The Setting: Technology, Market Trends, Critical Issues

Lecture #2: Wednesday, 3 April 2000
Lecturer: Bill Dally
Scribe: Mattan Erez

In this session we discussed three papers:

1. S. Hamilton, *Taking Moore's Law into the Next Century*- presents some of the challenges future generation designers and architects will face.

2. C. Kozyrakis and D. Patterson, *A New Direction for Computer Architecture Research*- evaluates a number of proposed architectures for future processors under very specific figures of merit

3. K. Diefendorff, *PC Processor Microarchitecture* - provides a survey of current techniques employed by architects to achieve high performance in superscalar processors.

1 Taking Moore's Law into the Next Century

Hamilton's view is that the two challenges that must be met to keep apace with Moore's Law are interconnects and design. He also views Moore's law regarding the number of transistors on a chip and not for performance. The connection is in turning transistors into "doing something good", i.e. increasing performance.

- **Interconnect Challenges** - interconnects are becoming a problem since the relative distances that must be traversed are growing. That is, local interconnects scale with technology but the global interconnects do not. The solution is to consider physical aspects at the architecture level - "... a different breed of designer - a combined computer architect and physicist ...". One possible direction is to **exploit locality** by having a modular architecture. But modularity requires that the application be malleable to it. Applications that are highly parallel can usually be mapped on to a modular architecture, whereas sequential control-intensive programs do not map well to such architectures. Other options are new technologies such as 3D semiconductors and optical interconnects that undo some of the detrimental effects currently observed (non-scalable global interconnects).

- **Design Challenges** - The greatest challenge facing designers is increasing complexity. Complexity is increasing due to the larger number of units and more advanced algorithms afforded by the increasing number of transistors. A good solution is to, again, rely on modularity to partition the design. But for some applications modularity is not appropriate, ten two options are available:
  - Adding more people to the project - probably not the answer. Must be able to create hierarchy in the design and spend much effort on interfaces.
  - Better Tools - better synthesis, simulation, and validation tools will help, but cannot handle the architectural aspects of the design.

2 A New Direction for Computer Architecture Research

This was a response to the "What to Do with a Billion Transistors on a Chip" papers. The tone of the paper is set by the fact that the authors considered the previous "measure of goodness" to be inapplicable to future applications. Instead of addressing workstation workloads they focus on multi-media workloads. As a result their architecture uses a much smaller number of transistors for memory caches. The workstation targeted architectures devoted 50% - 90% of the transistor budget to caches, but caches are not effective for streams which have very little temporal locality.

We then went on to discuss what future architectures should be optimized for - what should be the *figures of merit* (the authors list their figures of merit on page 29).

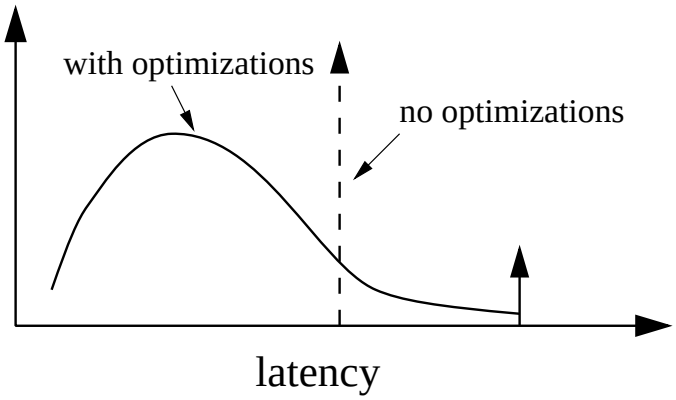
- Performance (MIPS) - for various domains (benchmarks):
  - Scientific Computing and Desktop - Not necessarily parallelisable but contain ILP, large programs, latency is critical.
  - Multi-media - Coarse grain parallelism, small programs with high instruction locality, streaming data, bandwidth is more important than latency.
  - OLTP (On Line Transaction Processing) - Medium to Fine grained parallelism (little ILP), low instruction and data locality, non-predictable control and data flows.
- Power Efficiency
- Ease of Programming
  - High level programming language
  - Binary Compatability - FX!32 and Transmeta address the compatibility issue by translating binaries, but apparently people don't trust such technologies.

- Area / Physical design complexity

The authors address multi-media only whereas architectures should probably target multiple domains. Such as targets which have commercial market demand or that have not been well researched.

Many current high performance architectural techniques are not suitable to the properties of multi-media applications(page 28):

- **Real-time Constraints** - Most optimizations are tuned to improving the common case and may have nearly unbounded worse case performance. The fact that humans can't tell even when the longest latency is incurred is not a valid argument for using these techniques for applications such as data communication.



  Unpredictable latencies such as for cache misses or mispredicted branches make these techniques less appealing for real-time applications.

- **Stream Data Types**- caches are not appropriate for streaming applications due to the lack of temporal locality.

- **High Memory Bandwidth Requirements** - current memory systems are tuned for low latency and not high bandwidth.

- **Many Levels of Parallelism** - multi-media applications have both fine-grain and coarse-grain parallelism, whereas most high-end processors take advantage of fine-grain only.

In summary, the paper's philosophy is *Keep It Simple*. Simple design and simple software will have greater performance while requiring lower power (less energy). The two main complaints the authors have for current architectures with respect to multi-media applications are:

1. The memory systems of general purpose processors are not tuned (perhaps tuned against) multi-media applications

2. The power-performance trade off is also not right for multi-media. Speculation and high-complexity burn power while affording little to no performance gains for these applications.

3 PC Processor Microarchitecture

This is a survey paper of current superscalar techniques and we did not spend much time on it in class. Two main points were made:

- **Three Ways to Improve Performance** -
  - Increase processor frequency (most of the performance gain)
  - Pipelining
  - Parallel execution

  Both pipelining and parallel execution rely on inherent parallelism in the applications, but are not exactly the same (throughput vs. latency).

- **Out-Of-Order Execution** - "Something is wrong" with instruction set architectures if the hardware needs to re-learn information that the compiler already had. For example,data dependencies are obscured by a small number of architectural registers. However, certain things, such as exact control flow, are known only at run-time. Also, it may be easier for the hardware to re-learn information than to decode complex instructions.

4 An Additional Note

It was agreed that ***instruction dispatch*** refers to moving instructions from the fetch/decode stage to the instruction window, and ***instruction issue*** is the process of moving instructions to the execution units.