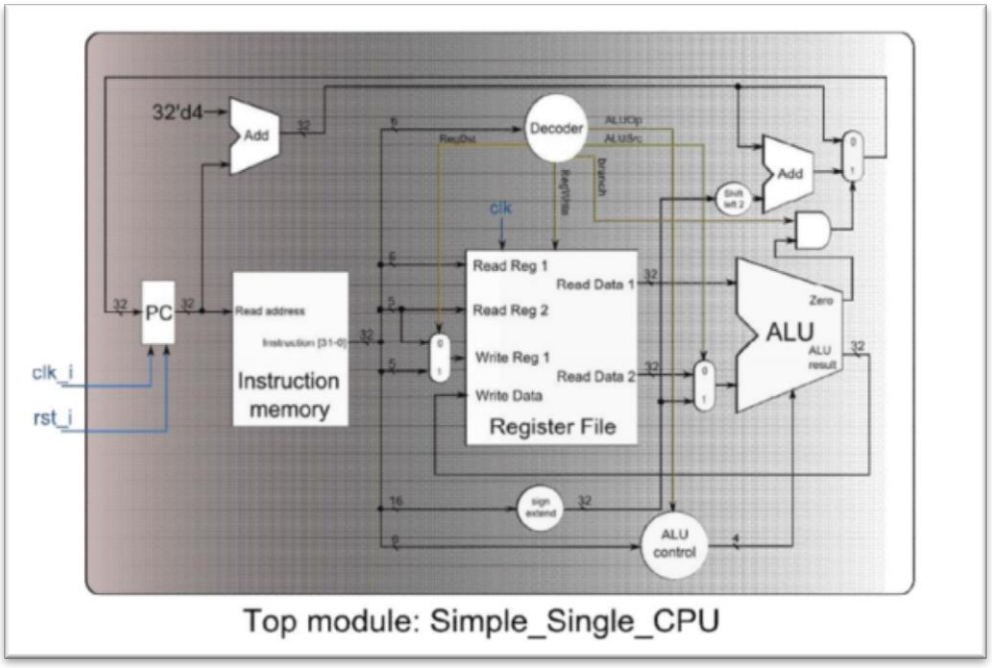
Lab 2: Simple Singal CPU

1. 系統架構

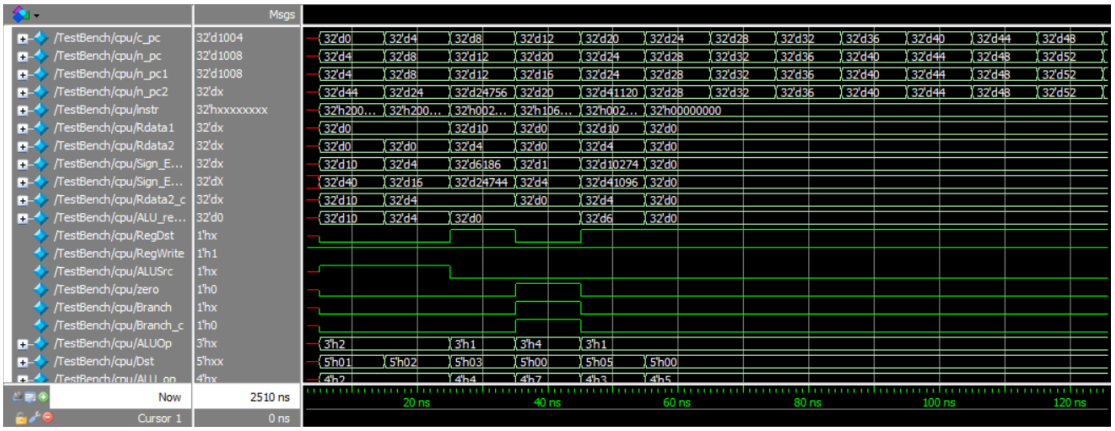


2. 設計模組分析

- **Simple_Single_CPU**： 包含所有的 module，把全部的功能連結在一起。
- **Instr_Memory** ： 讀取 PC 位置，產生 instruction。
- **MUX_2to1 - Mux_Write_Reg** ： 依據 RegDst 選擇哪個 register 被輸入至 Write Reg 1。
- **Reg_File** ： 讀取 Read Register 和把 data 寫入 Write Register。
- **Decoder** ： 將 Instruction [31:26]中的 bit 轉換成各種控制選擇的 output。
- **ALU_Ctrl** ： 依據 function 的不同和 ALUOp 的不同，產生出對應的 ALUCtrl 控制 ALU 的動作。
- **Sign_Extend** ： 將 16 bits 的常數延長至 32 bits。
- **MUX_2to1 - Mux_ALUSrc** ： 依據 ALUSrc，選擇由常數或 Read Data2 當 ALU 的元素。
- **ALU** ： 依據 ALU_Ctrl 所給的 ALU_op 決定其所做的動作。
- **Shift_Left_Two_32 Shifter** ： 將常數*4。
- **MUX_2to1 - Mux_PC_Source** ： 依據 Branch 決定 PC 的下一個位置。

