EECS 2070 02 Digital Design Labs 2019 Lab 6
學號：107062314　姓名：陳柏均

0. 前言

這一次 lab06 要我們實作達文西密碼，雖說以前經常聽到這一個名詞， 但其實一直都不知道它在幹嘛; 也以為要理解它的意思就很困難了。 但之後我才發現它其實就是猜數字，我們都知道達文西密碼其實是一本著作，背後更隱藏了諸多宗教上、文學上的道理; 而這當然不是此次需要深究的部分。

就猜數字遊戲的部分，其實我們從小就常常在玩，給定一個範圍然後慢慢地去縮小一直到猜到正確數字為止，它真的滿貼近生活的。 同時，這也讓我聯想到資工系其他軟體課所學的找數字的許多方式，在這邊真的可以去做一些想法的結合。 這次 lab 也是初次操作鍵盤，我想這同時也是一大新挑戰，但我最終仍克服了它。

1. 實作過程

(i)理解鍵盤操作原理

最一開始由於對鍵盤的不熟悉，去看了講義的敘述稍懂它操作的道理，然後再去跑助教所給的 Sample code，我認為若這次沒給這個程式碼，將導致我做這一題時有一定的困難性。

將 Sample code 在 FPGA 版上顯示出來再慢慢看 Sample code 去理解它運作的原理，我認為這是這次的軸心之一，也是我花最多時間的部分。但我必須說 Sample code 真的十分有幫助，不會讓我一直窮想而打不出任何東西也學不到新觀念。

(ii)畫 State Diagram

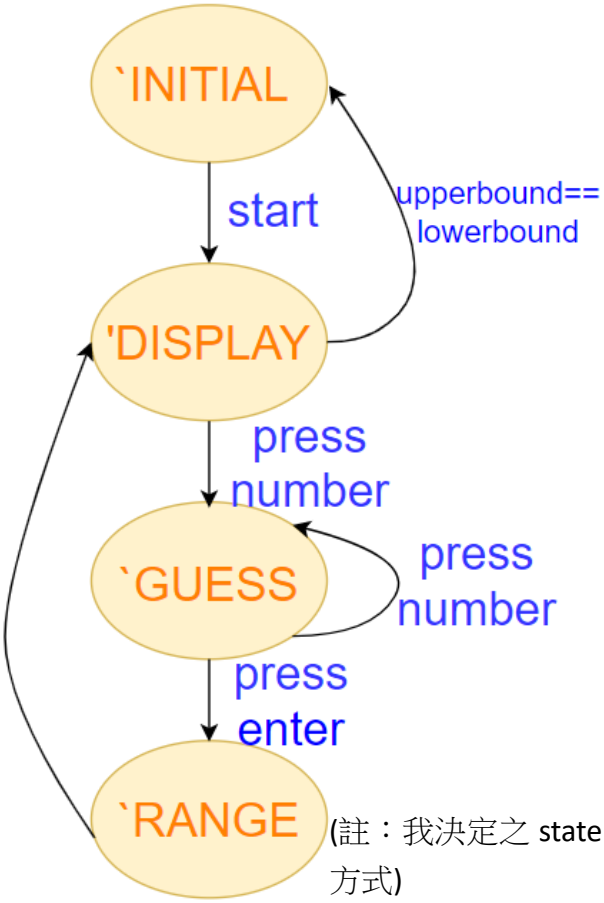
這次的題目很明顯要用 State Transition 的技巧，若不使用我相信要打出來有一定的困難性，同時若使用 State 來做轉換，程式碼不只能看起來很清晰且有條理許多，在進行 debug 的時候相信也輕鬆許多。

在這一題 lab 中，我總共使用了四個不同的 states，分別是 `INITIAL 代表初始狀態、`DISPLAY 代表顯示目前的數字範圍、`RANGE 是用來判斷目前所猜的數

字跟目前的範圍之關係而最後一個 state `GUESS 則是正在猜的狀態。

`INITIAL state 之時為初始以及 rst 後的狀態，如題目所要求，即在四組 seven segments 顯示四個一樣的中間橫線。 當按下 start 之後，進到 `DISPLAY state， 表示要顯示範圍的意思。

