Intel Technology Journal Q1, 2002

Hyper-Threading Technology Architecture and Microarchitecture

Motivation

- The growth of the Internet and telecommunications in general
  - Thread Level Parallelism
- Increasing number of transistors and power…
- … faster than performance is
- Looking for a way for performance to ‘keep up’
- Hyper-Threading is one solution

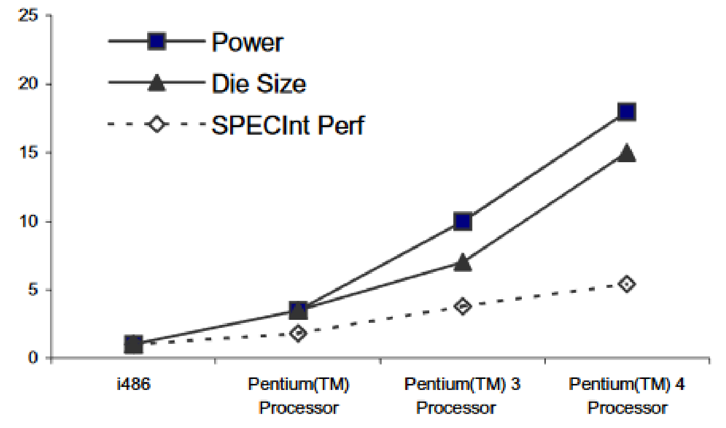


Figure 1: Single-stream performance vs. cost

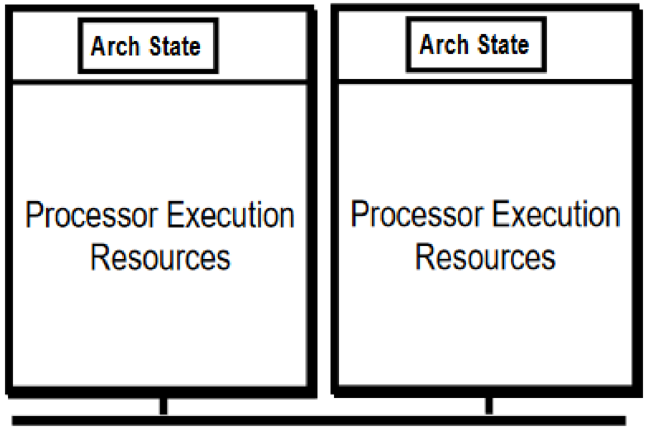
Existing Solutions

- Time-Slicing Multithreading
  - Processor swaps threads after fixed amount of time
- Switch-on-event Multithreading
  - Switches processes on events which have a lot of latency, e.g. cache miss
- Neither of these achieve optimal overlap of resource usage
- Instruction Level Parallelism
  - Need to find instructions to parallelise
- Chip Multiprocessing was still a very new idea

Hyper-Threading: Idea

- Two ‘logical’ processors per physical processor
- Higher levels can address these as if they were separate processors
- But they will share (most) execution resources

Figure 2: Processors without Hyper-Threading Tech



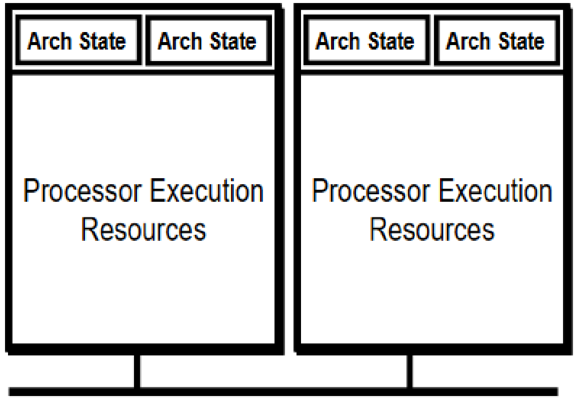


Figure 3: Processors with Hyper-Threading Technology

Hyper-Threading: Goals

1. Minimise die-size cost
   - Resource sharing
2. When one logical processor is stalled, the other continues
   - Limited or partitioned buffers and queues
3. When only one thread is running, performance should be the same as on a processor without HT
   - Re-combining shared resources

