Multithreading Architectures

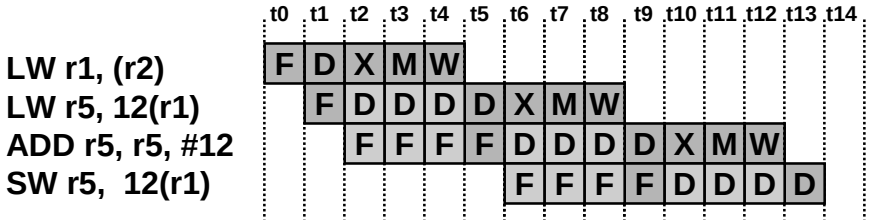
Krste Asanovic
Laboratory for Computer Science
M.I.T.

http://www.csg.lcs.mit.edu/6.823

Pipeline Hazards

```
LW r1, (r2)
LW r5, 12(r1)
ADD r5, r5, #12
SW r5,  12(r1)
```

- Each instruction may depend on the next
- Without bypassing, need interlocks

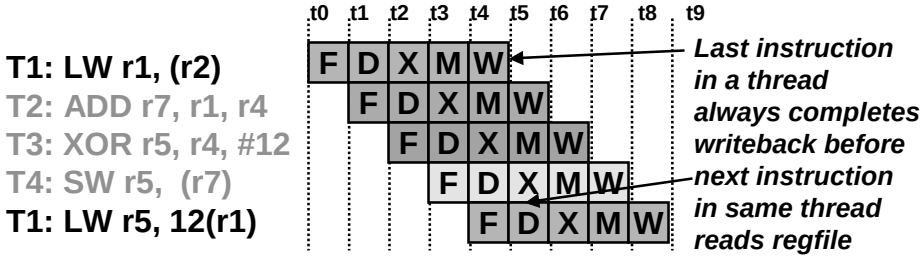


- Even with bypassing, still get some interlocks or delay slots

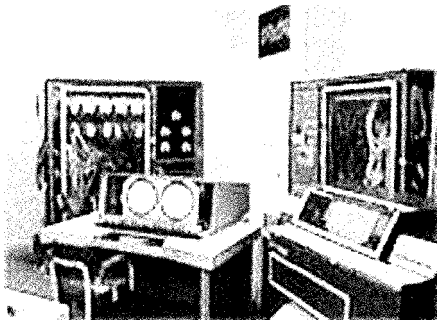
Multithreading

- **Problem is that instructions within a single sequential program thread depend on each other**
- **One solution is to interleave execution of instructions from different program threads on same pipeline, guaranteeing no dependencies between instructions in pipeline**

Interleave four threads, T1-T4, on non-bypassed 5-stage pipe



CDC 6600 Peripheral Processors
(Cray, 1965)

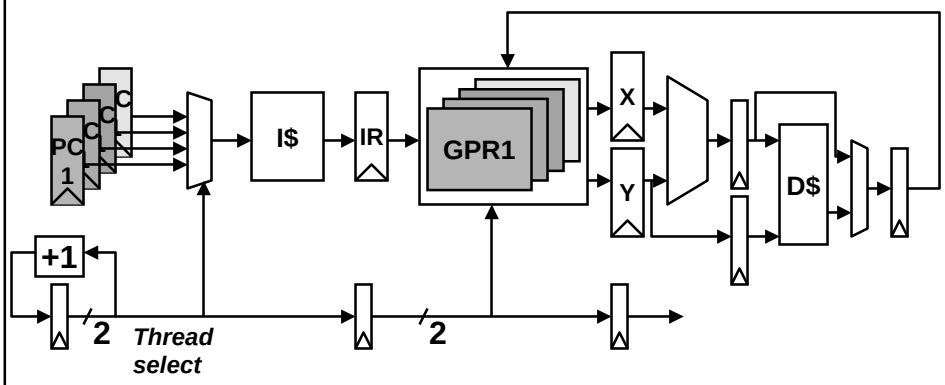


- **First multithreaded hardware**
- **10 “virtual” I/O processors**
- **fixed interleave on simple pipeline**
- **pipeline has 100ns cycle time**
- **each processor executes one instruction every 1000ns**
- **accumulator-based instruction set to reduce processor state**

