|  |
| --- |
| **Computer-Aided Design for VLSI Design** |
| Homework4（Student ID: 61047046S｜Name: 林宣佑） |

1. Provide a simple explanation of your code.

這個專案的目標是實現一個隨機數生成器，使用 CA.vhd、RNG.vhd 和 test\_bench\_3.vhd 三個元件共同合作。

CA.vhd：這個元件代表一個有限自動機，根據輸入的 adr（地址）信號選擇相應的輸出 Y。

RNG.vhd：這個元件利用 CA.vhd 元件組成一個反饋移位寄存器，通過將前一個輸出作為輸入並進行適當的位元操作，生成隨機數。

test\_bench\_3.vhd：這個元件是測試環境，用於驅動 RNG 元件並模擬數據的輸入和輸出。

CA：

在entity CA中定義了它的實體。它具有五個port：

* adr (位址)：輸入訊號，用於選擇特定的輸入資料。
* clk (時脈)：輸入訊號，用於驅動整個系統的時脈。
* sel (選擇)：輸入訊號，用於控制是否進行初始化。
* init (初始值)：輸入訊號，用於設定初始化值。
* Y (輸出)：輸出訊號，代表電路的輸出結果。

在architecture behavioral of CA中，先是定義了訊號X，根據adr的值，選擇輸出並存入X中。利用with的語法，可以根據不同的adr利用查表的方式找出對應的輸出。以我這次的程式碼為例，adr為0000時，便會將X的值設為0。

接下來我們定義了一個process，並偵測clk訊號。當rising\_edge時，若sel為1即會將X的值指派給輸出信號Y；反之若sel為0則會將初始值init指派給Y。

RNG：

在entity RNG中定義了它的實體。它具有四個port：

* seed (種子)：輸入訊號，64 位元的種子輸入，用於初始狀態。
* clk (時脈)：輸入訊號，用於驅動整個系統的時脈。
* sel (選擇)：輸入訊號，選擇輸入，用於控制亂數生成。
* rng\_out (初始值)：輸出訊號，代表隨機數生成器產生的隨機數值。

在architecture structure of RNG中，主要是將既有電路組合起來，故RNG為structure model。首先宣告了一個型別為CA\_input，用於儲存64個CA的陣列，並隨之將先前介紹的CA作為component引入。

接著定義兩個signal：作為儲存陣列的my\_CA\_input以及接收所有CA輸出的my\_CA\_output。

在processs中我們偵測my\_CA\_output若有變化時，使用for迴圈根

我們的規則產生i、j、k、l四個變數並將之組合成一個std\_logic\_vector寫入my\_CA\_input的每個CA中。

結束proccess的定義後透過CA\_generate區塊的程式碼將所有CA將其對應的訊號組合，CA的adr由my\_CA\_input(n)提供，clk和sel由外部輸入的clk和sel提供，init由seed(n)提供，最後所有CA的Y連接到my\_CA\_output(n)並組合成my\_CA\_output。

最後將my\_CA\_output寫入rng\_out並作為RNG.vhd的輸出。

test\_bench\_3：

Test\_bench\_3的設計是為了用於驗證RNG的元件功能，在entity Test\_bench\_3中定義了它的實體。

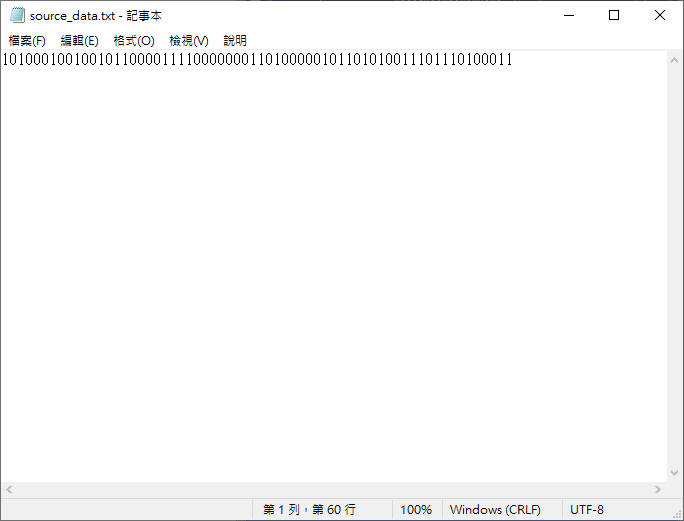
在architecture arch\_test of CA中。我們定義了四個訊號：

