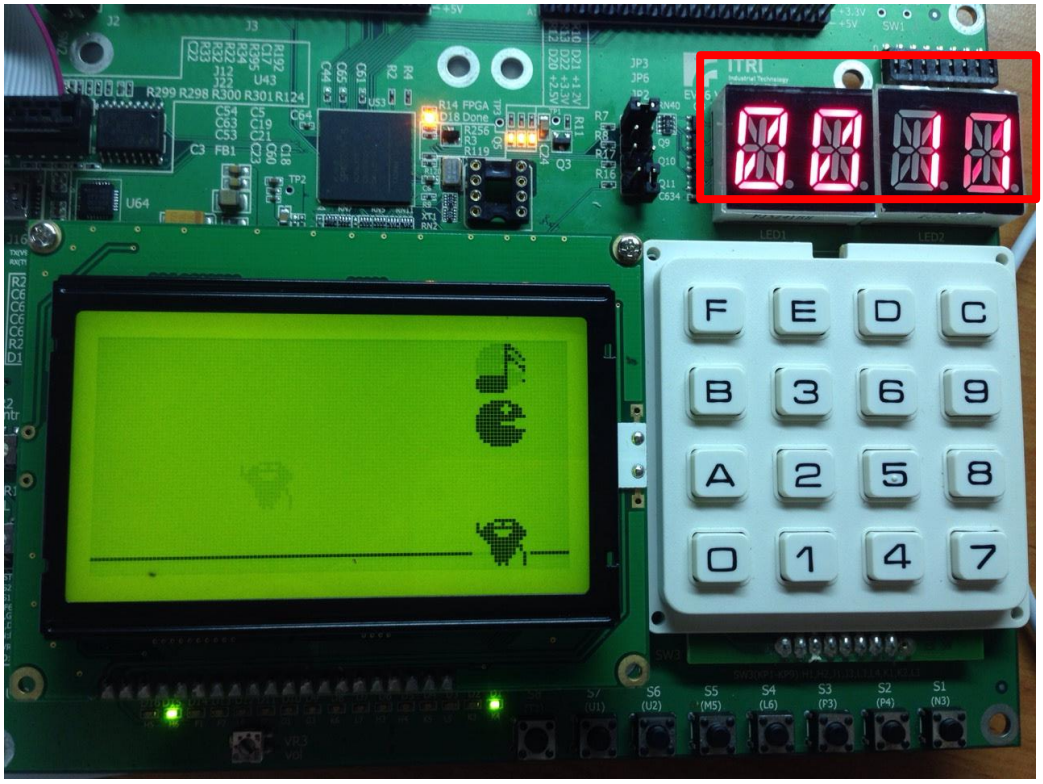
When the game starts, different pictures will appear from the top of the LCD screen. There are three different pictures. One is the **note** of music, another one is a cute **mushroom character**, and the last one is the **PAC-MAN**. These three characters are designed to make the game more entertaining and fun. The pattern of their appearance is chosen randomly. They would each fall down from the top of the screen to the line on the below and then disappear. The player should press down the corresponding keys on the keypad in order to earn points in this game. The keys are 0,1,4,and 7. The keys that the players are required to press are relative to the situation of the characters on the LCD screen. If the character is now situated on the leftmost of the screen, then the player should press 1, in order to earn points.



The player should press the key 0,1,4,7 when the characters are on the line

Where the players should press the key

During the game, the 14-segment display situated on the top of the keypad would then show the points that you've earned until the music stops and end the game.



Display of the score the players have earned.

2. **Special Features –**

- The situations where the characters are shown are randomly picked.

- The pictures of the characters are also randomly picked. Not always the same.

- The amounts of characters that appear in every tempo of the music aren't the same.

- Two songs for the players to pick.

The above features causes the game to be more diverse and produces excitement and unpredictability to the players and also increase the difficulty of the game, making it much more challenging than normal tempo games.

Also to prevent boredom, we provide two songs for the players to pick, so they wouldn't get sick of the game too easily.