Multithreading Costs

- Appears to software (including OS) as multiple slower CPUs
- Each thread requires its own user state
  - GPRs
  - PC
- Also, needs own OS control state
  - virtual memory page table base register
  - exception handling registers

Simple Multithreaded Pipeline



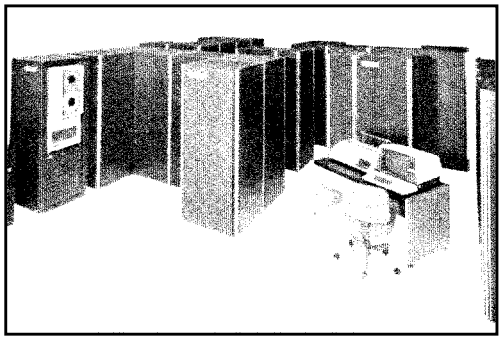
- Have to carry thread select down pipeline to ensure correct state bits read/written at each pipe stage

Thread Scheduling Policies

- **Fixed interleave** ***(CDC 6600 PPUs, 1965)***
  - **each of N threads executes one instruction every N cycles**
  - **if thread not ready to go in its slot, insert pipeline bubble**
- **Software-controlled interleave** ***(TI ASC PPUs, 1971)***
  - **OS allocates S pipeline slots amongst N threads**
  - **hardware performs fixed interleave over S slots, executing whichever thread is in that slot**
- **Hardware-controlled thread scheduling** ***(HEP, 1982)***
  - **hardware keeps track of which threads are ready to go**
  - **picks next thread to execute based on hardware priority scheme**

Denelcor HEP

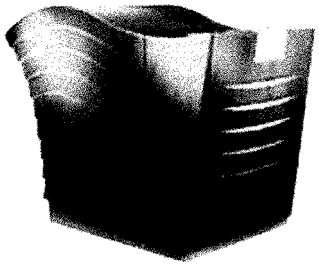
(Burton Smith, 1982)



First commercial machine to use hardware threading in main CPU

- **120 threads per processor**
- **10 MHz clock rate**
- **Up to 8 processors**
- **precursor to Tera MTA (Multithreaded Architecture)**

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 (short VLIW)
- 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

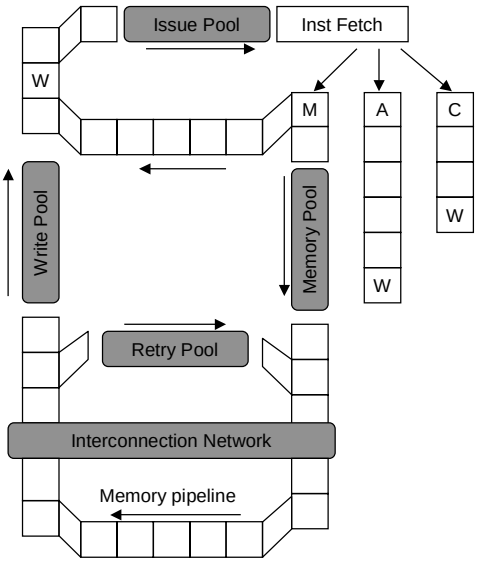
MTA Memory Tags

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



MTA 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

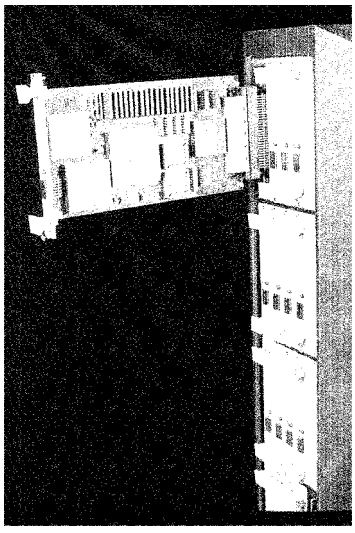
MTA Operating System Support

- 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

Coarse-Grain Multithreading

- Tera MTA designed for supercomputing applications with large data sets and low locality
  - No data cache
  - Many parallel threads needed to hide large memory latency
- Other applications are more cache friendly
  - Few pipeline bubbles when cache getting hits
  - Just add a few threads to hide occasional cache miss latencies
  - Swap threads on cache misses