* seed(種子)：64-bit的random seed，從source\_data.txt提取。
* clk (時脈)：用於驅動整個系統的時脈，100ns為一個週期。
* sel (選擇)：起先設定0以初始化，在100ns時改變為1。
* rng\_out (輸出)：經測試由RNG元件輸出的64-bit隨機數。

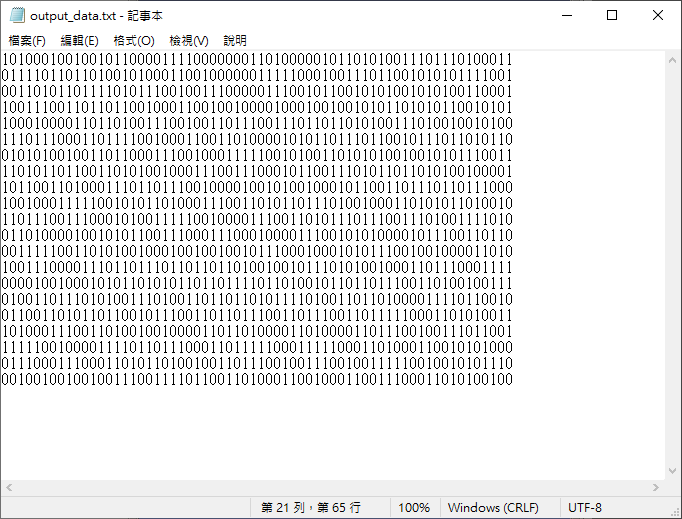
接著將以上訊號連接到RNG元件相對應的端口上，以模擬對RNG元件的輸入以及記錄他的輸出，測試過程必須先打開source\_data.txt，透過readline函式從其讀取一行數字，並使用read將其轉換為64-bit的型態指派給temp，最後將temp指派給seed。

待RNG運作後輸出隨機數，我們先檢查rng\_out是否有發生變化且不全等於U。若滿足條件則透過write函式將rng\_out寫入out\_data，再透過writeline函式將out\_data輸入至output\_data.txt中。