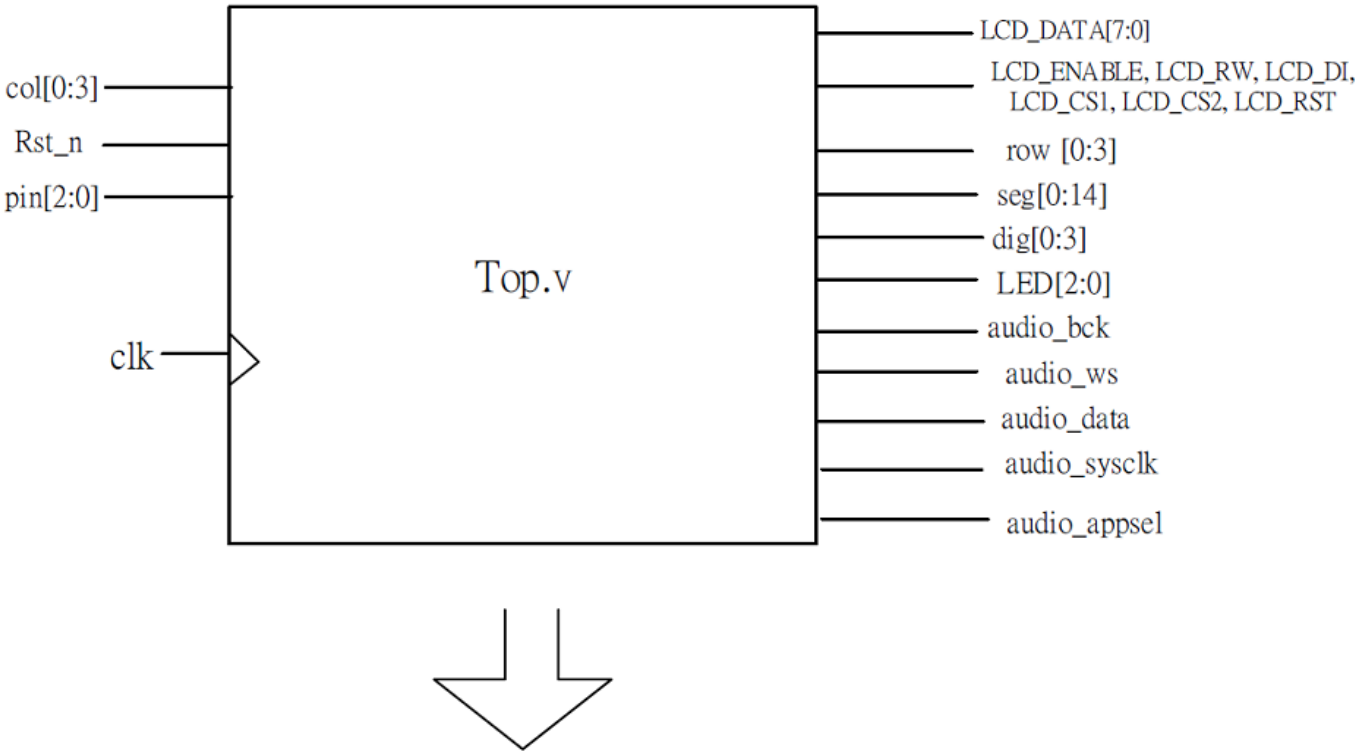
Part II. Implementation details (explanation with block diagram, state transition diagram, etc.)

