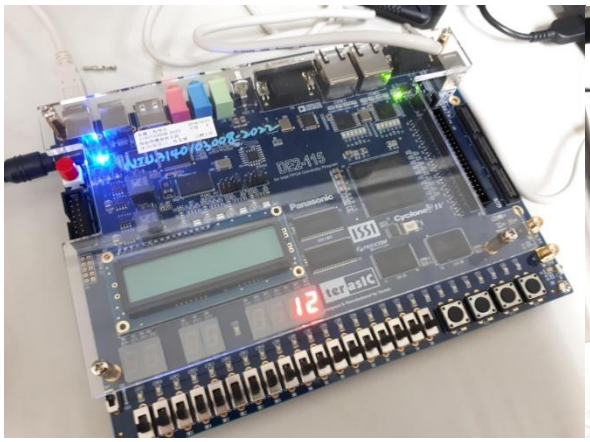
數位電路實驗 Lab1 report
亂數產生器

組別：team09
組員：鐘民憲(B06901017)
吳睿哲(B06901018)
謝兆和(B06901026)

一、使用器材與架設方式



FPGA 板

筆電

傳輸線

電源線

二、使用方式與詳細步驟

1.按下 KEY1 進行 reset。

2.按下 KEY0 後，亂數產生器便會開始運作，亂數會顯示於 HEX0 和 HEX1 上，數字範圍為 0~15，數字跳動的頻率會逐漸變慢，最後停在一個數字上。

3.(1)在亂數產生後按下 KEY0 的話，便會重複步驟 2 產生下一個亂數，且此亂數產生過程的 sequence 與前次不同。

(2)在亂數產生後按下 KEY1 的話，數字便會被 reset 成 0。

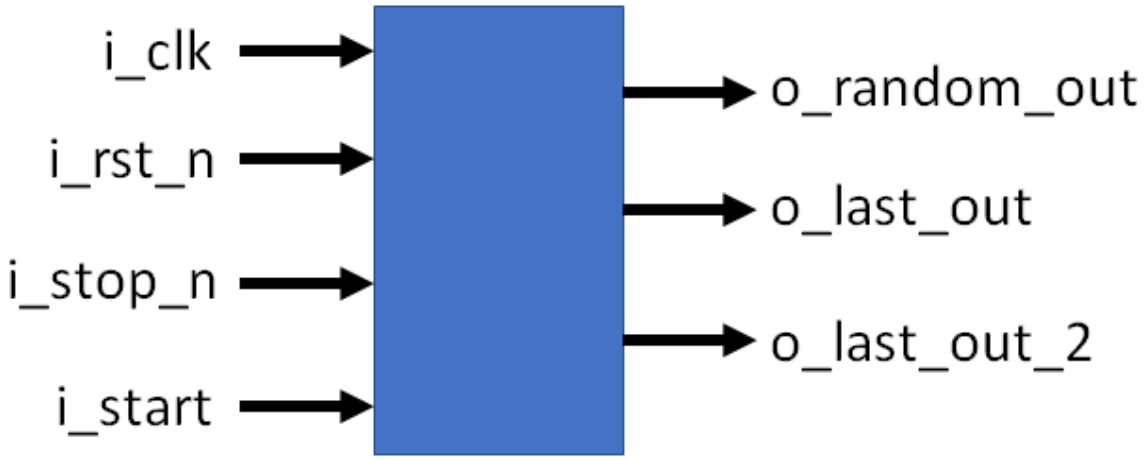
bonus1-擷取亂數：在亂數跳動的過程中按下 KEY2，會立刻停在當前的數字上。

