**計算機組織**

**期末報告 - Pipelined CPU Design**

**109學年度第二學期**

**老師 : 朱守禮 老師**

**班級 : 資訊二甲**

**組別 : 18**

**組員 : 10827124 莊朝翔**

**10827138 廖翊中**

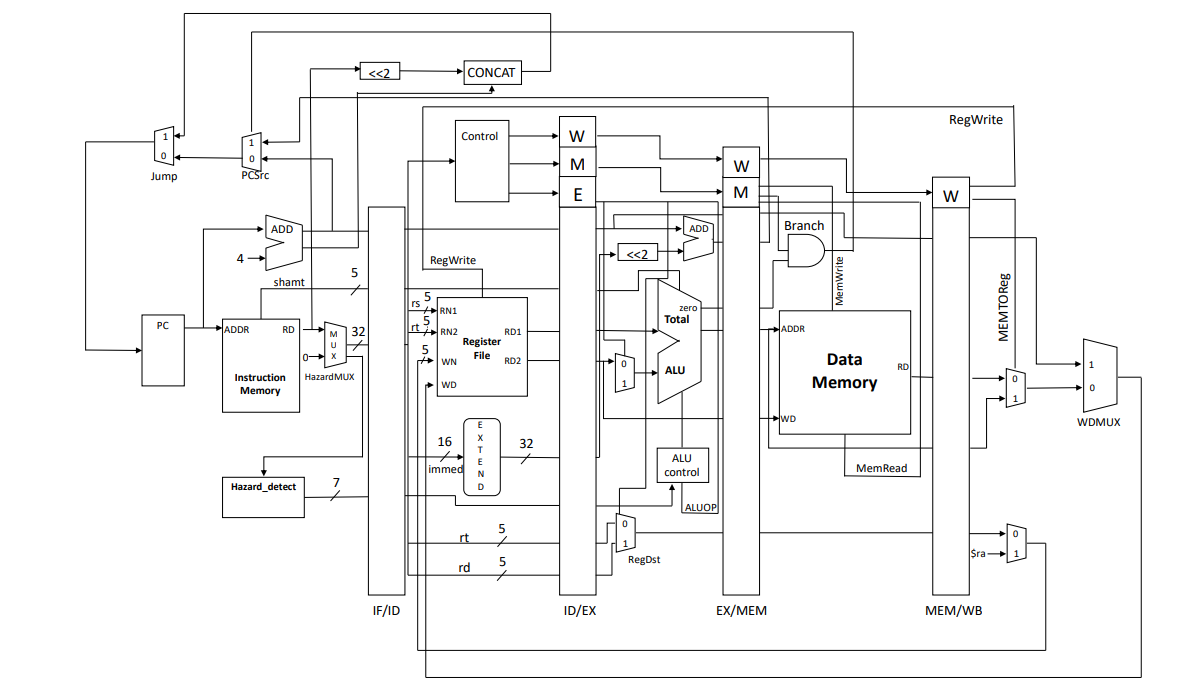
**10827139 黃云潔**

**10827152 華君豪**

一 . 背景

本次project為設計一個 Pipelined MIPS-Lite CPU。 其中可實現指令包含add, sub, and, or, sll, slt, addiu, lw, sw, beq, j, jal, multu, mfhi, mflo, nop等16道MIPS指令，參考自課本 Chapter 4 與課程講義之 Pipelined Datapath。

二 . Datapath 與詳細架構圖



三 . 設計重點說明

1. IF :

① PC多工器選擇下一個PC位址後傳入PC。

② 根據PC的位址讀取下一道指令後，進入hazard多工器，判斷該指令是否需要加入bubble，避免pipeline時不同指令同時讀取及存放暫存器或記憶體的內容。若需要加入bubble，則多工器在下一道指令傳入時選擇傳入32位元皆為0的指令，使該指令根據需要延後幾個cycle後再讀入。