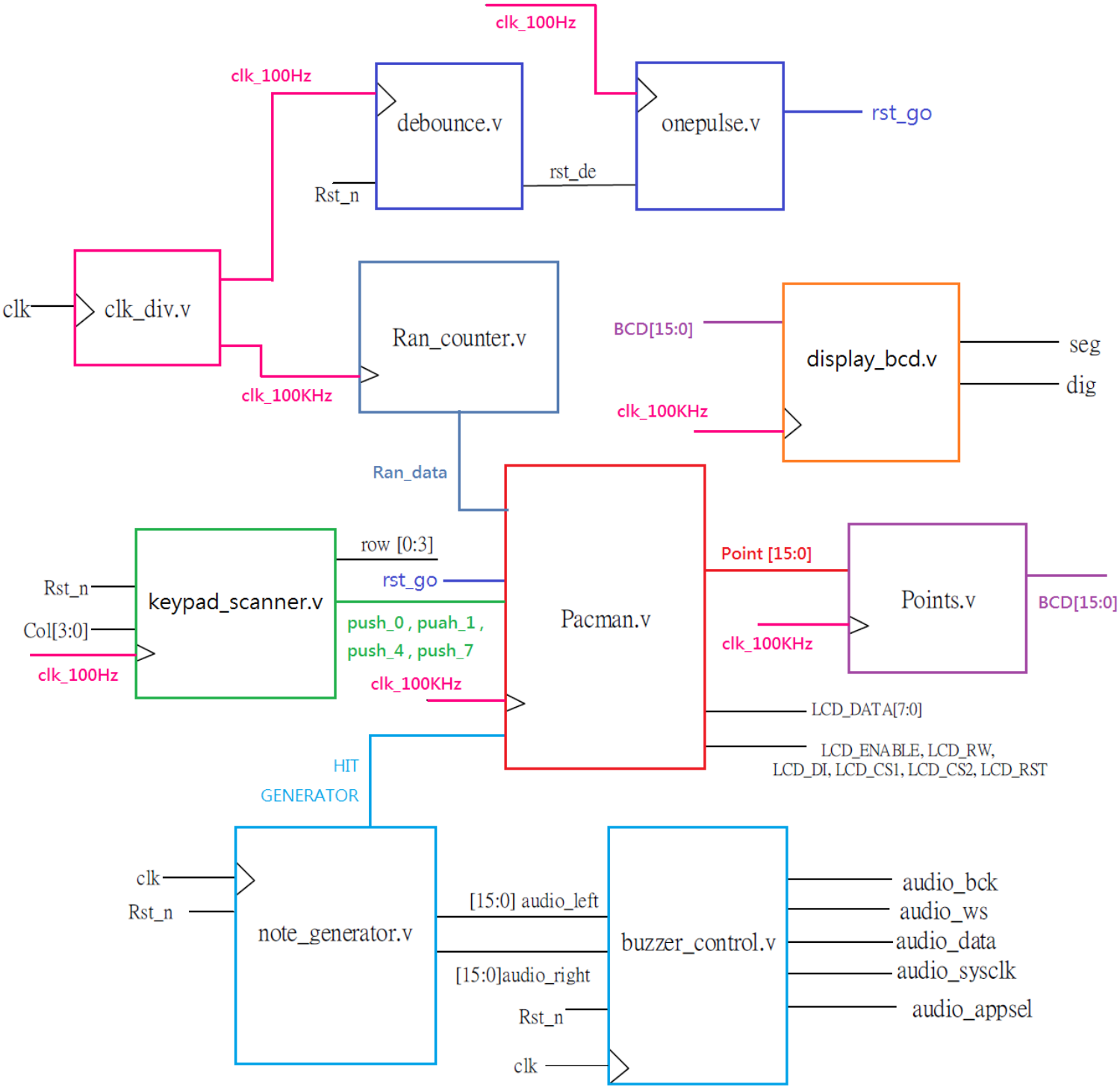
1. block diagram

首先這張圖，是用來說明 UCF 檔需要連接 FPGA 版上哪些 connection，基本上我們的 final 作品會用到 FPGA 上這些部分：

i. keypad (col & row)
ii. 14-segment display (seg & dig)
iii. buzzer (audio control)
iv. LCD display (LCD control)
v. PIN (pin)
vi. Push button(rst_n)
vii. LED (LED)
viii. Clock (clk)

我們的音樂節奏遊戲，會由 buzzer 產生音樂，同時將訊號傳給 LCD control 藉由 lcd 亂數隨機產生音符，往下掉，使用者可藉由 keypad 按下對應音符獲得加分，其中加分會顯示在 14-segment display。另外可藉由 PIN 來選歌曲以及控制是否開始遊戲，會對應到 LED 亮燈，若要重新開始，則可按下 Push button。
箭頭以下，則在下頁詳細說明各個 module 的連接情況。





