

Multithreading Models

- Coarse-grain – switch contexts (typically several cycles) on long-latency event.
 - MIT Alewife
- Fine-grain – switch contexts every cycle.
 - HEP, Tera, Sun Niagara
- Simultaneous Multithreading
 - Compaq 21464, Intel Pentium 4, Power 5, 6

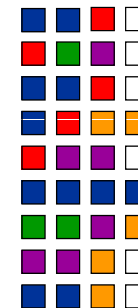
Coarse-grain multithreading



Fine-grain multithreading



Simultaneous Multithreading





MIT Alewife



Scalability

- Who are the arch enemies of scalability?



Scalability – Four Pronged Approach



Programming Model?

Coherent Shared Memory

- Unique features?

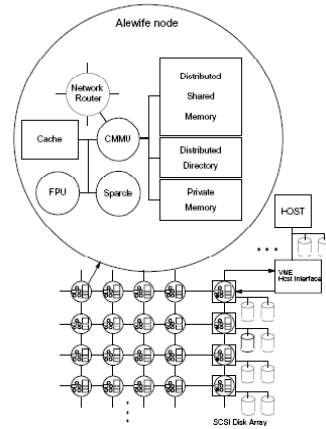


Figure 1: The Alewife architecture.

LimitLESS Directory

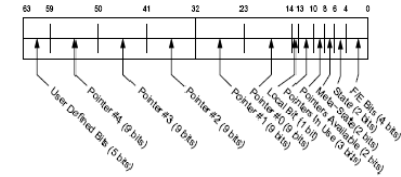


Figure 4: A hardware directory entry in Alewife.

- So why scalable?

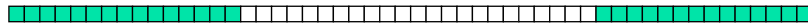
Full-Empty Bits

- Store, set full
- Load if full, set empty

Latency Tolerance

Block Multithreading

- Aka coarse-grain multithreading

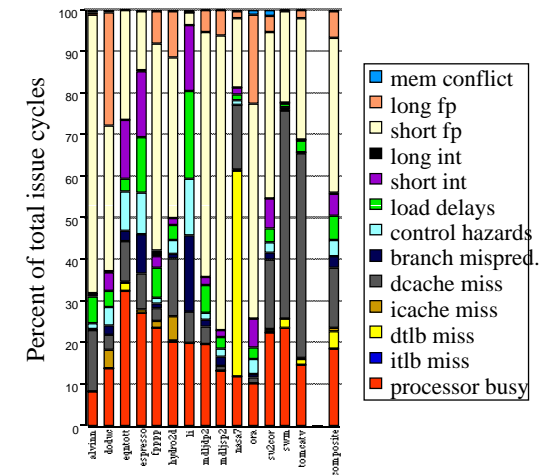


Block (CG) Multithreading

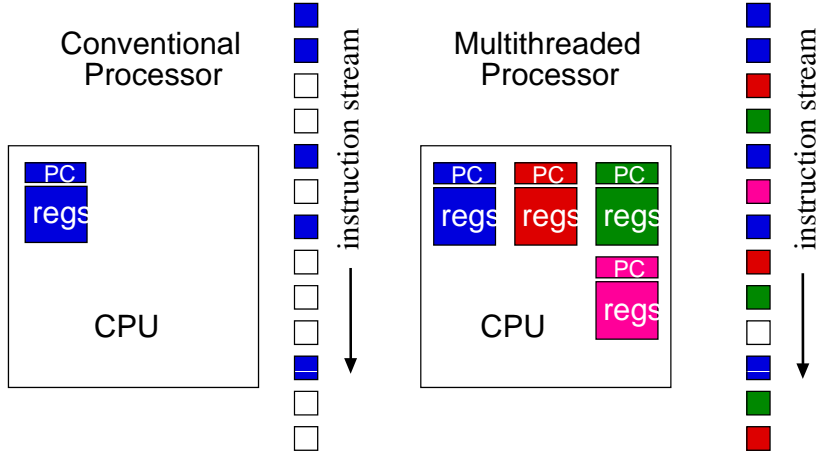
- What does it do well?
- What doesn't it do well?