③ 計算PC+4的值。

1. ID :

① 從IF階段傳入的指令中分割出opcode、funct、rs、rt、rd、shamt和立即值、offset等。

② 從Register File中找到相應的暫存器位址，根據控制訊號決定讀出或寫入內容，輸出線路為rd1、rd2。

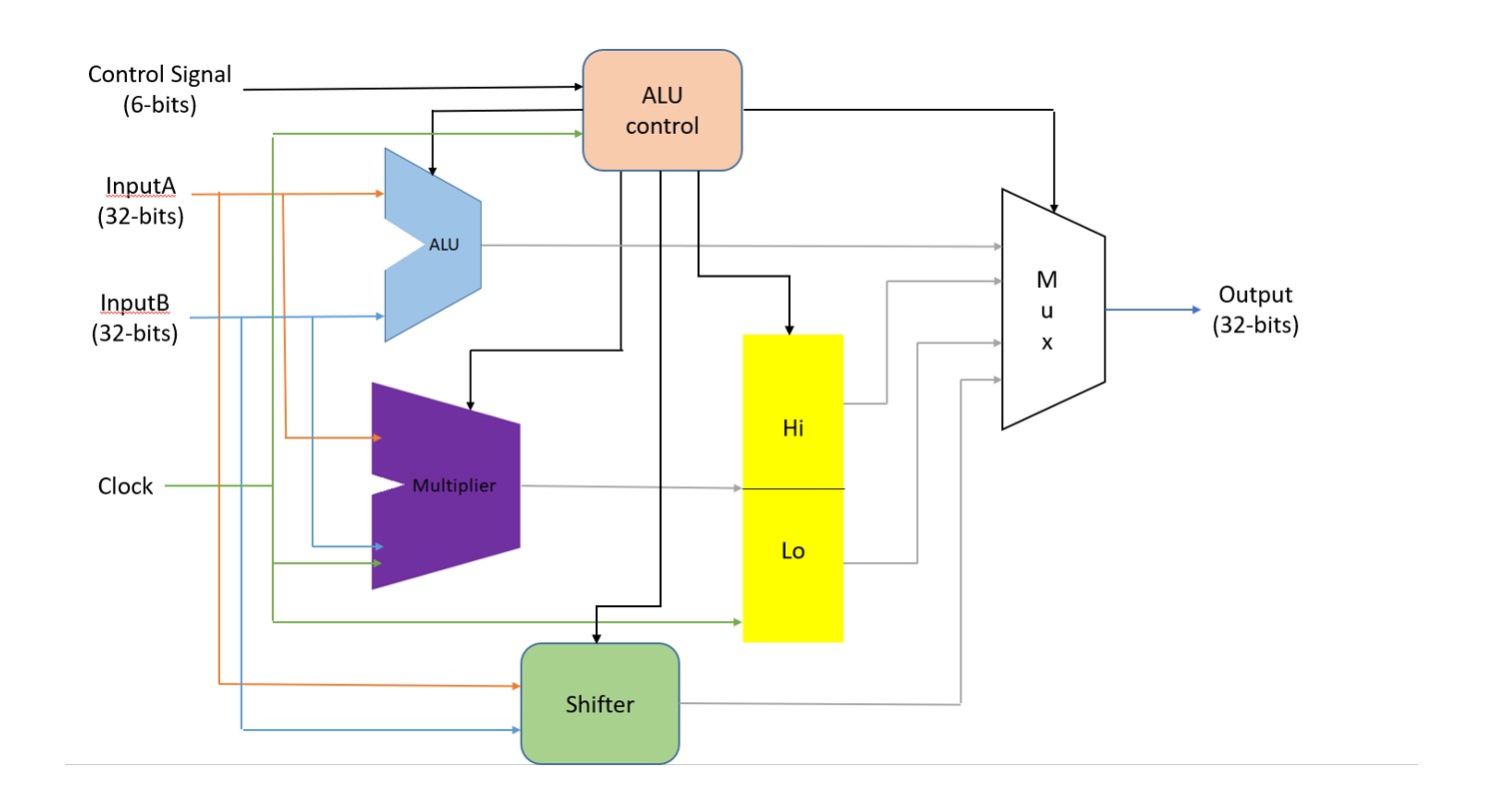
③ 將opcode和funct傳入Control中，根據指令給定控制訊號。

④ 將16位元的立即值擴展為32位元。

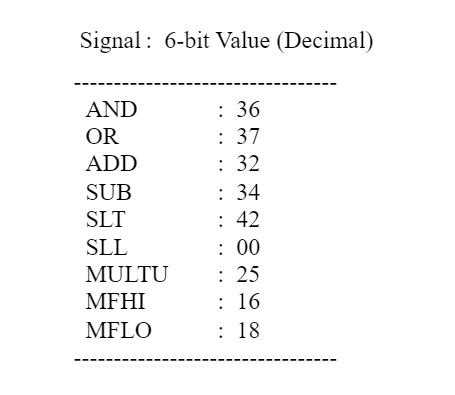
1. EX :

