CLASSNOTES LaTeX

Converter Bot

05/11/2017

Comp Architecture Notes

- Make sure you have the handout from yesterday
- 'X' value is the '?' value in gtkwave, means that we don't care what it is
- Cycles:
  - 1, filling
    - A PC is initialized (passed in), and does everything in the IF stage
    - Waiting at IF/ID
  - 2, decode
    - PC stuff happens as expected, normally, etc.
    - Control oval acts as MIPS decoder, reading from the 'real' register
    - Waiting at register ID/EX
  - 3, execute
    - Fill in the signals as you would expect. . .
    - (this is all pretty normal so far, except now it's spread over 5 steps)
    - Remember: the "barriers" are actually 'registers!' (in that they block on each clock cycle)ste
    - Wait at EX/MEM
  - 4, memory
    - Do stuff with memory (if you need to)
    - Waiting at MEM/WB
  - 5, write back
    - Basically just pass everything relevant back, write if need to
- We do the example provided on the handout from yesterday (10/23)

  - I'm going to need to scan this (eventually) since apparently I lost mine
  - Key idea: the stuff on the right side of each barrier is the values of those controls as it should be, since it was on the left side of the barrier last instruction!
- A "Hazard" of running through branch instructions
  - "We will spend the next 2 lectures on"
- "The state of your program will have dependencies"
  - Not everything can be massively parallel in either hardware OR software!
  - We'll draw dependency arrows from writebacks to accessing
    - Any that go backward in time are dangerous!
    - Any that go straight down or forward are fine
  - You can access the value as soon as it's available, not just after the writeback…
- Forwarding
  - One tactic to circumvent the hazards of branch instructions