

Portland State University

ECE 587/687

# **Simultaneous Multithreading (SMT)**

# Motivation

- ILP limitations of superscalar processors
  - ◆ Many control, data and functional dependences
- Wide superscalar pipelines cannot use all issue slots
  - ◆ Vertical Waste: All issue slots in a cycle are not used
  - ◆ Horizontal waste: Some issue slots in a cycle are not used
  - ◆ Paper Figure 1
- To increase throughput, we need to use thread-level parallelism (TLP)

# Multithreaded Programs

- Thread vs. process
  - ◆ Threads in a process share virtual address space
  - ◆ Processes have different virtual address spaces
- Design Issues:
  - ◆ Each thread needs its own set of registers (register address space is not shared)
  - ◆ Threads cause interference in instruction and data caches
  - ◆ Synchronization is necessary, may cause some threads to be idle (OS idle loop)

# Multithreading Alternatives

- Fine-grain multithreading
  - ◆ During each cycle, a single thread is allowed to issue instructions
  - ◆ Removes vertical waste
  - ◆ Still limited by ILP available within each thread
- Simultaneous Multithreading
  - ◆ During each cycle, any thread can issue instructions (instructions from different threads can be issued at the same time)
  - ◆ Addresses both horizontal and vertical waste

# Superscalar Processors: Where Have Cycles Gone?

- Discuss Paper Figure 2
  - ◆ Issue slots are utilized only 19% of the time
  - ◆ Lots of causes for issue stall cycles
  - ◆ Need aggressive latency-hiding techniques
- Multiple causes for stalls can be addressed using latency-hiding techniques
  - ◆ Paper Table 3

# Simultaneous Multithreading Models

- SM: Full Simultaneous Issue
  - ◆ Completely flexible model: All threads compete for each of the issue slots every cycle
  - ◆ Disadvantage: Hardware complexity
- SM: Single Issue
  - ◆ Each thread can issue at most one instruction every cycle
- SM: Dual Issue and SM: Four Issue
  - ◆ Each thread can issue at most two (Dual Issue) or four (Four issue) instructions every cycle

# Simultaneous Multithreading Models (Cont.)

- SM: Limited Connection
  - ◆ Each thread is connected to exactly one of each type of functional unit
  - ◆ Limits scheduling choices for functional units to reduce hardware complexity
- Hardware Complexity: Paper Table 4

# SMT Performance

- Paper Figure 3
- Fine-grain MT can only increase throughput by a factor of 2.1
- SMT has much higher speedup
- Alternatives to execute 4 instructions per cycle
  - ◆ Four issue or full SMT with 3-4 threads
  - ◆ Dual issue SMT with 4 threads
  - ◆ Limited Connection SMT with 5 threads
  - ◆ Single issue SMT with 6 threads

# SMT Performance Side Effects

- Lowest priority thread runs much slower than high priority thread
- Highest priority thread sees degraded performance as more threads are added
  - ◆ Sharing of resources (e.g., caches, TLB, BP tables)
- Caches are more strained by an MT workload vs. ST workload due to a decrease in locality
  - ◆ Different cache configurations explored in Paper Figure 4

# SMT vs. Multiprocessors

- Paper Figure 5
- SMT outperforms multiprocessing for all scenarios compared
- Advantages of SMT vs. MP
  - ◆ Area efficiency
  - ◆ Reducing number of threads (i.e., threads becoming idle) allows other threads to progress faster in SMT processors, no change in MP
  - ◆ Granularity and flexibility of design: Unit of design is a whole processor for MP, more flexible in SMT
- Disadvantages? (discuss)

# SMT Design Issues

- Hardware complexity
  - ◆ Scheduling hardware requirements increase with threads
  - ◆ Register file size increase
  - ◆ May need more ports
- Pipeline depth
  - ◆ Bigger structures (e.g., register file) require longer access time
  - ◆ Leads to increasing the number of pipeline stages
- Issue policy
  - ◆ Fixed thread priority
  - ◆ Round-Robin priority
  - ◆ ICOUNT
  - ◆ Others?

# Reading Assignment

- Project Progress Report Due on Monday
- No reviews due next week
- Papers for Monday
  - ◆ Sohi et al., “Multiscalar Processors,” ISCA 1996 (Read)
  - ◆ Roth & Sohi, “Speculative Data-Driven Multithreading,” HPCA 2001 (Skim)