① 計算32位元立即值\*4 + PC+4 (branch用)。

② TotalALU : top module - 用以整合所有module，包含加法、減法、乘法、左移、AND、OR等功能。



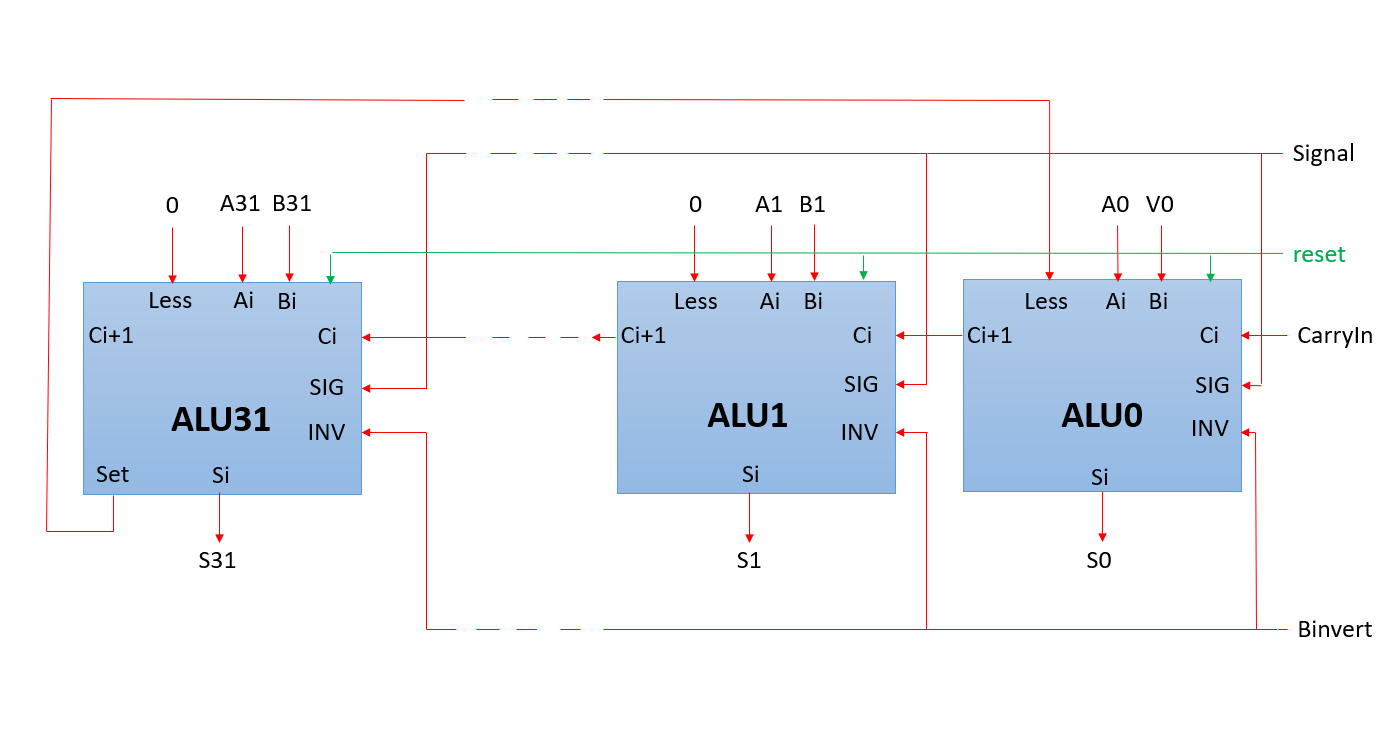
(1) ALU Control : 根據輸入的 6-bit Signal控制訊號，決定該完成AND, OR, ADD, SUB, SLT, SLL, MULTU, MFLO, MFHI哪一種運算。控制訊號與功能對應如下 :