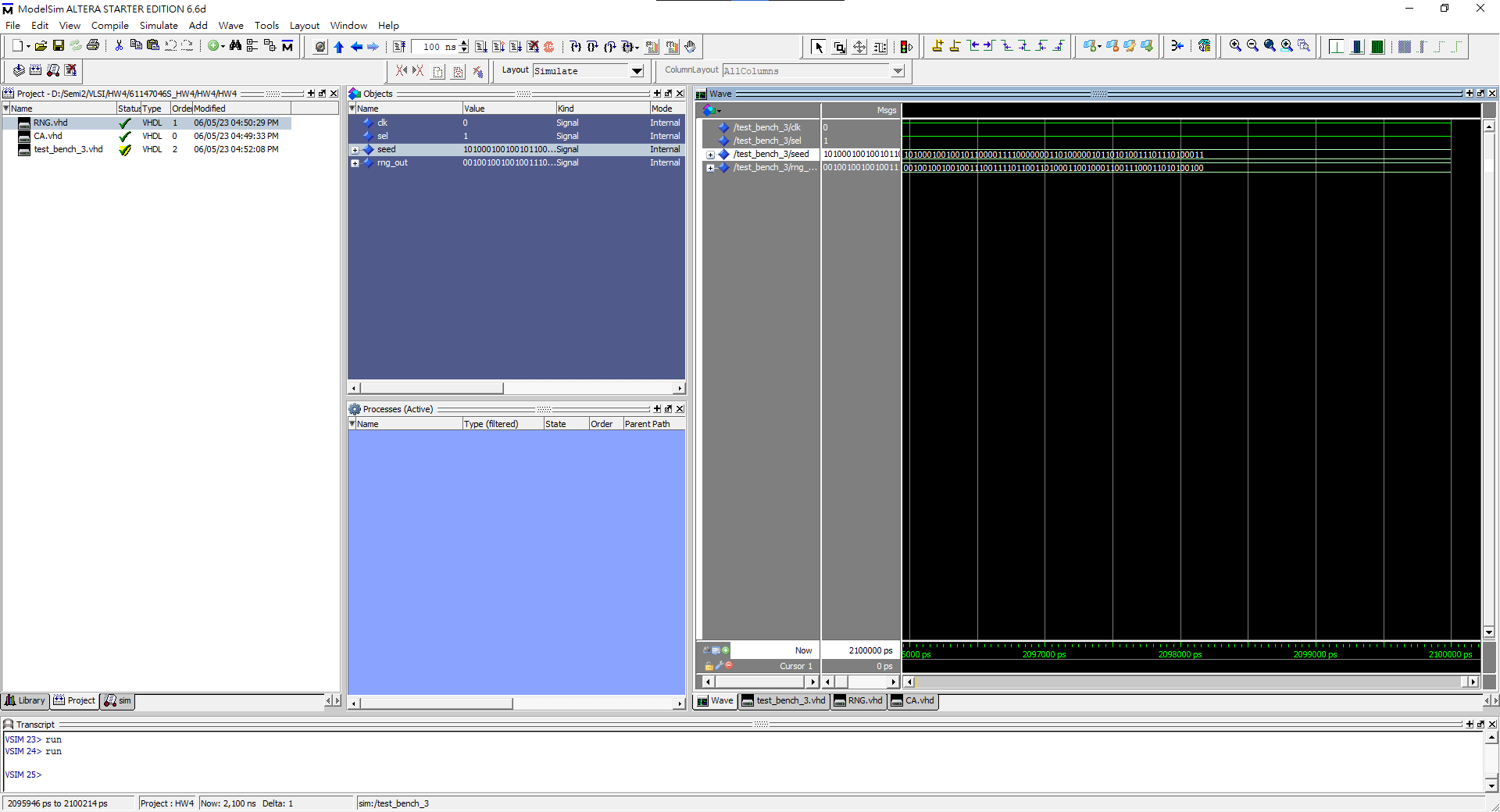
1. Waveform diagram here (Simulation Results)



圖表 1 Source\_data



圖表 2 Output\_data



圖表 3 Modelsim截圖

1. Reflections and discussions

在這次的專案中，我們建立了一個包含三個主要元件的組合。首先，我們有 CA.vhd，它代表了元件自動機（Cellular Automaton）的行為，根據給定的位址（adr）產生對應的輸出信號（Y）。接下來，我們有RNG.vhd，它使用多個 CA 元件組成的環形結構，從而生成隨機的輸出信號（rng\_out）。最後，我們透過test\_bench\_3.vhd測試這些元件的功能。它使用一個時脈信號（clk）來驅動模擬的時序行為，並在一開始將 sel 訊號設定為0將所有CA元件進行初始化。同時，從一個文字檔中讀取我們設定的random seed，將其作為輸入提供給 RNG 元件，並將 rng\_out 的結果寫入另一個文字檔作為模擬結果。

這次作業使用到我們這學期所學到所有的VHDL技能，從設計單個元件到可以將許多元件組成為得以實現隨機數產生器的電路，同時可以透過test bench來驗證元件的正確性以及隨機數生成的結果。