Simultaneous Multithreading

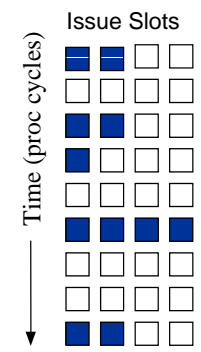
Motivation



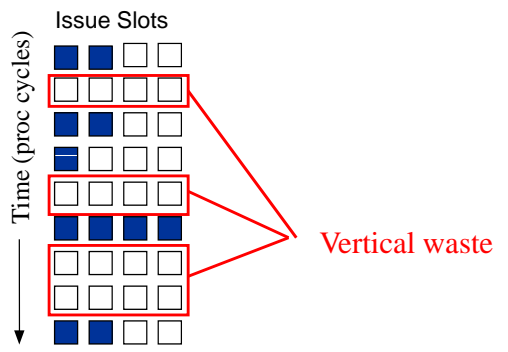
Hardware Multithreading



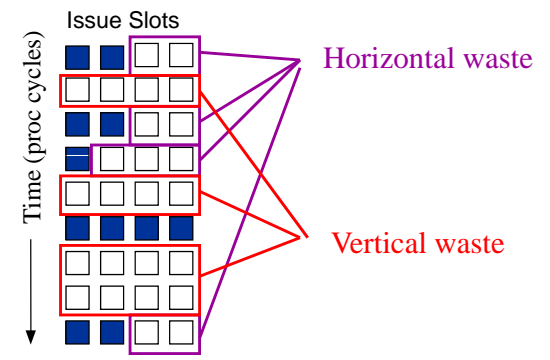
Superscalar Execution



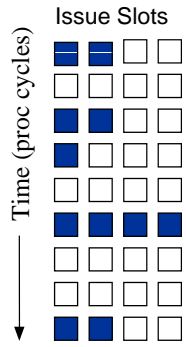
Superscalar Execution



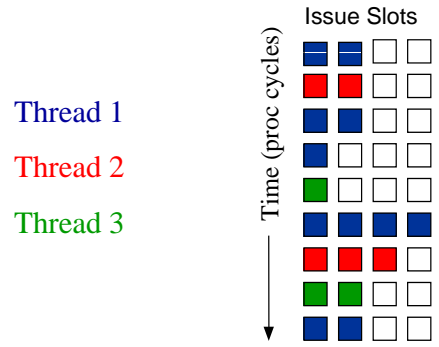
Superscalar Execution



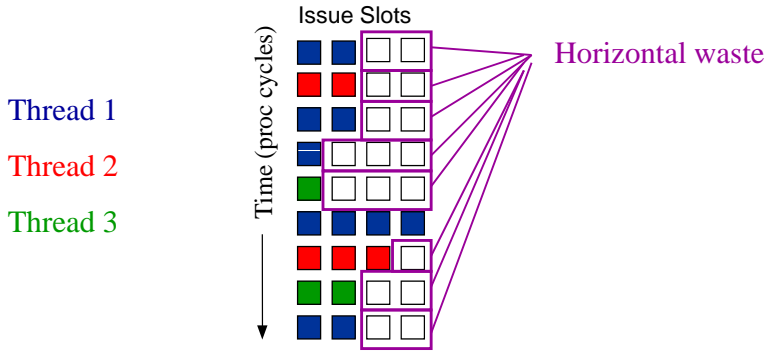
Superscalar Execution



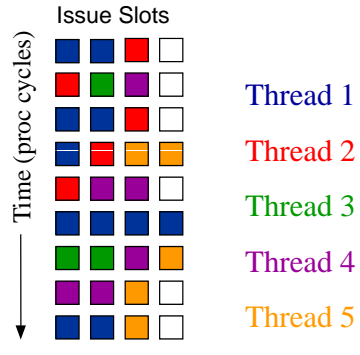
Superscalar Execution with Fine-Grain Multithreading



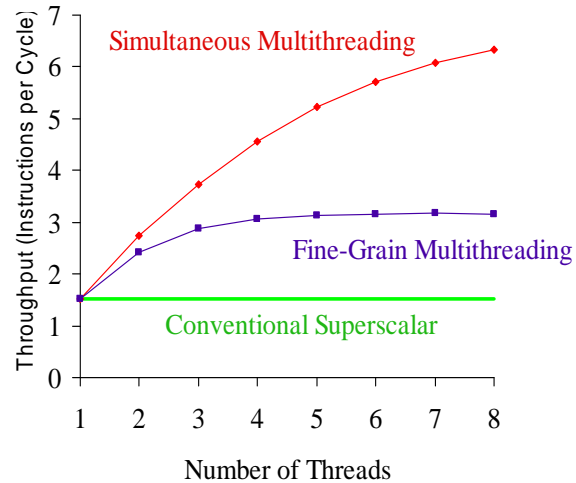
Superscalar Execution with Fine-Grain Multithreading