MIT Alewife

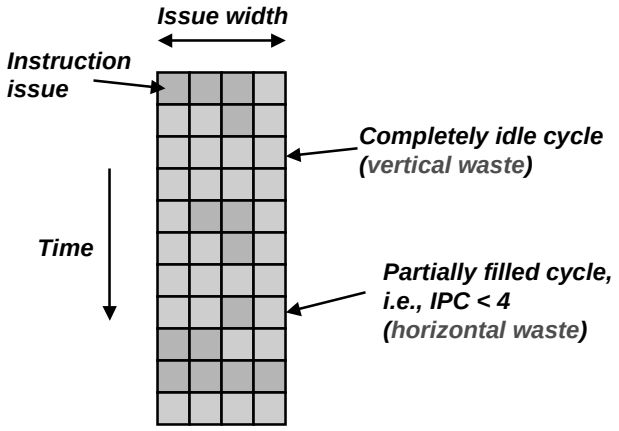


- Modified SPARC chips
  - register windows hold different thread contexts
- Up to four threads per node
- Thread switch on local cache miss

IBM PowerPC RS64-III (Pulsar)

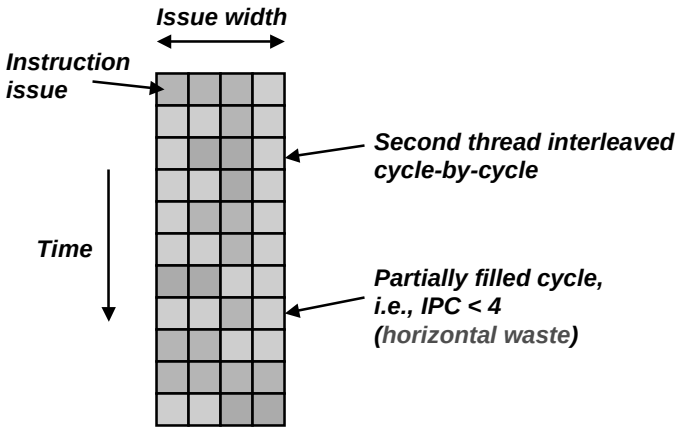
- Commercial coarse-grain multithreading CPU
- Based on PowerPC with quad-issue in-order five-stage pipeline
- Each physical CPU supports two virtual CPUs
- On L2 cache miss, pipeline is flushed and execution switches to second thread
  - short pipeline minimizes flush penalty (4 cycles), small compared to memory access latency
  - flush pipeline to simplify exception handling

Multithreading Superscalar Machines



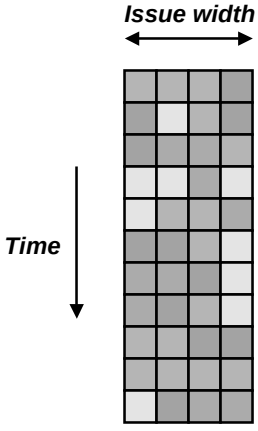
- Difficult to keep a superscalar processor busy, IPC 1-1.5 typical for 4-8 issue machine