以各個顏色來區分傳進與傳出的訊號，**黑色字體**表示接到 FPGA 版上的線。

以下說明各個 module 的功用與相互間關聯：

- 首先在**桃紅色的 clk_div** 先分割出需要用到的包含 clk 三種 clock : 100hz , 100khz , 40Mhz。並依桃紅色的線傳進各個 module 中。
- 接著，需要藉由 pacman 的 LCD 來顯示音符，同時會結合 keypad & 14seg

display。也就是遊戲的主體，主要計算都在 pacman，因此有許多訊號經由各 module 處理過後會傳進 pacman，相當複雜。

i. 淺藍色的 note_generator & buzzer_control 輸出聲音與 generator(需要產生幾個音符)、HIT(哪幾個拍子有音符被按到可加分)進 pacman 中。

ii. 藍色的 debounce & onepluse 產生可傳進 pacman 的 rst_go，可以同步 reset 整個遊戲（LCD , buzzer , display）。

iii. 暗藍色 Ran_counter 產生亂數的數字，用於新增音符的圖片與位置。

iv. 綠色的 keypad 會將按下 0 , 1 , 4, 7 等按鈕的訊號傳入 pacman，以此判斷有沒有按到該音符，並統計加分。

v. 同時會輸出紅色的 point 分數，由於這個分數是二進位顯示，因此傳入紫色的 points 轉換為十進位表示，並將處理過的 BCD 訊號傳入橘色的 display 顯示。

- 此外，在 top module 裡面，加上 pin 作歌曲選擇，傳入 note_generator，選擇歌曲，並會亮相對應的 LED 燈。

2. Module Design

使用的 module 中，幾乎都是 lab 做好的再加上修改的功能，除了 pacman 與 note_generator 幾乎是重寫，以及加了新的 module : points （十進位顯示）& ran_counter（利用 shift & xor 做類似 random 的效果）。
以下簡述主要的兩個 module : pacman 與 note_generator。

⊙ note_generator：

要讓 buzzer 完整的播完一首歌，也就要做出完整的譜，然後利用 clk 將 tone 一次加一，跑完所有 tone，當然整首歌的音調都要先算好，並加上哪幾個拍子要產生幾個音符、哪幾個拍子按下對應按鈕會加分，這些都在這邊算好，傳入 pacman。並加上 reset 的設計，讓歌曲可以重播。

要注意的是，因為必須讓 pacman 與 note_generator 同步，加上 delay 的設計，讓 counter 加到等於 delay 時 tone 才會加一，讓兩邊的 clk 可以做到類似同步的效果，並且可以控制 delay 讓歌曲播慢一點。

