計算機組織期末報告 108 學年度第二學期

第9組

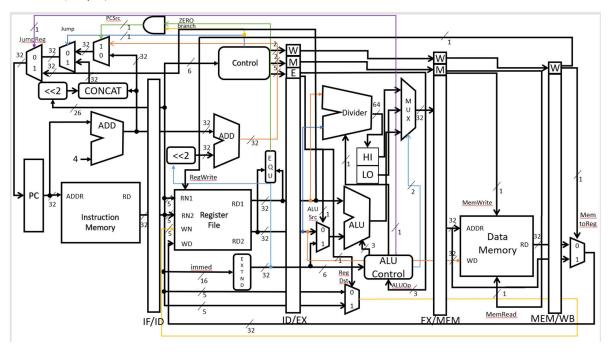
老師:朱守禮 老師 學生: 10727214 洪友祥

> 10727215 沈家丞 10727221 邱晨凱 10727248 鄭珮慈

一、背景

使用 Verilog HDL 與 Modelsim 模擬器,以 Midterm Project 所設計之 ALU Design 為基礎,參考課本 Chapter 4 與課程講義之 Pipelined Datapath,設計一個 Pipelined MIPS Lite CPU 。

二、架構圖



三、方法

(1)ALU

沿用 Midterm Project 的設計,以 32 個 1-bit ALU 組成 32-bits ALU,具有 32-bits AND、OR、ADD、SUB、SLT 的功能,並加入 Shifter 來滿足 SRL 指令,而從 alu_crl 接收到的 crl 訊號共三位元,最低兩個位元當作 alu_one 的 sel,而最高為元為 binvert。

(2)Divider

設計一個 32-bits 無號數除法器,設計時設置餘數和商數為 32 位元,除數為 64 位元,最一開始程式會先將記數的變數 i 設置為十進位的 0 ,再來重設商數,並將餘數設為被除數,除數向前移動 32 位元,並且 i 會+1,這是 i = 0 的狀況,其他在 i 還沒到 32 之前的動作都是,i 先+1,餘數-除數,接著判斷餘數是正是負,若是正就將商左移一位最後一位設 1,反之則左移一位最後一位設 0,並且餘數=餘數+除數。兩種情況之後除數都會又移 1,如跑完 33 次,i=33,就將商數存是後 32 位元的 32 位元的 32 位元的 32 位元的 32 位元的 32 位元的 33 位元的 33 位元的 33 位元的 34

(3)alu_crl

判斷 ALUOp 決定要值行的指令是 Add、Sub、Or、And、S1t, 並將對應訊號輸入電路中。

(4)control_pipelined

利用輸入的指令代號 opcode,產生對應的控制訊號,並於內部做好暫存器 EX、MEM、WB 之設定。

控制訊號如下表所示:

	RegDst	ALUSrc	MemtoReg	RegWrite	MemRead	MemWrite	Branch	Jump	ALUOp	ExtendSel
R_FORMAT	1	0	0	1	0	0	0	0	010	1
ORI	0	1	0	1	0	0	0	0	100	1
LW	0	1	1	1	1	0	0	0	000	0
SW	X	1	X	0	0	1	0	0	000	0
BEQ	X	0	X	0	0	0	1	0	001	0
J	X	0	X	0	0	0	1	1	001	0

(5)add32

實現 32 位元加法,此模組用於 PC 及 branch 的位址加法。

(6)branch equ

將原本在 ALU 中做判斷的 beq 指令拉出,當 opcode 為 beq 且 Register File 兩輸出值相等時,便將 ZERO 設為 1,若否,則設為 0。

(7)memory

用於 Instruction Memory 與 Data Memory, 主要根據 MemRead 和 MemWrite 去 決定對指定記憶體位置的讀取、寫入。

(8)mips_pipelined

整合所有 module 以及處理訊號傳送。

(9) IF/ID

為 IF (Instruction Fetch) 與 ID (Instruction Decode) 之間的暫存器。當 clock 正緣觸發,若rst為true,將輸出值設為0;若rst為false,將輸入值傳送給輸出值,藉此完成訊號傳遞。

(10) ID/EX

為 ID (Instruction Decode) 與 EX (Execute Calculation) 之間的暫存器。當 clock 正緣觸發,若rst為true,將輸出值皆設為0;若rst為false,將輸入值傳送給輸出值,藉此完成訊號傳遞。

