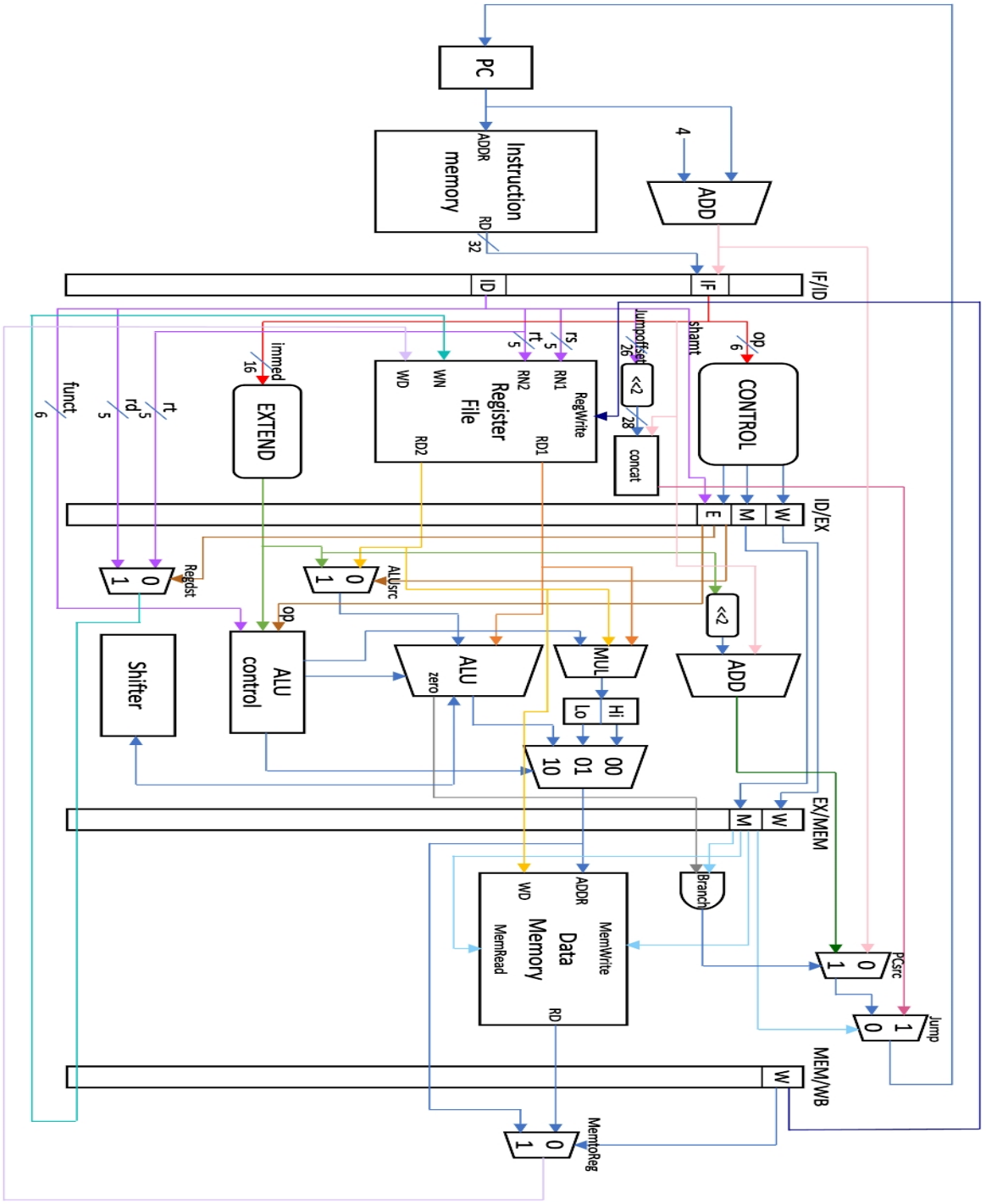
計算機組織期末Project

教授: 朱守禮

組別:15

組員:
11027102仇瀚慶
11027142 林蕙郁
11027144 陳佳音
11027131 李虹昀

一、Datapath與詳細資料圖:



二、設計重點說明

使用了5種暫存器(分別為IF_ID、ID_EX 、EX_MEM 、MEM_WB)五種部份。

第一階段: IF

IF_ID存了PC, INS, INS, reset, clk, enable
這個MUDULE(IF_ID)存了兩個值: PC(指令位置)及INS(指令數據)，當enable信號為高電位時，PCout輸出的值等於PCin輸入的值，儲存了指令位置。同樣的，INSout輸出的值就會等於INSin輸入的值，儲存了指令數據。

第二階段: ID

ID_EX存了WB, MEMORY, EX, IDEX, RD1, RD2, Extnd, shamt, RT, RD, funct, IDEX_ctlbrn, jump, jumpaddr, IDEX, RT, functin, reset, clk, enable

這個MODULE會從第一階段(IF_ID)階段接收PC（指令位置）和INS（指令數據）。如果enable信號為高電位，則將PCin的值傳遞到PCout，保存指令位置。同樣地，將INSin的值傳遞到INSout，保存指令數據。從Register File中讀取第一個RD1的值，並將其傳遞到IDEX_Rd1out再從Register File中讀取第二個數據寄存器RD2的值，並傳到IDEX_Rd2out。對立即數進行符號擴展，並將結果傳遞到SignEXTNDout。對它進行無符號擴展，並將結果傳到UNEXTNDout。根據控制單元的信號(control signal)，從UNEXTNDout和SignEXTNDout中選擇擴展結果，再將其傳遞到extendmuxout。從IF_ID做出的動作再把結果進入ID_EX存起，例如ALU操作類型和分支控制信號。將控制信號傳遞到IDEX_ctlbrnout，以供後續階段使用。最後檢查是否存在跳躍指令，並根據需要計算跳躍地址。將跳躍信號和跳躍地址傳遞到IDEX_jump和jumpaddr1，以供後續階段使用。

第三階段: EX

EX_MEM存了WB, MEMORY, TOTALALU, RD2, WN, ZERO, add, jump, jumpaddr, reset, clk, enable

在執行（Execute）階段，模組會從第一階段（IF_ID）階段接收指令位置和指令數據。它將這些值保存起來並從REGISTER FILE中讀取數據寄存器的值。同時，它對值進行擴展，選擇擴展結果並傳遞控制信號。這些值和控制信號被傳遞到ID_EX（指令執行）寄存器中。ID_EX它保存了從前一階段獲取的數據和控制信號，以供後續階段使用。其中包括了PC和INS從IF_ID階段接收並保存到ID_EX寄存器中。數據寄存器值（RD1和RD2）。從REGISTER FILE中讀取的數據寄存器值被傳遞到ID_EX寄存器的相應字段中。擴展結果，對值進行符號擴展和無符號擴展，然後根據控制信號選擇相應的擴展結果，最終傳遞到ID_EX寄存器中。
控制信號，從控制單元獲取的控制信號被傳遞到ID_EX寄存器中，包括ALU操作類型和分支控制信號等。跳躍指令檢測：在執行階段中檢查是否存在跳躍指令，如果存在，則計算跳躍地址並將跳躍信號和跳躍地址傳遞到ID_EX寄存器的相應字段中。

第四階段: MEM

MEM/WB存了WB, DM, ADDR, WN, DMin, reset, clk, enable

從前一階段（EX_MEM）接收數據和控制信號。將接收到的數據和控制信號保存在相應的寄存器中。如果收到重置信號(reset)，則將寄存器的值重置為初始狀態。如果(enable)為高電位，就會根據控制信號和數據進行相應的操作。將操作結果保存在寄存器的相應字段中，以供後續階段使用。
將需要寫入的數據、存儲器地址和寫入寄存器編號從寄存器輸出（DMin、ADDRin、WNin）傳遞到寄存器輸出（DMout、ADDRout、WNout）。
根據需要進行寫回操作的控制信號（WBin），將寄存器輸出（DMout、ADDRout、WNout）傳遞到寄存器輸出（WBout）。