Code design:（僅擷取部分）

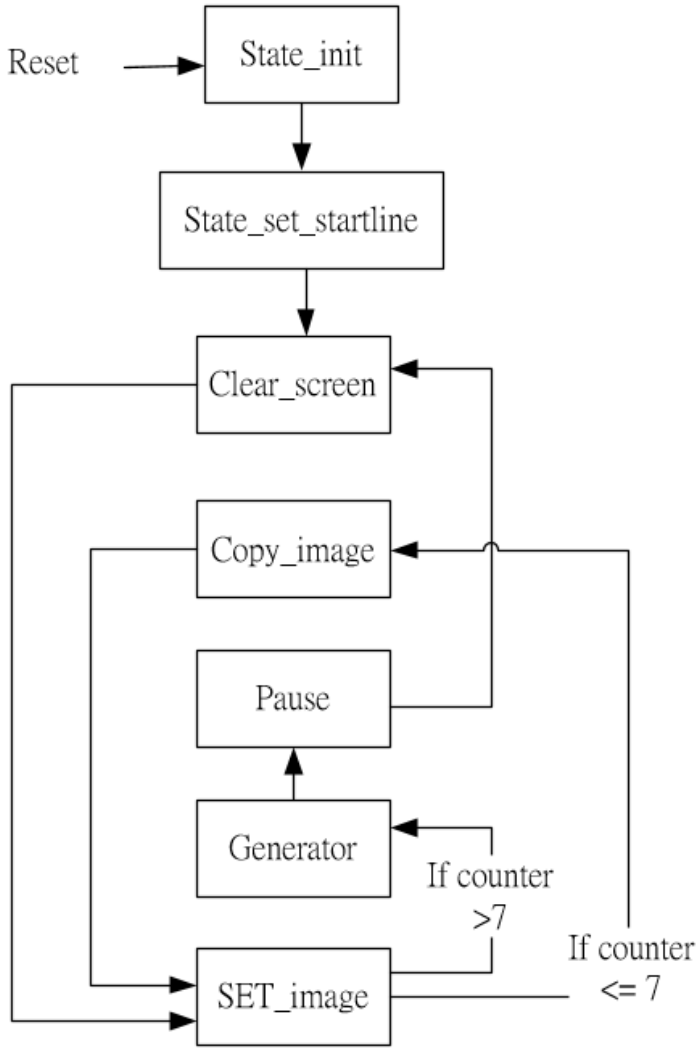
```
if (counter != ( 24'hDF_FFFF)) begin
        counter_next = counter + 1'd1;
        tone_next = tone;                    end
else tone_next = tone +1;
```

```
case (tone)
        10'h000: begin freq_div = RE; HIT = 1 ; GENERATOR = 0;end
        10'h001: begin freq_div = RE; HIT = 0 ; GENERATOR = 0;end ……
```

⊙ pacman：

以下為 pacman 主要的 **state transition**：

圖解： 基本上分成七種 state，除了助教原本就寫好的五種 state : initial, set_startline , Clear_screen , Copy_image , Pause 之外，另外加上兩個 state : Generator , SET_image。所有的 code 運算為了避免混亂與不同步，都會在這七種 state 中做完，加上加入換頁功能（助教給的 code 不能換頁），以及三種 16*16 圖片的畫法，整個程式碼變兩千行左右，十分龐大。



流程說明：init -> set_startline -> Clear_image -> SET_image -> Copy_image -> SET_image .... -> Copy_image ->Generator -> Pause -> Clear_screen （到此畫完一拍內的，最多八個音符）。

以下說明各七種 state 功能：

i. Initial : 與助教的 code 基本相同，加上了 CS2 =1，讓 LCD 能同時 initial 兩個頁面，還有當 reset 時將所有 XXXX_next 的變數 initial 成 0。

ii. Set_startline：與助教 code 相同，一樣加上 CS2 = 1。

iii. Clear_image：清掉整個頁面的同時，加上在最後 page 7 畫上一條線，next_state 改為 SET_image。

iv. SET_image：畫圖的部分在 Copy_image，而 set X_PAGE , Y 則移到 SET_image 這個 state。因此在進入 Copy_image 前都必須先進入 SET_image。控制八種圖案我使用了類似物件的概念，每種圖案都由一個物件控制，依序 set 八張圖，跳入 Copy_image 畫圖，在跳回 SET_image。運用 counter，當 counter 等於八時，表示八張圖都畫完了，即跳入 Generator state。

v. Copy_image：一樣用於畫圖，這邊還加上了判斷換頁的功能。要注意的是在 START 這個 if 裡面，會判斷他是否有按下 keypad 的按鈕。並在此處加分。接著在最後 PAGE_COUNTER = 1 時（表示將圖完整畫完），會跳回 SET_image 這個 state。

vi. Generator ：用於新增音符，音樂的部分有限制最多四拍內只有八個圖案，因此控制的物件也只有八個。會這樣寫是因為 verilog 無法動態產生物件，只能寫死。使用物件的順序是輪流使用，最先被 new 的一定會最先被用掉。我用了 note_X , note_Y , note_image 三種變數來控制一組物件。新產生的物件 X 一定會是 0（從上往下掉）。而 Y 與 image 的值都由亂數產生。Generator 完該拍子的音符後會進入 Pause 階段。

vii. Pause ：用於停頓，與助教的 code 相同，next_state 為 Clear_image，而這邊我有更改 Delay 的部分，讓音樂能跟 LCD 同步。