(11)EX/MEM

為 EX (Execute Calculation) 與 MEM (Memory Access) 之間的暫存器。當 clock 正緣觸發,若 rst 為 true,將輸出值皆設為 0;若 rst 為 false,將輸入值傳送給輸出值,藉此完成訊號傳遞。

(12)MEM/WB

為 MEM (Memory Access) 與 WB (Write Back) 之間的暫存器。當 clock 正緣觸發,若 rst 為 true,將輸出值皆設為 0;若 rst 為 false,將輸入值傳送給輸出值,藉此完成訊號傳遞。

(13)mux2

二對一多工器,用於選擇 ALU 的 operand2 輸入、PC 要 fetch 下一道指令的位址、欲寫回的暫存器以及內容。

(14)mux3

三對一多工器,用於選擇 EX 階的運算結果(Hi/Lo/ALU)。

(15)reg_file

用於 Register File,主要根據 RegWrite 決定暫存器是否寫入。

(16)reg32

32 位元暫存器,在此當作 PC 使用。

(17) sign extend

將 16 位元的值進行有號數擴充成 32 位元。

(18) unsign extend

將 16 位元的值進行無號數擴充成 32 位元。

(19)tb_Pipelined

根據時脈不斷地執行程式碼,而當 CPU. funct 在不同值的時候,分別執行不同的指令。

四、結果

我們根據以下指令來做測試

- (1) lw \$s1, \$t7, 0
- (2) srl \$t7, \$t6, 3
- (3) beq \$s1, \$s2, 6
- (4) add \$s2, \$s0, \$s2
- (5) sub \$s2, \$s0, \$s2
- (6) add \$s1, \$s0, \$s1
- (7) or \$s2, \$s0, \$s2
- (8) add \$s1, \$s0, \$s1
- (9) sub \$s2, \$s0, \$s2
- (10) or \$s2, \$s0, \$s2
- (11) lw \$s1, \$t5, 0
- (12) ori \$s2, \$s0, 4
- (13) sw \$zero, \$s2, 24
- (14) divu \$t7, \$v1
- (15) mfhi \$s1
- (16) mflo \$s0

```
0, reading data: Mem[ x] => x
# 18446744073709551615, PC: x
                                           0] => 238
0 (Port 2)
0 (Port 1)
                                              0] => 2385444864
                   0, reading data: Mem[
                   0, reg_file[ 0] =>
                   0, reg_file[ 0] =>
                   0, PC:
                                0
                   O, NOP
                   1, reading data: Mem[ 4] =>
1 reg_file[15] => 21 (Port 2)
2 (Port 1)
                                                             0
                   1, PC:
                   1, LW
                   2, reg_file[ 0] => 0 (Port 2)
2, reg_file[ 0] => 0 (Port 1)
                   2, PC:
                   2, NOP
                   3, reading data: Mem[
                                              2] => 256
                   3, PC: 12
                   3, NOP
                                              16] => 1011906
                   4, reading data: Mem[
                                          256 (Write)
                   5, reg_file[15] <=
                           16
                   4, PC:
                   4, NOP
                   5, reading data: Mem[ 20] => 30
256 (Port 2)
                                             20] => 305201158
                   5, PC: 20
                   5, SRL
                   6, reading data: Mem[
                                             24] => 0
                                           24] =>
2 (Port 2)
                   6, reg_file[17] =>
                   6, reg_file[17] =>
                                            2 (Port 1)
                   6, PC: 24
                   6, BEQ
                                          48] => 36735008
0 (Port 2)
0 (Port 1)
                   7, reading data: Mem[
                   7, reg_file[ 0] =>
                   7, reg_file[ 0] =>
                   7, PC:
                              48
                   7. NOP
```

```
8, reading data: Mem[ 52] =>
8, reg_file[16] => 1 (Port 2)
2 (Port 1)
22 (Write)
                                             32 (Write)
                    8, PC:
                                 52
                    8, ADD
                    9, reg_file[ 0] =>
                                             0 (Port 1)
0 (Port 2)
                    9, reg_file[ 0] =>
                    9, PC:
                                 56
                    9, NOP
                   10, reading data: Mem[ 60] => 38834213
run
                   10, PC:
                                60
                   10, NOP
                   11, reading data: Mem[
                                               64] => 36735008
                                             1 (Port 2)
3 (Port 1)
                   11, reg_file[16] =>
                   11, reg_file[18] =>
                                              3 (Write)
                   12, reg_file[17] <=
                   11, PC:
                               64
                   11, OR
                   12, reading data: Mem[ 68] =>
                                              3 (Port 1)
                   12, reg_file[17] =>
                   12, PC:
                            68
                   12, ADD
                   13, reg file[ 0] =>
                                              0 (Port 2)
                   13, reg_file[ 0] =>
                                              0 (Port 1)
                              72
                   13, PC:
                   13, NOP
                   14, reading data: Mem[ 76] => 38834210
15, reg_file[18] <= 3 (Write)
                   14, PC: 76
                   14, NOP
                   15, reading data: Mem[
                                               80] =>
                                                               0
                                             1 (Port 2)
3 (Port 1)
                   15, reg_file[16] =>
                   15, reg_file[18] =>
                                              4 (Write)
                   16, reg_file[17] <=
                               80
                   15, PC:
                   15, SUB
```

```
16, reg_file[ 0] => 0 (Port 1)
16, reg_file[ 0] => 0 (Port 2)
                   16, PC:
                                84
                   16, NOP
                   17, PC:
                                88
                   17, NOP
                   18, reading data: Mem[ 92] => 2 (Write)
                                              92] => 38834213
                            92
                   18, PC:
                   18, NOP
                                              96] => 2385313792
                   19, reading data: Mem[
                                            96] => 230
2 (Port 1)
                   19, reg_file[18] =>
                   19, reg_file[16] =>
                                              1 (Port 2)
                   19, PC:
                            96
                   19, OR
                   20, reading data: Mem[
                                              100] =>
                   20, reg_file[13] =>
                                            19 (Port 2)
                   20, reg_file[17] =>
                                              4 (Port 1)
run
                   20, PC:
                                100
                   20, LW
                   21, reg_file[ 0] =>
                                             0 (Port 2)
                   21, reg_file[ 0] =>
                                             0 (Port 1)
                   21, PC:
                             104
                   21, NOP
                                             108] => 911212548
                   22, reading data: Mem[
                   22, reading data: Mem[
                                               4] => 33554432
                   23, reg_file[18] <=
                                              3 (Write)
                            108
                   22, PC:
                   22, NOP
                   23, reading data: Mem[
                                             112] => 2886860824
                   23, reg_file[18] => 3 (Port 1)
23, reg_file[16] => 1 (Port 2)
                   24, reg_file[13] <= 33554432 (Write)
                   23, PC:
                              112
                   23, ORI
```