第五階段: WB

根據WBin的控制信號決定是否需要寫回。

三、驗證結果

```
#                    0, reading data: Mem[         x] =>          x
# 18446744073709551615, PC:          x
#                    0, reading data: Mem[         0] => 2385444864
#                    0, reg_file[ 0] =>          0 (Port 2)
#                    0, reg_file[ 0] =>          0 (Port 1)
#                    0, PC:          0
#                    0, wd:          0
#                    0, NOP
#
#                    1, reading data: Mem[         4] =>  305201155
#                    1, reg_file[15] =>         21 (Port 2)
#                    1, reg_file[17] =>          2 (Port 1)
#                    1, PC:          4
#                    1, LW
#
#                    2, reading data: Mem[         8] =>   38834208
#                    2, reg_file[17] =>          2 (Port 2)
#                    2, PC:          8
#                    2, BEQ
#
#                    3, reading data: Mem[        12] =>   38834210
#                    3, reading data: Mem[         2] =>        256
#                    3, reg_file[16] =>          1 (Port 2)
#                    3, reg_file[18] =>          3 (Port 1)
#                    3, PC:         12
#                    3, wd:          x
#                    3, ADD
#
#                    4, reading data: Mem[        16] =>   36735008
#                    5, reg_file[15] <=        256 (Write)
#                    4, PC:         16
#                    4, wd:        256
#                    4, SUB
#
#                    5, reading data: Mem[        20] =>  134217735
#                    5, reg_file[17] =>          2 (Port 1)
#                    5, PC:         20
#                    5, wd:          x
#                    5, ADD
#
#                    6, reading data: Mem[        24] =>   38834213
#                    6, reg_file[ 0] =>          0 (Port 2)
#                    6, reg_file[ 0] =>          0 (Port 1)
#                    7, reg_file[18] <=          4 (Write)
```

```
 7, reading data: Mem[         28] =>   36735008
 7, reg_file[18] =>          4 (Port 1)
 7, reg_file[16] =>          1 (Port 2)
 8, reg_file[18] <=          2 (Write)
 7, PC:         28
 7, wd:          2
 7, OR

 8, reading data: Mem[         32] =>   38834210
 8, reg_file[17] =>          2 (Port 1)
 9, reg_file[17] <=          3 (Write)
 8, PC:         32
 8, wd:          3
 8, ADD

 9, reading data: Mem[         28] =>   36735008
 9, reg_file[18] =>          2 (Port 1)
 9, PC:         28
 9, wd:          x
 9, SUB

10, reading data: Mem[         32] =>   38834210
10, reg_file[17] =>          3 (Port 1)
11, reg_file[18] <=          5 (Write)
10, PC:         32
10, wd:          5
10, ADD

11, reading data: Mem[         36] =>   38834213
11, reg_file[18] =>          5 (Port 1)
12, reg_file[17] <=          3 (Write)
11, PC:         36
11, wd:          3
11, SUB

12, reading data: Mem[         40] => 2886860824
13, reg_file[18] <=          1 (Write)
12, PC:         40
12, wd:          1
12, OR

13, reading data: Mem[         44] =>   36737060
13, reg_file[ 0] =>          0 (Port 1)
13, reg_file[18] =>          1 (Port 2)
14, reg_file[17] <=          4 (Write)
13, PC:         44
13, SW
```

```
14, reading data: Mem[        48] =>  640679937
14, reg_file[17] =>          4 (Port 1)
14, reg_file[16] =>          1 (Port 2)
15, reg_file[18] <=          4 (Write)
14, PC:         48
14, wd:           4
14, AND

15, reading data: Mem[        52] =>    1151042
16, reg_file[18] <=          5 (Write)
16, writing data: Mem[        24] <=          1
15, PC:         52
15, ADDIU

16, reading data: Mem[        56] =>   38895658
16, reg_file[ 0] =>          0 (Port 1)
16, reg_file[17] =>          4 (Port 2)
16, PC:         56
16, wd:           x
16, SRL

17, reading data: Mem[        60] => 1915879425
17, reg_file[18] =>          5 (Port 1)
18, reg_file[18] <=          0 (Write)
17, PC:         60
17, wd:           0
17, SLT

18, reading data: Mem[        64] =>          0
18, reg_file[17] =>          4 (Port 1)
18, reg_file[18] =>          0 (Port 2)
19, reg_file[16] <=          x (Write)
18, PC:         64
18, MADDU

19, reg_file[ 0] =>          0 (Port 1)
19, reg_file[ 0] =>          0 (Port 2)
20, reg_file[18] <=          0 (Write)
19, PC:         68
19, wd:           0
19, NOP

21, reg_file[16] <=          1 (Write)
20, PC:         72
20, wd:           1
20, NOP
```

```
21, PC:         76
21, wd:           x
21, NOP

22, PC:         80
22, wd:           x
22, NOP

23, PC:         84
23, wd:           x
23, NOP

24, PC:         88
24, wd:           x
24, NOP

25, PC:         92
25, wd:           x
25, NOP

26, PC:         96
26, wd:           x
26, NOP

27, PC:        100
27, wd:           x
27, NOP

28, PC:        104
28, wd:           x
28, NOP

29, PC:        108
29, wd:           x
29, NOP

30, PC:        112
30, wd:           x
30, NOP

31, PC:        116
31, wd:           x
31, NOP

32, PC:        120
32, wd:           x
32, NOP
```

```
33, PC:        124
33, wd:          x
33, NOP

34, PC:        128
34, wd:          x
34, NOP

35, PC:        132
35, wd:          x
35, NOP

36, PC:        136
36, wd:          x
36, NOP

37, PC:        140
37, wd:          x
37, NOP

38, PC:        144
38, wd:          x
38, NOP

39, PC:        148
39, wd:          x
39, NOP

40, PC:        152
40, wd:          x
40, NOP

41, PC:        156
41, wd:          x
41, NOP

42, PC:        160
42, wd:          x
42, NOP

43, PC:        164
43, wd:          x
43, NOP

44, PC:        168
44, wd:          x
44, NOP

45, PC:        172
45, wd:          x
45, NOP

46, PC:        176
46, wd:          x
46, NOP

47, PC:        180
47, wd:          x
47, NOP

48, PC:        184
48, wd:          x
48, NOP

49, PC:        188
49, wd:          x
49, NOP

50, PC:        192
50, wd:          x
50, NOP

51, PC:        196
51, wd:          x
51, NOP

52, PC:        200
52, wd:          x
52, NOP

53, reading data: Mem[       204] =>      17424
53, PC:        204
53, wd:          x
53, NOP

54, reading data: Mem[       208] =>      36882
54, PC:        208
54, wd:          x
54, Hi

55, reading data: Mem[       212] =>   36831257
55, PC:        212
55, wd:          x
55, Lo
```