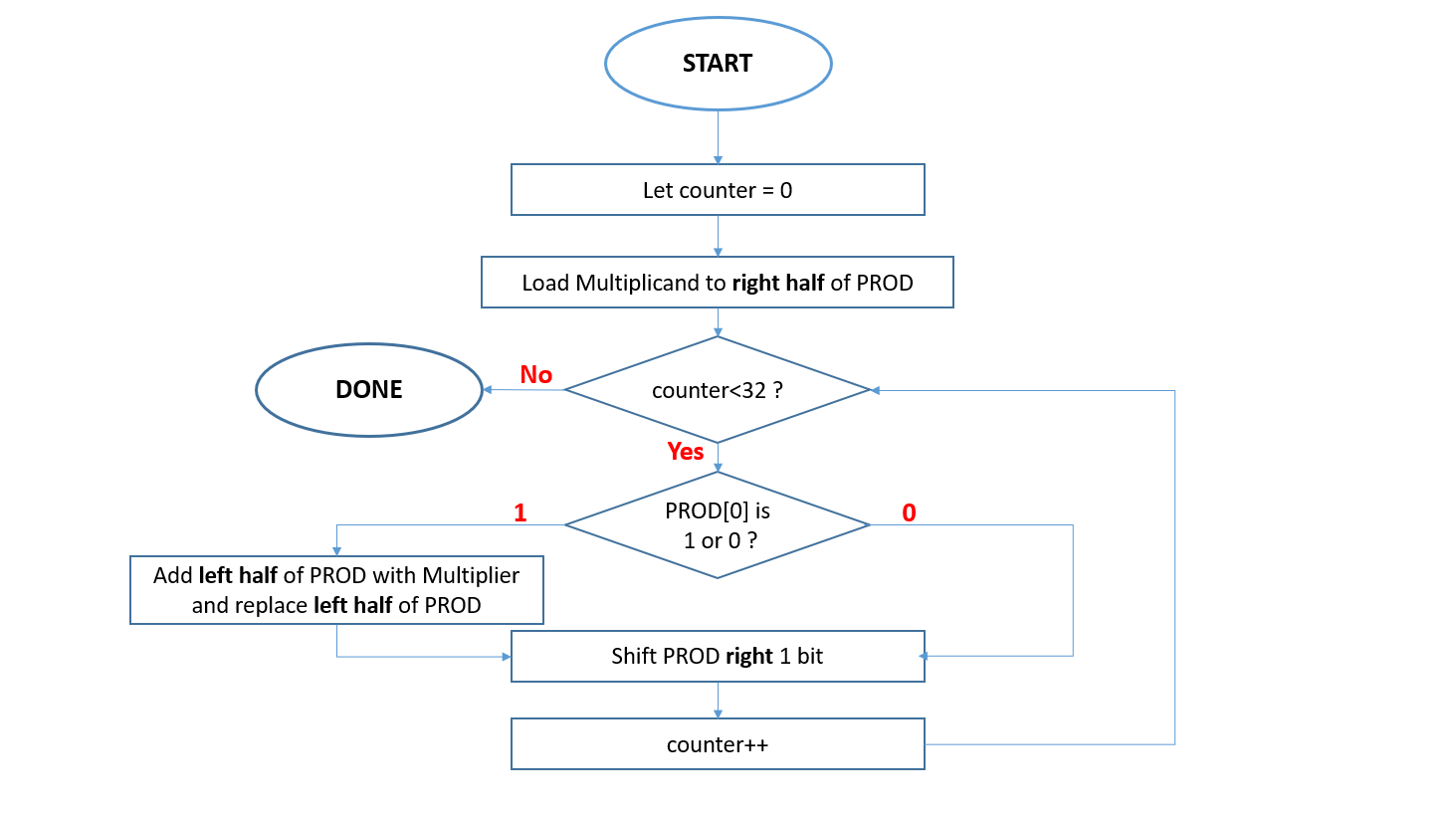
本模組為循序邏輯(Sequential Logic)，因此須以Clock訊號

同步。

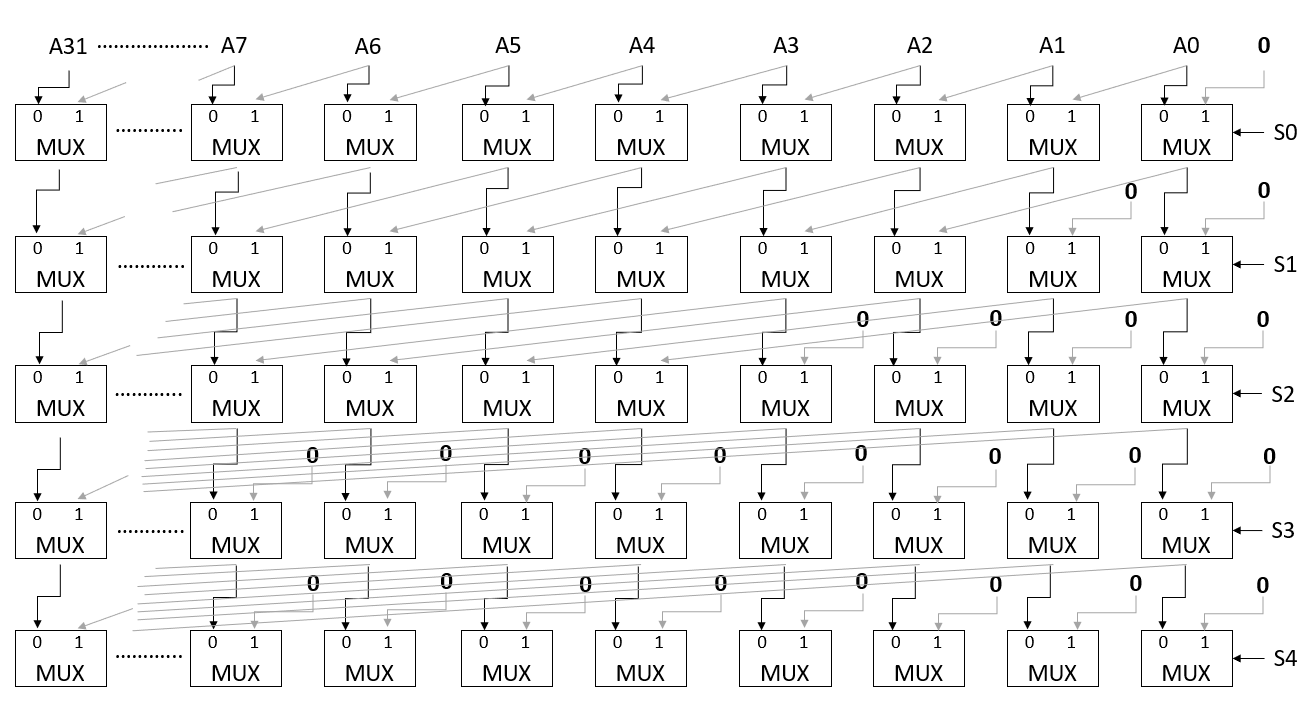
(2) ALU :包含 AND, OR, ADD, SUB, SLT 等功能，從 Full Adder 做起，以 Ripple-Carry 的進位方式， 連接 32 個 1-bit ALU Bit Slice，成為 32-bit ALU。接收來自ALU Control的Signal訊號決定輸出哪種運算結果，其中第0個ALU Slice的Less輸入為第31個ALU Slice的Set輸出，第2到31個ALU Slice的Less輸入則為0，即可達到SLT之輸出結果。



(3) Multiplier：第三版的乘法器一開始將乘數放在PROD 64位元暫存器的右半部分，這樣可以從第0位元判斷該被乘數需不需要加上去，做完後做32次右移，可以同時消去乘數，並得出乘積，省去另外分配暫存器給乘數的動作。

