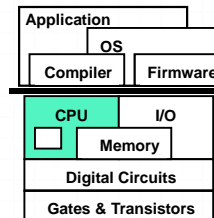


ECE 152
Introduction to Computer Architecture
Pipelining
Copyright 2012 Daniel J. Sorin
Duke University

Slides are derived from work by
Amir Roth (U. Penn)
Spring 2012

1

This Unit: Pipelining



- Basic Pipelining
 - Pipeline control
- Data Hazards
 - Software interlocks and scheduling
 - Hardware interlocks and stalling
 - Bypassing
- Control Hazards
 - Fast and delayed branches
 - Branch prediction
- Multi-cycle operations
- Exceptions

© 2012 Daniel J. Sorin from Roth

ECE 152

2

Readings

- P+H
 - Chapter 4: Section 4.5-end of Chapter 4

© 2012 Daniel J. Sorin from Roth

ECE 152

3

Pipelining

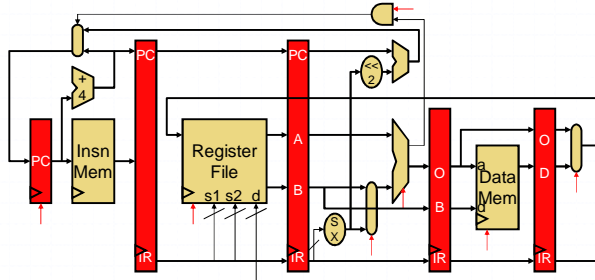
- Important performance technique
 - **Improves insn throughput (rather than insn latency)**
- Laundry / SubWay analogy
- Basic idea: divide instruction's "work" into stages
 - When insn advances from stage 1 to 2
 - Allow next insn to enter stage 1
 - Etc.
- Key idea: each instruction does same amount of work as before
 - + **But insns enter and leave at a much faster rate**

© 2012 Daniel J. Sorin from Roth

ECE 152

4

5 Stage Pipelined Datapath



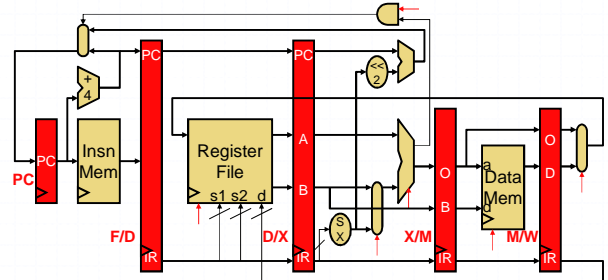
- Temporary values (PC,IR,A,B,O,D) re-latched every stage
 - Why? 5 insns may be in pipeline at once, they share a single PC?
 - Notice, PC not re-latched after ALU stage (why not?)

© 2012 Daniel J. Sorin from Roth

ECE 152

5

Pipeline Terminology



- Five stage: **F**etch, **D**ecode, **eX**ecute, **M**emory, **W**riteback
 - Latches (pipeline registers) named by stages they separate
 - **PC, F/D, D/X, X/M, M/W**

© 2012 Daniel J. Sorin from Roth

ECE 152

6

Aside: Not All Pipelines Have 5 Stages

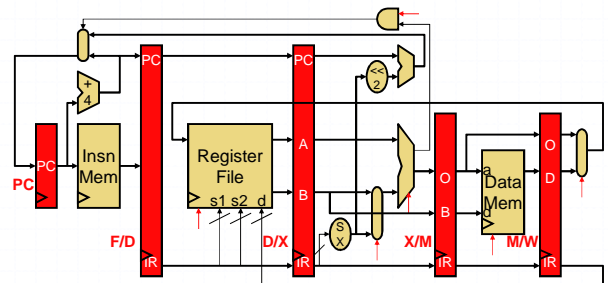
- H&P textbook uses well-known 5-stage pipe != all pipes have 5 stages
- Some examples
 - OpenRISC 1200: 4 stages
 - Sun UltraSPARC T1/T2 (Niagara/Niagara2): 6/8 stages
 - AMD Athlon: 10 stages
 - Pentium 4: 20 stages (later 32 stages!)
- **ICQ**: why does Pentium 4 have so many stages?
- **ICQ**: how can you possibly break "work" to do single insn into that many stages?
- Moral of the story: in ECE 152, we focus on H&P 5-stage pipe, but don't forget that this is just one example