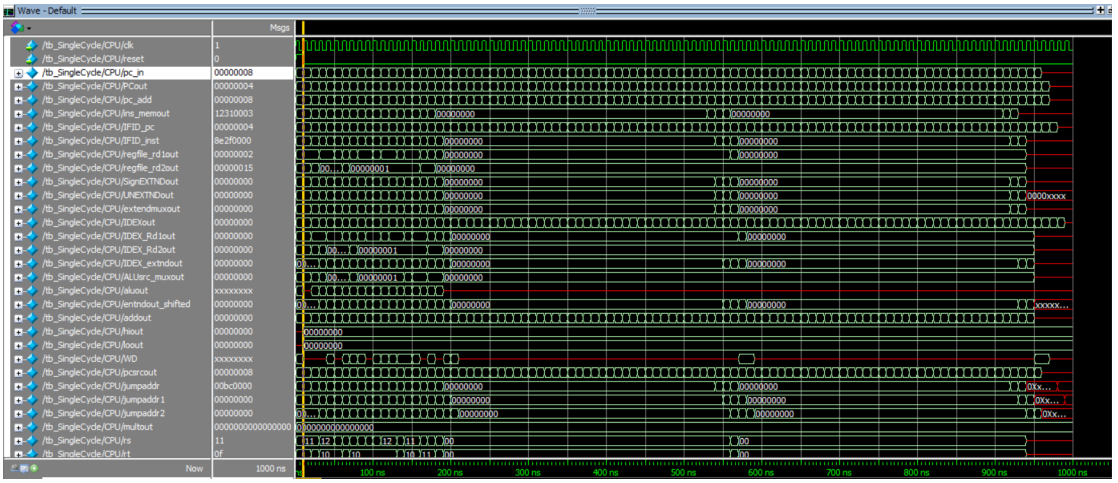
```
56, reg_file[18] =>          0 (Port 2)
56, reg_file[17] =>          4 (Port 1)
56, PC:       216
56, wd:          x
56, MULTU

57, reg_file[ 0] =>          0 (Port 2)
57, reg_file[ 0] =>          0 (Port 1)
58, reg_file[ 8] <=          0 (Write)
57, PC:       220
57, wd:          0
57, NOP

59, reg_file[18] <=          0 (Write)
58, PC:       224
58, wd:          0
58, NOP

59, PC:       228
59, wd:          x
59, NOP

60, PC:       232
60, wd:          x
60, NOP

61, PC:       236
61, wd:          x
61, NOP

62, PC:       240
62, wd:          x
62, NOP

63, PC:       244
63, wd:          x
63, NOP

64, PC:       248
64, wd:          x
64, NOP

65, PC:       252
65, wd:          x
65, NOP
```