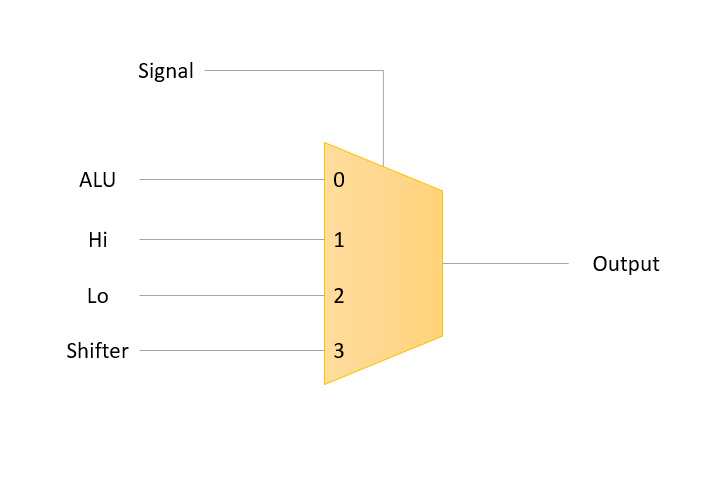


(4) Shifter : 以DataFlow Modeling 設計32-bits Barrel Shifter，以完成邏輯左移運算。總共需要設計出五層，每層32個2對1多工器，共計160個2對1多工器來實現Shifter的功能。



(5) HiLo：將乘法器得到的乘積分為Hi、Lo(32位元)輸出，需與clk訊號同步。

(6) MUX：需以DataFlow Modeling的方式撰寫，因此使用(?:)條件判斷，做出if-else和case選擇的動作。



1. MEM :

① 判斷rd1、rd2是否相等，若指令為beq且rd1、rd2相減為0，則控制下一個讀入的指令位址為PC+4+(立即值\*4)。

② 根據控制訊號決定Data Memory的讀出和寫入，輸出線路為rd。

1. WB :

① 判斷是否為執行jal指令，是則將目的暫存器WN設為$ra，否則將WN設為RedDst的輸出。

② 根據各指令所給定的控制訊號，決定MemtoReg的輸出為Total ALU的輸出或Memory的輸出，並將結果傳至WDMUX。

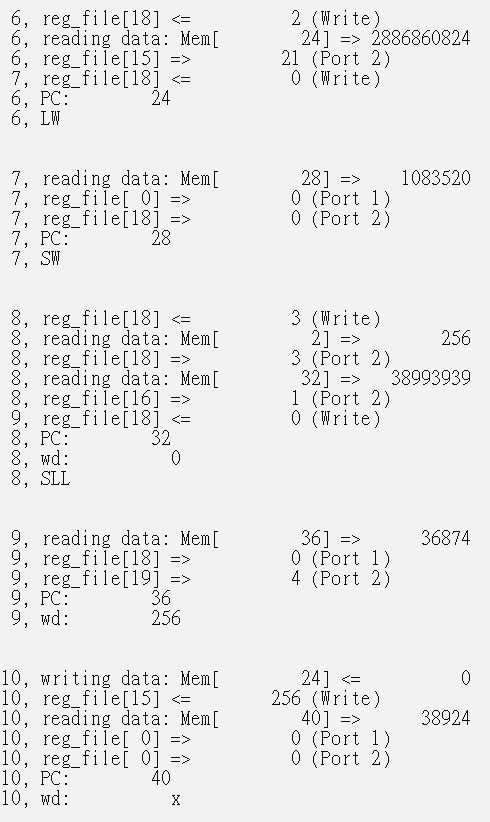
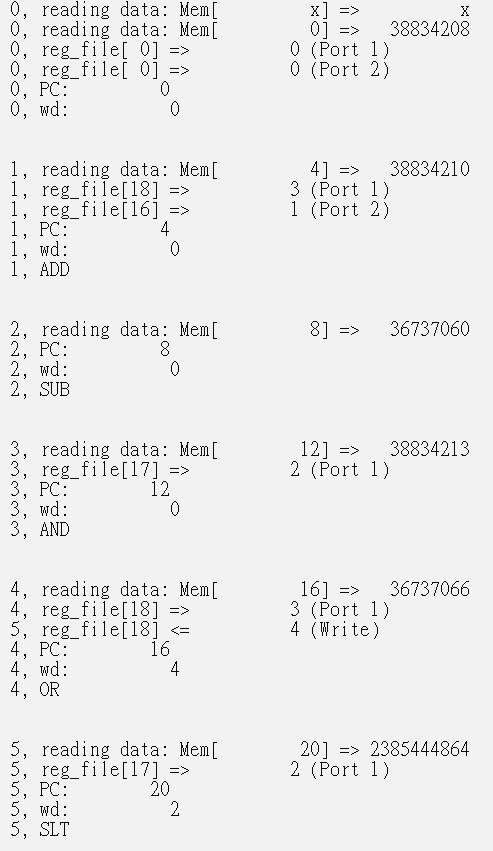
③ 判斷是否為執行jal指令，是則將WDMUX的輸出設為PC+4的位址，否則將WDMUX的輸出設為MemtoReg的輸出，並將結果寫回Register File的WD。