開始猜數字時跳到 `GUESS state 並一直留在此 state 直到使用者按下 enter 鍵或 rst button 為止。 而若按下 enter 鍵，則會從 `GUESS state 跳回 `RANGE state 比較數字再到 `DISPLAY 去顯示更新後的範圍。 以此方法直至猜到數字後到`RANGE state 再到`DISPLAY state 最終再將全部 LED 燈亮起並回到初始 `INITIAL state。下圖為我畫的簡易 State Transition Graph。



(註：我決定之 state 以及各個 state 之間大致轉換方式)

(註：以上是每種狀態大致的情形)

(iii)處理完大致的 states 之後，最終再回去處理一些 LED 以及每個 state 之中的小問題，包括許多在數字判斷以及顯示上的問題、或是對於 random number 的產生方式 clk 長短的控制，這些都是到最後在去深究。

雖說一邊寫就把所有的 bug 都更正也是一個方法，但如今我選擇採用這種大架構為主的方式我認為也不錯。更何況用清楚的 state diagram 既可以使我思路清晰、更幫助我打程式最難的部分，也就是 debug。

雖然說不全然可以確定錯的地方，但通常只要是那個地方的顯示方式有錯，往往歸咎於該 state 的寫法，也許是不夠嚴謹，也可能是想法完全錯誤。總之 state diagram 能助我較輕鬆解決這些複雜的問題。 以上是我這題 lab 大致的打法。

(註：這次 lab 的大概方向)

2. 學到的東西與遇到的困難

(i)verilog 數字底線的用法

在 verilog 中，當我要表達一個很長的 binary 數字時，我往往容易算錯位數，尤其是在很多重複數字出現的時候，很容易因為不小心就算錯到底要給幾個０或１。之前為了解決這一個問題，我使用的方法都是在前面加上一個純量以代表幾倍並且在後面用大括號括住我想要重複的數字。

之前的那種方法固然方便，但一直到這次我才發現一個新的方法也使自己不容易錯亂，那就是在數字中間加上底線。每四個 bits 便加上一個底線，即使是很長一串在算數量的時候較不易出錯，也不用因為這種不必要的錯誤再去花很多時間更正。

以往當打連續數個重複的０或１時，每次當想要 debug 之時又得再算一次，這明顯的不優，無論是用以上哪一種方式，我深信皆比最初時打的那種一長串無底線的 bits 的方式好很多。雖說這只是一個小技巧，但我認為實質上其實幫助頗大的，也是值得學習的。

```
=10'b1111100000;
```

(註：不易數數量的表示方式)

```
 nextmr1pos={1'b1,{15{1'b0}}};
nextmr3pos={{3{1'b1}},{13{1'b0}}};
```

```
nextupbd=8'b1001_1001;
nextlwbd=8'b0000_0000;
```

(註：後來認為比較好的兩種表示方式)

(ii)關於鍵盤的操作

其實我一直覺得 FPGA 版能接鍵盤滑鼠是一件很有趣的事，一個小小看似不起眼的板子竟然能有那麼多的功能。而這次 lab 很明顯的老師及助教要我們學的就是操作鍵盤的概念。

除了講義以外，我同時也稍微研究了助教給的 code，雖說真的不能全部都看得懂，但是對照講義還是能明白一些道理的。譬如之前其實從沒想過鍵盤在

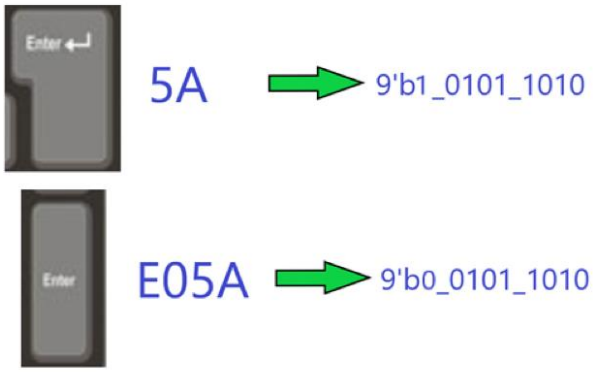
接上 FPGA 板時，是只傳 clk 跟 data 兩個東西；我原本竟然沒想到鍵盤上那麼多按鍵，其實只要賦予它們不一樣的值，就可以讓 FPGA 板了解使用者所真正想輸入的。

