Microprocessor Evolution: Past, Present, and Future

Guri Sohi
University of Wisconsin

Outline

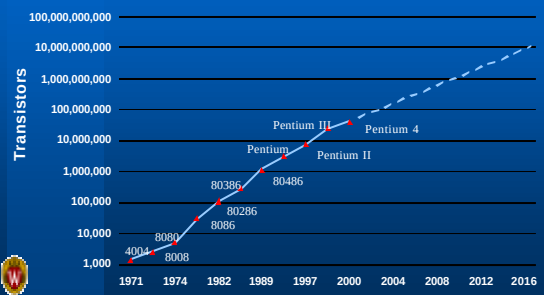
- The enabler: semiconductor technology
- Role of the processor architect
- Micro-architectures of the past 20 years
  - From pipelining to speculation
- Micro-architectures of the next 10 years

Semiconductor Technology

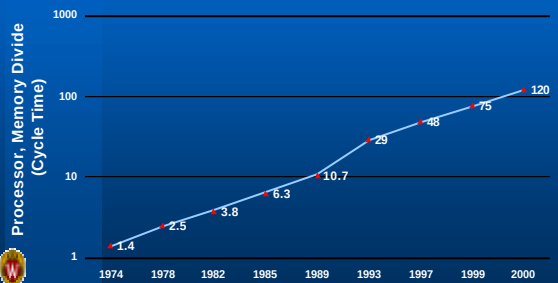
- Many more available transistors
- Imbalances due to disparate rates of performance improvement
  - E.g., logic and memory speeds

How does this impact the architecture of microprocessors?

Number of Transistors



Relative Memory Speed



Intel Microprocessors

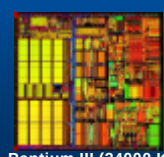

386 (275 K)


486 (1180 K)


Pentium (3100 K)


Pentium II (7500 K)


Pentium 4 (42000 K)


Pentium III (24000 K)

What is being done with all the transistors?

Role of Computer Architect

- Get desired level of performance
  - Single-thread "desktop" applications
- Determine functionality needed
- Determine how functionality should be implemented

Role of Computer Architect…

- Defining functionality
  - Functionality to deal with increasing latencies (e.g., caches, wires)
  - Functionality to increase parallelism and its exploitation
- Implementing functionality
  - Balancing various technology parameters
  - Ease of design / verification / testing

The Performance Equation

Time = Number of Instructions x Cycles per Instruction x Clock Cycle Time

- Not much can be done a…
  - But, …
- Logic speed increase - decreases 3rd term
  - Watch out for possible increase in 2nd term
- Use micro-architectural innovations to decrease 2nd and 3rd terms
  - Reduce latencies
  - Exploit parallelism

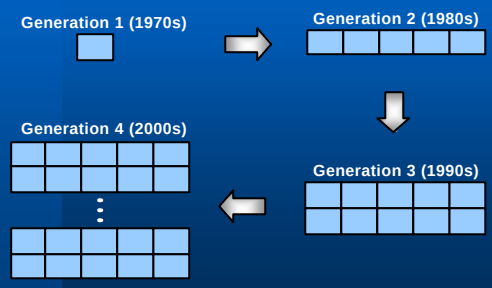
Microarchitectural Functionality

- Functionality to cope with increasing memory latencies
- Functionality to exploit parallelism

Memory Hierarchies

- Reducing access latency and improving access bandwidth
- Single-level caches
- Multi-level caches
- Non-blocking caches
- Multi-ported and multi-banked caches
- Trace caches

The March of Parallelism



Exploiting Parallelism

- Little change in programming model
  - still write programs in sequential languages
- Automatic parallelization not widely successful
- Great investment in existing software

Resort to low-level, Instruction Level Parallelism (ILP)

Instruction Level Parallelism (ILP)

- Determine small number (e.g., < 100) instructions to be executed
- Determine dependence relationships and create dependence graph
  - Use to determine parallel execution
- Can be done statically (VLIW / EPIC) or dynamically (out-of-order superscalar)

Limitations to ILP

- Branch instructions inhibit determination of instructions to execute: control dependences
- Imperfect analysis of memory addresses inhibits reordering of memory operations: ambiguous memory dependences
- Program/algorithm data flow inhibits parallelism: true dependences
- Increasing latencies exacerbate impact of

Use speculation to overcome impact of dependences

Speculation

Speculation: ".. to assume a business risk in hope of gain"

Webster

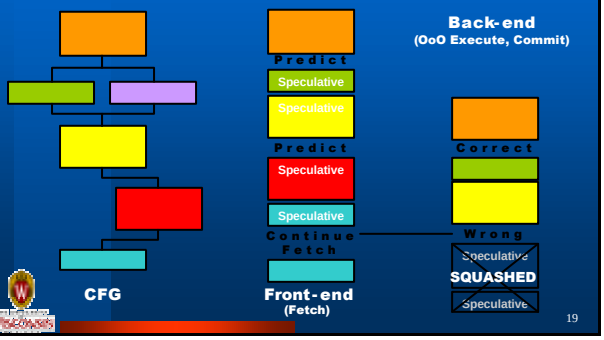
Speculation and Computer Architecture

- Speculate outcome of event rather than waiting for outcome to be known
  - Program behavior provides rationale for high success rate
- Functionality to support speculation
- Functionality to speculate better
- Functionality to minimize mis-speculation penalty

