Multithreading

6.911 Architectures Anonymous
The Problem

• Lots of silicon
  – higher clock rates
  – same old speed of light
  – bandwidth, latency problem

• Extracting ILP is hard
  – Hazards and stalls
  – dependencies, structural limitations
  – static optimization limits
One Solution

• OOE superscalar
  – dynamically re-order instructions to fill multiple execution units
  – must preserve sequential semantics
    • dependency checking req’d
  – complexity grows as product of instructions in flight and number of execution units
  – recent work by Sun, IBM, Compaq indicates that a superscalar width of about 4 is the current cost vs. Performance point
What Went Wrong?

• Preserving sequential semantics while reordering instructions is hard--esp. in hardware

• Limits to reordering
  – branches
  – loads and stores
Enter Multithreading

- Observation: many tasks are divisible into multiple threads
  - but, requires a different coding style
- These threads are independent
  - except for the dependencies you put in
- Executing multiple threads allows you to fill wide execution units without the hardware dependency checking!
- Additional benefit: latency hiding
Cost of Multithreading

• Hardware cost
  – a copy of PC, register file
  – cache pollution issues

• Software cost
  – need to write for a multithreaded paradigm
  – synchronization issues pushed up into the user domain
Papers

• Two TERA papers
  – massively multithreaded processor

• *T (“start”) architecture
  – massively parallel processor design, intro to dataflow

• MPR Report on Alpha EV8
  – SMT (simultaneous multithreading)