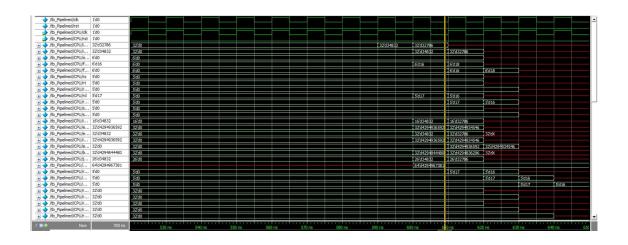
```
24, reading data: Mem[ 116] => 31653915
                   24, reg_file[ 0] =>
                                              0 (Port 1)
                   24, reg_file[18] =>
                                               3 (Port 2)
                   24, PC:
24, SW
                               116
                                             120] =>
3 (Port 2)
                   25, reading data: Mem[
                   25, reg_file[ 3] =>
                                            256 (Port 1)
                   25, reg_file[15] =>
                               120
                   25, PC:
                   25, DIVU
                   26, reg_file[ 0] =>
                                             0 (Port 2)
                   26, reg_file[ 0] =>
                                             0 (Port 1)
                   27, writing data: Mem[
                                               24] <=
                                                                3
                   27, reg_file[16] <=
26, PC: 124
26, NOP
                                              7 (Write)
                   27, PC:
                                128
                   27, NOP
                   28, PC:
                                 132
                   28, NOP
                   29, PC:
                               136
                   29, NOP
run
                   30, PC:
                                 140
                   30, NOP
                   31, PC:
                                 144
                   31, NOP
                   32, PC:
                               148
                   32, NOP
                   33, PC:
                               152
                   33, NOP
                   34, PC:
                                156
                   34, NOP
```