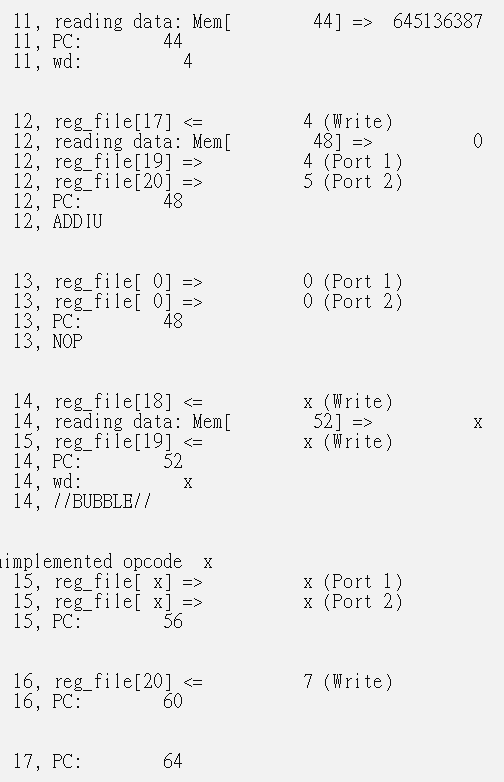
1. Testbench：

原本Single Cycle為 #10 一個cycle，因Pipeline將一個cycle切成五段，因此clk改為 #2 取一次posedge。

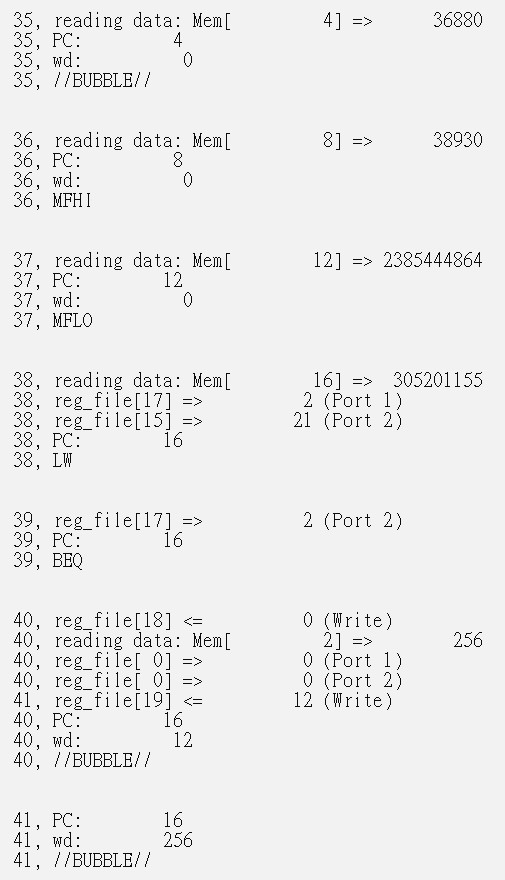
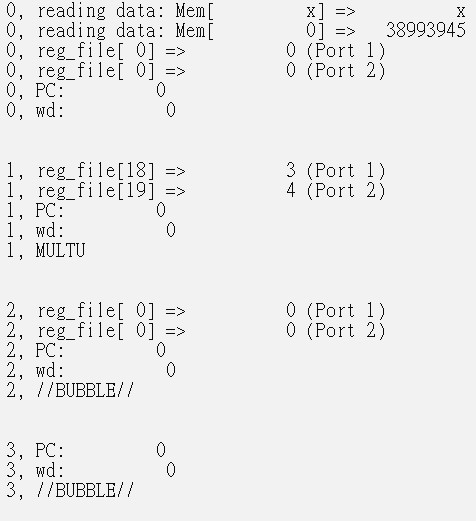
四 . 執行結果

輸出結果： **ADD、SUB、AND、OR、SLT、LW、SW、SLL、ADDIU、NOP**

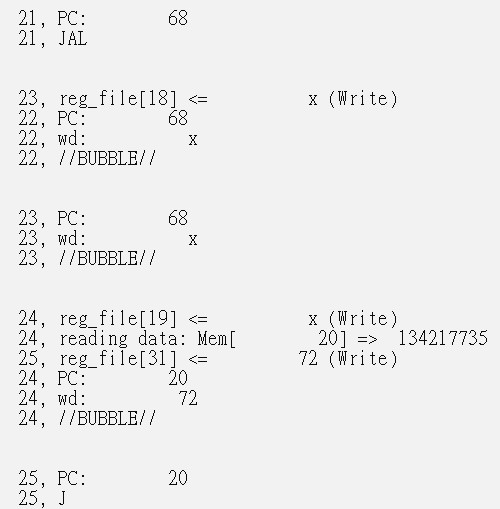
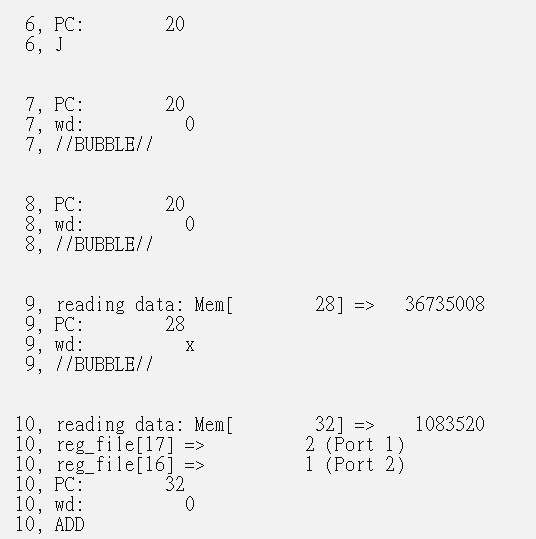
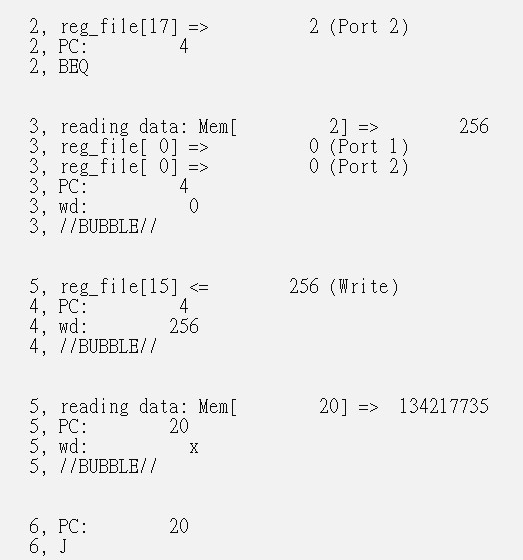




