Integrated Circuit Design
Final project
Recurrent Unit Circuit Design

1. 問題描述

本題請完成一遞迴神經網路單元電路設計 (Recurrent Unit Circuit Design，以下稱 RNN 電路)，RNN 電路輸入為持續 t 時間的特徵描述 (input feature description)，電路需包含 RNN 的數學運算及暫存，最後輸出計算後的特徵描述 (output feature description)，簡化的運算處理流程，如下圖 1. 所示。

首先在時間 t 時，RNN 的輸入為 x_t 及 $h_{(t-1)}$，輸出為 h_t，而被稱為隱藏狀態 (hidden state) 的 h_t，會作為輸入傳入下一個時間點 t + 1。系統的運算由以下算式定義：

$$h_t = \sigma(W_{ih}x_t + b_{ih} + W_{hh}h_{(t-1)} + b_{hh})$$

其中，輸入的特徵描述 x_t 為長度 32 之向量，t 的總長度由系統給定；W_{ih} 及 W_{hh} 是運算的權重，而 B_{ih} 及 B_{hh} 是運算的偏差，它們的大小分別是 64 x 32、64 x 64、64 x 1、64 x 1，這四組參數會由系統給定並儲存在記憶體中供使用者圖取；σ 是 activation function，本題目中以簡化的 hard tanh 取代，其計算方式為三區段的線性方程式，如下圖 2. 所示；最後，電路的輸出 h_t 則為長度 64 之向量。

特別注意，在時間 t = 0 時，此時的系統尚未計算出 $h_{(t-1)}$，使用者需將其以 0 填滿作為隱藏狀態的初始值。另外，上述所有參數除 x_t 只會輸入一次外，W_{ih}、W_{hh}、B_{ih} 及 B_{hh}，都儲存在測試系統的記憶體中，使用者可以隨時存取。但同時是輸出結果及中間產物的 h_t 只可寫出不可讀取，使用者如有計算需要應自行留存。