4	35, PC:	160	
	35, NOP		
	00, 1101		
	26 pc.	164	
l.	36, PC:	164	
!	36, NOP		
*		2.22	
#	37, PC:	168	
#	37, NOP		
#			
•	38, PC:	172	
	38, NOP		
	2.74		
	39, PC:	176	
i.	39, NOP	2.0	
I.	SS, HOP		
*			
run			
+	40, PC:	180	
#	40, NOP		
#			
#	41, PC:	184	
#	41, NOP		
į.	42, PC:	188	
	42, NOP	200	
1	42, 1101		
	40 00.	100	
lī.	43, PC:	192	
*	43, NOP		
*			
	44, PC:	196	
+	44, NOP		
#			
#	45, PC:	200	
#	45, NOP		
i i	,		
4	46, PC:	204	
4		204	
	46, NOP		
!			
#	47, PC:	208	
#	47, NOP		
*			
#	48, PC:	212	
#	48, NOP		
#			
	49, PC:	216	
	49, NOP		
I.	10, 1101		

```
50, PC:
                               220
                  50, NOP
                  51, PC:
                               224
                  51, NOP
                  52, PC:
                               228
                  52, NOP
                  53, PC:
                               232
                  53, NOP
                  54, PC:
                               236
                 54, NOP
                  55, PC:
                               240
                  55, NOP
                  56, PC:
                               244
                  56, NOP
                 57, PC:
                               248
                 57, NOP
                  58, PC:
                               252
                  58, NOP
                  59, reading data: Mem[
                                           256] => 34832
                  59, PC:
                             256
                  59, NOP
                  60, reading data: Mem[
                                            260] =>
                                                         32786
VSIM 5> run
                  60, PC:
                              260
                  60, MFHI
                  61, reading data: Mem[
                                            264] =>
                                                           ×
                  61, PC:
                             264
                  61, MFLO
                  62, reg_file[ x] =>
                                            x (Port 2)
                  62, reg_file[ x] =>
                                            x (Port 1)
# control_single unimplemented opcode x
                  62, PC:
                             268
                  64, reg_file[17] <=
                                            1 (Write)
                  63, PC:
                  65, reg_file[16] <=
                                      85 (Write)
```