Vertical Multithreading



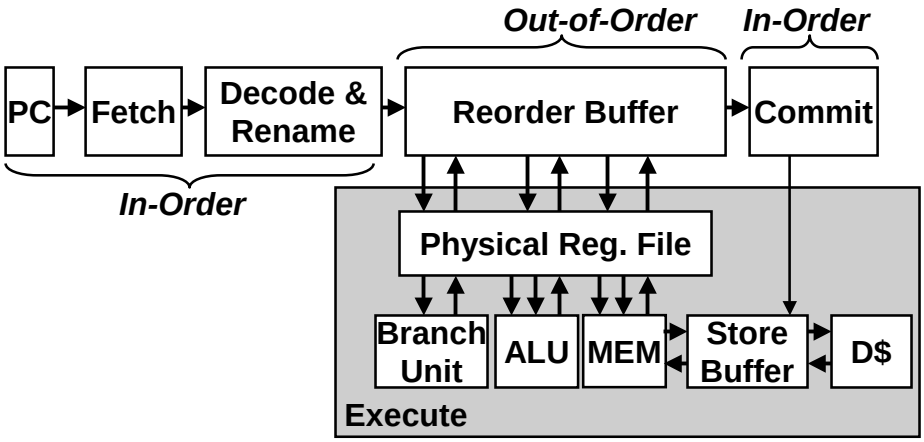
- **Cycle-by-cycle interleaving of second thread removes vertical waste**

Ideal Multithreading for Superscalar



- **Interleave multiple threads to multiple issue slots with no restrictions**

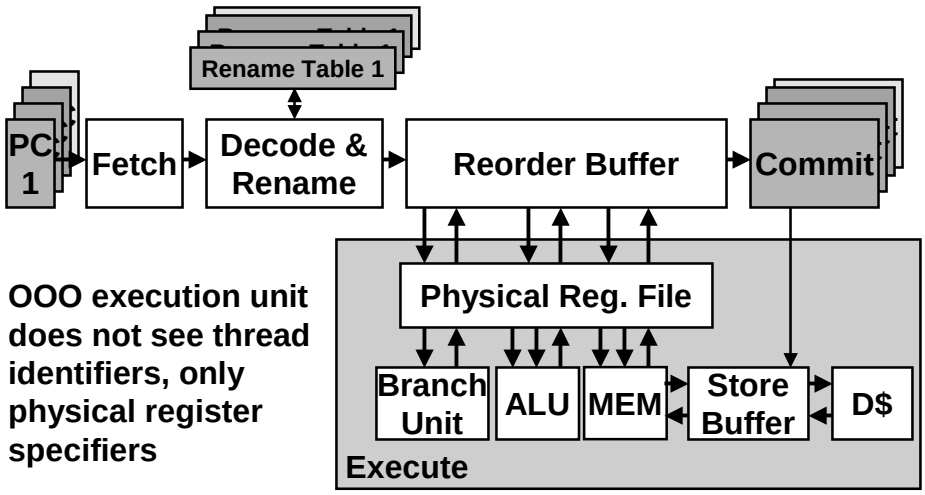
Speculative and Out-of-Order Superscalar Processor



Simultaneous Multithreading

- **Add multiple contexts and fetch engines to wide out-of-order superscalar processor**
  - **[Tullsen, Eggers, Levy, UW, 1995]**
- **OOO instruction window already has most of the circuitry required to schedule from multiple threads**
- **Any single thread can utilize whole machine**

Simultaneous Multithreaded Processor



OOO execution unit does not see thread identifiers, only physical register specifiers

SMT Design Issues

- **Which thread to fetch from next?**
  - **don't want to clog instruction window with thread with many stalls → try to fetch from thread that has fewest insts in window**
- **Locks**
  - **virtual CPU spinning on lock executes many instructions but gets nowhere → add ISA support to lower priority of thread spinning on lock**

Upcoming SMT Processor Designs

- **Compaq Alpha 21464**
  - **Announced plans to introduce SMT processor with four threads sharing an 8-issue out-of-order processor**
  - **Claim only 10-15% die area overhead on top of 8-issue core**
- **Intel Pentium-4 SMT follow on**
  - **Rumors of second logical CPU being added to Pentium-4 core**