請參考附錄中所列的要求，附上評分所需要的檔案。
設計檔案說明也請參考附錄。

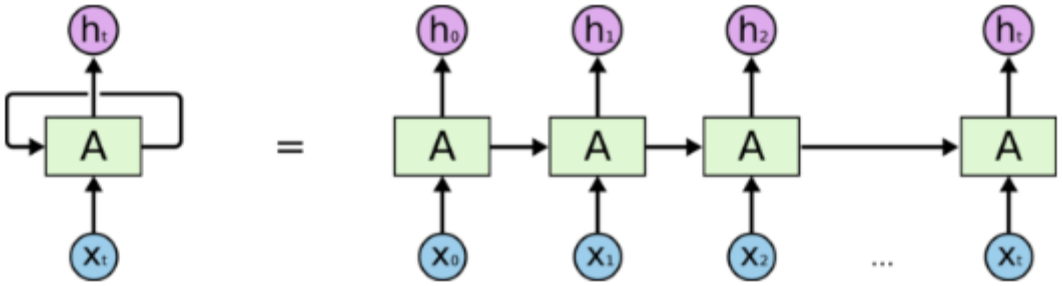

圖 1. 遞迴神經網路單元電路系統架構

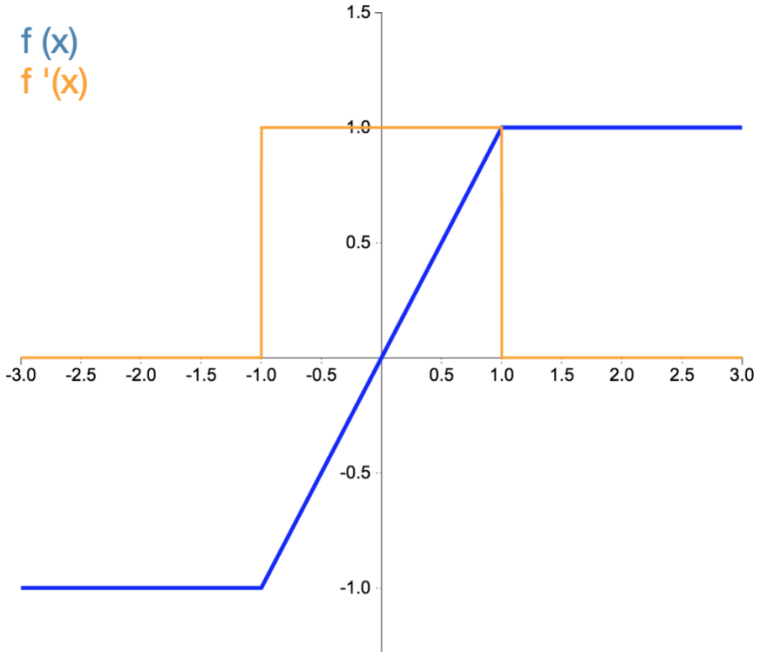

圖 2. 用以作為 activation function 的 hard tanh function (藍色線)