上方為 16 道指令於 terminal 輸出的結果,能觀察出指令之暫存器數值與輸出結果相符合。下方為 16 道指令顯示之 waveform 圖形。

												·
/tb_Pipelined/CPU/dk 1'd0												
	(32'd2385444864 \ 32'd0	,		[32'd 10 1 1906	32'd305201158		32'd36735008	32'd0		32'd38834213	32'd36735008	I32d0
	(32d0 32d2	2385444864 【32'd0				32'd305201158 6'd4	32'd0 6'd0	32d36735008	32'd0		32'd38834213	32'd36735008
	(6'd0 [6'd35	, ,,,,,,					6'd0	[6'd32	6'd0		6'd37	[6'd32
	(6'd0 (5'd0 [5'd17	7 [5d0				6'd2 5'd17	[6'd6 [5'd0	6'd0 5'd17	6'd32 5'd0	(6'd0	[5'd18	[6'd37 [5'd17
→ /tb_Pipelined/CPU/rt 5'd0	(5'd0 [5'd15				[5'd15		5'd0	5'd16	5'd0		5'd16	1501/
	(5'd0 (5'd0	5'd15	5°d0			5'd15 5'd0	5d17	5'd0 5'd17	5'd16	5'd0	[5'd18	[5'd16 [5'd17
→ /tb_Pipelined/CPU/r 5'd0	(5'd0					5d14	[5'd0	13017		5'd0	13010	15'd18
	(5'd0 (5'd0				5'd3	5'd0 5'd3	[5'd0					
★ /tb_Pipelined/CPU/i 16'd34832	(16'd0				16'd28866	16'd6	16'd0	16'd34848	16'd0		16'd36901	16'd34848
 → /tb_Pipelined/CPU/e 32'd4294936592 → /tb_Pipelined/CPU/e 32'd34832 	(32d0 (32d0				32'd28866 32'd28866	32'd6 32'd6	32'd0 32'd0	32'd4294936608 32'd34848	32'd0 32'd0		32'd4294938661 32'd36901	32'd4294936608 32'd34848
★ /tb_Pipelined/CPU/i 32'd4294936592	(32'd0					32'd6	32'd0	32'd4294936608	32'd0		32'd4294938661	32'd4294936608
	(3Zd0 (3Zd0				[32'd115464		32'd6 32'd0	32'd0 32'd4294844544	32d4294936608	32'd0		3Zd4294938661 3Zd4294844544
★ /tb_Pipelined/CPU/j 26'd34832	(26'd0 26'd3	36634624 【26'd0					26'd0		26'd0		26'd38834213	
 ★ /tb_Pipelined/CPU/ 64'd4294967381 ★ /tb_Pipelined/CPU/r 5'd0 	(5'd0	5d15	[5'd0			5'd14	5'dX	(5'd0	5'd17	5'd0		I5'd18
+ > /tb_Pipelined/CPU/ 5'd0	(5'd0	3013	5d15	[5'd0		13011	5d14	5'dX	5'd0	5'd17	[5'd0	
	(5'd0 (32'd0 [32'd2	2 I 32'd0		(5'd15	[5'd0	[32'd2	[32'd0	[5'd14 [32'd2	5'dX 132'd0	5'd0	5'd17 132'd3	[5'd0
→ /tb_Pipelined/CPU/r 32'd0	(32'd0	[32'd2	(32'd0				32'd2	32'd0	32'd2	32'd0		[32d3
	(3Zd0 [3Zd2 (3Zd0	21 32'd0 32'd21	32'd0		32'd256	32'd2 32'd256	32'd0 32'd2	32'd1 32'd0	32d0 32d1	32'd0	[32d1	[32d1
→ /tb_Pipelined/CPU/r 32'd0	(3Zd0		32'd21	32'd0			32'd256			32'd1	32'd0	-
25 € Now 700 n		20 ns	30 ns 4	Ons 5	0 ns 60	ins 70	ns 8	Ons 9	ns 10	0 ns 1	10 ns 12	10 ns 130
Cursor 1 609 n	5											
→ /tb_Pipelined/dk 1'd0 → /tb_Pipelined/rst 1'd0												-
	3Zd0 [3Zd38	8834210 I 32'd0			3Zd38834213	32'd2385313792	32'd0		37d911212549	37/d2886860924	32'd31653915	37d0
 /tb_Pipelined/CPU/i 32'd34832 	(32d0	32d38834210	32'd0		2233037213	32'd38834213	32'd2385313792			32d911212548	32'd2886860824	32d316S3915
	(e,qo	I6'd34	[6'd0				6'd35	[6'd0		6'd13 6'd4	6'd43	6'd0 6'd27
	(6'd32 [6'd0		6'd34	6'd0			6d37	6'd0			[6'd4	6'd24
	(5,90 (2,90	I 5'd18 I 5'd16	5'd0 5'd0				5d17 5d13	5'd0		I 5'd18 I 5'd16	[5'd0 [5'd18	5'd15 5'd3
	5'd16 5'd0		5'd16	5'd0			5d16	5'd13	5'd0		[5d16	5d18
	(5'd0 (5'd17 15'd0	I 5'd 18	5'd0 5'd18	15d0			[5'd0 [5'd18	[5'd0				
+ > /tb Pipelined/CPU/s 5'd0	5'd0											
	5'd0 (16'd0	I 16'd36898	16'd0			16'd3690'1	I 16'd0			I 16'd4	[16'd24	16'd27
+ > /tb Pipelined/CPU/e 32'd4294936592	(32d0	32'd4294938658	32'd0			32'd4294938661	32'd0			32'd4	32'd24	32'd27
