April 8rd/2020

ENCM 369 - B04

Laboratory # 9

Michael Tagg

30080581

Exercise A

initially:

$17 = 0x0 ; @ t = 90ns, lw finished, add on write stage

① at t = 91ns, beq is on it's decode cycle and has not decided to branch yet.

PC = 0x0040_0090

shortly after { InstrD = 0x1220_fffa (beq instruction in decode)
PCPlus4D = 0x0040_0090 (beq + 4)

② at t = 92ns, beq is on execute stage. No branch decision still.

PC = 0x0040_0094

shortly after { InstrD = 0x0274_4024 (add instruction)
PCPlus4E = 0x0040_0078 (L1 address calculated)

③ at t = 93ns, beq has finished executing & will decide to branch to L1 next clock.

PC = 0x0040_0098 (L1 address in queue still)

shortly after { InstrD = 0x0274_4825 (or instruction decoded)
PCBranchM = 0x0040_0078 (carried through PCPlus4E)
ZeroM? = 1 (because $0 == 0 from beq instruction)

④ at t = 94ns, the L1 instruction will be fetched, add is on Memory, or instruction being executed and slt is decoding.

PC = 0x0040_0078 (L1 instruction)

shortly after: InstrD = 0x0274_4825 (From "or" decode stage)

⑤ at t = 95ns, PC = 0x0040_007c which is the instruction following L1 as no other branches were calculated. the lw instruction is now on decode stage.

thus, InstrD = 0x8e11_0000

Exercise 5

Part 1

- Hazard 2 occurs during the Execute phase of the addi instruction (line 7) the line 5 "sub" instruction is on its memory stage. The Hazard unit notices WriteregW & RsE both contain 11000_2 ($24) with RegWriteM = 1; in response, ForwardAE = "01" to accept ResultW.

- Hazard 3 occurs when the second add (line 9) is executing $9 + $10, Hazard unit sees RtE & WriteregW both contain: 01001_2 ($9) and MemtoRegW == 1; in response, ForwardB = "01" to use the memory value to be saved to $9 in the lw instruction.

- Hazard 4 occurs when sw (line 9) enters its execute stage, The Hazard Unit sees RsE & WriteRegM both contain: 01010_2 ($10) with RegWriteM and MemWriteE both = 1, the Hazard unit will forward the ALUOutM by setting ForwardAE = "10."

Part 2

- The circuit in Figure 7.50 does not contain signals or outputs to let the hazard unit stall, which could help here. The Hazard Unit would still see MemtoRegM and MemWriteE both on but without a way of forwarding RD from data memory, an out of date $10 value will be saved. Unless the Hazard Unit crashes and an unexpected value is saved.

Exercise 6

```
L1:   lw   $t1, ($a0)
      addiu $a0, $a0, 4
      slt  $t3, $zero, $t1
      sub  $t2, $a1, $a0
      bne  $t2, $zero, L1
      addu $t0, $t0, $t3
```

] this is weird...

Exercise D

address	tag	set	Action
0x0040_5650	0x00405	724	I-cache hit - no update to I-cache
0x0040_5654	0x00405	725	Icache miss - Instruction 0x0280_2021 will be copied into set 725's instruction field, V bit will change to 1, tag to 0x00405
0x0040_5658	0x00405	726	I-cache miss - Instruction 0x0c10-12d6 copied into set 726, tag changes to 0x00405, v-bit remains 1.
0x0040_4658	0x00404	726	I-cache miss, 0x8c88_0000 written to set 726, Tag changed to 0x00404
0x0040_465c	0x00404	727	I-cache hit - no update
0x0040_4660	0x00404	728	I-cache hit - no update
0x0040_4664	0x00404	729	I-cache hit - no update
0x0040_565c	0x00405	727	I-cache miss - Set 727 updates with instruction: 0x2694_0004, Tag changes to: 0x00405.
0x0040_5660	0x00405	728	Icache miss - set 728 updates with Instruction: 0x1695_888c and tag: 0x00405
0x0040_5654	0x00405	725	Icache hit - no update
0x0040_5658	0x00405	726	Icache miss - set 726 updates with instruction: 0x0c10_12d6; Tag: 0x00405
0x0040_4658	0x00404	726	Icache miss - set 726 updated with instruction: 0x8c88_0000; Tag: 0x00404
0x0040_465c	0x00404	727	Icache miss - Set 727 updated with Instruction: 0x2509_ffc0; Tag: 0x00404
0x0040_4660	0x00404	728	Icache miss - set 728 updates with Instruction: 0xac89_0000; Tag: 0x00404
0x0040_4664	0x00404	729	Icache hit - no update.
0x0040_565c	0x00405	727	Icache miss - set 727 updates with Instruction = 0x2694_0004; Tag = 0x00405
0x0040_5660	0x00405	728	Icache miss - set 728 updates Instruction = 0x1695_fffc; Tag = 0x00405
0x0040_5664	0x00405	729	I cache miss - set 729 updates Instruction = 0x0000_a021; Tag = 0x00405