Announcements

Homework 3 returned
Project Status report in a couple of weeks!
• Initial set of numbers ready by then
Page Fault

1 2 3 4 5 6 7 8 9
i   IF ID EX MEM WB
i+1  IF ID EX MEM WB  ← page fault
i+2  IF ID EX MEM WB  ← abort
i+3  IF ID EX MEM WB
i+4  IF ID EX MEM WB
i+5  trap  ←→ IF ID EX MEM WB
i+6  trap handler  ←→ IF ID EX MEM WB

Arithmetic Exception

1 2 3 4 5 6 7 8 9
i   IF ID EX MEM WB
i+1  IF ID EX MEM WB  ← exception
i+2  IF ID EX MEM WB  ← abort
i+3  IF ID EX MEM WB
i+4  trap  ←→ IF ID EX MEM WB
i+5  trap handler  ←→ IF ID EX MEM WB
i+6  IF ID EX MEM WB
Instruction Fetch Page Fault

1 2 3 4 5 6 7 8 9
i  IF  ID  EX  MEM WB
i+1 IF  ID  EX  MEM WB
i+2 IF  ID  EX  MEM WB  ← page fault
i+3 IF  ID  EX  MEM WB
i+4 trap  → IF  ID  EX  MEM WB
i+5 trap handler  → IF  ID  EX  MEM WB
i+6 IF  ID  EX  MEM WB

Interrupt Order

Let preceding instructions complete
Abort succeeding instructions
What if multiple instructions fault?
- ith instruction gets a data page fault
- i+1th instruction gets an instruction page fault
In-Order Interrupt Handling

Post interrupts
- check interrupt bit on entering WB
- precise interrupts
- longer latency

Handle immediately
- restartable but not fully precise
- interrupt may occur in order different from sequential
- may cause implementation headaches!

Interrupt Complications

Odd bits of state
- e.g., implicit condition codes
- e.g., branch delay slot PC

Early writes (e.g., auto-increment)
Instruction buffers and prefetch logic
Dynamic scheduling and out-of-order execution (later)
Interrupts come at random times
- frequent cases not everything
- must get the rare cases (e.g., interrupts) correct
What Breaks Our Simple Pipeline?

So far
- Simple in-order issue, in-order completion pipeline

Multicycle operations
- not all operations complete in 1 cycle
- out-of-order completion?

Long-latency operations delay the pipeline
- can we find, decode, and execute inst with ready operands?
- out-of-order issue? (later)

Multicycle Operations

How long do operations take?
- Integer operations typically 1 cycle
- 2-4 cycles for FP multiply and add
- 20-50 cycles for FP divide

Extend DLX pipeline
- EX stage repeated multiple times
- multiple parallel functional units
  - can pipeline long-latency ops
  - for now assume non-pipelined!
Handling Multicycle Operations

Four functional units
- EX: integer
- E*: FP/integer multiply
- E+: FP add
- E/: FP/integer divide

Assume
- EX takes 1 cycle and all FP take 4 cycles
- separate integer and FP registers
- all FP arithmetic in FP registers

Worry about hazards
- RAW, WAR, and WAW between integer and FP

Assume in-order issue

Simple Multicycle Example

1 2 3 4 5 6 7 8 9
i  IF  ID  EX  MEM  WB
i+1  IF  ID  E*  E*  E*  E*  MEM  WB
i+2  IF  ID  EX  MEM  WB
i+3  IF  ID  E/  E/  E/  MEM  WB
i+4  IF  ID  EX  MEM  WB
i+5  IF  ID  --  --  E/  E/
i+6  IF  --  --  ID  EX
Simple Multicycle Example

Notes
- I+2 no WAW but complicates interrupts
- I+4 no WB conflict
- I+5 stall forced by structural hazard
- I+6 stall forced by in-order issue

But different FP operation times are possible
- makes FP WAW hazards possible
- further complicates interrupts

FP Instruction Issue

Check for structural hazards
- wait until functional unit is free

Check for RAW (stall)
- source regs are not used as destinations by instructions in EX

Check for forwarding
- bypass data from EX, MEM, or WB if needed

What about overlapping instructions?
- Contention in WB
- possible WAR/WAW hazards and interrupts headaches
Overlapping Instructions

Contestion in WB
- static priority
- e.g., FU with longest latency
- instructions stall after issue

WAR hazards
- always read sregisters at same pipe stage

WAW hazards
- divf f4, f2, f0 followed by subf f10, f8, f0
- stall subf or baot divf's WB

Multicycle (Cont.)

Problems with interrupts
- divf f4, f2, f0
- addf f10, f8, f2
- subf f10, f4, f6

addf completes before divf
- out-of-order completion

what if divf excepts after addf/subf complete?