	(32'd0 (32'd0	32'd36898 32'd4294938658	32'd0 32'd0			32'd36901 32'd4294938661	32'd0 32'd0			32'd4 132'd4	32'd24 32'd24	32'd27 32'd27
→ /tb_Pipelined/CPU/e 32'd0	(3Zd4294936608		3Zd4294938658	[32'd0			32'd4294938661	32'd0			[32'd4	32'd24
	(32'd0 (26'd0	32'd4294852744 26'd38834210	26'd0			32'd4294852756 26'd38834213	26'd36503552	26'd0		32d16 26'd38797316	32'd96 26'd1179672	3Zd108 26'd31653915
	(5d17 I5d0		5d18	[5'd0			[5d18	5'd13	15'd0		[5d16	5'dX
+ /tb_Pipelined/CPU/ 5'd0	(5d18 5'd17	[5'd0	2018	5'd18	[5'd0		5018			I5'd0		5'd16
	5d0 5d18 (32d0	[5d17 [32d3	5'd0 32'd0			5'd0 32'd2	32'd4	32'd0		5'd13 32'd3	[5'd0 [32'd0	32'd256
+ > /tb_Pipelined/CPU/r 32'd0	32'd3 32'd0		32'd3	32'd0			32'd2	32d4	32'd0		32'd3	32d0
	32'd0 32'd1 [32'd0	I 32'd1	32'd0 32'd1	32'd0		32'd1	32d19 32d1	32'd0 32'd19	32'd0	32d1	[32'd3 [32'd1	[32d3
	(32'd1	32'd0		32d1	32d0			[32d1	32d19	32'd0		13Zd1
	140 mg					THE RESERVE AND ADDRESS OF THE PERSON NAMED IN	-	-				
	14016	150 ns 1	60 ns 17	0 ns 18	0 ns 19	ns 20	0 ns 21	0 ns 22	Ons 23	0 ns 2	10 ns 25	i0 ns 260
/tb_Pipelined/dk 1'd0		150 ns 1	60 ns 17	0 ns 18	0 ns 19	ns 20	0 ns 21	0 ns 22	ns 23	0 ns 2	90 ns 25	i0 ns 260
/tb_Pipelined/rst 1'd0 /tb_Pipelined/CPU/ck 1'd0		150 ns 1	60 ns 17	0 ns 18	0 ns 19	ns 20	0 ns 21	0 ns 22	ons 23	Ons 2	90 ns 25	0 ns 260
/tb_Pipelined/cPU/ck 1'd0 /tb_Pipelined/CPU/ck 1'd0 /tb_Pipelined/CPU/rst 1'd0	37/d0	150 ns 1	60 ns 17	0 ns 18	Ons 19	ns 20	0 ns 2:	0 ns 22) ns 23	0 ns 2	00 ns 25	0 ns 26(
/tb_Pipelined/rst 1'd0 /tb_Pipelined/CPU/ck 1'd0 /tb_Pipelined/CPU/st 1'd0 /tb_Pipelined/CPU/i 32'd32786 /tb_Pipelined/CPU/i 32'd34932	32'd0 (32'd0	150 ns 1	60 ns 17	0 ns 18	0 ns 19	ns 20	0 ns 21	0 ns 22) ns 23	0 ns 2:	00 ns 25	0 ns 26(
/b_Ppelned/rst 1d0 1d0	(3Zd0 6d0	150 ns 3	60 ns 17	0 ns 18	0 ns 19	ns 20	Ons 2:	0 ns 22	Ons 23	0 ns 2:	0 ns 25	0 ns 266
//tb. Ppelined/PU/dk 1/d0	(32'd0 6'd0 (6'd0 (6'd27 [6'd0	150 rs 2	60 ns 17	0 ns 18	0 ns 19	ns 20	Ons 21	0 ns 22) ns 23	0 ns 2	0 ns 25	0 0 20 20 20 20 20 20 20 20 20 20 20 20
Np. Ppelined/rst 150	(32'd0 6'd0 (6'd0	150 rs 1	60 ns 17	0 ns 18	0 ns 19	ons 20	Ons 2:	0 ns 22	ons 23	0 ns 2-	90 ns 25	0 ns 250
Ps. Ppelined/rst 150	(3Zd0 6d0 (6d0 (6d27 (5d0 (5d0 (5d3	150 es 1	60 ns 17	0 ns 18	0 ns 19	ns 20	Ons 21	0 ns 22	ns 23	0 ns 2	0 ns 25	0 ns 250
№ Pedindd91 100 № Pedindd91 100 № Pedindd7U[s] 100 № Pedindd7U[s] 100 № Pedindd7U[s] 100 10	(3Zdo 6do (6do (6do (6d2Z 6do (Sdo (Sdo (Sdo (Sdo Sdo	150 re 1	50 ns 17	0 ns 18	0 ns 19	ns 20	Ons 21	0 ns 22	ns 23	0 ns 2	0 ns 25	io na 260
10 PoplandcH1 10 10 10 10 10 10 10	(32'd) 640 (640 (6427 640 (540 (540 (543 540 540 540 540	150 re 1	50 ms 17	0 11	0 ns 19	ns 20	Ons 21	0 ns 22	ns 23	Ors 2-	0 ns 2:	b ns 26K
10 Poplandcht 10 10 10 10 10 10 10 1	(32d) 6d0 (6d0 (6d0 (6d27 (5d0 (5d0 (5d3 (5d3 (5d0 (5d3	150 re 1	60 ns 17	0 11	Ons 19	ns 20	Ons 21	0 ns 22	ns 23	Ors 2	9) ns 21	2 A
№ Pedindd91 100 1	(3740 (3740	10%	60 ns 17	Ons 18	0 ns 19	ns 20	Ors 21	0 ns 22	ns 23	Oris 2:	9) ns 21	Dris 2K
№ Pedindrist 100	(32'00 (32		60 ms 17	en 16	O'ns 19	ns 20	ors 21	One 22	ne 23	One 2	9 ns 25	A