Simultaneous Multithreading



The Potential for SMT

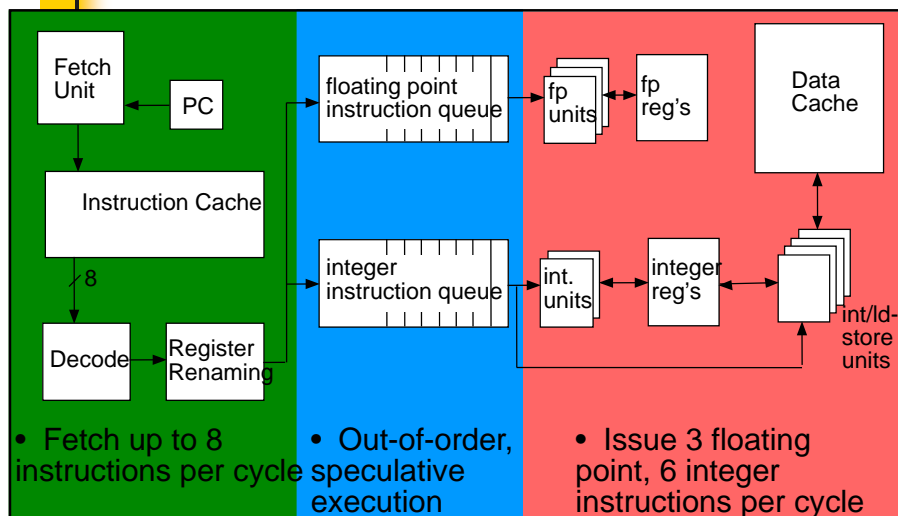


Goals

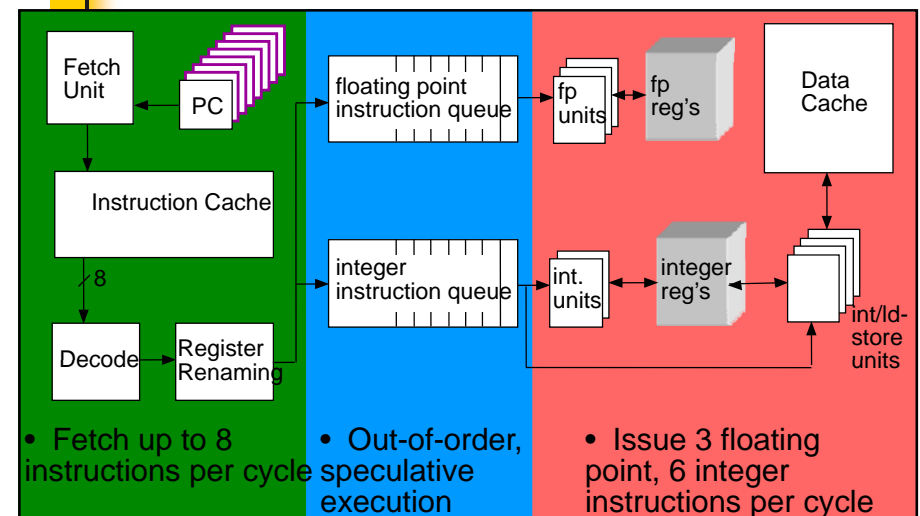
We had three primary goals for this architecture:

1. Minimize the architectural impact on conventional superscalar design.
2. Minimize the performance impact on a single thread.
3. Achieve significant throughput gains with many threads.