© 2012 Daniel J. Sorin from Roth

ECE 152

7

Pipeline Example: Cycle 1



add \$3,\$2,\$1

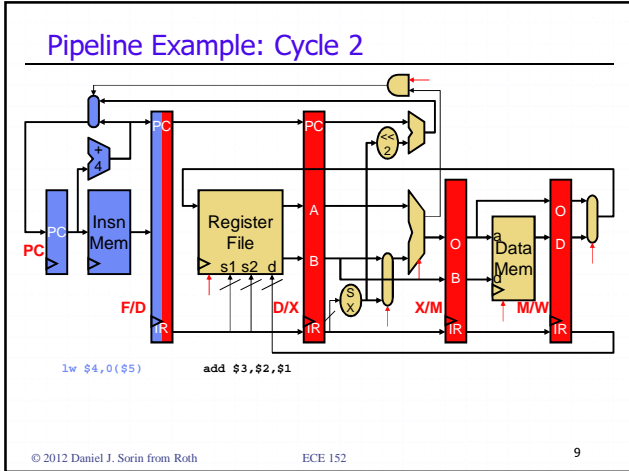
- 3 instructions

© 2012 Daniel J. Sorin from Roth

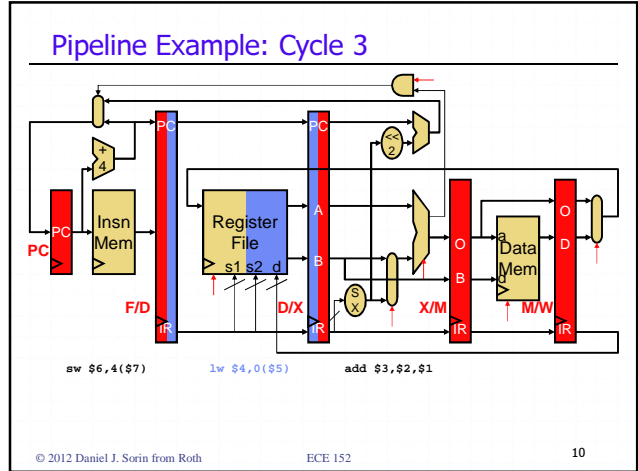
ECE 152

8

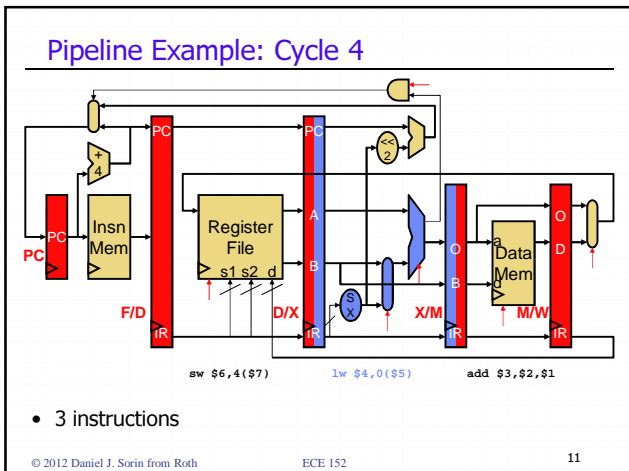
Pipeline Example: Cycle 2



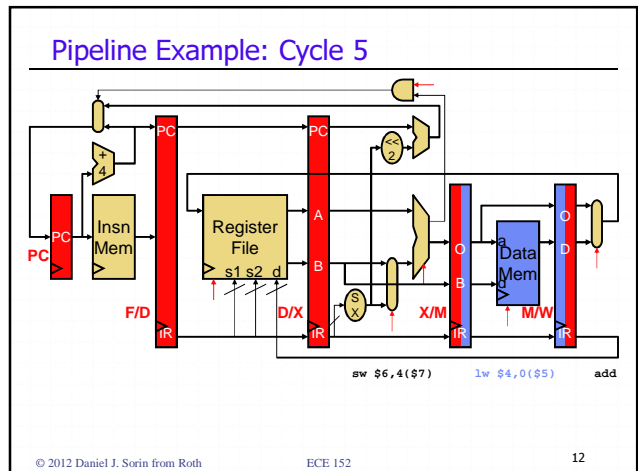
Pipeline Example: Cycle 3



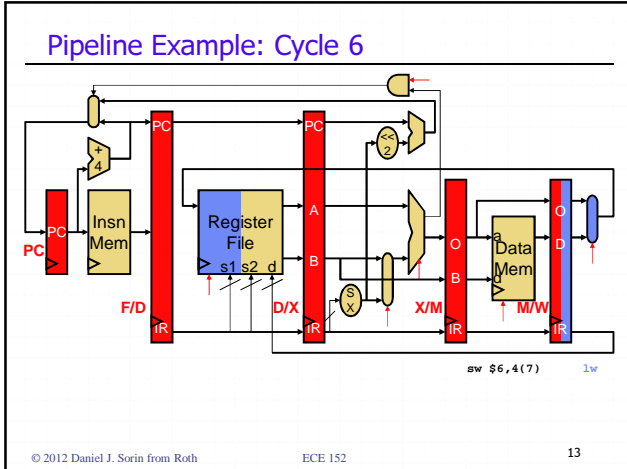
Pipeline Example: Cycle 4



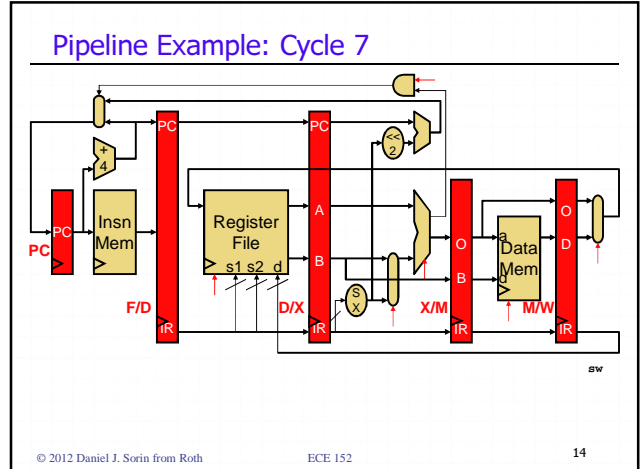
Pipeline Example: Cycle 5



Pipeline Example: Cycle 6



Pipeline Example: Cycle 7



Pipeline Diagram

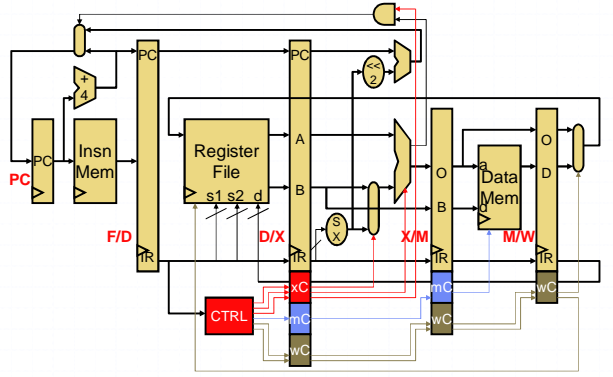
- **Pipeline diagram:** shorthand for what we just saw
 - Across: cycles
 - Down: insns
 - Convention: **X** means `lw $4, 0($5)` finishes execute stage and writes into X/M latch at end of cycle 4

	1	2	3	4	5	6	7	8	9
<code>add \$3, \$2, \$1</code>	F	D	X	M	W				
<code>lw \$4, 0(\$5)</code>		F	D	X	M	W			
<code>sw \$6, 4(\$7)</code>			F	D	X	M	W		

What About Pipelined Control?