10 Pepinedrist 100	(3260 (3260 (360		60 ms 17	en se	One 199	ne 200	ns 2:	5 es 22	nei 23	Om 2	0:00 25	5 m 26
№ Pednedd91 100 1	(32'00 (32		0 to 10	200 20	One 19	rs 200	ns 2:	5 ne 22	ore 23	Om 2	0.00	2 no 26
10	(32'00 (32		0 D	200 18	On 19	re 200	21	5 ne 22	re 23	200 2	0.00 25	0 ns 264
10 Pepinedrist 100	3250 560 660		O m 17	200 U	en 19	rei 200	21	S re 223	one 23	200	919 25	S on
10 Pepineddyst 100	\$250 \$250		17 T	00 15	50 to 100	re 200	21	6 m 22	72 22 22 22 22 22 22 22 22 22 22 22 22 2		20	5 m
10	\$250	550	O 0 1	100 U	Sec 19	re 200	21	6 m 22	72 22 22 22 22 22 22 22 22 22 22 22 22 2		20	2 m
10 Periodicity 100	1500 1500	\$350	Om 17			re 200	21	Sre 223	m 23		20	S on
10 Periodicity 100	3269	\$500	Om 17			76 JO	21	6 m 22	120 220 220 220 220 220 220 220 220 220		20	5 m
10 Periodicity 100	3269	\$350	00m 17	Om 3	6-m 19-	re 200	n 21	6 m 22	n 22	200	20 m 22 m	2 m 2 m
10	3269	\$350	200 E	Om 3	9 to 19 to 1	re 200	on 21	6 ne 223	ne 23	0m 3	30 ns 25	
10	3269	\$350	One 17	00 ms 3	0 cs 32	re 30	21 21 21 21 21 21 21 21 21 21 21 21 21 2	6 m 22	n 20	5 ns 3	20 mm	
10	3290 560	\$350	17 17 17 17 17 17 17 17 17 17 17 17 17 1	7 m	0 ne 32	70 20 m 33 m	on 21	8 m 22	m 23	000 20	No. 20	
16. Peolindid14 100	\$250	\$350	Om 17	00 U	0 cs 32	re 200	21 21 21 21 21 21 21 21 21 21 21 21 21 2	8 m 22	nt 22	On 2	20 cm	
10	\$250	\$350	00 17	One 3	Dom 32	re 30	on 21	8 m 22	22 22 23 24 25 25 25 25 25 25 25 25 25 25 25 25 25	200 200 200 200 200 200 200 200 200 200	20 m 20 m	
10	\$250	\$350	00m 17	0 w 10 w	0 es 10 es 1	re: 200	Ons >	8 m 22	m 23	00 2 2 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3	30 m 20	
10	1329 1490	\$350	Om 17	904 3	0 19 19 19 19 19 19 19 19 19 19 19 19 19	re 200	21 21 21 21 21 21 21 21 21 21 21 21 21 2	8 m 22	12 22 22 22 22 22 22 22 22 22 22 22 22 2		200	
10	\$250	\$350	On 17	900	Ber 19	re 200	on 21	6 m 22				
10	3250	\$350	909s X	009 3	Description 100	na 20	000 21	6 m 22	ns 22	000 20 20 20 20 20 20 20 20 20 20 20 20	70 mm 32 mm	
10	3200	\$350	17 (17 (17 (17 (17 (17 (17 (17 (17 (17 (200	0 co 32	10 20 20 20 20 20 20 20 20 20 20 20 20 20	21 21 21 22 22 22 22 22 22 22 22 22 22 2	8 m 22	72 22 23 24 25 25 25 25 25 25 25 25 25 25 25 25 25			
10	\$250	\$350	17 (17 (17 (17 (17 (17 (17 (17 (17 (17 (Des 12	re 200	on 21	6 nm 222				
10	1500 1500	\$350	00 V	0 m	0 or 22	ns 20	000 3-	6 m 22		0 m 3	20 Oct 20	
10	\$200	\$350	20 See 3	7 m	Beet 199	200	on 21	6 m 22	m 23	000	None 3	
10	13260	\$350	17 Table 18		Dom 22	re 200	on 21	on 22				
10	\$1200	\$350	000 X	00m 3	0 cs 32	re Joe	on 21	6 m 22		on 3		
10	\$250	\$350	000 X	0 cs 3		ne 20	on 21	on 22		000 3000		
10	\$200	\$350	17 (17 (17 (17 (17 (17 (17 (17 (17 (17 (200		ne 200	21	8 m 22				
10	Size	\$350		0 m 3		re 30	on 21	on 22				
10	Size	\$350	00m 17	0 os 3		ne 20	000 321	on 22		Down 3		
10	1500	\$350	00 17 18 18 18 18 18 18 18 18 18 18 18 18 18	7 m	Done 32	20 20 20 20 20 20 20 20 20 20 20 20 20 2		S n		Oran 2	9 / 2 / 2 / 2 / 2 / 2 / 2 / 2 / 2 / 2 /	
10	Size	\$350				re 200	on 21	on 22				
10	Size	\$350	On 17	0 cs 3	0 es 12	ns 200	on 21	Or 33		5 ns 3	70 Oct 31 Oct 31 Oct 32	