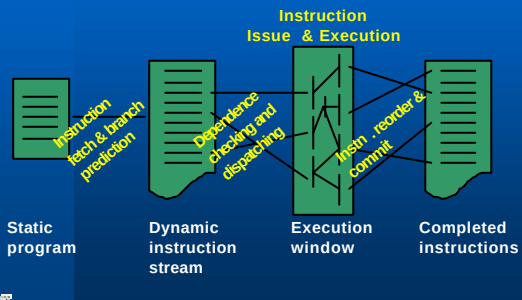
Control Speculation

- Predict outcome of branch instructions
- Speculatively fetch and execute instructions from predicted path
  - Increase available parallelism
- Recover if prediction is incorrect

Control Speculation Example



Model for Speculative Execution



Supporting Control Speculation

- Techniques to predict branch outcome: **branch predictors**
  - Initiating speculation
  - Improving accuracy of speculation
- Techniques to support speculative execution: **reservation stations**, **register renaming** etc.
  - Supporting speculative execution
- Techniques to give appearance of sequential execution: **reorder buffers**, etc.
  - Doing it transparently

[illegible]servation

Basic mechanisms to support control speculation can support other forms of speculation as well

Performance-Inhibiting Constraints

- **Control dependences:** inhibit creation of instruction window
  - Use **control speculation**
- **Ambiguous data dependences:** inhibit parallelism recognition
  - Use **data dependence speculation**
- **True data dependences:** inhibit parallelism
  - Use **value speculation**
- Common mechanisms may support different forms of speculation
- Different techniques to improve accuracy of speculation

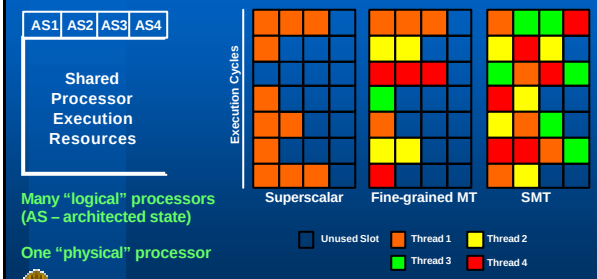
Speculation in Use Today

- Address calculation and translation (especially if 2-step process)
- Cache hit
- Memory ordering violation in multiprocessors
- Load/store dependences
- Many others on the design board

Multithreading

- Microprocessor can execute multiple operations at a time
  - 4 or 6 operations per cycle
- Hard to achieve this level of parallelism from single program
- Can we run multiple programs (threads) on (single) processor without much effort?
  - Simultaneous multithreading (SMT) or Hyperthreading

SMT Overview



Many "logical" processors (AS – architected state)

One "physical" processor

Multithreading

- Today many high-end microprocessors are multithreaded (e.g., Intel Pentium 4)
- Support for 2-4 threads but expect to get only 1.3X improvement in throughput

Microprocessors – the next 10 years

- Factor of 30 increase in semiconductor resources
  - How to use it?
- New constraints
  - Power consumption
  - Wire delays
  - Design / verification complexity
- New applications?
  - Throughput-oriented workloads
  - Coarse-grain multithreaded applications

Technology Trends

- Design and verification of large number of transistors becoming unwieldy
- Wires getting relatively slower
  - Short wires for fast clock
  - Implies increase latencies; exploit locality of communication
- Power issues becoming very important

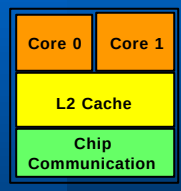
Architect's Role Revisited

- Defining functionality
  - New models needed to further increase parallelism exploitation
- Implementing functionality
  - Becoming a dominating factor?
- Speculation is likely to be the key to overcoming constraints

Implications of Trends

- Implementation considerations will imply computing chips with multiple (replicated?) processing cores
  - "multiprocessor" or "multiprocessor-like" or "multithreaded"
  - Will start out as "logical" replication (e.g., SMT)
  - Will move towards "physical" replication (e.g., CMP)
- How to assign work to multiple processing cores?
  - Independent programs (or threads)
  - Parts of a single program

CMP Overview



- Several processor cores in one die
- Shared L2 caches
- Chip Communication to build multi-chip module with many CMPs + memory

Throughput-Oriented Processing

- Executing multiple, independent programs on underlying parallel micro-architecture
  - Similar to traditional throughput-oriented multiprocessor
  - Significant engineering challenges, but little in ways of architectural / micro-architectural

Can we use underlying "multiprocessor" to speed up execution of single program?

Parallel Processing of Single Program

- Will the promise of explicit / automatic parallelism come true?
- Will new (parallel) programming languages take over the world?

Parallel Processing of Single Program

- Multiple processors in chip will encourage writing of parallel programs
- Reduced (on-chip) latencies may make parallel processing fruitful

Speculative Parallelization

- Sequential languages aren't going away
- Use speculation to overcome inhibitors to "automatic" parallelization
  - Ambiguous dependences
- Divide program into "speculatively parallel" portions or "speculative threads"

Speculative Threads

- Subject of extensive research today
  - Different speculative parallelization models being investigated

Generic circa 2010 Microprocessor

- 4 – 8 general-purpose processing engines on chip
  - Used to execute independent programs
  - Explicitly parallel programs (when possible)
  - Speculatively parallel threads
  - Helper threads
- Special-purpose processing units (e.g., DSP functionality)
- Elaborate memory hierarchy
- Elaborate inter-chip communication facilities
- Extensive use of different forms of speculation

Summary

- Semiconductor technology has, and will continue to, give computer architects new opportunities
- Architects have used speculation techniques to overcome performance barriers; will likely continue to do so
- Future microprocessors are going to have capability to execute multiple threads of code
- New models of speculation (e.g., thread-level speculation) will be needed to extract more parallelism