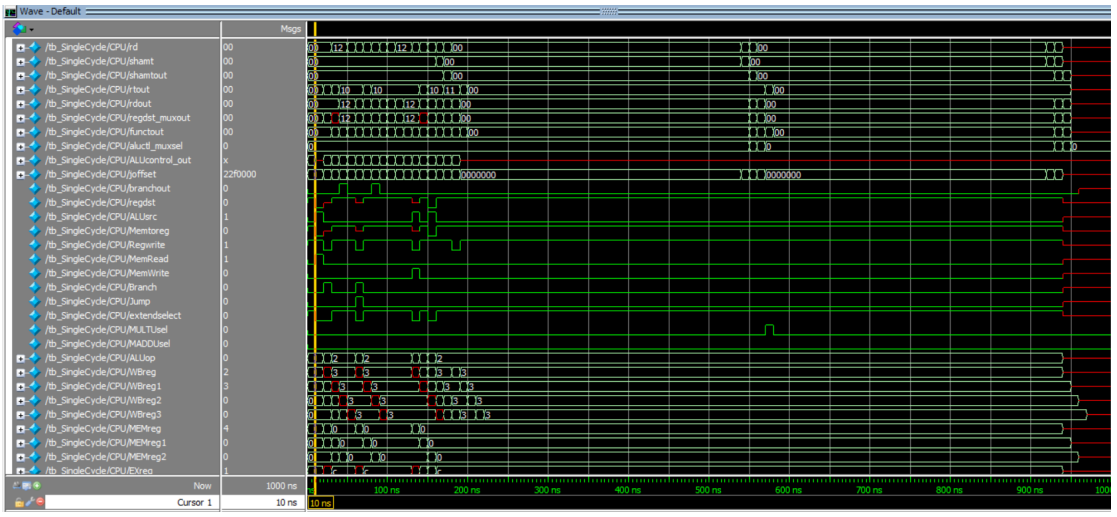
```
66, PC:       256
66, wd:          x
66, NOP

67, PC:       260
67, wd:          x
67, NOP

68, PC:       264
68, wd:          x
68, NOP

69, PC:       268
69, wd:          x
69, NOP

70, PC:       272
70, wd:          x
70, NOP

71, PC:       276
71, wd:          x
71, NOP

72, PC:       280
72, wd:          x
72, NOP

73, PC:       284
73, wd:          x
73, NOP

74, PC:       288
74, wd:          x
74, NOP

75, PC:       292
75, wd:          x
75, NOP

76, PC:       296
76, wd:          x
76, NOP

77, PC:       300
77, wd:          x
77, NOP
```

