Katya Thorup

Pipelined CPU with branch and jump controls

Figure: The pipeline diagram of the instruction execution (I do not know why my tracker would not go above cycle 9, when Professor Grodstein runs it it goes to cycle 12)

```
 PC:   instruction   | cycle.....
  0: beq R16,R17=>16|IF |ID |   |   |   |   |   |   |   |   |
  4: add R8=R8,R8   |   |IF |ID |EX |MEM|WB |   |   |   |   |
  8: beq R18,R19=>20|   |   |IF |ID |   |   |   |   |   |   |
 12: add R9=R9,R9   |   |   |   |IF |   |   |   |   |   |   |
--------------------
 20: add R11=R11,R11|   |   |   |   |IF |ID |EX |MEM|WB |   |
 24: J 32           |   |   |   |   |   |IF |ID |   |   |   |
 28: add R16=R16,R16|   |   |   |   |   |   |IF |   |   |   |
 32: add R17=R17,R17|   |   |   |   |   |   |   |IF |ID |   |
--------------------
 36: nop            |   |   |   |   |   |   |   |   |IF |   |
vm-hw06[kthorup01]71:
```

Instruction add R9=R9, R9 (add $t1, $t1, $t1) is squashed because saved registers $s2 and $s3 are equal

Instruction add R16=R16, R16 (add $s0 $s0 $s0) is squashed because of the jump.

Figure: Testing if instruction is squashed due to branch/jump

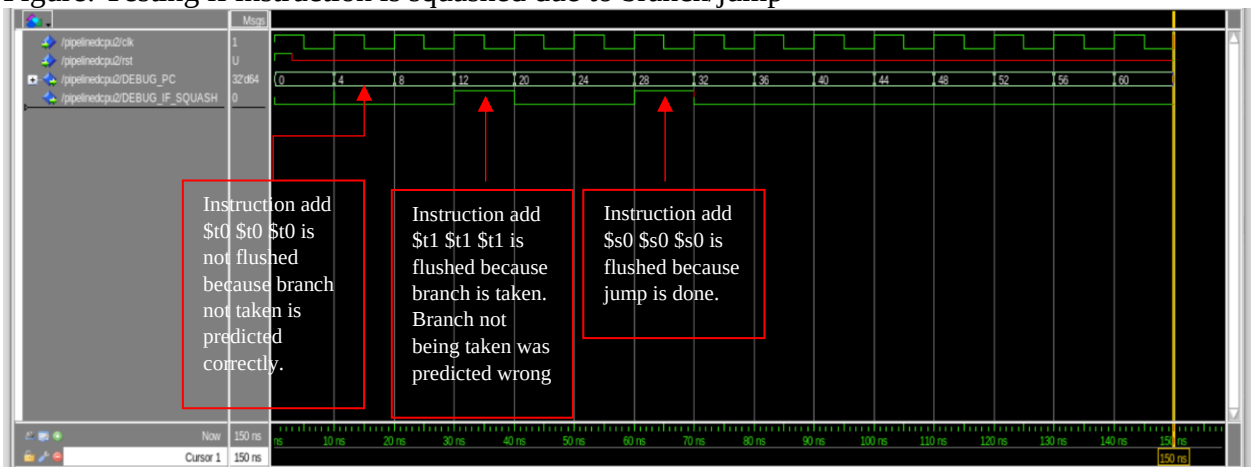
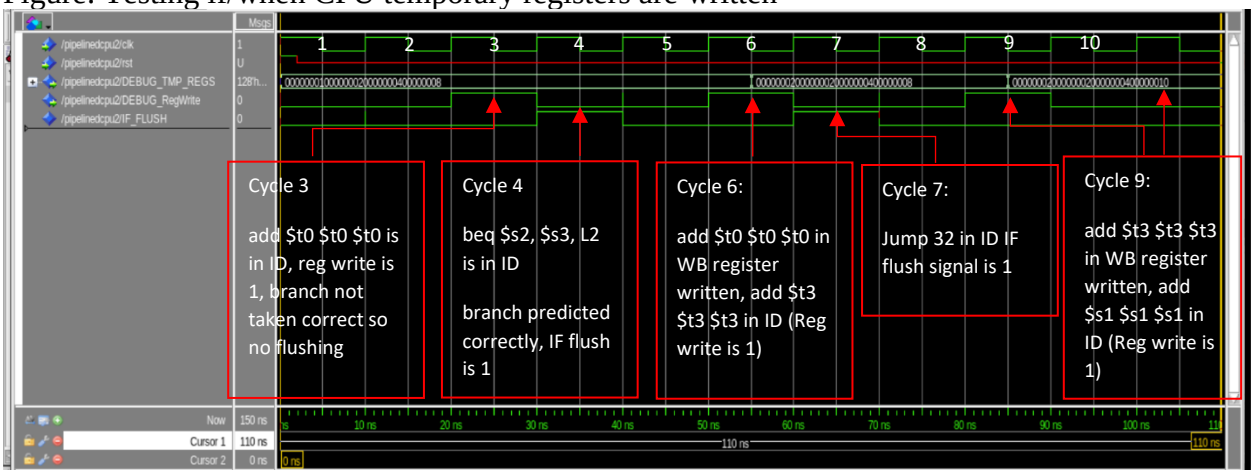


Figure: Testing if/when CPU temporary registers are written



In cycle 6, $t0 <= 0x00000002 (0x00000001 + 0x00000001)
In cycle 9, $t3 <= 0x00000010 (0x00000008 + 0x00000008)

Figure: Testing if/when CPU saved registers are written