bonus2-記憶前次亂數結果：按下 KEY0 後，在 HEX4 及 HEX5 會顯示上次產生的亂數。

bonus3-reset 後的亂數序列不同：藉由測量燒錄進去到按 KEY1 的 clock cycle 作為亂數種子。

三、實作設計技術細節及巧思

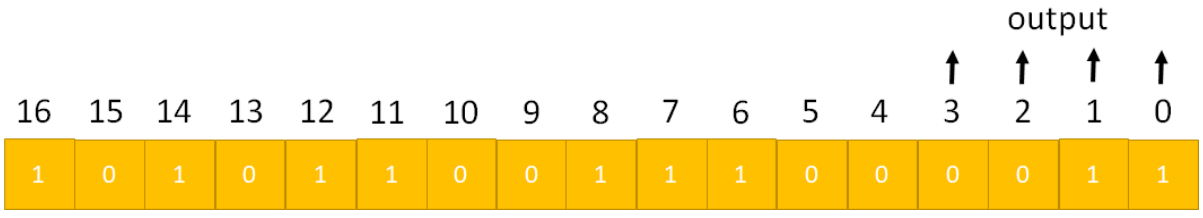
```
module Top (
        input   i_clk,
        input   i_rst_n,
        input   i_stop_n, //KEY2
        input   i_start,
        output [3:0] o_random_out,
        output [3:0] o_last_out //記憶上次的 output
        output [3:0] o_last_out_2 //記憶上上次的 output
);
```



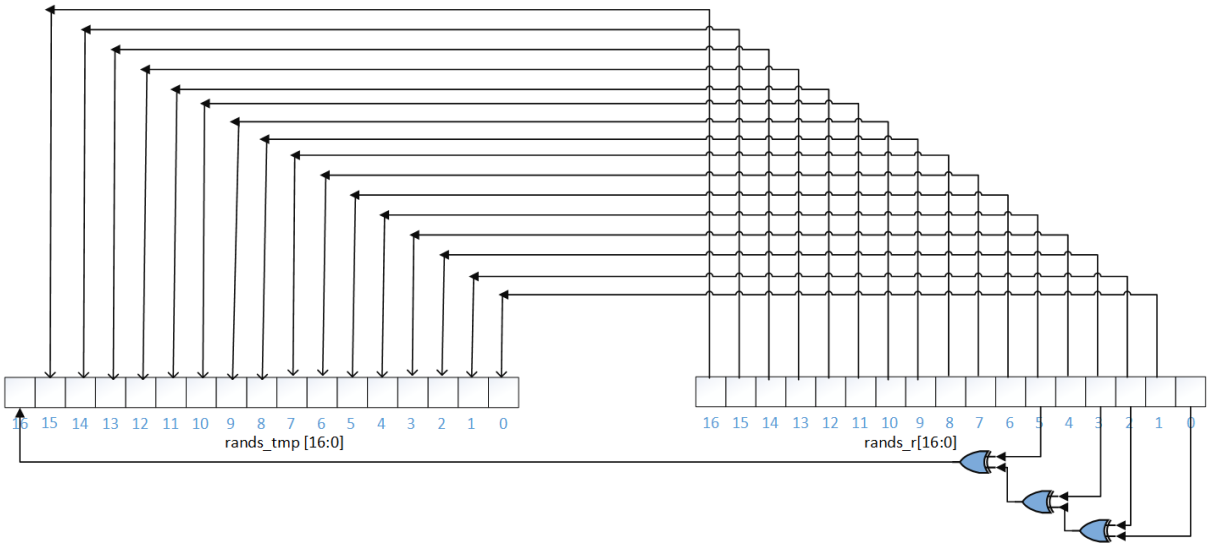
Registers：

(0)output buffer：在 clock positive edge 的時候，register 會傳遞數值，整個系統進行運算，此時若沒有 output buffer 的話，結果可能會在 clock middle 的時候才 output，若是有 output buffer 的話，便可以讓結果剛好在每個 clock positive edge 的時候 output。

(1)rands：17 個 bits，儲存 0101 的 random sequence，output 取用最後 4 個 bits。



(2)rands_tmp：將 rands 的 sequence 用 linear feedback shift register 的方式計算後儲存在此 register 中，也就是下一個 random sequence，可視為 rands 的 buffer。



