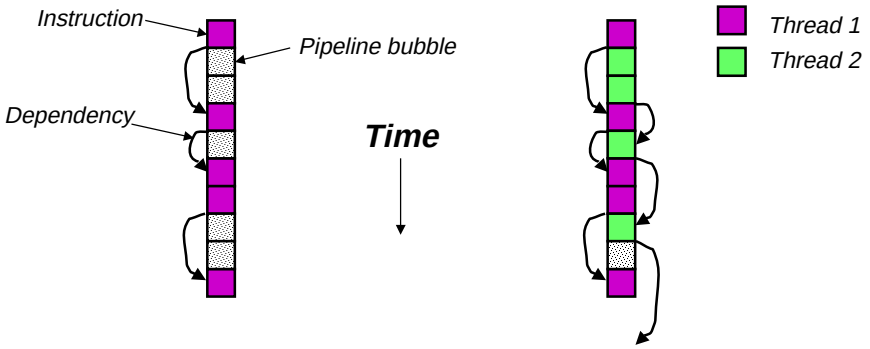
Multithreading and the Tera MTA

Krste Asanovic

krste@lcs.mit.edu

http://www.cag.lcs.mit.edu/6.893-f2000/

Multithreading for Latency Tolerance



- **Problem: Dependencies limit sustainable throughput of single instruction stream**
- **Solution: Interleave execution of two or more instruction streams on same hardware to increase utilization**

Multithreading History

- CDC 6600 Peripheral Processors (Cray, shipped in 1965)
  - 10 “virtual” I/O processors
  - fixed interleave on simple pipeline
  - pipeline has 100ns cycle time
  - each processor executes one instruction every 1000ns
  - accumulator-based instruction set to reduce processor state
- Denelcor HEP (Burton Smith, shipped in 1982)
  - 120 threads per processor
  - 10 MHz clock rate
  - Up to 8 processors
  - precursor to Tera

Tera MTA Overview

- Up to 256 processors
  - 8 in current prototype
- Up to 128 active threads per processor
- Processors and memory modules populate a sparse 3D torus interconnection fabric
- Flat, shared main memory
  - No data cache
  - Sustains one main memory access per cycle per processor
- GaAs circuitry in prototype, 1KW/processor @ 260MHz
  - CMOS prototype in development, 50W/processor

MTA Thread (Stream) State

- 32 64-bit general-purpose registers (R0-R31)
  - unified integer/floating-point register set
  - R0 hard-wired to zero
- 8 64-bit branch target registers (T0-T7)
  - load branch target address before branch instruction
  - T0 contains address of user exception handler
- 1 64-bit stream status word (SSW)
  - includes 32-bit program counter
  - four condition code registers
  - floating-point rounding mode

MTA Instruction Format

- Three operations packed into 64-bit instruction word
- One memory operation, one arithmetic operation, plus one arithmetic or branch operation
- "Skip" instructions provide short forward branches without using branch target registers
- Memory operations incur ~150 cycles of latency
- Explicit 3-bit "lookahead" field in instruction gives number of subsequent instructions (0-7) that are independent of this one
  - c.f. Instruction grouping in VLIW
  - allows fewer threads to fill machine pipeline
  - used for variable-sized branch delay slots
- Thread creation and termination instructions

MTA Multithreading

- Each processor supports 128 active hardware threads
  - 128 SSWs, 1024 target registers, 4096 general-purpose registers
- Every cycle, one instruction from one active thread is launched into pipeline
- Instruction pipeline is 21 cycles long
- At best, a single thread can issue one instruction every 21 cycles
  - Clock rate is 260MHz, effective single thread issue rate is 260/21 = 12.4MHz!
- If lookahead is 0 (following instruction needs this memory value), then around 150 cycles of latency

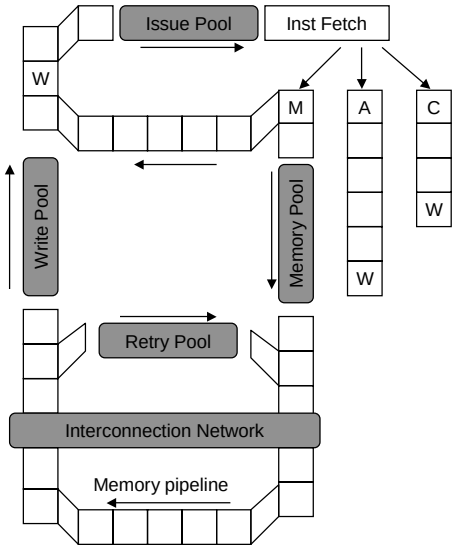
Memory Tags

Each 64-bit word in memory has four extra tag bits (also has separate SECDEC bits)

- full-empty bit
  - Used for light-weight synchronization between threads
  - various flavors of read/write action on full-empty bit provided
- two data trap bits
  - available to software
  - user-level trap handler pointed to by T0 branch target register
- forwarding bit (indirection bit)
  - if set, use data value as pointer for fetch
  - allows processor-invisible data forwarding

Also provides fetch-and-add synch primitive executed at memory module

MTA Pipeline



Exceptions

- **Many exceptions handled at user-level in same thread, avoids large OS overhead for 100s of threads trapping**
- **Branch register T0 holds address of user-level exception handler**
- **Exceptions are precise to the lookahead value, i.e., all exceptions will be taken before marked dependent instruction is executed**

MTA Operating System Support

- Single address space shared by all active processes
- Four hardware privilege levels: IPL, KERNEL, SUPER, USER
- Each processor has 16 protection domains which delimit segments accessible by each job
  - Each thread belongs to one domain
- One protection domain reserved for OS, other 15 used by user programs
- Two types of process scheduler
  - single-processor scheduler for small interactive jobs that run on one processor (allocated most domains)
  - multi-processor scheduler for large batch jobs that run across multiple processors (allocated a few domains)
- No interrupts, instead dedicated thread polls for events

Performance Studies

- Only two working installations currently:
  - San Diego
  - Tera
- First Paper: Snavely et al, NAS parallel kernels
- Second Paper: Brunett et al, C3I parallel benchmarks

Coarse-Grain Multithreaded Machines

- MIT Alewife
  - four threads per node
  - used modified SPARC chips (used register windows for different thread contexts)
  - thread switch on local cache miss
- Commercially available in IBM Power chips used in AS400 series
  - two active threads
  - context switch on L2 cache miss
  - context switch takes 4 cycles (drains pipeline)

Simultaneous Multithreading

- Add multiple contexts and fetch engines to wide out-of-order superscalar processor
  - [Tullsen, Eggers, Levy, UW]
- OOO instruction window already has most of the circuitry required to schedule from multiple threads
- Any single thread can utilize whole machine
- Alpha 21264 will support 4-way simultaneous multithreading in an 8-issue OOO core