2. 設計規格

2.1 系統方塊圖

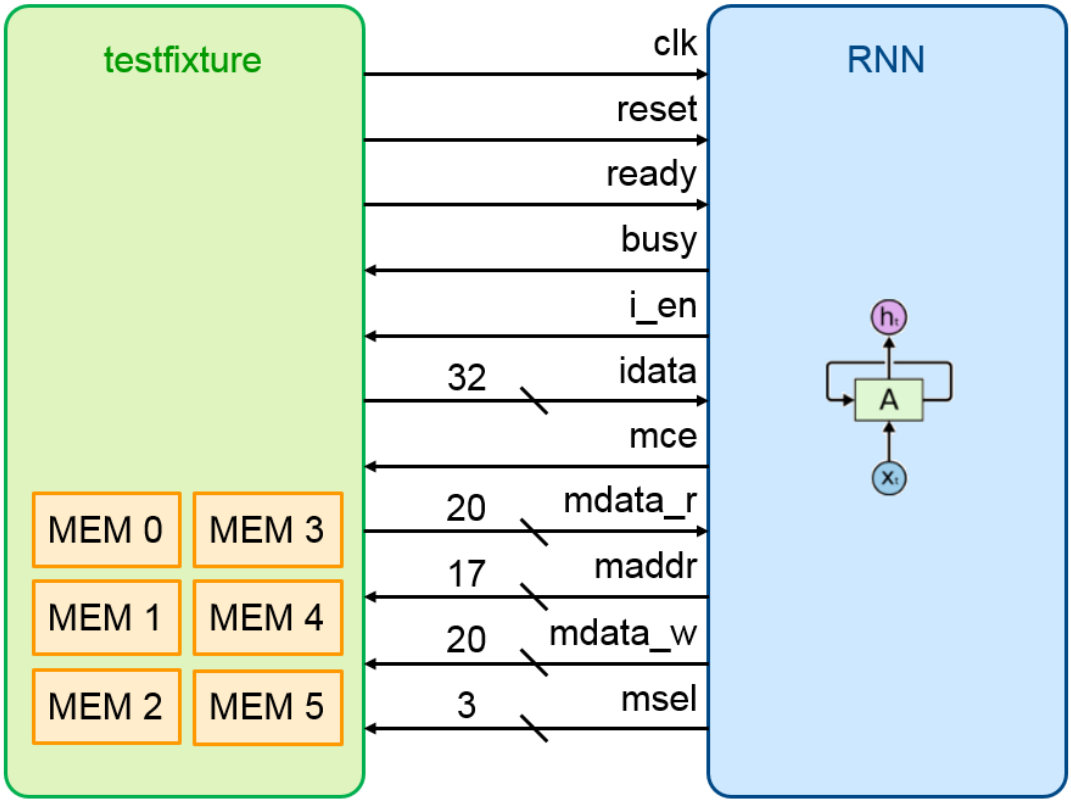

圖 3. 系統方塊圖

2.2 輸出入訊號和記憶體描述表一、輸入/輸出信號

Signal Name	I/O	Width	Simple Description
clk	I	1	系統時脈訊號。本系統為同步於時脈**正緣**之同步設計。
reset	I	1	**高**位準”非”同步(active low asynchronous)之系統重置信號。
ready	I	1	輸入及記憶體資料準備完成指示訊號。當訊號為 High 時，所有資料準備完成，此時 RNN 才可以開始向 testfixture 發送訊號要求。
busy	O	1	系統忙碌指示訊號。當 RNN 接收到 ready 訊號為 High，且 RNN 準備開始動作時，需將此訊號設為 High，表示準備開始進行輸入資料索取；待所有運算處理完成且輸出結果寫回 testfixture 後，需再將訊號設為 Low 表示動作結束。
i_en	O	1	RNN 模組輸入訊號控制訊號，當訊號設為 High 時，表示準備開始接收輸入資料，testfixture 會在下一個 cycle 將輸入資料送給 RNN。
idata	I	32	RNN 模組輸入訊號，由 32 個 1 bit 整數組成。i_en 為 High 時，testfixture 會將輸入資料用此訊號送給 RNN。輸入資料的總長度存在記憶體 4 內。
mce	O	1	RNN 外部記憶體控制訊號。當時脈正緣觸發時，若此訊號為 High，表示要進行讀寫動作。testfixture 會將 maddr 位址指示之資料讀取/寫入到記憶體中。
mdata_r	I	20	RNN 記憶體**讀取**訊號，由 4 bits 整數 (MSB) 加上 16 bits 小數 (LSB) 組成，**多為有號數**。當選擇的記憶體是記憶體 0~4 時，testfixture 會將記憶體上的資料傳送至 RNN 電路。
maddr	O	17	記憶體讀取/寫入位址。testfixture 會根據 msel 選擇記憶體，並決定要讀取/寫入該記憶體，該記憶體的讀取/寫入位址由此訊號決定。
