Revisiting the Future 😊

Erik Hagersten
Uppsala University
Sweden

DARK2 in a nutshell

1. Memory Systems (caches, VM, DRAM, microbenchmarks, ...)
2. Multiprocessors (TLP, coherence, interconnects, scalability, clusters, ...)
3. CPUs (pipelines, ILP, scheduling, Superscalars, VLIWs, embedded, ...)
4. Future: (physical limitations, TLP+ILP in the CPU, ...)

How do we get good performance?

- Creating and exploring:
  1) Locality
     a) Spatial locality
     b) Temporal locality
     c) Geographical locality
  2) Parallelism
     a) Instruction level (ILP)
     b) Thread level (TLP)
     c) Memory level (MLP)

Doubling (or Halving) times

- Dynamic RAM Memory (bits per dollar) 1.5 years
- Average Transistor Price 1.6 years
- Microprocessor Cost per Transistor Cycle 1.1 years
- Total Bits Shipped 1.1 years
- Processor Performance in MIPS 1.8 years
- Transistors in Intel Microprocessors 2.0 years
- Microprocessor Clock Speed 2.7 years
Old Trend 1: Deeper pipelines
Exploring ILP (instruction-level parallelism)

Old Trend 2: Wider pipelines
Exploring more ILP

Old Trend 3: Deeper memory hierarchy
Exploring access locality

Are we hitting the wall now?

Pop: Can the transistors be made even smaller and faster?

Performance [log]

Possible path, but requires a paradigm shift

Business as usual...

Well uhm ...
The transistors can be made smaller and faster, but there are other problems 😐
Instruction-Level Parallelism (ILP) in Superscalar Pipelines

START:
J=K+L
G=H+I
D=E+F
A=B+C
MEM[X]=MEM[X]+14
X=X+1
IF X < 1000 GOTO START:

K+L START:

Issue Logic

Microprocessors today: Whatever it takes to run one program fast.

Exploring ILP (instruction-level parallelism):
- Faster clocks → Deep pipelines
- Superscalar Pipelines
- Branch Prediction
- Out-of-Order Execution
- Trace Cache
- Speculation
- Predicate Execution
- Advanced Load Address Table
- Return Address Stack

Bad News #1: We have already explored most ILP

Bad News #2: Long wire delay → slow CPUs

Quantitative data and trends according to V. Agarwal et al., ISCA 2000
Based on SIA (Semiconductor Industry Association) prediction, 1999
Bad News #2: How much of the chip area can you reach in one cycle?

<table>
<thead>
<tr>
<th>Year</th>
<th>Feature size (micron)</th>
<th>Span -- Fraction of chip reachable in one cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>0.25</td>
<td>1.0</td>
</tr>
<tr>
<td>2005</td>
<td>0.18</td>
<td>0.95</td>
</tr>
<tr>
<td>2010</td>
<td>0.13</td>
<td>0.9</td>
</tr>
<tr>
<td>2015</td>
<td>0.10</td>
<td>0.85</td>
</tr>
<tr>
<td>2020</td>
<td>0.07</td>
<td>0.8</td>
</tr>
<tr>
<td>2030</td>
<td>0.05</td>
<td>0.8</td>
</tr>
<tr>
<td>2050</td>
<td>0.035</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Bad News #3: Looong wire delay \(\Rightarrow\) slow CPUs

<table>
<thead>
<tr>
<th>Feature</th>
<th>Span</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIA</td>
<td>0.75</td>
</tr>
<tr>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td>0.25</td>
<td></td>
</tr>
</tbody>
</table>

Bad News #4: Power is the limit

- Power consumption is the bottleneck
  - Cooling servers is hard
  - Battery lifetime for mobile computers
  - Energy is money

- Dynamic effect is proportional to
  \[ \sim \text{Frequency} \]
  \[ \sim \text{Voltage}^2 \]

Now What?