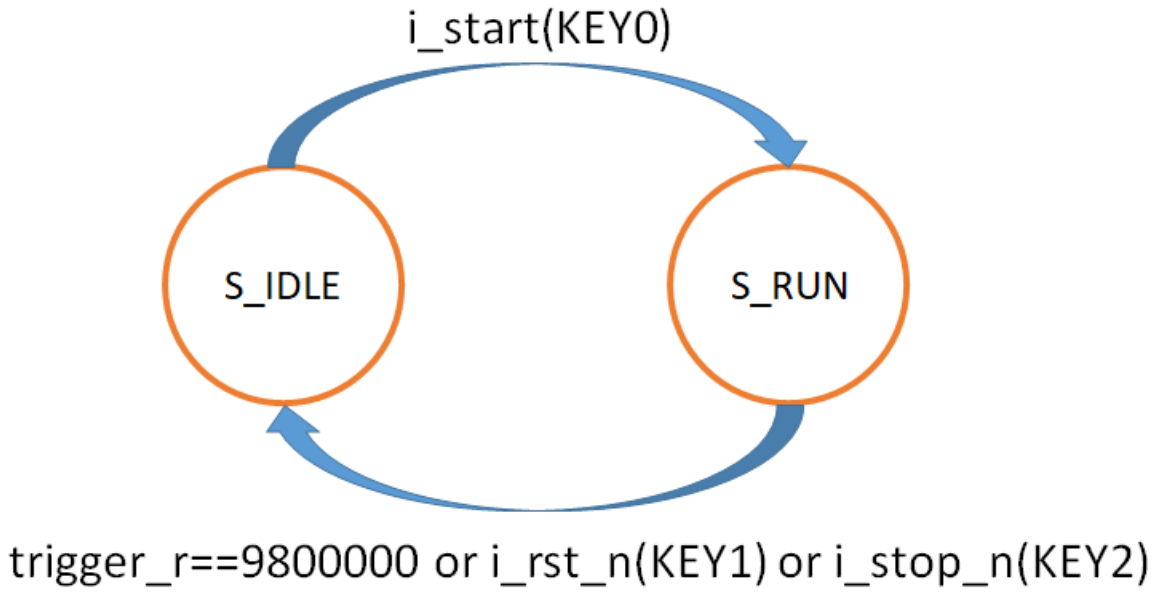
(3)state：用於 finite state macine

(4)counter：每個 clock cycle+1，記錄經過了幾個 clock cycle。

(5)trigger：用於實現亂數跳動減慢以及停止，當 counter==trigger 時，改變 rands 和 rands_tmp 的值，counter 的值歸零，trigger 的值等差增加

(6)seed：亂數種子，從燒錄進去後就不斷+1，不會被 reset，以此作為 rands_tmp initial 的值。

Finite State Machine：



四、碰到的問題或挑戰與解決方式

1. 一開始，我們無法正確把 example code 燒到 FPGA 板裡面，主要是因為對 Quartus II 工作環境及 systemverilog 的不熟悉，後來發現是因為沒有 import qsv 檔，因此很快就解決了。

2. 本來我們希望能把電路拆成 sequential part 及 combinational part 分工完成，然而本次實驗只有一個 module, 電路各部份關係太緊密，導致最後無法明確分工。
3. 我們本來把一部分的邏輯運算寫在 sequential part, 一部分寫在 combinational part, 後來所有計算都移到 combinational, sequential 只負責 register 的 reset 或資料傳遞。
4.在產生亂數時，我們一開始使用固定的初始值，可是這樣每次 reset 後的亂數就會一樣，因此決定自己產生 seed。然而以前寫 c++慣用的 time(NULL)在 FPGA 上不能使用，我們決定使用程式燒進 FPGA 後，計算經過的 clock cycle 產生 seed。雖然 seed 在 reset 時不能 initialization, 但 seed 本來就是要亂的，所以此方案可行。
5. ncsim 的 clock cycle 大約比電路板的 clock cycle 慢的 4 個數量級，我們的 trigger 設定是配合電路板，因此模擬時亂數看起來不會跳(第一次跳動從 0.1 sec 變成 1000 secs)，一度以為是 code 問題，後來才找到癥結點。
6. 即使電腦模擬沒有 error, 用 Quartus ii 燒 FPGA 時卻還是出現 error. 原因是在 conbinational part 的 case 除了 S_IDLE 跟 S_RUN 之外，對於每一個 logic 都要給 default 值，這樣在 FPGA 中才不會產生 latch。
7.我們有進行實驗去調整數字跳動的頻率，我們測試 trigger 以指數增加時，前後的時間間隔落差太大，數字一下子就停下來，所以最後我們決定以等差增加，看起來跳的比較賞心悅目。