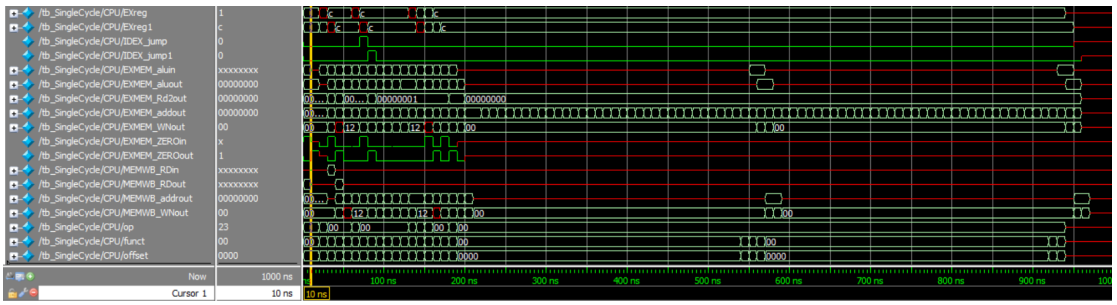
```
78, PC:         304
78, wd:           x
78, NOP

79, PC:         308
79, wd:           x
79, NOP

80, PC:         312
80, wd:           x
80, NOP

81, PC:         316
81, wd:           x
81, NOP

82, PC:         320
82, wd:           x
82, NOP

83, PC:         324
83, wd:           x
83, NOP

84, PC:         328
84, wd:           x
84, NOP

85, PC:         332
85, wd:           x
85, NOP

86, PC:         336
86, wd:           x
86, NOP

87, PC:         340
87, wd:           x
87, NOP

88, PC:         344
88, wd:           x
88, NOP

89, PC:         348
89, wd:           x
89, NOP
```

```
#                90, PC:       352
#                90, wd:         x
#                90, NOP
#
#                91, reading data: Mem[       356] =>      17424
#                91, PC:       356
#                91, wd:         x
#                91, NOP
#
#                92, reading data: Mem[       360] =>      36882
#                92, PC:       360
#                92, wd:         x
#                92, Hi
#
#                93, reading data: Mem[       364] =>          x
#                93, PC:       364
#                93, wd:         x
#                93, Lo
#
#                94, reg_file[ x] =>          x (Port 2)
#                94, reg_file[ x] =>          x (Port 1)
# control_pipelined unimplemented opcode  x
#                94, PC:       368
#                96, reg_file[ 8] <=          0 (Write)
#                95, PC:       372
#                97, reg_file[18] <=          0 (Write)
#                96, PC:       376
#                97, PC:         x
#                98, PC:         x
#                99, End of Simulation
```

四、waveform


Wave - Default
Msgs
/tb_SingleCycle/CPU/clk
/tb_SingleCycle/CPU/reset
/tb_SingleCycle/CPU/pc_in
/tb_SingleCycle/CPU/PCout
/tb_SingleCycle/CPU/pc_add
/tb_SingleCycle/CPU/ins_memout
/tb_SingleCycle/CPU/IFID_pc
/tb_SingleCycle/CPU/IFID_inst
/tb_SingleCycle/CPU/regfile_rd1out
/tb_SingleCycle/CPU/regfile_rd2out
/tb_SingleCycle/CPU/SignEXTNDout
/tb_SingleCycle/CPU/UnEXTNDout
/tb_SingleCycle/CPU/extendmuxout
/tb_SingleCycle/CPU/IDEXout
/tb_SingleCycle/CPU/IDEX_Rd1out
/tb_SingleCycle/CPU/IDEX_Rd2out
/tb_SingleCycle/CPU/IDEX_extndout
/tb_SingleCycle/CPU/ALUsrc_muxout
/tb_SingleCycle/CPU/aluout
/tb_SingleCycle/CPU/entndout_shifted
/tb_SingleCycle/CPU/addout
/tb_SingleCycle/CPU/hiout
/tb_SingleCycle/CPU/loout
/tb_SingleCycle/CPU/WD
/tb_SingleCycle/CPU/pcsrcout
/tb_SingleCycle/CPU/jumpaddr
/tb_SingleCycle/CPU/jumpaddr1
/tb_SingleCycle/CPU/jumpaddr2
/tb_SingleCycle/CPU/multout
/tb_SingleCycle/CPU/rs
/tb_SingleCycle/CPU/rt
Now
1000 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns
800 ns
900 ns
1000 ns


Wave - Default
Msgs
/tb_SingleCycle/CPU/rd
/tb_SingleCycle/CPU/shamt
/tb_SingleCycle/CPU/shamtout
/tb_SingleCycle/CPU/rtout
/tb_SingleCycle/CPU/rdout
/tb_SingleCycle/CPU/regdst_muxout
/tb_SingleCycle/CPU/functout
/tb_SingleCycle/CPU/aluctl_muxsel
/tb_SingleCycle/CPU/ALUcontrol_out
/tb_SingleCycle/CPU/joffset
/tb_SingleCycle/CPU/branchout
/tb_SingleCycle/CPU/regdst
/tb_SingleCycle/CPU/ALUsrc
/tb_SingleCycle/CPU/Memtoreg
/tb_SingleCycle/CPU/Regwrite
/tb_SingleCycle/CPU/MemRead
/tb_SingleCycle/CPU/MemWrite
/tb_SingleCycle/CPU/Branch
/tb_SingleCycle/CPU/Jump
/tb_SingleCycle/CPU/extendselect
/tb_SingleCycle/CPU/MULTUsel
/tb_SingleCycle/CPU/MADDUsel
/tb_SingleCycle/CPU/ALUop
/tb_SingleCycle/CPU/WBreg
/tb_SingleCycle/CPU/WBreg1
/tb_SingleCycle/CPU/WBreg2
/tb_SingleCycle/CPU/WBreg3
/tb_SingleCycle/CPU/MEMreg
/tb_SingleCycle/CPU/MEMreg1
/tb_SingleCycle/CPU/MEMreg2
/tb_SingleCycle/CPU/EXreg
Now
1000 ns
Cursor 1
10 ns
100 ns
200 ns
300 ns
400 ns
500 ns
600 ns
700 ns
800 ns
900 ns



