EECS 2070 02 Digital Design Labs 2019 Lab 6
學號：104021215　姓名：熊磊

1. 實作過程

- 先利用 clock_divider 做出需要的三種頻率，然後將 input 中的 rst, cheat, start 做 debounce 及 one pulse 的處理。

```
87  clock_divider #(.n(16)) c16(.clk(clk), .clk_div(clk_d));
88  clock_divider #(.n(26)) c26(.clk(clk), .clk_div(clk_led));
89  clock_divider #(.n(13)) c13(.clk(clk), .clk_div(clk_display));
90
91  debounce dreset(.pb_debounced(rst_d), .pb(rst), .clk(clk_d));
92  debounce dcheat(.pb_debounced(cheat_d), .pb(cheat), .clk(clk_d));
93  debounce dstart(.pb_debounced(start_d), .pb(start), .clk(clk_d));
94  OnePulse oreset(.signal(rst_d), .signal_single_pulse(rst_1pulse), .clock(clk_d));
95  OnePulse ostart(.signal(start_d), .signal_single_pulse(start_1pulse), .clock(clk_d));
```

- 設一個 clk_used 的 wire，當 state 為 SUCCESS 時，就跟著 clk_led，否則就跟著 Basys3 的 clk 跑。

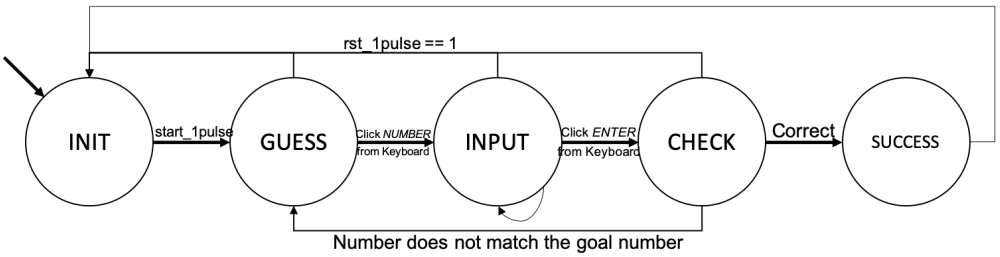
```
84  assign clk_used = (state == `SUCCESS) ? clk_led : clk;
```

- 把 posedege clk_used 及 posedge rst_1pulse 當作一個 always block 被觸發的條件，在裡面會將①state, ②min1, min0, ③max1, max0, ④goal1, goal0, ⑤D0, D1, D2, D3 這些 signal 做 reset 或是 update。

  其中{min1,min0}表示目前小於 goal 猜測的最大整數；{max1,max0}表示目前大於 goal 猜測的最小整數。

```
208 always@(posedge clk_used, posedge rst_1pulse) begin
209     if(rst_1pulse == 1) begin
210         state = `INIT;
211         min1 = 4'b0;
212         min0 = 4'b0;
213         max1 = 4'b1001;
214         max0 = 4'b1001;
215         goal1 = 4'b0;
216         goal0 = 4'b0;
217         D0 = next_D0;
218         D1 = next_D1;
219         D2 = next_D2;
220         D3 = next_D3;
221     end
222     else begin
223         state = next_state;
224         min1 = next_min1;
225         min0 = next_min0;
226         max1 = next_max1;
227         max0 = next_max0;
228         goal1 = next_goal1;
229         goal0 = next_goal0;
230         D0 = next_D0;
231         D1 = next_D1;
232         D2 = next_D2;
233         D3 = next_D3;
234     end
235 end
```

- 這次總共用到五個 state，大致的 FSM 切換 trigger 如圖所示



- 用一個@*的 always block 來維護所有我需要用到的值，包含上述 8 個值，以下將針對各個 state 討論。

- 在 **INIT** 時，在進到這個 state 時把 min 設為 00，max 設為 99，並把 7-segment Display 的直設成 `----`。此時會利用 LFSR 的 module 來生成亂數，在 INIT 時如果按下 start_1pulse，會把 random 的值儲存到 next_goal，並會將 state 切換到 **GUESS**。

```
next_goal1 = (start_1pulse) ? random%4'd10 : goal1;
next_goal0 = (start_1pulse) ? ~random%4'd10 : goal0;
```

- 在 **GUESS** 時，顯示器會顯示目前已猜測到的數字範圍，左邊兩個 Digits 是 min，右邊兩個 Digits 是 Max。開始進行猜數字時，意即當收到 Keyboard 數字鍵的 Signal 時，就會把 state 切到 **INPUT**。

- 在 **INPUT** 時，只要使用者鍵入任何數字，個位數都會替換掉十位數，而鍵入的數字會放在個位數。當使用者按下 Enter 時，即會將 state 切入 **CHECK**。

- 在 **CHECK** 時，如果使用者鍵入的數字只有一個位數，則不更改 min 和 max 值，直接重回 **GUESS** 的 state。而其他情況會跟 goal 做比較，如果跟 goal 的數字相同則進入 **SUCCESS** 的 state；其餘，如果大於 goal，小於 max，則把 next_max 設為鍵入的數字。如果小於 goal，大於 min，則把 next_min 設為鍵入的數字。其他情況則不更新 next_min、next_max，直接回到 **GUESS** 的 state。

```
164     `CHECK: begin
165         next_state = (nums[7:0] == {goal1,goal0}) ? `SUCCESS : `GUESS;
166         next_min1 = (nums[7:4] != 4'b1111 && nums[7:0] > {min1,min0} && nums[7:0] < {goal1,goal0}) ? nums[7:4] : min1;
167         next_min0 = (nums[7:4] != 4'b1111 && nums[7:0] > {min1,min0} && nums[7:0] < {goal1,goal0}) ? nums[3:0] : min0;
168         next_max1 = (nums[7:4] != 4'b1111 && nums[7:0] < {max1,max0} && nums[7:0] > {goal1,goal0}) ? nums[7:4] : max1;
169         next_max0 = (nums[7:4] != 4'b1111 && nums[7:0] < {max1,max0} && nums[7:0] > {goal1,goal0}) ? nums[3:0] : max0;
170         next_D0    = (nums[7:0] == {goal1,goal0}) ? goal0 : max0;
171         next_D1    = (nums[7:0] == {goal1,goal0}) ? goal1 : max1;
172         next_D2    = (nums[7:0] == {goal1,goal0}) ? goal0 : min0;
173         next_D3    = (nums[7:0] == {goal1,goal0}) ? goal1 : min1;
174         next_goal1 = goal1;
175         next_goal0 = goal0;
176     end
```

- 在 SUCCESS 時，把左邊兩個 digits 和右邊兩個 digits 都顯示 goal 的 numbers。

2. 學到的東西與遇到的困難

   這次作業做起來都蠻順利的，但是後面在測試的時候出現很奇怪的 bug，就是如果輸入的值不是 goal，而會更新 min 的值時，漸入的值會蓋在 max 的值上面，造成 max 值顯示的很奇怪。後來發現是 clk 設定的問題，把 clk 換成一樣的就解決了。

3. 想對老師或助教說的話

   謝謝老師與助教!

   在 demo 時，助教幫我找出上面那個 bug 可能發生的原因，所以才能解決並順利完成 demo。還在思考期末有沒有有趣的 idea 可以做出來，想做娃娃機和拉霸這些有趣的 project，但是做一樣的沒什麼意思，希望有點創新，不知道助教有沒有有趣的想法。