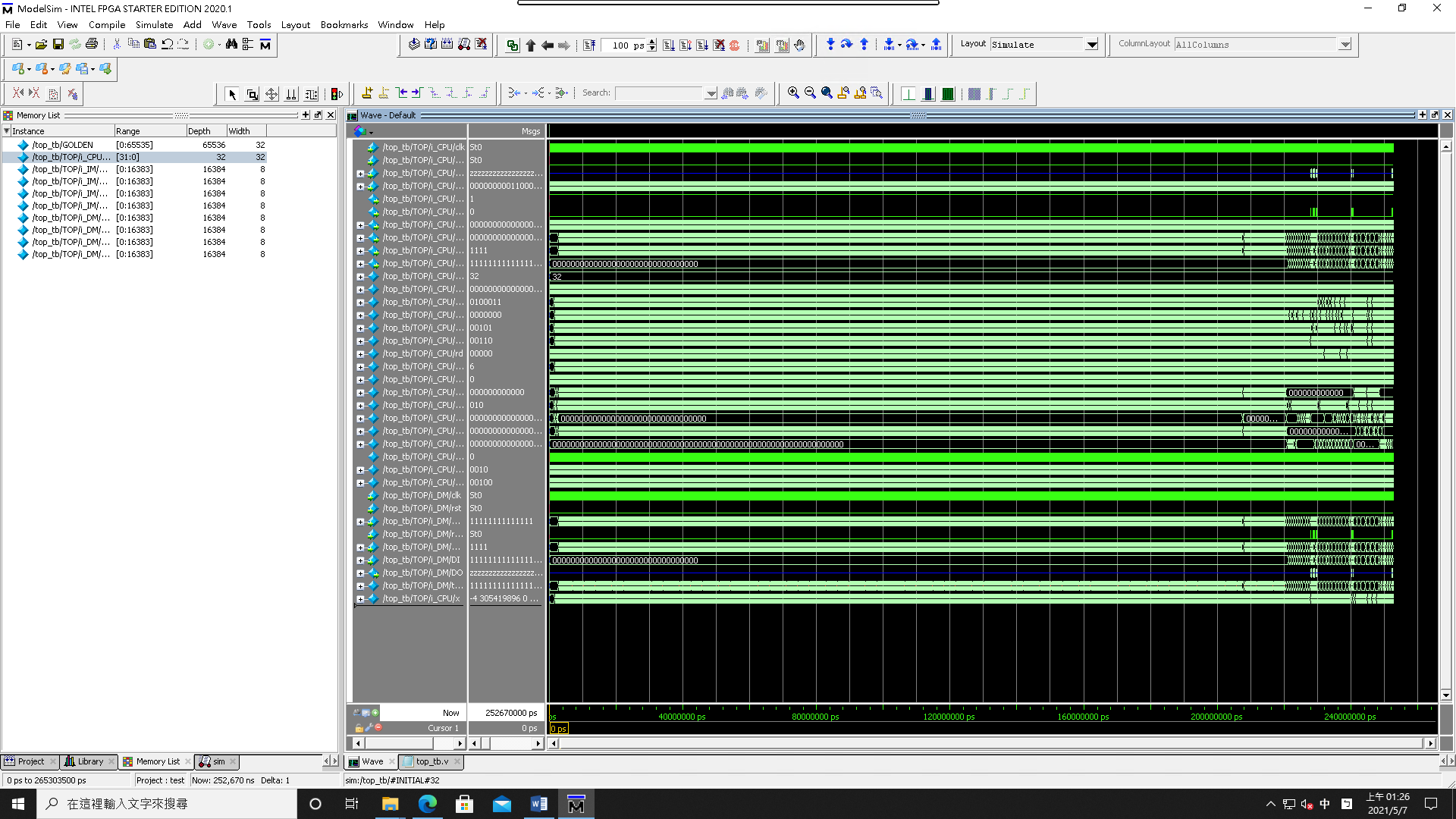
**Computer Organization 2019**

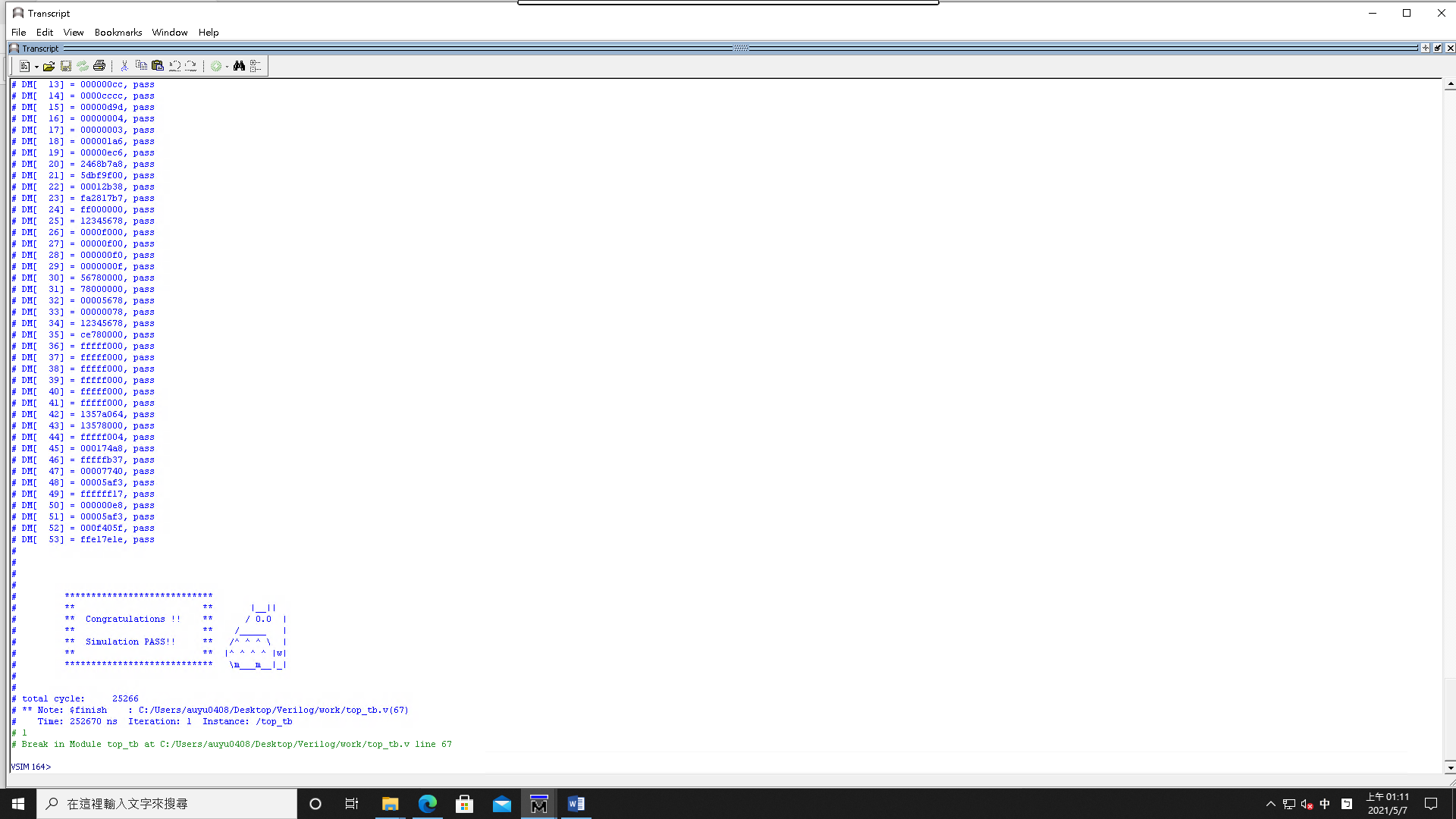
**HOMEWORK 4**

系級: 資訊112 學號: E94086107 姓名: 張娟鳴

**實驗結果圖:**

(波形圖及模擬完成截圖)



****

**程式運作流程:**

(簡單說明波形變化的意義)

利用FSM的概念處理每一步動作，分成5個State(1個Instruction使用5個cycle)，不可諱言這部分我有參考前人的code:

State(READ): 2cycle，載入指令以及讀取指令，並將opcode賦值；有的指令因為動作比較多，像是乘法我會先乘在一個temp再存進register，這種就會在這裡先乘。

State(DECODE): 利用 case 判斷 opcode 將每個 Type 各自所需的變數賦值。

State(HALF): 運算各自 type 裡面的式子，並將 case 為 lw 的 data\_read <= 1; 將 case 為 sw 的 data\_write <= 1。

State(LOAD): 當 State(HALF) 中的 case 為 lw 時，會跳入這個 stste， 並實作 rd = M[rs1 + imm]。

State(NEXT): 除了 case 為 lw 的其他指令都會跳入這個 state，將新的 instr\_addr 賦值並把 instr\_read <= 1。

但除了state外，我還有一個FSM是optype，這可以讓處理各個變數時我不用寫opcode(opcode好長，我直接用R, I0, I1, I2…等代替。)

**心得**

(請寫下完成本次作業的心得、學到哪些東西、困難點的部分。)

寫東西不難，debug才是一切的開始，一開始其實有一個指令漏處理，結果一直沒辦法看到側資結果，後來才發現是錯誤；還有在紅綠燈的時候沒有很明顯，但在這次的作業我可以明顯感受到要assign時要注意clock觸發和分配順序的問題，因為很多是錯在這裡。

再來就是sign extension的部分了，有的immediate會用到但有的其實沒有那麼大的差別，有的時候我是看到我的data\_addr值怪怪的才發現的。

然後一開始覺得modelsim非常難用，但再經過這個作業以後我有發現了一點點小訣竅!