輸出結果：**MULTU、MFHI、MFLO**



輸出結果：**BEQ、J、JAL**



a) Integer Arithmetic:

add : add $s2, $s0, $s2, 0

r18的值為3，r16的值為1，兩者執行add指令後得4，並將答案寫入r18。

sub : sub $s2, $s0, $s2, 0

r18的值為3，r16的值為1，兩者執行sub指令後得2，並將答案寫入r18。

and : and $s1, $s0, $s2, 0

r17的值為2，r16的值為1，兩者執行and指令後得0，並將答案寫入r18。

or : or $s2, $s0, $s2, 0

r18的值為3，r16的值為1，兩者執行or指令後得3，並將答案寫入r18。

slt : slt $s1, $s0, $s2, 0

r17的值為2，r16的值為1，因為2 > 1，因此兩者執行slt指令後得0，並將答案

寫入r18。

addiu : addiu $s3, $s4, 3

r19的值為4，偏移量為3，兩者執行addiu指令後得7，並將答案寫入r20。

sll : sll $zero, $s0, $s1, 2

r16值為1，往左偏移2位元得到4後，將結果寫回r17。

b)Integer Memory Access:

lw : lw $s1, $t7, 0

r17的值為2，加上偏移量0後得2，抓取Memory位址2的值得到256後，將256寫入

r15。

sw : sw $zero, $s2, 24

r0的值為0，加上偏移量24後得到24，再將r18的值0寫入Memory位址24中。

c) Integer Branch:

beq : beq $s1, $s2, 3

r17的值為2，r18的值為2，兩者相等，執行beq指令，位址跳至beq的下一道指

令開始數3個指令偏移量。

j : j 7

指令位址為 20，要跳到的下一個指令位址為 PC+4 的[31:28]位元，和偏移量

7 左移2位(7\*4)放在[27:0]位元，因此跳到指令位址 28。

jal : jal 5

指令位址為 68，要跳到的下一個指令位址為 PC+4 的[31:28]位元，和偏移量

5 左移2位(5\*4)放在[27:0]位元，因此跳到指令位址 20，並將jal下一道指令

的位址存到$ra。

d) Integer Multiply:

multu : multu $s2, $s3

r18值為3，r19值為4，乘法經過32個clk得出值為12。

e) Other Instructions :

