LECTURE - 08

Announcements

- Please submit scribe notes within a week
- Homework to be handed out by tomorrow
  - Due in a week

Today's Topic

- Pipelining complications
  - Exceptions
  - Multi-cycle operations

Exceptions and Pipelining

- What are exceptions?
  - I/O interrupt
  - System call
  - Tracing instruction execution, breakpoint
  - Integer/FP anomaly
  - Page fault
  - Misaligned memory access
  - Memory protection violation
  - Undefined instruction
  - Hardware malfunction/Power failure
- Also called interrupts or faults

Exceptions: The Nemesis of Pipelining

- While taking exceptions, ensure that machine is in a "consistent" state
- Exceptions can occur:
  - In many pipeline stages
  - Out of order

	CC1	CC2	CC3	CC4	CC5	CC6
LW	IF	ID	EX	MEM	WB	
ADD		IF	ID	EX	MEM	WB

Classification of Exceptions

- **Synchronous vs. Asynchronous**
  - Asynchronous usually caused by devices external to the processor
  - Asynchronous ==> can be handled after current instruction (easier)
- **User requested vs. Coerced**
  - User requested ==> can be handled after current instruction
  - Coerced ==> unpredictable
- **User maskable vs. Non-maskable**

Classification of Exceptions (continued)

- **Within vs. Between instructions**
  - Within ==> instruction cannot be completed, usually synchronous (harder)
- **Resume vs. Terminate**
  - Terminate process ==> easier

Exception Classification

Exception type	Synchronous?	Coerced?	Maskable?	Within instn.?	Resume?
I/O request	No	Yes	No	No	Yes
Sys. call	Yes	No	No	No	Yes
Tracing/Brk.pt.	Yes	No	Yes	No	Yes
ALU excpn.	Yes	Yes	Yes	Yes	Yes
Page fault	Yes	Yes	No	Yes	Yes
Misaligned. mem. access	Yes	Yes	Yes	Yes	Yes
Protecn. violn.	Yes	Yes	No	Yes	Yes
Undefined instns.	Yes	Yes	No	Yes	No
H/w malfn./ power failure	No	Yes	No	Yes	No

Restarting Execution

- **Restartable:** take exception, save state, restart without affecting execution
- Restarting
  - Force a *trap* instruction into pipeline
  - Until trap, disable all writes for faulting instruction and all subsequent ones
  - Trap into exception handling routine (OS)
  - Need to save more than one PC for delayed branches
- **Precise Exceptions:** all instructions prior to faulting one completed, but not any other

Exceptions in DLX

	CC1	CC2	CC3	CC4	CC5	CC6
LW	IF	ID	EX	MEM	WB	
ADD		IF	ID	EX	MEM	WB

- Exceptions can occur:
  - In same cycle, or even out-of-order
- Cannot handle an exception when it occurs in time
  - Carry an instruction status in the pipeline latches
  - In WB stage, exception corresponding to earliest instruction will be handled

More Complications in Pipelining

- Multiple write stages
- Or, changing processor state in the middle of an instruction
  - E.g., Auto-increment addressing mode in VAX
- Updating memory state during instruction
  - E.g., String copy instruction in VAX

More Complications in Pipelining (continued)

- Implicitly set condition codes
  - Problems in scheduling the delay slot, and during exceptions
- Self-modifying code in 80x86!
- Multi-cycle operations

```
MOVL     R1, R2
ADDL3    42(R1), 56(R1)+, @(R1)
SUBL2    R2, R3
MOVC3    @(R1)(R2), 74(R2), R3
```

Data hazards very complicated to determine!

VAX pipelines micro-instructions