- Should it be like single-cycle control?
 - But individual insn signals must be staged
- How many different control units do we need?
 - One for each insn in pipeline?
- Solution: use simple single-cycle control, but pipeline it
 - Single controller
 - **Key idea:** pass control signals with instruction through pipeline

Pipelined Control



© 2012 Daniel J. Sorin from Roth

ECE 152

17

Pipeline Performance Calculation

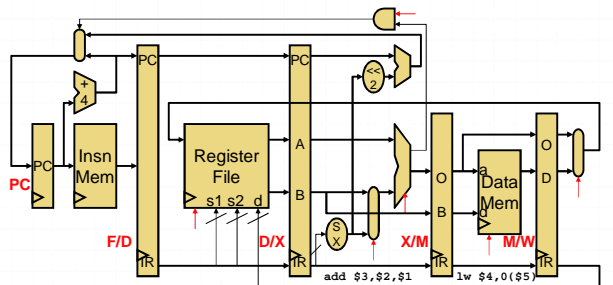
- Single-cycle
 - Clock period = 50ns, CPI = 1
 - Performance = 50ns/insn
- Pipelined
 - Clock period = **12ns (why not 10ns?)**
 - CPI = **1** (each insn takes 5 cycles, but 1 completes each cycle)
 - Performance = **12ns/insn**

© 2012 Daniel J. Sorin from Roth

ECE 152

18

Why Does Every Insn Take 5 Cycles?



- Why not let `add` skip M and go straight to W?
 - It wouldn't help: peak fetch still only 1 insn per cycle
 - **Structural hazards**: not enough resources per stage for 2 insns

© 2012 Daniel J. Sorin from Roth

ECE 152

19