mdata_w	O	20	RNN 運算結果記憶體**寫出**訊號，由 4 bits 整數 (MSB) 加上 16 bits 小數 (LSB) 組成，為**有號數**。當選擇的記憶體是記憶體 5 時，RNN 電路的運算結果會利用此訊號將結果輸出至 testfixture 並寫入記憶體 5。
msel	O	3	RNN 運算處理寫入/讀取記憶體選擇訊號。此訊號指示目前寫入/讀取資料的目標為哪一塊記憶體。說明如下: 3’b000: 表示選擇並**讀取**記憶體 0，內存有權重參數 W_{ih}。 3’b001: 表示選擇並**讀取**記憶體 1，內存有偏差參數 B_{ih}。 3’b010: 表示選擇並**讀取**記憶體 2，內存有權重參數 W_{hh}。 3’b011: 表示選擇並**讀取**記憶體 3，內存有偏差參數 B_{hh}。 3’b100: 表示選擇並**讀取**記憶體 4，內存有輸入資料總長度，會以 20 bits 整數，為**無號數**，經由 mdata_r 傳回，此記憶體無須 maddr 控制。 3’b101: 表示選擇並**寫入**記憶體 5，執行 RNN 的輸出結果。

2.3 系統功能、時序及記憶體對應方式描述

系統時序如圖 4. 所示。當 reset 啟動結束後，testfixture 會將 ready 訊號設為 High 表示輸入資料及記憶體資料都已經準備就緒，接著 RNN 電路須將 busy 訊號設為 High 表示開始動作 (如圖 4. t1 時間點)，而 testfixture 偵測到 busy 訊號為 High 後就會將 ready 訊號設為 Low 表示等待 RNN 電路處理題目所要求的動作 (如圖 4. t2 時間點)。待所有時間點的計算都實現完成並輸出後，RNN 電路就可將 busy 訊號再次設為 Low 表示所有動作已經完成 (如圖 4. t3 時間點)，此時 testfixture 就會準備下一組資料並將 ready 訊號設定為 High；並且 testfixture 一偵測到 busy 訊號再次被設定為 Low 就會立刻進行資料驗證比對，因此針對每一組輸入資料的運算處理，busy 訊號只允許在 RNN 電路開始動作時被設定為 High 一次，RNN 運算處理結束時設定為 Low 一次。在 busy 為 High 的過程中，RNN 電路可重複不限次數對 MEM0、MEM1、MEM2、MEM3 及 MEM4 進行讀取。