五、心得:

1027102仇瀚慶:

在這次的PROJECT中，我認為比期中的難度差了好多，在期中project時做出來的東西，結果現在只是變成一個架構圖的一小部分而已，而且趕在期中PROJECT完就要寫期末了，一開始我們還沒有接流水線的時候都以為會順利的做完，結果其實最難的地方就是接線了，我們一開始就在PIPELINE花了不少的時間，看了課本，聽老師上課，回去複習，再到實作出來。在流水線寫出來的時候，以為可以快快結束了，並沒有。直接COMPILE ERROR一整行，讓我們頭非常痛，有夠麻煩。之後底霸個也讓我們戰了好幾個夜晚，不過雖然做這件事的過程很煎熬，但最後有弄出一些小成果出現，讓我也很開心。不過我以後應該不會想碰硬體了哈哈哈哈。

11027131李虹昀:

我們一開始還沒做pipeline的時候，感覺做起來很順，但碰到了pipeline的時候，error就跳出一大堆，各種的問題==，幾乎每一個模組都有毛病，連compile都過不

了。印出來答案時wd都是xx，跑wave的時候一堆都是紅線，我們在要交作業前每天都meeting超過了5個小時，有時候甚至沒吃晚餐，就為了抵出bug。

在交前一個禮拜睡的時數都不超過6小時，很常不知道不覺看到天亮，有種壽命又減10歲的感覺，但在學習的過程中也默默的學到了許多東西，從一開始不知道怎麼接到了解整個架構圖，雖然學習過程是辛苦的，但也學到了不少，對CPU的架構又更加了解。

11027142林蕙郁:

這次的期末PROJECT我覺得感覺和期中的難度真的差很多，在我們期中的時候至少課本翻一翻也能有大概的想法可以寫出來或者是網路上可以參考，但到期末PROJECT之後發覺，除了了解他的流程，在實現程式碼又是另外一回事了，寫PIPELINE，新指令，一開始還用HAZARD都遇到了一堆困難(但在聽老師說用NOP就得救了因為HAZARD不用寫，耶)，我們在交前一整個禮拜每天都MEETIING了不知道有幾個小時，感覺每天都快要天亮的時候才跑去休息，而且還不小心引來了壞習慣，在半夜一看著螢幕底霸個，手就會很癢想要吃零食。不過雖然真的很辛苦，但也學到了許多。

11027144陳佳音:

在寫程式碼的過程中，遇到各種錯誤和困難真的是一段煎熬的旅程。然而，我們通過堅持和持續的努力，我們最終成功還是克服了這些問題，真的非常開心。在寫程式碼的時候常常感覺快好了，但後來發覺問題越來越多，好不容易debug結束，結果跑模擬又出現其他問題，每天花超多的時間在討論新出現的bug是因為什麼原因造成的，還好每一次問題最後都有被解決。

我們在實作pipelining的時候真的超挫折，因為常常搞不清楚問題出在哪裡，pipelining引入了時序和數據相依的複雜性都有可能導致各種錯誤，所以要對每個module仔細的檢查和調整，才能夠確保數據和控制信號的正確傳輸，但我們在逐行看程式的時候還是常常會看到迷路。

雖然在接的時候過得很不順，但在我們堅持與努力之下，東西最後還是有看起來還ok的結果，雖然不敢確定答案一定是正確的，但我們在寫得過程中也學習到許多的知識。

六、負責項目

報告、繪製、程式撰寫: 一起完成