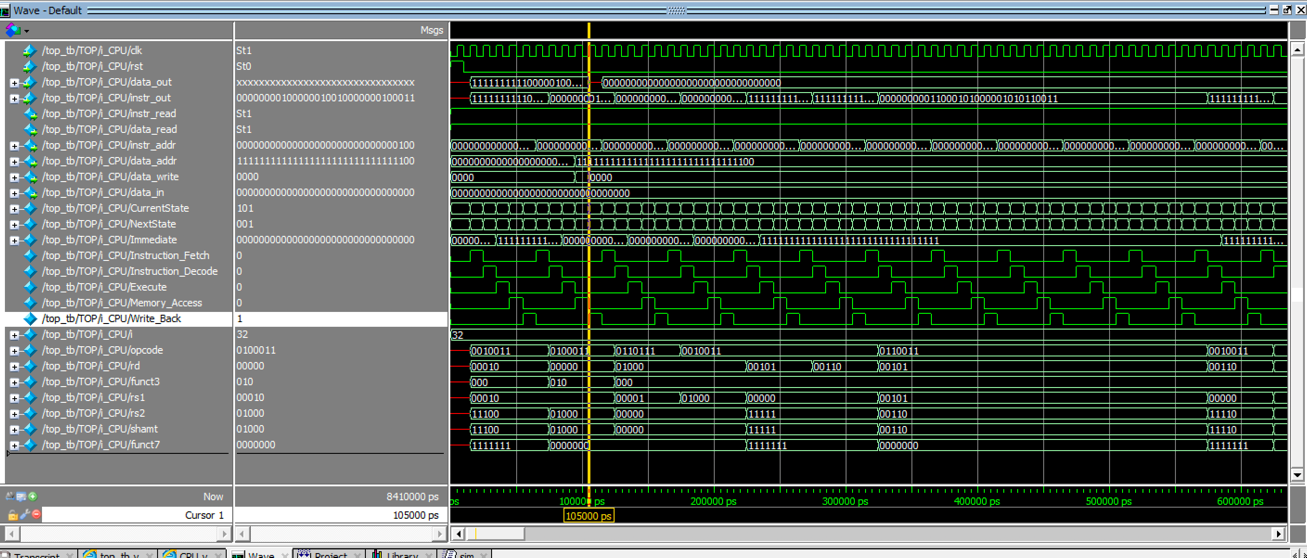
**Computer Organization 2022**

**HOMEWORK 4**

系級: 資訊113 學號: F74096247 姓名: 陳映伃

**實驗結果圖:**

(波形圖及模擬完成截圖)

****

**一張含有 文字 的圖片

自動產生的描述**

**程式運作流程:**

(簡單說明波形變化的意義)

1. 首先是immediate (32 bits)，I-type, S-type, U-type的MIPS 都需要這一項，比較需要注意的是，因為 immediate 是設定成 32 bits 所以沒有用instr\_out表示的地方需要補齊。（例如：S-type的12-31 bit需要用第11 bit 進行延伸，而 LUI 指令是補齊 0-11 bit 的部分，只需補0即可）
2. 再來是暫存器 Register 的部分。我們將指令計算存在Register中
   * R-type：只需在rd的位置放上「位置rs1, rs2暫存器」的運算結果即可。
   * I-type：在rd的位置放上「位置rs1暫存器與immediate」的運算結果。
   * U-type的LUI指令：在rd的位置放上immediate即可。
3. instr\_addr即是PC。我們將「指令走到哪裡」紀錄在instr\_addr中。除了reset外，instr\_addr每次都要加4。（要跳往下一個指令）
4. data\_addr是服務於S-type的一個數。紀錄我們想要值的位置。
   * data\_addr是「位置rs1暫存器與immediate」之和

（我們如何判斷是什麼type、什麼指令則需要opcode, funct3, funct7來判斷）

（其他指令跟參數會在之後作業再來說明）

**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。)

我覺得這次作業非常良心了。複雜的部分助教已經寫好了，我們只需要填寫部分的code而已。

雖然感覺蠻容易的（使需要照著隨附的表格寫就好），但還是有些小地方困擾了我一下。主要是immediate的部分。一開始忘記補齊32 bits一直error（根本跑不動）；隨後補齊32 bits 還補錯 ⋯⋯

主要用non-blocking寫的時候我都會不小心用blocking 的方式在思考。一開始我是直接拿immediate的第11個數去補，但因為是用non-blocking寫法，immediate根本還沒更新，才會一直出錯。最後找了很久才發現這個小問題，直接用instr\_out的第31個bit 去補就對了！

其他部分都比較好懂、還算順利的完成了這次的作業。以上是我的心得。