※ 助教評分時將視情況決定是否增加輸入樣本數或調整數入樣本長度進行測試。

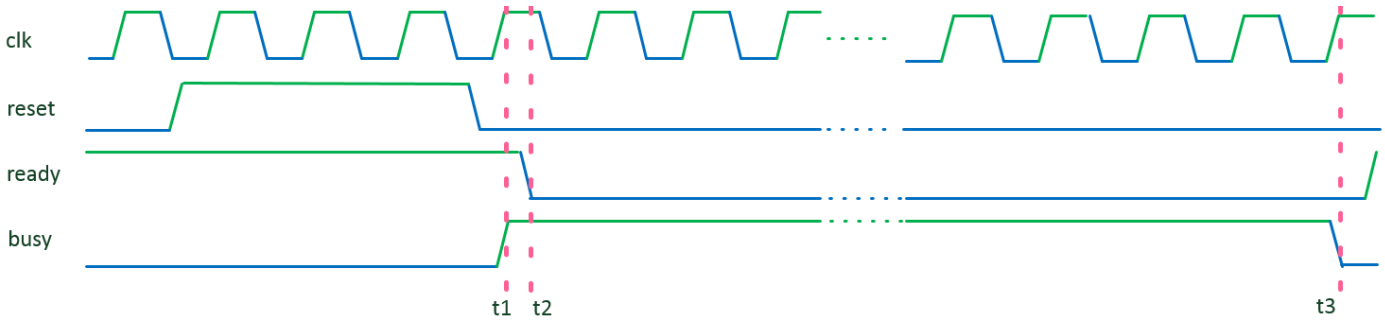

圖 4. 系統啟動及結束時序

1. x_t 為本題輸入資料，其尺寸為 32 bits (特徵向量長度) x 1000 (時間)，存放於 testfixture 的記憶體中。動作時序上，RNN 電路需利用 i_en 發送欲索取資料的訊號到 testfixture (如圖 5. t1 時間點)，testfixture 在每個時脈負緣後，會將每筆共 32 bits 的 x_t 資料利用 idata 送入 RNN 電路 (如圖 5. t2 時間點)，若 testfixture 在時脈負緣後，偵測到 i_en 為 Low 的情況下，將不會繼續變化 idata (如圖 5. t3 時間點)。

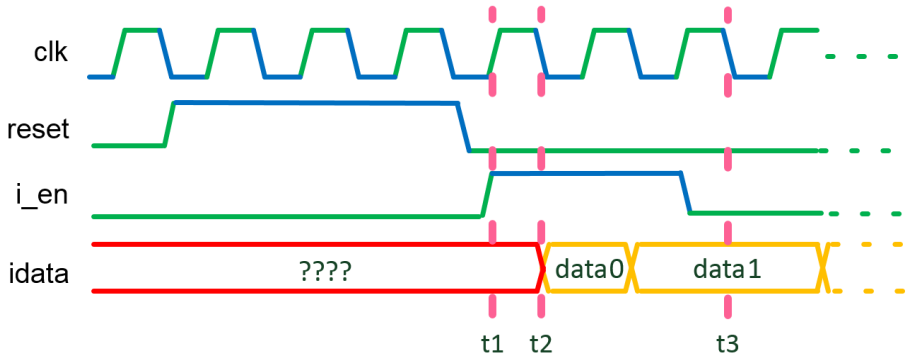

圖 5. 輸入資料 x_t 資料經 idata 輸入之時序

2. MEM0、MEM1、MEM2 及 MEM3 的記憶體內分別儲存著 W_{ih}、B_{ih}、W_{hh} 和 B_{hh}，其對應的記憶體大小分別是 64 x 32、64 x 1、64 x 64 和 64 x 1，其中每筆資料都是以 20 bits 表示 (4 bits 整數+16 bits 小數，**有號數**)。圖 6. 是以 64 x 32 的二維矩陣為例，矩陣座標與記憶體儲存的位址對應關係圖，每格的第一行為矩陣的座標，第二行為對應的記憶體位址，每格中都是一個 20 bits 定點小數 (fixed-point number)。