1. Optional

在加分題項目中必須透過HW4的RTL元件，將之產生的隨機結果經過module後轉換為0~20的數字，並取連續三個做為樂透號碼。

我撰寫了一隻Optional.vhd的主程式，在entity Optional中定義了它的實體。它具有五個port：

* clk (時脈)：輸入訊號，用於驅動整個系統的時脈。
* en (開啟)：輸入訊號，用於選擇特定的輸入資料。
* sel (選擇)：輸入訊號，用於控制是否進行初始化。
* seed (種子)：輸入訊號，64 位元的種子輸入，作為random seed。
* result (輸出)：輸出訊號，代表輸出的樂透號碼。

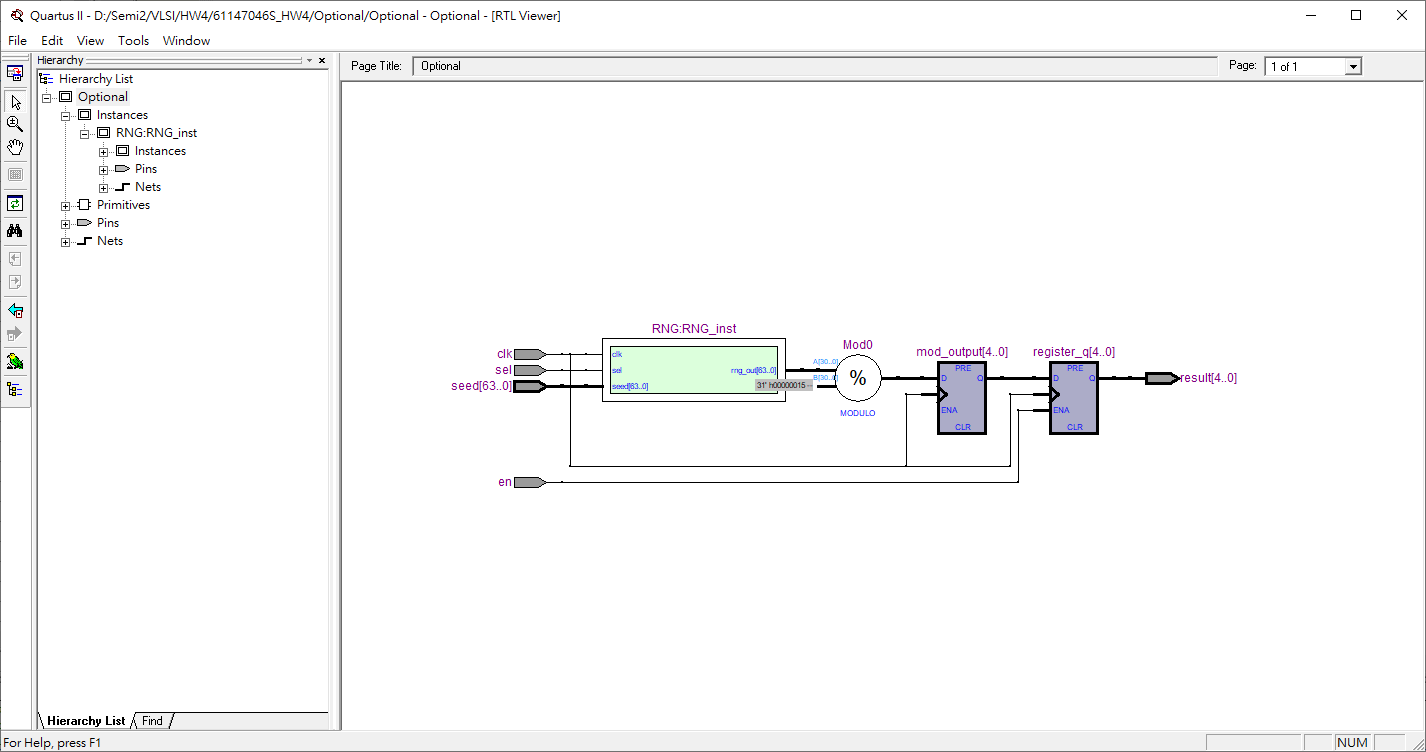
在architecture behavior of Optional中定義了它的行為。首先將RNG作為component引入，再來定義了三個訊號：rng\_output紀錄RNG的輸出、mod\_output紀錄rng\_output經module計算後的結果、regitster\_q作為D型正反器紀錄最終的輸出結果。