另外，對於按鍵的表達方式我也覺得十分的特別，使用了 extend code、break code、make code 分別表示重複的按鍵、是否放開(按鍵)以及按鍵之值。其中以 extend 我個人認為最令我印象深刻，因為我最初也沒想過若鍵盤的表示有所重複那該如何處理。

由於助教有給許多按鍵的寫法，因此我只要稍微模仿即可，像這次 lab 之中需要自己新加的按鍵不外乎就是 enter 鍵；而 Sample code 之中有 shift 這一個也有重複的按鍵，因此 enter 我也看著講義上給的鍵盤對照圖依樣畫葫蘆，當然過程中也有學到一定程度的知識以及獲得收穫。

```
9'b0_0110_1100, // right_7 => 6C
9'b0_0111_0101, // right_8 => 75
9'b0_0111_1101,  // right_9 => 7D
9'b1_0101_1010,  //left_enter
9'b0_0101_1010   //right_enter
```

(註：照著原本的 code 新增兩個 enter 鍵)



(註：兩種 enter 鍵表達方式)

(iii)分開寫左邊與右邊兩位

由於這次左邊兩位以及右邊兩位大部分情況是兩個獨立的東西，因此比起每次都指定數字給 16bits 的 nums，我覺得將前面兩個數字與後面兩個數字分開較清楚且對於我自己在打 code 及 debug 時相對較方便。然後最終再將

lownum(我左兩位數字表示方式)， highnum(我右兩位數字表示方式)指定給我需要 display 的 nums.

(註：如圖，我將 nums 再細分為左邊的 lownum 跟右邊的 highnum)

(iv)Cheat button 的做法

我認為有 cheat button 這個操作不只方便自己直接看到答案，也讓這個 lab 多了一個有趣的部份。不同於其他狀況，我的 cheat button 情形比較特別一點，由上面 state diagram graph 中可看出我並無設 cheat 這一個 state。一方面原因是我到後來才去做這個功能，另一方面是由於我認為 cheat 應該要與其他種分開且只有`INITIAL 這一個 state 無法使用，其他種 states 皆可以用。

我決定在 always block 之外另外再做操作，我的方法是當目前的 state 非 `INITIAL 且 cheat button 被按下時，則顯示出目標，也就是答案，不然的話就顯示 lownum 跟 highnum 的組合。如下圖所示。

```
assign nums=(state!=`INITIAL&&cheat==1'b1)?
        {goal,8'b1011_1011}:
        {lownum,highnum};
```

(註:如果 cheat 並且 state!= `INITIAL 便顯示答案，不然就照各 state 顯示應該顯示的數字)