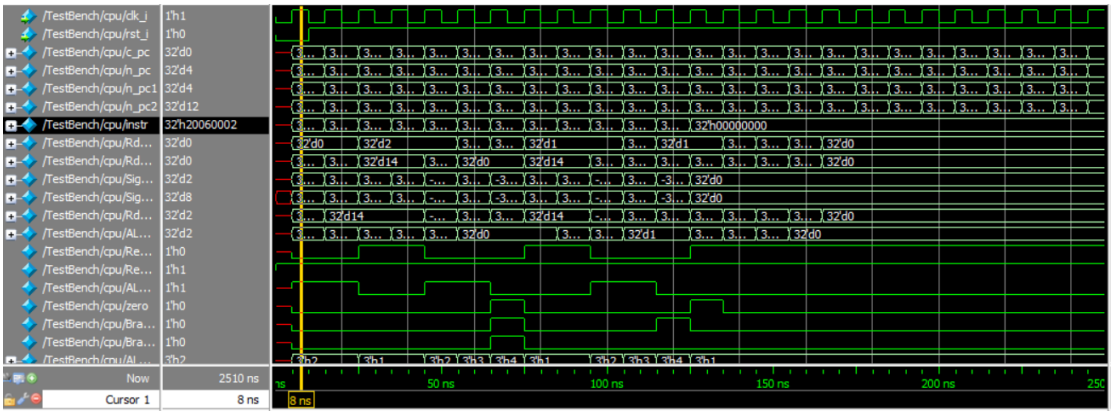
3. 設計結果

Data1 ：




```
VSIM 33> run -all
# r0=          0, r1=         10, r2=          4, r3=          0, r4=          0, r5=          6, r6=          0, r7=          0
# r8=          0, r=9         0, r10=         0, r11=         0, r12=         0, r=13         0, r14=         0, r15=         0
# r16=         0, r17=         0, r18=         0, r19=         0, r20=         0, r21=         0, r22=         0, r23=         0
# r24=         0, r25=         0, r26=         0, r27=         0, r28=         0, r29=         0, r30=         0, r31=         0
# ** Note: $finish    : C:/Users/yu-chen/Desktop/Test_Bench.v(38)
#    Time: 2510 ns  Iteration: 0  Instance: /TestBench
```

Data2 ：




```
VSIM 30> run -all
# r0=          0, r1=          1, r2=          0, r3=          0, r4=          0, r5=          0, r6=          0, r7=         14
# r8=          0, r=9         15, r10=         0, r11=         0, r12=         0, r=13         0, r14=         0, r15=         0
# r16=         0, r17=         0, r18=         0, r19=         0, r20=         0, r21=         0, r22=         0, r23=         0
# r24=         0, r25=         0, r26=         0, r27=         0, r28=         0, r29=         0, r30=         0, r31=         0
# ** Note: $finish    : C:/Users/yu-chen/Desktop/Test_Bench.v(38)
#    Time: 2510 ns  Iteration: 0  Instance: /TestBench
```

4. 遭遇困難與解決方法

1. 一開始打完程式碼之後，圖形一直跑不出來，不知道為什麼 adder 一直跑不進去，後來重開 ModelSim 之後 adder 可以成功開啟了。
2. ALUOp 一直沒有訊號，把跟 ALUOp 有關的 module 打開一個一個檢查，才發現是 assign 的時候，ALU 的名字打錯。
3. 確認全部的數值都有輸入之後，答案卻是錯的。觀察波形圖，發現可能是 control 出錯。就把 Decoder 跟 ALU_crtl 打開檢查，每個 function 都把他的 control signal 寫出來，才成功找到。

5. 作業心得討論

覺得真的是自己寫過之後，才真的了解 control 的功能。也比較清楚 instruction 中的哪個指令要走哪一條路，也能理解 CPU 中各個 Block 的任務及如何分工合作。