在process中我們會定義一個名為dec\_value的integer，以作為計算rng\_output的中繼型態，當rising\_edge觸發時將rng\_output轉換為無符號數再將之轉換為integer記錄至dec\_value中，接下來將dec\_value作module運算並將之轉換回unsigned型態最後再轉回std\_logic\_vector型態寫入mod\_output中。

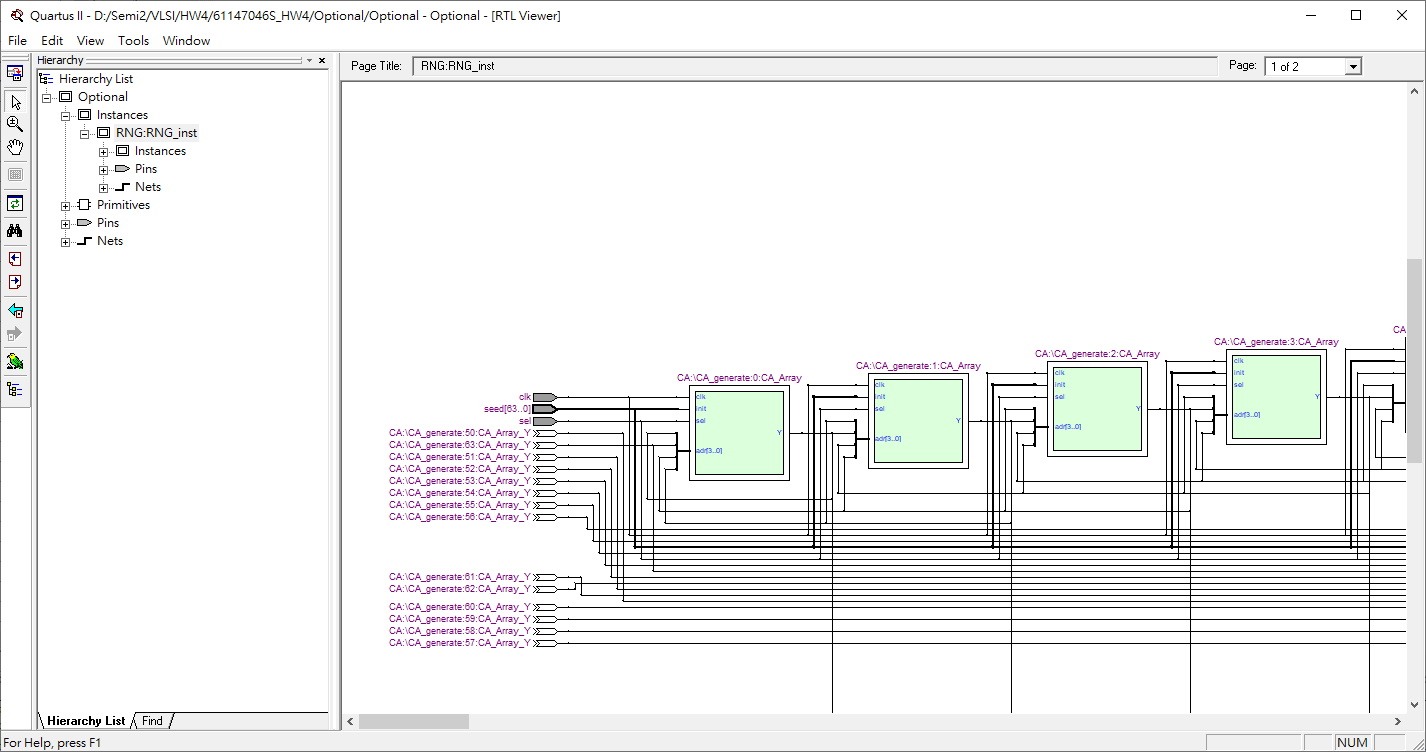
下個階段的process則是在rising\_edge觸發時確認en訊號是否允許將運算結果寫入D型正反器，若en為一就將結果寫入register\_q中。