Goal 1: Resource Sharing

- Logical processors do not share: interrupt controllers, Instruction TLBs, Instruction Pointers, and Register Alias Tables
  - All of which are small structures
- Logical processors do share: cache, execution units, branch predictors, control logic, and busses

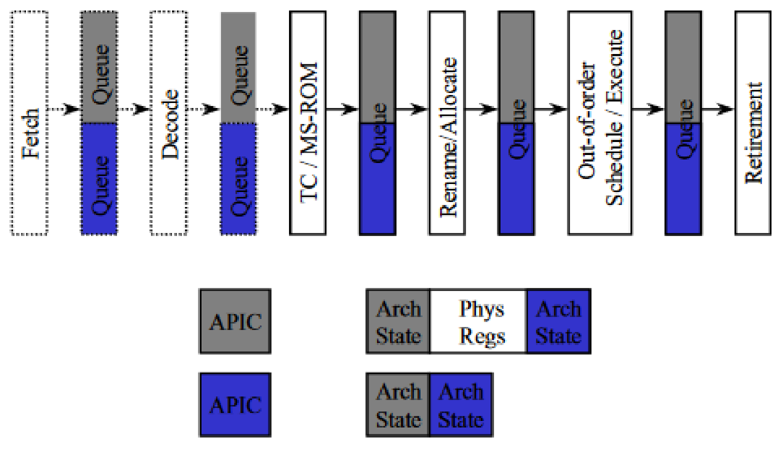


Figure 4 Intel® Xeon™ processor pipeline

Goal 2: Independent execution

- By splitting queues and cache buffers, and alternating priority, fairness is ensured
- In case one logical processor is stalling, simply stop alternating the priority
  - But keep the partitioning of the resources

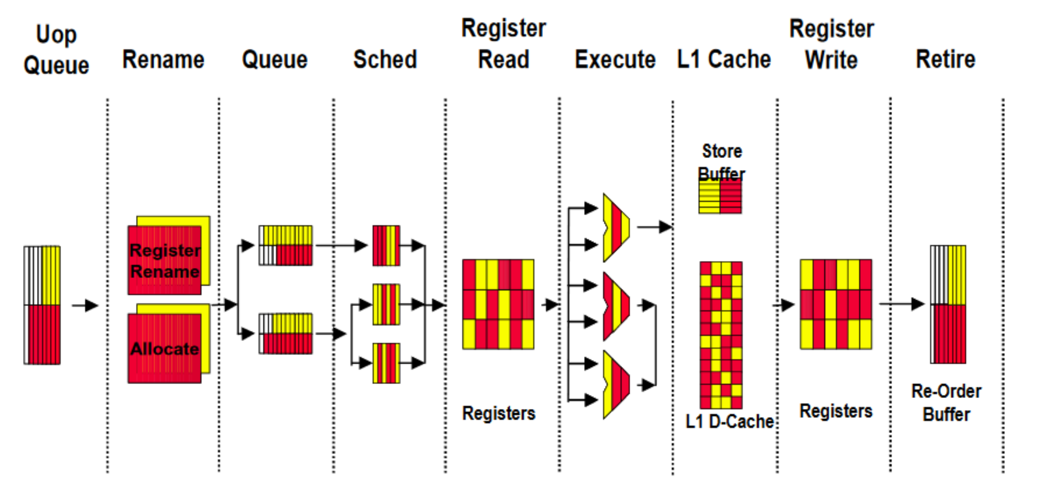


Figure 6: Out-of-order execution engine detailed pipeline

Goal 3: When there is only one thread

- The HALT instruction
- Puts processor in ‘power-saving’ mode
- Only the OS and similar can execute this
- When it is executed on a multithreaded processor, put one of the logical processors to sleep and combine the resources
- OS is responsible for managing transition