#1: Running out of ILP

#2: Wire delay is starting to hurt

#3: Memory is the bottleneck

#4: Power is the limit
Solving all the problems: exploring threads parallelism

#1: Running out of ILP
   ➔ feed one CPU with instr. from many threads

#2: Wire delay is starting to hurt
   ➔ Multiple small CPUs with private L1$

#3: Memory is the bottleneck
   ➔ memory accesses from many threads (MLP)

#4: Power is the limit
   ➔ Lower the frequency ➔ lower voltage

Bad News #1: Not enough ILP 1(2)
   ➔ feed one CPU with instr. from many threads

Sloooow Memory

Bad News #1: Not enough ILP 2(2)
   ➔ feed one CPU with instr. from many threads

SMT: Simultaneous Multithreading
   “Combine TLP&ILP to find independent instr.”
Thread-interleaved
- Each thread executes every n:th cycle in a round-robin fashion
- Historical Examples (1984..)
  Denelcor, HEP, Tera Computers [B. Smith]
  -- Poor single-thread performance
  -- Expensive (due to early adoption)

Bad News #2: wire delay
- Multiple small CPUs with private L1$

L2 Cache
L3 Ctrl
Externa L3
Externa L3
L1 cache
CPU

Bad News #3: memory latency/bandwidth
- memory accesses from many threads (MLP)
- Sloooow memory

CMP: Chip Multiprocessor
more TLP & geographical locality

TLP $\Rightarrow$ MLP
Thread-Level Parallelism $\Rightarrow$ Memory-Level Parallelism (MLP)
**Bad News #4: Power consumption**

- Lower the frequency
- Lower voltage

\[ P_{\text{dyn}} = C \cdot f \cdot V^2 \approx \text{area} \cdot \text{freq} \cdot \text{voltage}^2 \]

<table>
<thead>
<tr>
<th>CPU freq=f</th>
<th>CPU freq=f/2</th>
<th>CPU freq=f/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_{\text{dyn}}(C, f, V) = CfV^2 )</td>
<td>( P_{\text{dyn}}(2C, f/2, &lt;V) &lt; CfV^2 )</td>
<td>( P_{\text{dyn}}(C, f/2, &lt;V) &lt; \frac{1}{2} CfV^2 )</td>
</tr>
</tbody>
</table>

**Throughput < 2x**

**Around the corner...**

- Chip Multiprocessors (CMP)
- Simultaneous Multithreading (SMT)

**Darling, I shrunk the computer**

- **Mainframes**
- **Super Minis**:
- **Microprocessor:** Paradigm Shift
- **Chip Multiprocessor (CMP): A multiprocessor on a chip!**

**Now, everyone is doing it!**

"Intel have 10 projects in the works that contain four or more computing cores per chip"

[Paul Otellini, Intel Chief Executive at IDF fall 2005]

"Today, processors with multiple CPUs and a large cache on a single chip are becoming common. Attempts to tease the parallelism out of a sequential program automatically haven’t worked out very well. We need better education, better languages, and better tools, since building concurrent programs is hard"

[Andrew Herbert, Director of Microsoft Cambridge Research Lab, May 2005]
What thread parallelism?

![Diagram showing thread parallelism]

- Prog1
- Prog2
- OS
- Thread1
- Thread2
- Thread3

Capacity Computing
Capability Computing

Seq. Prop.
Parallel Prop
Parallelizing compiler
Thread speculation

Thread1
Thread2
Thread3

Prog1
Prog2
OS

CMP: Chip Multiprocessor

- External I/F
- Mem I/F
- L2$
- S1
- S1
- S1
- S1

CPU
CPU
CPU
CPU

Simple fast CPU core (w/ SMT?) + local caches

Looks and Smells Like an SMP?

- Memory
- Interconnect
- L2
- L2
- L2
- L1
- L1
- L1

... 32 ...

Well, how about:
- Cost of parallelism?
- Cache capacity per thread?
- Memory bandwidth per thread?
- Cost of thread communication? ...

Trends (my guess!)

- Threads/Chip
- Transistors/Thread
- Memory/Chip
- Cache/Thread
- Bandwidth/Thread
- Thr. Comm. Cost (temporal)
Design Issues for CMPs

Erik Hagersten
Uppsala University
Sweden

Niagara Chip

UltraSPARC-Core

Niagara

Sun Microsystems

UltraSPARC T1 Processor

System Interface
Buffer Switch Core

4 x DDR-2 = 25GB/s (!)