(0,0) 0	(0,1) 1	(0,2) 2	(0,3) 3	(0,4) 4	…	(0,31) 31
(1,0) 32	(1,1) 33	(1,2) 34	(1,3) 35	(1,4) 36	…	(1,31) 63
(2,0) 64	(2,1) 65	(2,2) 66	(2,3) 67	(2,4) 68	…	(2,31) 95
(3,0) 96	(3,1) 97	(3,2) 98	(3,3) 99	(3,4) 100	…	(3,31) 127
⋮	⋮	⋮	⋮	⋮	⋮	⋮
(63,0) 2016	(63,1) 2017	(63,2) 2018	(63,3) 2019	(63,4) 2020	…	(63,31) 2047

圖 6. 矩陣儲存於記憶體中的位址對應方式

3. 當要讀取記憶體 MEM0、MEM1、MEM2 及 MEM3 中的資料時，RNN 電路需將 mce 設為 High，並以 msel 選擇記憶體 0 ~ 3 (如圖 7. t1 時間點)，testfixture 在每個時脈負緣後，依照 msel 所選擇的記憶體及 maddr 所指示的位址取記憶體值，並利用 mdata_r 送入 RNN 電路 (如圖 7. t2 時間點)。圖 8. 為部分實際波形參考圖。

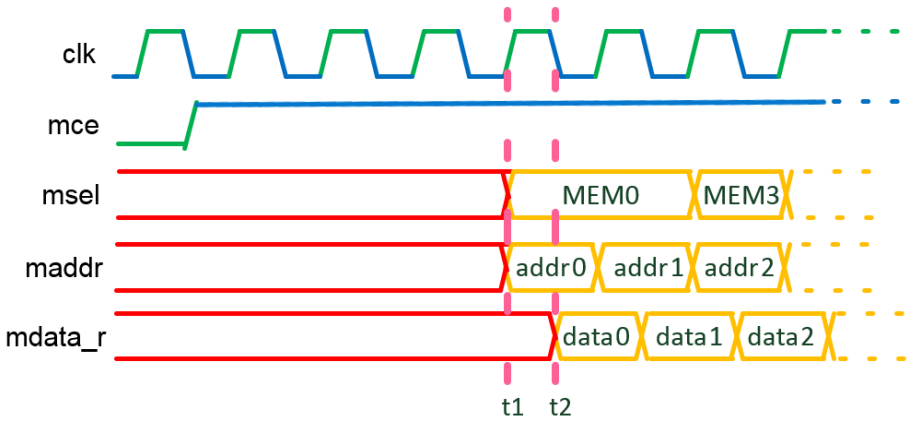

圖 7. 讀取記憶體資料並經 mdata_r 輸入 RNN 系統之時序

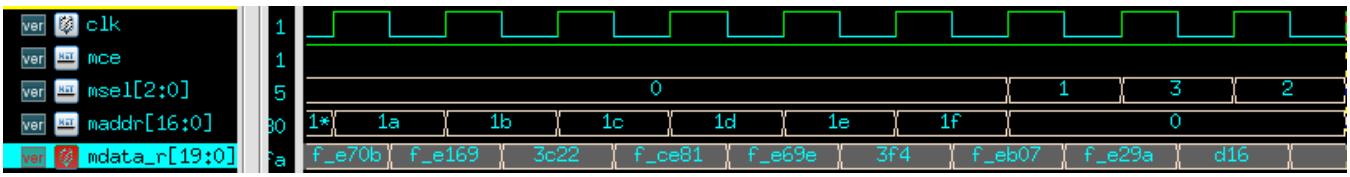

圖 8. 讀取記憶體資料時，實際波形參考圖

4. 當欲將資料寫入記憶體 MEM5 時，RNN 電路需先將 mce 設為 High，並以 msel 選擇記憶體 5，(如圖 9. t1 時間點)，testfixture 在每個時脈負緣後，會將 mdata_w 的資料，依照 maddr 所指示的位址寫入記憶體中 (如圖 9. t2 時間點)。

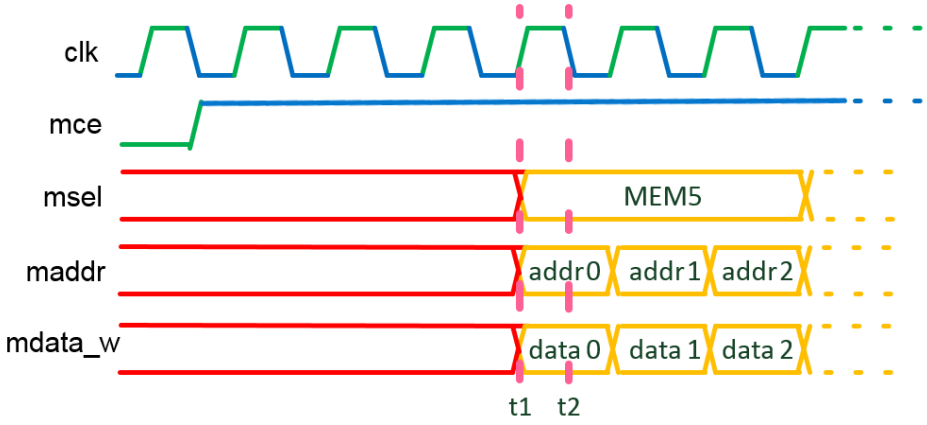

圖 9. RNN 系統欲寫入記憶體資料之時序

3. 評分標準

助教將依照同學提供之系統時脈進行 RTL simulation、Gate-level simulation 及 Transistor-level simulation，以驗證設計正確性。同學應於 report 中定義的系統時脈下，確保輸出結果無設置與保持時間 (setup/hold time) 的問題，並完全符合助教所提供的標準設計結果。

✧ 評分方法及標準：評分方式會依設計完成程度，分成 S、A、B、C 共四個等級，排名優先順序為 **S > A > B > C**。助教將根據設計內容的完成度與正確性給予記分。

1. **S** 等級：RTL、Gate-level 及 Transistor-level simulation 結果完全正確。
2. **A** 等級：RTL 及 Gate-level simulation 結果完全正確，Transistor-level simulation 結果不正確。
3. **B** 等級：RTL simulation 結果完全正確，Gate-level 及 Transistor-level simulation 結果不正確。
4. **C** 等級：所有 simulation 結果都不正確。

✧ 同等級中，AT 值 (Area 值 x Time 值) 越小者名次越好。

✧ Area 值：S 級取 layout 之 total area of core 為準；A 級取 synthesis 之 total cell area 為準，B、C 等級 Area 值為 1。