最後將D型正反器的內容輸出。

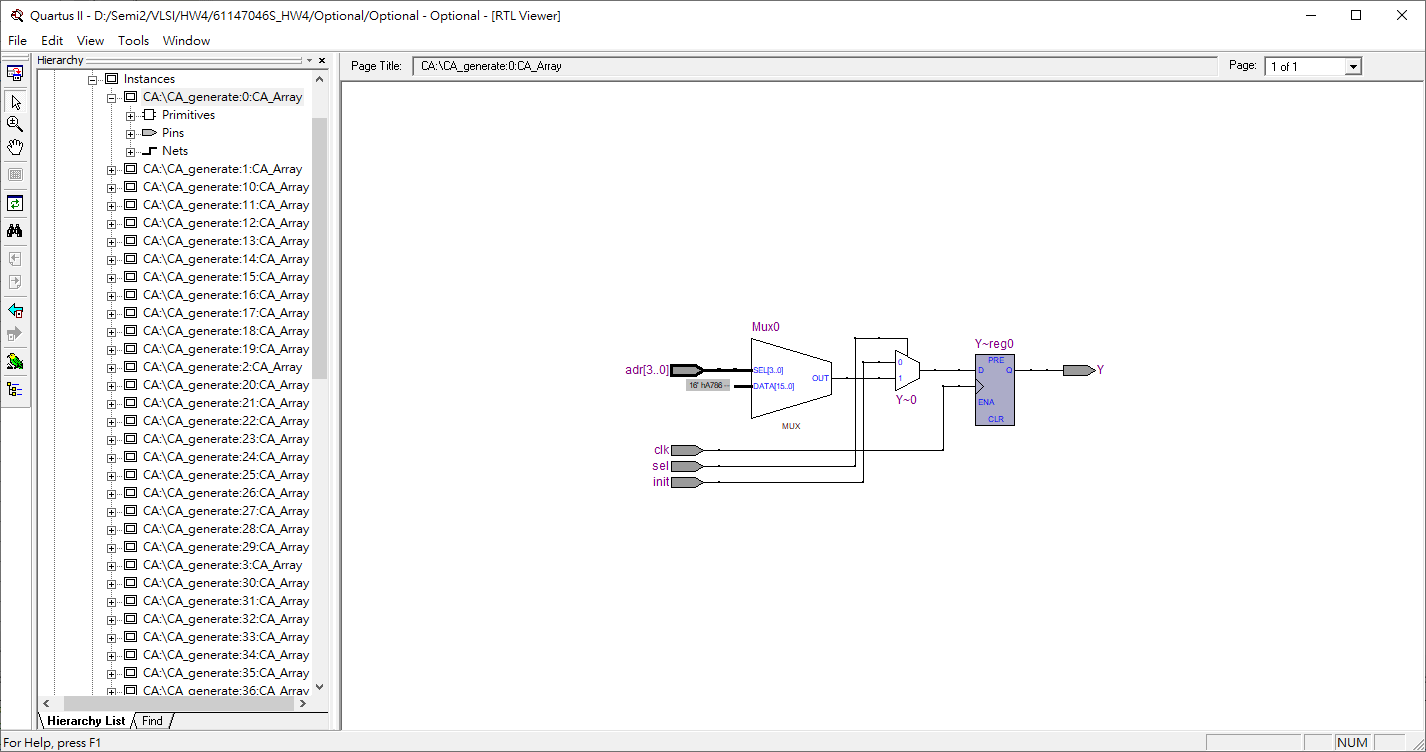
附圖為產生出的RTL電路圖：



圖表 4 Optional

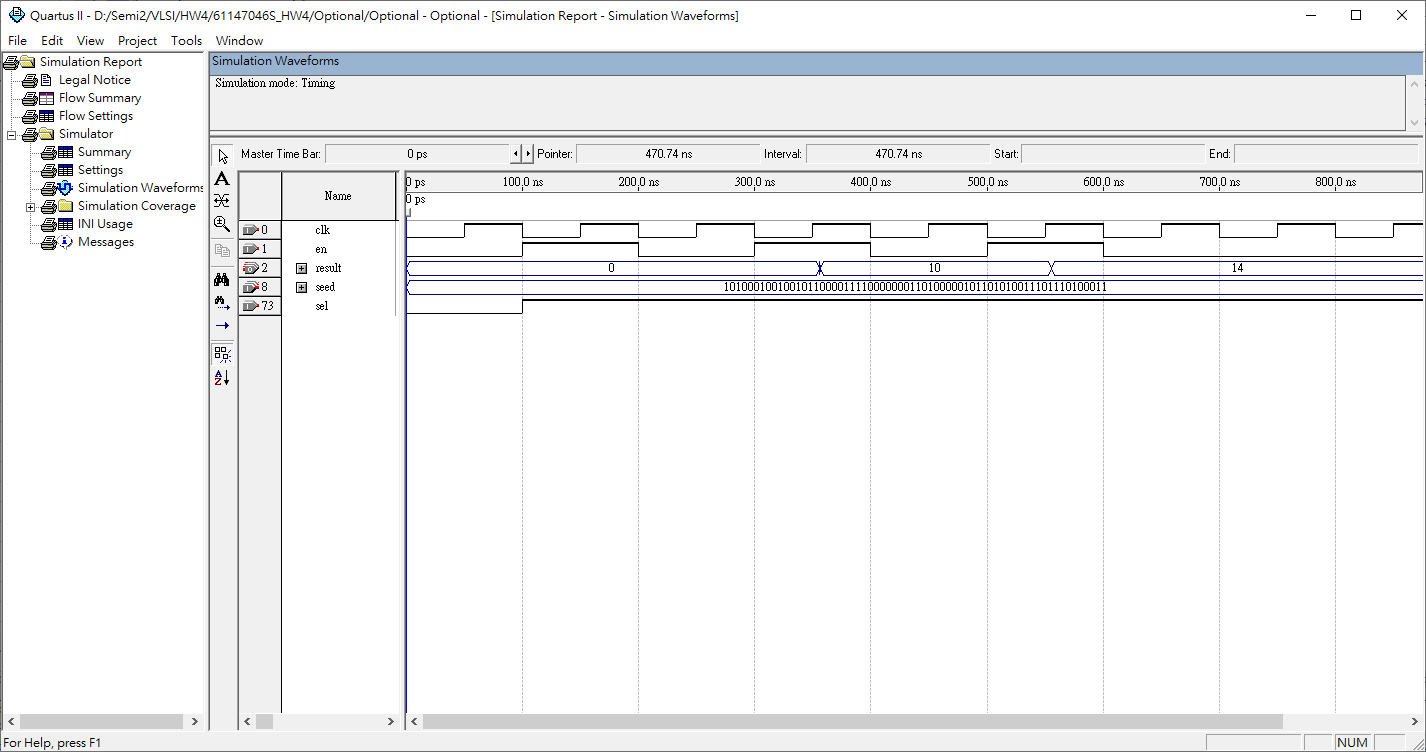


圖表 5 RNG



圖表 6 CA

最後是Optional產生的波形圖：



圖表 7 波形圖

可以觀察到當en=’1’時，result即會有不同的隨機數輸出，且經過3個en的rising\_edge後，因為en=’0’，所以result就不會再有變化，此次實驗輸出的樂透號碼為：0、10、14。