4 x DDR-2 = 25GB/s (!)

Shared L2!

Shared caches: Good or bad?
CMP bottlenecks/points of optimization

- Performance per Watt?
- Performance per memory byte?
- Performance per bandwidth?
- Performance per $?
- ...

- How large fraction of a CMP system cost is the CPU chip?
- Should the execution (MIPS/FLOPS) be viewed as a scarce resource?

DRAM issues

“Rock will have more than 1000 memory chips per Rock chip” [M. Trembley, Sun Fellow at ICS 2006]

- Memory will dominate cost?

- Fewer “open pages” accessed due to interleaving of several threads

- Far memory ➔ long latency & low per-pin BW

Pushing dense memory technology/packaging?

Bandwidth issues

- #pins is a scarce resource!
- every pin should run at maximum speed

- external memory controllers?
- off-chip cache?
- is there room for multi-CMP?
- is this maybe a case for multi-CMP?

More transistors ➔ More Threads

- Warning:
  - # transistors grows exponentially
  - ➔ # threads can grow exponentially
  - Can memory BW keep up?
Transistors vs. off-chip bandwidth

SIA Prediction: #Transistors vs. pin bandwidth

A case for: off-chip memory controllers?

A case for off chip L3 cache controllers
A case for multiple small CMPs

Capacity or Capability Computing?
Capacity? (≈several sequential jobs)
or
Capability? (≈one parallel job)

Issues:
- Memory requirement?
- Sharing in cache?
- Memory bandwidth requirement?

Memory: the major cost of a CMP system!
How do we utilize it the best?
- Once the workingset is in memory, work like crazy!

→ Capability computing suits CMPs the best (in general)

Fat or narrow cores?
- Fat:
  - Fewer cores but...
  - wide issue?
  - O-O-O?

- Narrow: More cores but...
  - narrow issue?
  - in-order?
  - have you ever heard of Amdahl?
  - SMT, run-ahead, execute-ahead ... to cure shortcomings?

Read:
Maximizing CMP Throughput with Mediocre Cores
Davis, Laudon and Olukotun, PACT 2006

Cores vs. caches
- Depends on your target applications...

- Niagara’s answer: go for cores
  - In-order 5-stage pipeline
  - 8 cores a’ 4 SMT threads each → 32 threads,
  - 3MB shared L2 cache (96 kB/thread)
  - SMT to hide memory latency
  - Memory bandwidth: 25 GB/s
  - Will this approach scale with technology?

- Others: go for cache
  - 2-4 cores for now
Privat/Shared Caches? 
Early Commercial CMPs

- Cache sharing strategies:
  1. Fight it out!
  2. Fair share: 50% of the cache each
  3. Maximize throughput: who will benefit the most?

- Read:
  STATSHARE: A Statistical Model for Managing Cache Share via Decay
  Pavlos Petoumenos et al in MOBS workshop ISCA 2006

- Predicting the inter-thread cache contention on a CMP
  Chandra et al in HPCA 2005

Hiding Memory Latency

- O-O-O
- SMT
- Run-ahead/Execute-ahead

Questions for the Future

- What applications?
- How to get parallelism and data locality?
- Will funky languages see a renascence?
- Will automatic parallelizing return?
- Are we buying:
  - compute power,
  - memory capacity, or
  - memory bandwidth?
- Will the CPU market diverge into desktop/capacity CPUs again?
- How to debug? ...
- A non-question: will it happen?
Major points for CMP designers:

- Start from a clean slate – this is not an SMP!
  - bandwidth & memory capacity is king
  - a case for memory compression [M. Ekman, ISCA 2005]
  - on-chip memory controllers is not a given
  - off-chip cache is not a given

- New rules for coherence
  - snooping is still in the ballgame
  - directory-based only if needed
  - cache-line size is not a given, ...

- Consider cheap options for hiding memory latency

- Trade-off core-obesity/#cores/cache_size can be very application specific!

- Manage cache sharing

- Capacity/capability computing is very different

- ...

...