✧ Time 值：取 testbench 的 total simulation time 為準。

附錄

附錄 A 為助教所提供各檔案說明；附錄 B 為評分用檔案，亦即同學必須繳交的檔案資料。

附錄 A 設計檔案說明

1. 下表一是所提供的設計輔助檔案中較重要的部分

表一、設計輔助檔

檔名	說明
testfixture.v	測試樣本檔。此測試樣本檔定義了時脈週期與測試樣本之輸入及預期輸出信號。
RNN.v	同學所使用的設計檔，已包含系統輸/出入埠之宣告
./data/input1_hex.dat	測試樣本檔案
./data/weight_ih1_hex.dat	w_{ih} 之權重檔案
./data/weight_hh1_hex.dat	w_{hh} 之權重檔案
./data/bias_ih1_hex.dat	b_{ih} 之權重檔案
./data/bias_hh1_hex.dat	b_{hh} 之權重檔案
./data/golden1_hex.dat	測試結果比對樣本檔案
report.txt	結果報告範本 (請同學確實填寫各項結果)
./syn/RNN.sdc	Design Compiler 電路合成規範檔。 本規範檔除了 **cycle** 可修改外，其餘 **constraints** 皆不可修改。
./syn/tsmc13_neg.v	Gate-level simulation 所需要之 cell library file
./syn/.synopsys_dc.setup	Design Compiler 初始設定範例檔案
./layout/RNN_APR.sdc	Innovus APR 時間規範檔。 本規範檔除了 **cycle** 可修改外，其餘 **constraints** 皆不可修改。
./layout/	包含 APR 可使用的輔助檔案之資料夾。
./library/	包含 APR 可能用到的 library 檔案之資料夾。

2. 本題所提供的 Testbench 檔案，有多增加幾行特別用途的敘述如下：

```
`define SDFFILE    "./syn/RNN_syn.sdf"          // Modify your sdf file name

`ifdef SDF
    initial $sdf_annotate(`SDFFILE, CONV);
`endif
```

註:

1. SDF 檔案，請自行依照電路設計階段修改 SDF 實際檔案名稱及路徑後再模擬。

2. 在 Testbench 中，本題提供`ifdef SDF 的描述，目的是讓 Testbench 可做為 RTL 模擬、合成後模擬及 APR 後模擬皆可使用。當同學在合成後及 APR 後模擬時，請務必多加上一個參數+define+SDF，方可順利模擬。
3. RTL-Level Simulation 模擬指令範例如下：(請依實際檔案路徑調整)
   ncverilog testfixture.v RNN.v
4. 合成後的 Gate-Level Simulation 模擬指令範例如下：(請依實際檔案路徑調整)
   ncverilog testfixture.v RNN_syn.v -v tsmc13_neg.v +define+SDF
5. APR 後的 Transistor-Level Simulation 模擬指令範例如下：(請依實際檔案路徑調整)
   ncverilog testfixture.v RNN_APR.v -v tsmc13_neg.v +define+SDF +ncmaxdelays

附錄 B 評分用檔案及繳交

1. 請同學將下表二的所有檔案放入 final_project_<組別>資料夾中，並壓縮成 zip 檔上傳至 Ceiba。(例：第 96 組的資料夾名稱為 final_project_96)
2. 如果同學發現所有檔案無法一起用一個 zip 上傳 Ceiba 的話，請去除 gds 檔後，用同樣方式壓縮其他所有檔案並上傳 Ceiba，再另外將 gds 檔單獨寄給助教並在信件中註明。

表二、評分用檔案檔

檔名	說明
RNN.v	同學完成的 RTL 檔
RNN_syn.v	同學完成合成的 Gate-level netlist
RNN_syn.sdf	Gate-level 的 SDF 時間資訊檔，請注意要可以與 RNN_syn.v 一起通過 testbench
RNN_syn.ddc	Design Compiler 產生之 design database 檔
RNN_APR.v	同學完成 APR 後的 transistor-level netlist
RNN_APR.sdf	Transistor-level 的 SDF 時間資訊檔，請注意要可以與 RNN_APR.v 一起通過 testbench
RNN_APR.gds	Innovus 產生的 gds 檔，注意此檔案可能太大而導致無法上傳 Ceiba，見上面說明
report.txt	結果報告數據 (請依格式填寫)
report.**pdf**	說明設計概念及各結果的報告檔

3. report.pdf 檔分二部分。
   (i). 請同學按照作業二、三的截圖規定截圖
   (ii). 請同學描述自己的設計及遇到的問題。可以寫包含 FSM 設計圖、實做過程遇到的問題及解法、優化的方案及實現方式和使用到的技巧、優化前後的 trade-off 等等。