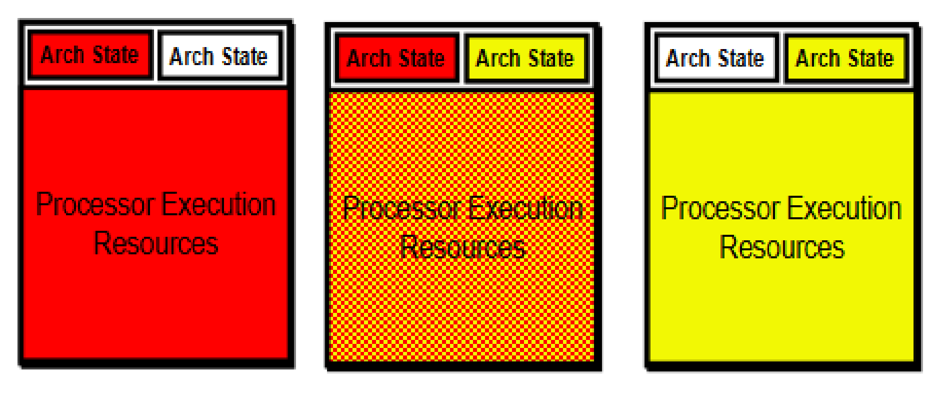


Figure 7: Resource allocation

Performance

- Die size: 5% increase
- Around 21% performance gain compared to non Hyper-Threaded processor systems
- 16-28% performance gain in server-side applications

- This was a new technology. And one that became very successful.

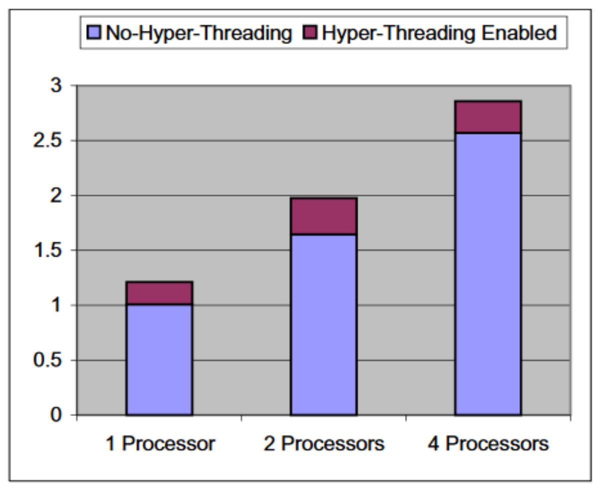


Figure 8: Performance increases from Hyper-Threading Technology on an OLTP workload

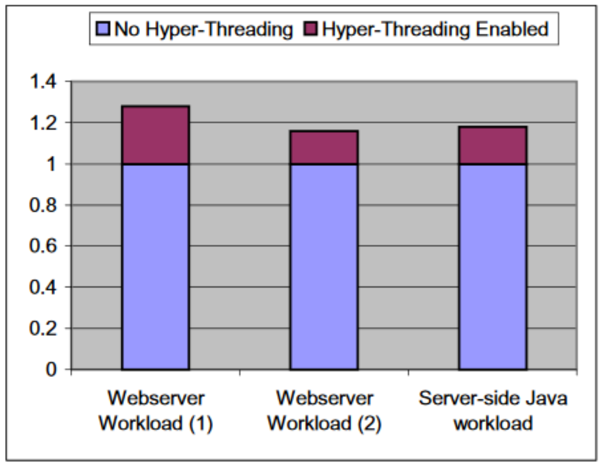


Figure 9: Web server benchmark performance

Credits

- Slides by 150015673
- Content based on https://www.cs.virginia.edu/~mc2zk/cs451/vol6iss1_art01.pdf
- All figures are taken from https://www.cs.virginia.edu/~mc2zk/cs451/vol6iss1_art01.pdf

Questions?