Computer Architecture Final Project Report

Single Cycle CPU

蔡帛恩石旻翰邱啓翰B08502169B0502141B08901024

Group Ghost, NTUEE

CPU Architecture

整體架構與投影片所述架構基本完全相同,主架構有分為reg_file, imm_gen, Control, ALU_Control, ALU 共5個block。而在主module中我們共寫了9個mux。除了投影片上的3個mux外,我們還增加了1個auipc相關的mux、1個multicycle相關的mux、1個load相關的mux、3個jalr和jal相關的mux。

Data Path

主要增加的指令為auipc、jal與jalr。

jal與jalr兩者的差別是前者在imm讀取address而後者在rs2讀取,因此我們在control增加jump1與jump2兩個signal。而我們在control輸出這兩個signal後便在主module作為3個mux的控制訊號來決定PC_nxt。而兩者在ALU端都屬於相加的操作,因此除了有使用reg_write且沒有使用mem_to_reg外,control其餘部分和ALU部分與save完全相同。

auipc在control和ALU表現上與addi、slti等除了rs1的輸入取決當前pc位置外完全相同,因此我們在control增加usepc_src這個signal。在control輸出該signal後便在主module作為一個rs1 mux的control signal。

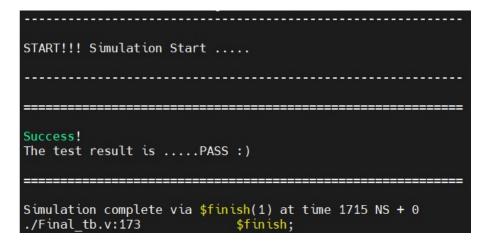
Multi-Cycle Instructions

與Hw2的内容大致相同,分成6個stage: 1. FSM(此併入hw2中counter的部分) 2.Load ALU input並傳遞給Control 3.ALU Output:在此輸出mul的結果 4.Shifter:在此做所有需要移位的工作 5.Sequential Part:產生Output所要求的波形

Simulation Time

```
START!!! Simulation Start .....
Success!
The test result is .....PASS :)
Simulation complete via $finish(1) at time 275 NS + 0
./Final tb.v:173
```

Leaf: a = 3, b = 9, c = 5, d = 17



Perm: n = 8, r = 5

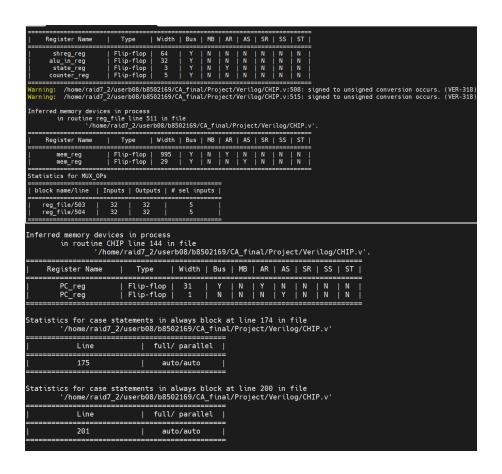
```
START!!! Simulation Start .....
Success!
The test result is .....PASS :)
Simulation complete via $finish(1) at time 5265 NS + 0
./Final tb.v:173
                     $finish;
```

HW1: n = 11

Observation

- 1) mul在hw2中還有一個input叫mode,但我們在這把他精簡了,因為我們只需要用到其中的mul的功能
- 2) counter其實是可以併入FSM一起算的,所以我們決定把他併入FSM的block裡
- 3) pc的部分也是十分重要的一環,畢竟要由pc去控制mem_I的輸入,一開始我們在寫控制pc的地方耗了不少時間,是因為對jal和jalr不是很熟悉,但了解後整個程式也就順利完成了

Register table



Distribution Table

Student	Verilog	Assembly	Report	Contribution
石旻翰	X	v	v	33%
蔡帛恩	v	v	X	33%
邱啓翰	v	\mathbf{v}	\mathbf{v}	33%