五、討論

Q1. 如何在做 branch 指令的運算時,下一道指令能夠 fetch 到正確的?

Ans: 在 register file 的兩個輸出 rs、rt 連接一個判斷是否相等的模組,在 ID 階就對立即值做有號數擴充,將是否做 branch 跳躍的判斷提前至 ID 階, stall 一個 cycle 不 fetch 指令,就能達到正確運算 branch 指令的效果。

Q2. 如何測試結果正不正確?

Ans: 必須將 Register file、Data memory 以及 instruction memory 設定完成後,一道一道檢驗輸出結果,追蹤該指令的 datapath,將訊號顯示於 waveform 後,可曉得在哪一個模組中的訊號不如預期,再針對該模組做修正。

Q3. 除法器做運算時,該如何處理 hazard 發生的問題?

Ans: 我們在做除法運算時,會在後方加入 34 個 nop 指令,使指令推遲避免發生 hazard \circ

六、結論(心得)

這次的 Final project,基本上是統整整個學期的東西了,延續了期中 project 較為簡單的除法器、位移器、ALU 等部分。而這次最困難的不外乎就是切 Pipeline 了,要去把所有指令,去切成五個部分,來達成同步執行指令的事情,以加速程式的 執行。而這次任務主要的東西,基本上都是在切 Pipeline,所以後來我們發現分工其 實很多人分到的程式碼部分,是可以不用做更改的,可能是因為我們一開始以為分工 可以跟期中 Project 一樣,只要更改程式碼達成教授的要求,那就可以了,結果後來 才發現全然不同,需要自行設計一個架構圖,來執行老師所要求需要的所有指令、規 範,我們最後也終於在幾乎快壓哨的時候完成了我們的 Final project。

七、未來展望

計算機組織這門課,我們從一開始陸續接觸了電腦系統組織,再來了解些 MIPS 有關的電腦語言,也因為經過之前的組合語言、嵌入式系統,更容易掌握這門課,接著陸續的 ALU 乘法器除法器,再到整個 CPU ,我們其實學得挺多的,希望未來的我們能學到更多 MIPS 指令,在硬體方面上能夠更熟悉。