(v)隨機亂數的作法

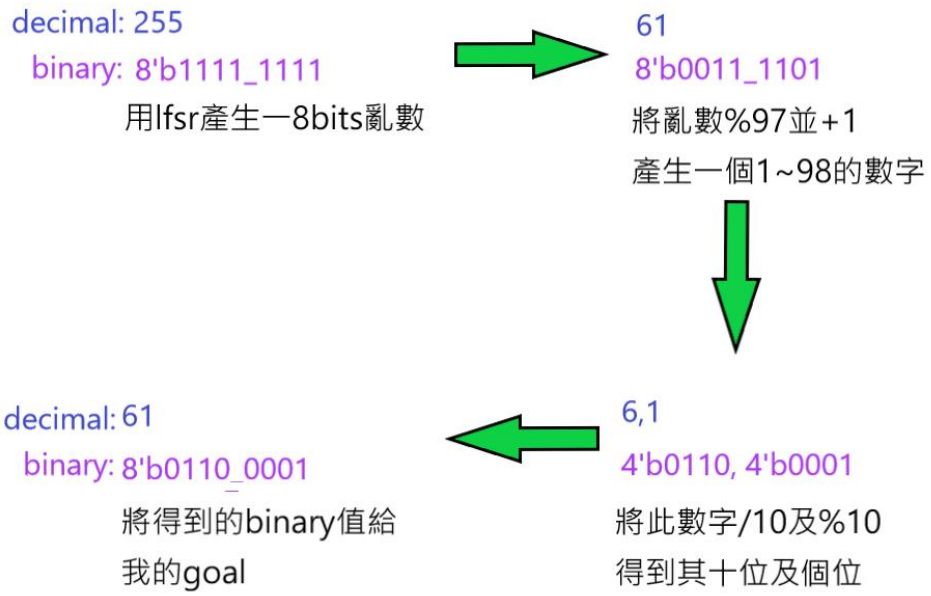
這是我這次 lab 之中覺得最有趣的部分之一，我將它放到最後才去做，之前就有在 lab 的 bonus 中學習到產生一個假亂數的方法，那就是 lfsr。以此方式我就可以照著目前的 clk 不斷去更新我這一個數字。由於題目要求數字必須到 98，因此我將此隨機亂數設了 8 個 bits。

接下來當我取得這一個數字後我將它%98(產生 0~97)再加上 1，如此一來我

便可以產生一個 1~98 的亂數。然後因為這一個 8bits 的亂數整串都是 binary 的表示法，而我的 goal(答案的那個數字)我希望是有點像 BCD 的表示方式，將個位與十位數分開以方便我去顯示以及做後面的諸多運算。

一開始的時候我不斷地會產生不顯示(我 seven segments 的 default)以及初始值的橫線(我 seven segments 中值為 4'b1011 的 val)，我不斷地去思考說我到底是哪裡寫錯了，因為那時感覺會亂數但卻又不知超出範圍的原因。

後來自己拿出紙筆仔細的去將思考的東西寫下來，才發現是表示方法的問題。原本我直接將亂數的值丟給我的 goal，以為這樣就對，後來才知道我是沒考慮到我的 goal 是用 BCD 的方式去表示而非整串代表一個數字。



(註:我產生一個亂數並指定給 goal)

3. 想對老師或助教說的話

這次 lab06 有一種互動的感覺，這就好像在玩一個有趣的遊戲一樣，更何況由於亂數的加入，又將使用者帶入一個更佳的境界。操作 FPGA 板的人也在猜對或是 cheat 之前不知道目前的目標為多少，我認為這真的是它好玩的地方。硬體課初次接觸鍵盤，覺得體驗十分佳，不斷地學習新事物並設計出一個個耐人尋味的結果真的帶給我無比的成就。

如今所學相信只是廣大學問中的一部分，但我相信這些知識便是通往

更佳境界的墊腳石。一次次 lab 下來，我不可否認每次都在進步，過程中難免會遇到 bug 一直除不掉而產生極大的挫折感，但同時也會在打的過程中得到一定程度的喜悅以及無比成就感。

達文西密碼應該是我到目前做過最有趣的一個題目了，完成的當下真的有一種筆墨也無法形容盡的快樂，其中包括獲得成就及知識還有能開始跟板子做有點像是互動性的事了；而非像之前好像只能按按鈕看著板子跑。

以前剛碰到邏輯設計這門課時，真的很排斥它，剛開始認為它抽象難懂，尤其是硬體的運作方式。但如今我學了許多才慢慢發現它非我曾經想像中那麼無趣，也非比軟體更難除錯，甚至有時候我覺得能從之中學到更多珍貴的東西。這次的 lab06 我經過了不少努力及挫敗終於得以完成它並獲得一個很棒的遊戲。

來分享一個可愛的笑話：
一天，小明說：「爸爸，講歷史故事給我聽呀！」
爸爸：「好～～從前，有一隻青蛙…。」
小明：「唉呀！我要聽歷史故事呀！」
爸爸：「好～～在宋朝，有一隻青蛙…。」