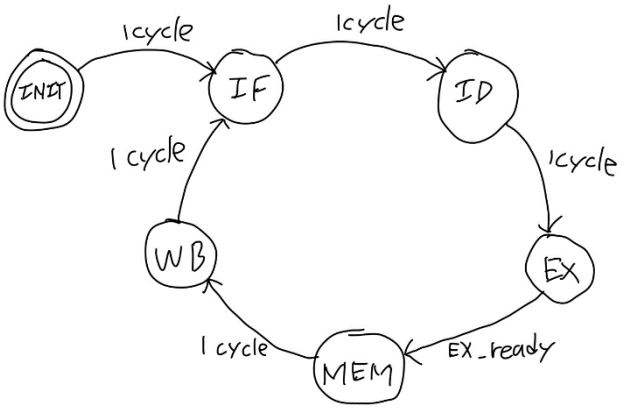
CA Final Project Report

Group 8
B08202054 物理三 李杰銘
B08901211 電機三 游耿睿

1. Briefly describe your CPU architecture

我們參考pipeline CPU的作法，分成IF、ID、EX、MEM、WB，IF從instruction memory取得instruction，ID將instruction轉換成控制訊號以及從Registers拿到對應的data1、data2，EX用ALU運算數值，MEM把數值存入或取出Data memory，WB將結果寫回Registers並更新PC。

FSM如下圖，除了MEM需要多個cycle運算以外，其他都只需要一個cycle來存取memory或registers，INIT是initial state。



2. Describe how you design the data path of instructions not referred in the lecture slides (jal, jalr, auipc, …)

(1)jal: 運算出immediate之後，將PC更新為PC+immediate，並將PC+4存至register中。

(2)jalr: ALU計算完之後，將PC更新為ALU_output，並將PC+4存至register中。

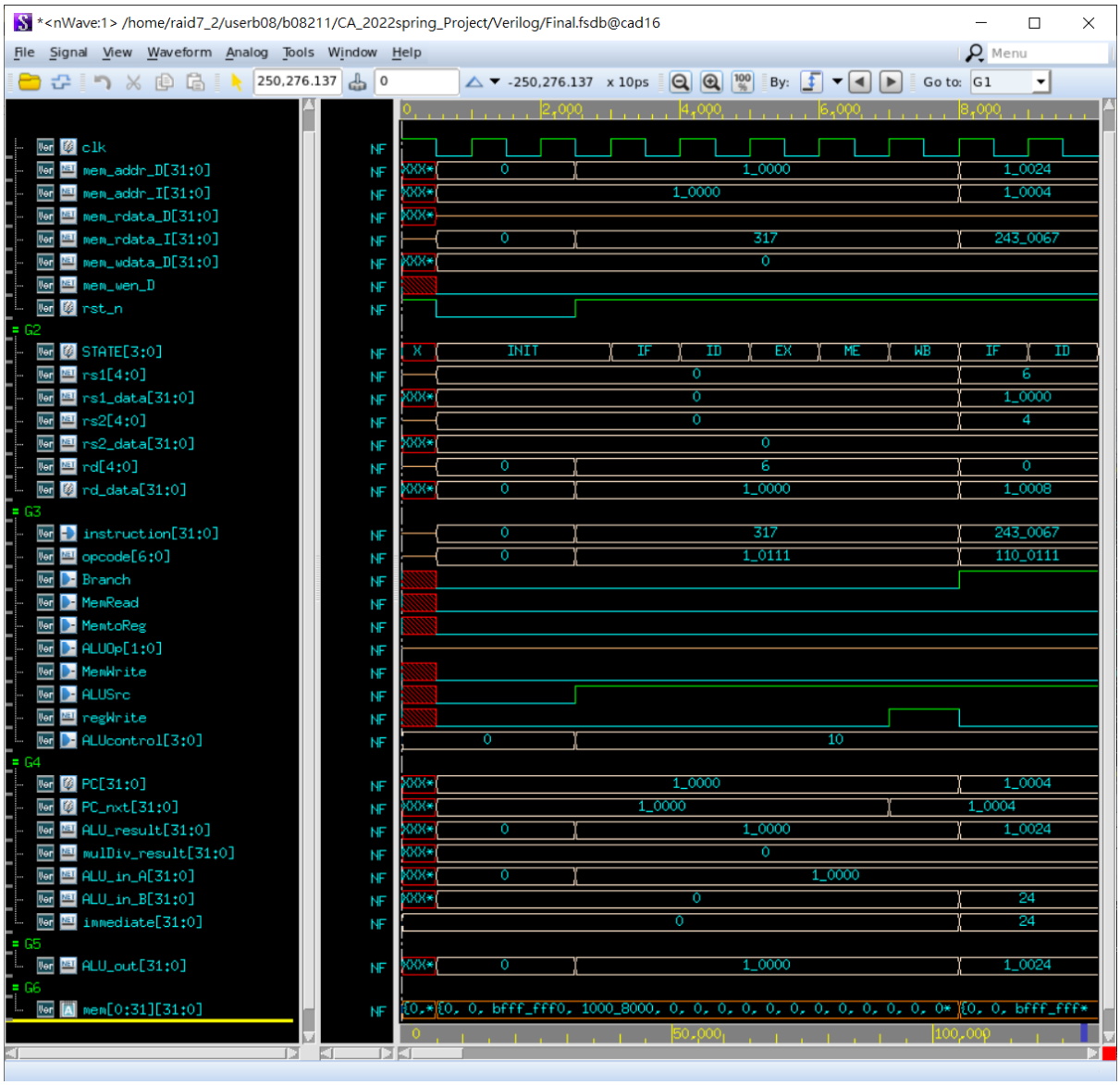
(3)auipc: 多加入一個MUX，讓ALU_in_A根據instruction變為PC，並與immediate在ALU中相加，最後再將結果存至register中。

3. Describe how you handle multi cycle instructions (mul)

在EX state進行計算時，當EX_ready=1才會跳到下一個state MEM，mul在運算完才會把EX_ready變成1，其他運算則是過一個cycle就變成1，繼續後面的步驟。

4. Record total simulation time (CYCLE = 10 ns)
   (1) Leaf: 1285 ns
   (2) Perm: 4495 ns
   (3) (Bonus) HW1: 9175 ns

5. Describe your observation



在debug的過程中可以看到每個clock都有做不同的事，像是IF時從Instruction memory得到instruction，ID時rs1、rs2、rd根據

instruction翻譯成對應的address，在翻譯出control訊號後，ALU、multiplexer、PC更新的部分、mem read/write、reg write，就會執行對應的動作。因為我們把每一個instruction分成五個部分，在波型圖上就會看到明顯的每五個cycle(如果不是mul的話)一個單位。

6. Snapshot the “Register table” in Design Compiler (p. 22)

Register Name	Type	Width	Bus	MB	AR	AS	SR	SS	ST
shreg_reg	Flip-flop	64	Y	N	Y	N	N	N	N
alu_in_reg	Flip-flop	32	Y	N	Y	N	N	N	N
state_reg	Flip-flop	3	Y	N	Y	N	N	N	N
counter_reg	Flip-flop	5	Y	N	Y	N	N	N	N

7. List a work distribution table

李杰銘：EX、WB及PC更新

游耿睿：IF、ID及MEM