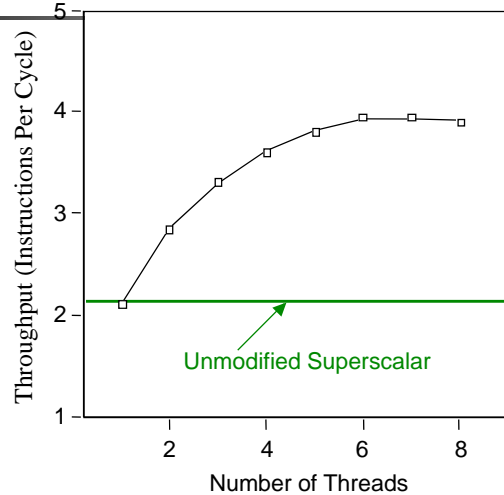
A Conventional Superscalar Architecture



An SMT Architecture



Performance of the Naïve Design



Bottlenecks of the Baseline Architecture

- Instruction queue full conditions (12-21% of cycles)
 - Lack of parallelism in the queue.
- Fetch throughput (4.2 instructions per cycle when queue not full)

Improving Fetch Throughput

- The fetch unit in an SMT architecture has two distinct advantages over a conventional architecture.
 -
 -

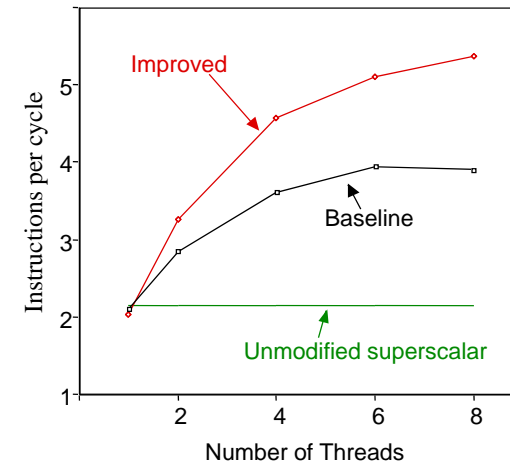
Improving Fetch Throughput

- The fetch unit in an SMT architecture has two distinct advantages over a conventional architecture.
 - Can fetch from **multiple threads** at once.
 - Can **choose which threads** to fetch.

Improved Fetch Performance

- Fetching from 2 threads/cycle achieved most of the performance from multiple-thread fetch.
- Fetching from the thread(s) which have the fewest unissued instructions in-flight significantly increases parallelism and throughput (ICOUNT fetch policy)

Improved Performance



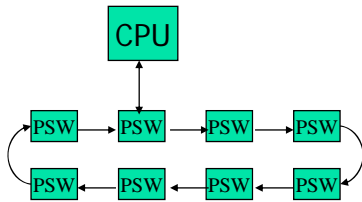
This SMT Architecture, then:

- Borrows heavily from conventional superscalar design.
- Minimizes the impact on single-thread performance
- Achieves significant throughput gains over the superscalar (2.5X, up to 5.4 IPC).

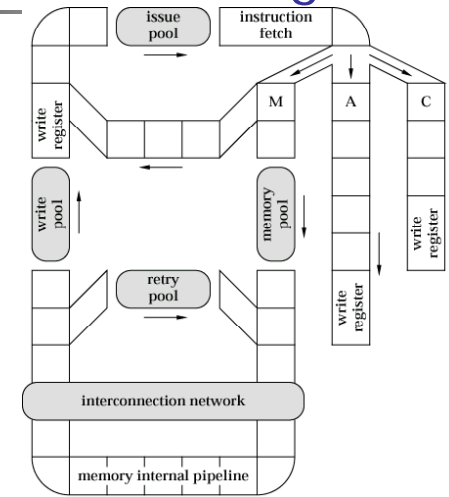
Multithreading Models

- Coarse-grain – switch contexts (typically several cycles) on long-latency event.
 - MIT Alewife
- Fine-grain – switch contexts every cycle.
 - HEP, Tera, Sun Niagara
- Simultaneous Multithreading
 - Compaq 21464, Intel Pentium 4, Power 5, 6
- Pros, cons, issues, ... ?

HEP Multithreading



Tera Multithreading



Cool Tera Features

- Full/empty bits on memory
- Randomized memory (why??)
- No bypassing
- Explicit-dependence lookahead
- LIW
- No caches
- High-bandwidth network

- Which of these are related to multithreading?

Other SMT Research (UCSD emphasis)

- Compilation issues, including instruction layout for icache performance.
- Multiple-path execution
- Symbiotic job scheduling
- Synchronization and parallelization
- Multithreading and power
- Helper threads – speculative precomputation
- Multicore/multithreaded, clustered multithreaded
- Multithreaded value prediction
- Event-driven compilation



Other interesting multithreaded research

- Speculative multithreading
- Fault tolerance
- slipstreaming



Summary

- Why simultaneous multithreading?
- Long-term solution?
- When won't it work?
- What next?