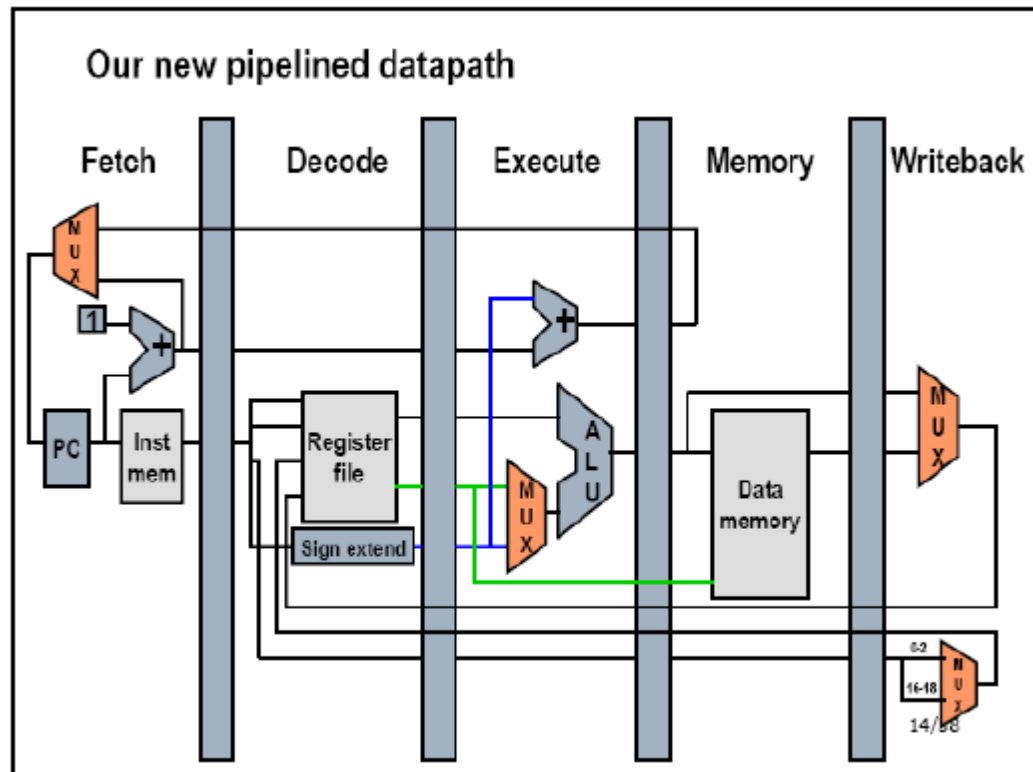


EECS 370

Discussion 7

Week of 3/13 – 3/19



- I) Control Hazards
 - a. Branch instructions may change PC
 - b. Need to decide what to fetch before we know result of branch
 - c. Target is sent to PC in MEM stage if equality test passes
- II) Methods
 - a. Avoidance
 - i. Don't use branches ☹
 - ii. Delay taking branch
 - iii. Problems: compatibility, useless instructions, CPI takes a hit
 - b. Detect and stall
 - i. Like data hazards, detect a beq instruction and insert noops
 - ii. Problems: CPI increases for every beq instruction
 - c. Speculate and squash
 - i. Assume either taken or not taken
 - 1. Wide array of techniques for predicting the result of branches!
 - ii. If we're wrong, replace bad instructions with noops

III) Examples

Consider a normal 5-stage LC-2K7 pipeline as discussed in class. Branches are resolved in the MEM stage and branches are predicted as not taken. 25% of load instructions incur a one cycle stall due to data hazards that can not be resolved via forwarding. Assume the instruction cache has a perfect hit rate, but the data cache has a 15% miss rate for both reads and writes. For a memory read, a data cache miss causes a 20 cycle stall. For a memory write, a data cache miss does not cause a stall. Analysis has shown the following dynamic instruction breakdown for program X: 20% not taken branches, 5% taken branches, 10% loads, 15% stores, 50% R-type instructions

- a) What is the CPI of this machine when running program X?
- b) Now assume the pipeline was stretched out to 7 stages (numbered 1 to 7, with instruction fetch beginning in stage 1, and instructions finally completing in stage 7). Branches are resolved in stage 6. The clock frequency is increased to twice that of the original processor with the 5 stage pipeline. Data cache misses cause a 40 cycle stall. 25% of load instructions still incur a one cycle stall due to data hazards that can not be resolved via forwarding. What is the CPI for this new pipeline design?
- c) Which of these machines performs better, the original 5 stage pipeline or the new 7 stage pipeline? How many times faster is it than the other one?

Consider three versions of a 5-state pipeline **IF/ID/EX/MEM/WB** with full data forwarding that stalls on branches. Version A resolves branches in the MEM stage and has clock period T . Version B resolves branches in EX and has clock period $1.5T$. Version C resolves branches in ID and has clock period $1.75T$. Which version yields the shortest runtime for a program with 25% branch instructions and no load hazards?