mfhi : mfhi $2

乘法結果較高的32位元為0，寫入r18。

mflo : mflo $3

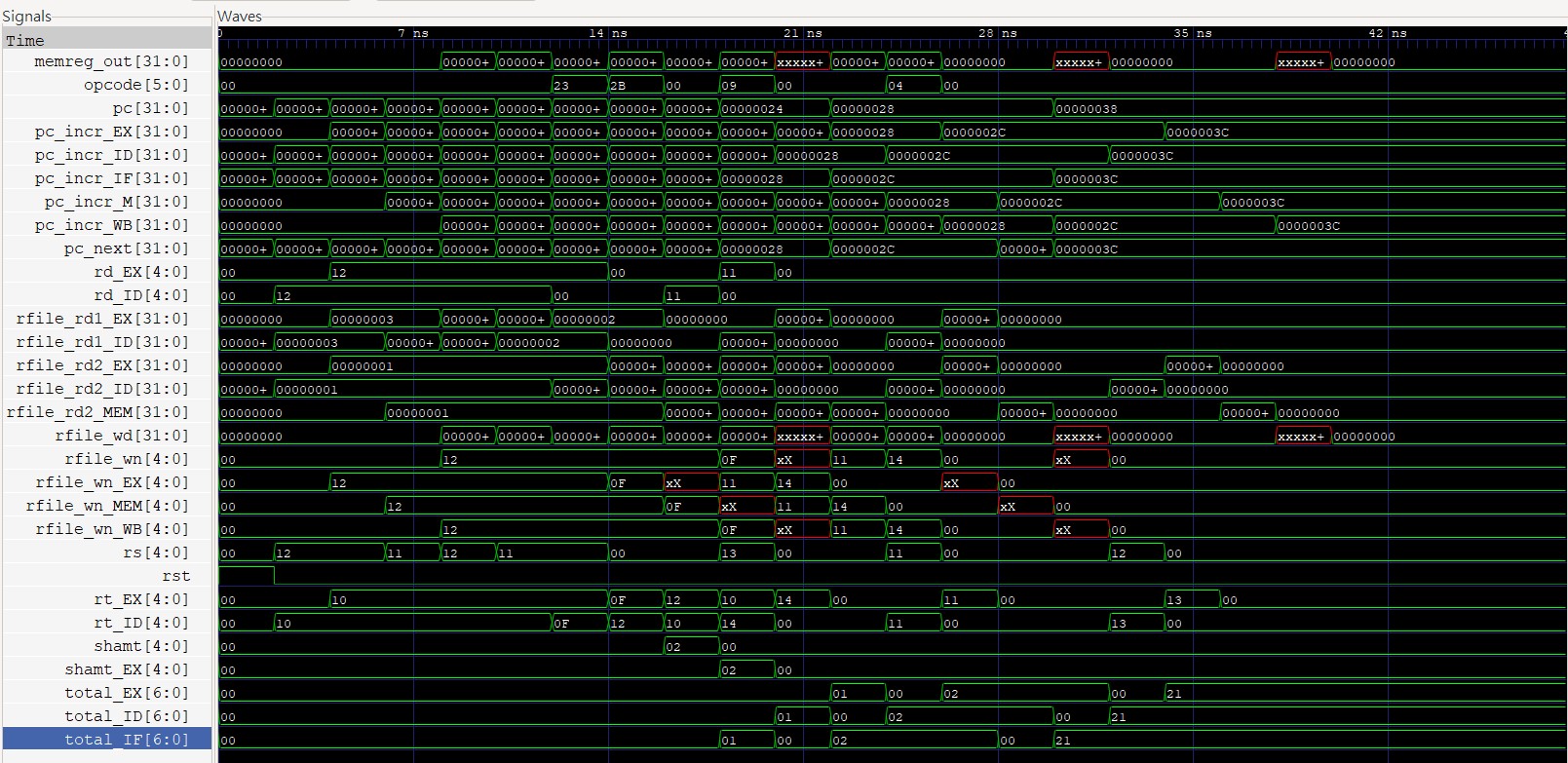
乘法結果較低的32位元為12，寫入r19。

nop : nop

do nothing，即為放入一個bubble。

波形圖 :





五 . 心得感想

10827124 莊朝翔 :

這次期末prroject為設計一個Pipelined CPU，我們除了會運用到期中學過的ALU外，還多了許多新觀念，從一開始將Single Cycle切成Pipelined，後面處理指令也遇到不少問題，例如jal要如何將下一道指令的位址寫到$ra，sll與nop指令如何區別，最困擾我們的便是hazards的問題了，什麼時候要放bubble？放幾個？和判斷是否為hazards，這些對我們來說都非常有挑戰性，但我們在老師影片和一些網路資料的幫助下還是將他寫了出來，在這些與組員合作的過程中，也學到了許多東西，算是收穫滿滿。

10827138 廖翊中 :

期末的project難度比起期中難了許多，要融合很多的觀念，從期中project的ALU架構，在融合進Pipelined CPU，再加上hazard的問題，涵蓋了整學期的範圍。也因為涵蓋的範圍很廣，導致verilog程式碼的量很龐大，要debug也很不容易，一個小錯，就要將整個程式碼重新檢視，所以在debug的階段花上我們很多時間。最終經過無數個日夜，我們將這次的project完成出來了。

10827139 黃云潔 :

期末project從分工開始就是一大難題，不像期中的時候可以分好幾個module大家同時進行，這次是需要階段性的，先把single cycle改成pipeline之後，再把期中的ALU放進來，再對每個指令所需要的接線和控制訊號做修改，一旦有人改了其中一部分，其他地方也可能都影響到，因此我們花很多時間在debug，每次都是一個小錯，但都要花上好幾個小時。並且這次還要處理新加入的指令的問題，如addiu、jal等，都要自己去研究，真的是結合了整個學期所學的知識，才完成這次的project，並且邊寫邊研究也讓我對課程內容有更多的理解。

10827152 華君豪 :

期末project需以期中為基礎，將single cycle改寫為pipeline，確保期中的指令在此架構上能夠成功執行，並新增這次所要求的新指令，需參考課本上不同部分的架構圖，了解不同的指令功能所需要的接線及控制訊號，也因為功能眾多導致整個程式量龐大，debug的過程非常辛苦，當結果不如預期時，就要從源頭將整個過程重新檢視，時常因為某個訊號接錯或是位元數算錯而導致指令無法成功執行，是一次需要耐心及細心才能完成的挑戰。

六 .未來展望

這次的project包含了整學期學的計組概念，要將整個計組的概念統整且釐清，才能完整的些出這次的 final project，也經過這次的訓練，我們不管對於計算機組織上的觀念理解又更進一步，verilog的程式能力也加強了很多，對於程式語言內容也更細心琢磨，對於未來每一次挑戰也要抱持初心，認真應對。當然，在小組合作方面組員間也變得更有默契，也希望現在累績的實力在未來也能讓我們好好的發揮我們自己的才能，大展身手。

七 . 分工方式與負責項目

10827124 莊朝翔 : verilog程式設計、書面報告

10827138 廖翊中 : verilog程式設計、書面報告

10827139 黃云潔 : verilog程式設計、書面報告

10827152 華君豪 : verilog程式設計、書面報告

PS:本次project採接力合作模式，沒有明確的分工，都是一起寫的！