Part III. Issues and problems worthy of note and solutions

1. Converting BCD from heximal to decimal

- **Problem -** When trying to convert BCD from displaying heximal numbers to decimal numbers, there were some really weird problems. When I was meant to add from 9 to 10, it suddenly jump from 9 to really huge numbers, way larger than the number ten. I couldn't find out the reasons, since I was pretty sure that the instructions couldn't run into if block that I have set. After a long time of studying the code, I figured that it actually did run into that block of code and the reason turns out to be the unsynchronization of the clock rate.

- **Solution** – The display of the scores was initially set to have the clock rate of 1KHZ and this block of code is running in the module where the LCD is displayed. The LCD, instead of 1KHZ, it runs at the rate of 100KHZ. Due to the reason, the display will then have some little trouble when displaying the digits. To solve this problem, I moved the function of calculating the points out of the **PACMAN** module, and created a new module called **points**. Then by throwing the points into this new module and adjust the clock rate of the **display** module to the same as the **PACMAN** module, the program could now run properly and the conversion of the points is completed.

2. Synchronizing music

- **Problem** – Since this is a music game, it requires the notes and the characters of the game to appear simultaneously with the tempo of the music. So, this means that the frequency and clock rate of the LCD has to be almost the same. Initially, the LCD can't display correctly and it seems that it's drawing really quickly thus resulting the display picture to be really lag, or even not showing the pictures.

- **Solution** – After figuring out that the problem seems to be the synchronization of the music and LCD, we started to calculate the frequency of both of the modules. It's really hard to grasp the right value, since the relative error rate is only in floating points. So after a big amount of trying and testing, we found a perfect frequency for both of the module to run and

perform fluently.

3. Characters appearing in a weird pattern

- **Problem** – While some characters fall down perfectly from top to the bottom, some suddenly disappear before reaching to the bottom of the screen. This phenomenon only appears when to many characters appears together at the same time. The functionality is good when less pictures are provided on the screen.

- **Solution** – We have searched for solutions but in vain. Unlike some people that drew the pictures onto the LCD screen row by row, we drew the pictures in bitmaps 16 x 16, thus, we couldn't perfectly control where the pictures should appear and also because we preferred the characters to appear in a random way, so we couldn't make the pictures and the location fixed. Thus, sometimes when the LCD draws to about 6 or 7 characters at a time, the characters on the screen starts jumping out and disappearing in a weird way. The Solution for this problem hasn't been found yet.

4. Wires problem

- **Problem –** We found a really weird phenomenon with the connecting of the wires. For example, when we assigned "startnext= 1'b1", the program couldn't run or couldn't run properly, but when we changed to "reset = 1'b1", "startnext= reset", then the program could run perfectly. This is a really confusing situation that we couldn't really get. It seems to have something to do with the connecting of the wires.

- **Solution –** We sometimes meet this problem, and when this problem happens, we try to set a new variable to turn another way and find another path for the circuits to be connected. Though this solution works sometimes, but it's still a bit weird for us to have to connect the wires together this way.

Part IV . A list of reference materials or manual

1. How to convert a picture that we want from bitmap to the format on the LCD

   Reference site - http://en.radzio.dxp.pl/bitmap_converter/

Part V . Suggestions for this class

1. The deadline for the final project is a bit early. The project could maybe contain more functions and run more perfectly if we could have more time to work on it.

2. It would be better if the teacher could release a reference solution for the each lab through the semester, so that after each lab assignments, we could take a look of a better way to complete our assignment and compare to the ones that we've written ourselves. Through studying the differences between the codes, we could improve and think in different aspects.

3. Could the school provide us better FPGA boards? At the beginning of the semester, a lot of us couldn't attain a board that functions normally and had to borrow boards from classmates in the other class. It's really hard to work without the FPGA board, especially when we want to test our programs in the dorm. This sometimes results the efficiency of teamwork, since we always have to wait for the other one to finish testing, in order to share the board together.