The Implications of Multi-core

What I want to do today

- ***Given that everyone is heralding Multi-core***
  - ***Is it really the Holy Grail?***
  - ***Will it cure cancer?***
- ***A lot of misinformation has surfaced***
- ***What multi-core is and what it is not***
- ***And where we go from here***

To whet the appetite

- ***Can multi-core save power via the freq cube law?***
- ***Is ILP dead?***
- ***Should sample benchmarks drive future designs?***
- ***Is hardware really sequential?***
- ***Should multi-core structures be simple?***
- ***Does productivity demand we ignore what's below?***

The Compile-time Outline

- ***Multi-core: how we got here***
- ***Mis-information***
- ***Where do we go from here***

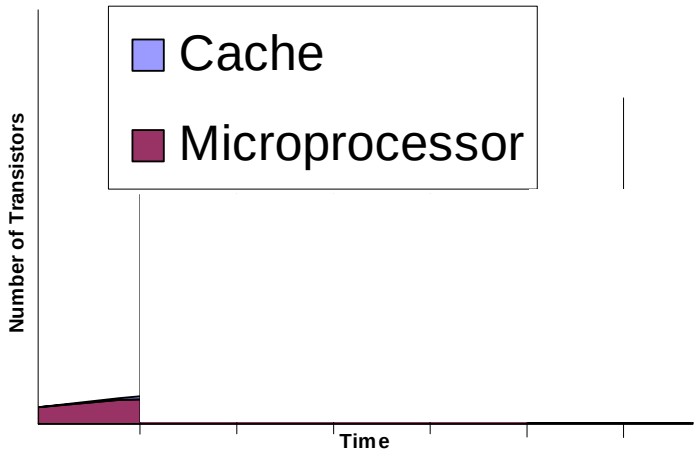
Outline

- ***Multi-core: how we got here***
- ***Mis-information***
- ***Where we go from here***

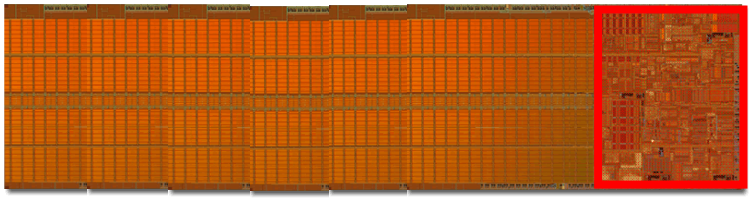
How we got here (Moore's Law)

- ***The first microprocessor (Intel 4004), 1971***
  - ***2300 transistors***
  - ***106 KHz***
- ***The Pentium chip, 1992***
  - ***3.1 million transistors***
  - ***66 MHz***
- ***Today***
  - ***more than one billion transistors***
  - ***Frequencies in excess of 5 GHz***
- ***Tomorrow ?***

How have we used the available transistors?

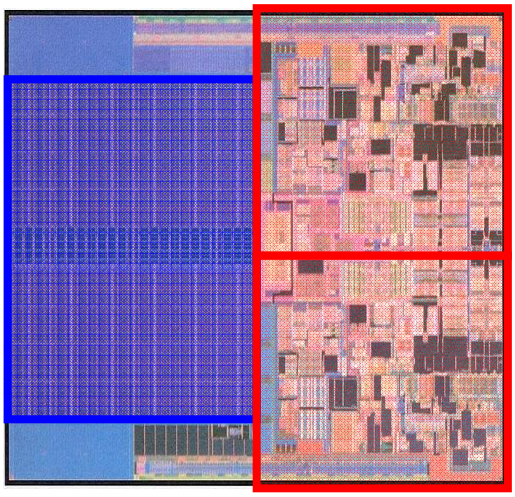


Intel Pentium M



Intel Core 2 Duo

- ***Penryn, 2007***
- ***45nm, 3MB L2***



Why Multi-core chips?

- ***In the beginning: a better and better uniprocessor***
  - ***improving performance on the hard problems***
  - ***…until it just got too hard***
- ***Followed by: a uniprocessor with a bigger L2 cache***
  - ***forsaking further improvement on the "hard" problems***
  - ***poorly utilizing the chip area***
  - ***and blaming the processor for not delivering performance***
- ***Today: dual core, quad core, octo core***
- ***Tomorrow: ???***

Why Multi-core chips?

- ***It is easier than designing a much better uni-core***
- ***It was embarrassing to continue making L2 bigger***
- ***It was the next obvious step***

So, What's the Point

- ***Yes, Multi-core is a reality***
- ***No, it wasn't a technological solution to performance improvement***
- ***Ergo, we do not have to accept it as is***
- ***i.e., we can get it right the second time, and that means:***
  What goes on the chip
  What are the interfaces

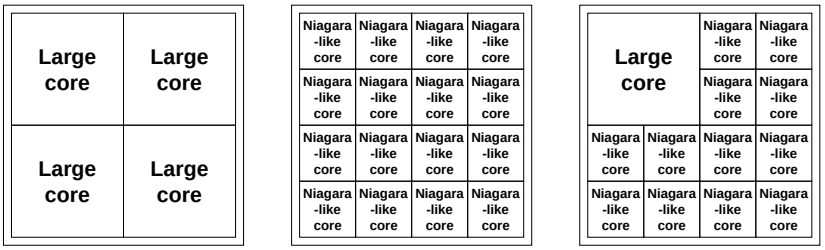
Outline

- ***Multi-core: how we got here***
- ***Mis-information, or more accurately: Multi-nonsense***
- ***Where do we go from here***

Multi-nonsense

- ***Multi-core was a solution to a performance problem***
- ***Hardware works sequentially***
- ***Make the hardware simple – thousands of cores***

The Asymmetric Chip Multiprocessor (ACMP)



Large core vs. Small Core

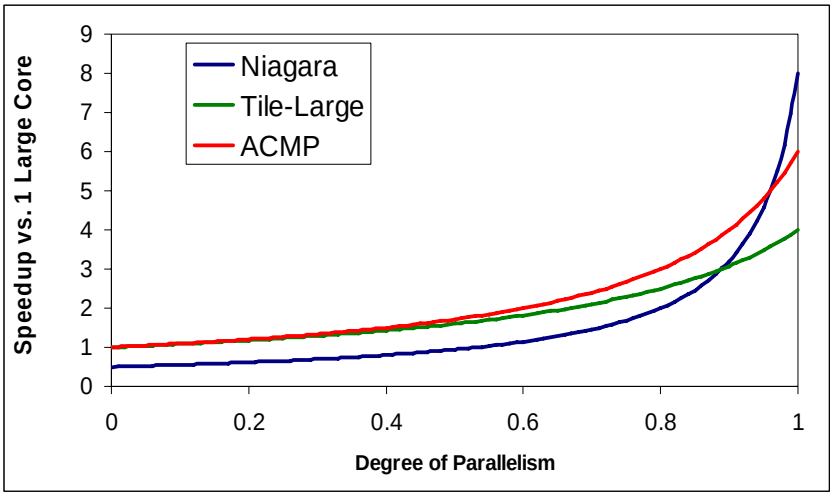
Large Core

- ***Out-of-order***
- ***Wide fetch e.g. 4-wide***
- ***Deeper pipeline***
- ***Aggressive branch predictor (e.g. hybrid)***
- ***Many functional units***
- ***Trace cache***
- ***Memory dependence speculation***

Small Core

- ***In-order***
- ***Narrow Fetch e.g. 2-wide***
- ***Shallow pipeline***
- ***Simple branch predictor (e.g. Gshare)***
- ***Few functional units***

Throughput vs. Serial Performance



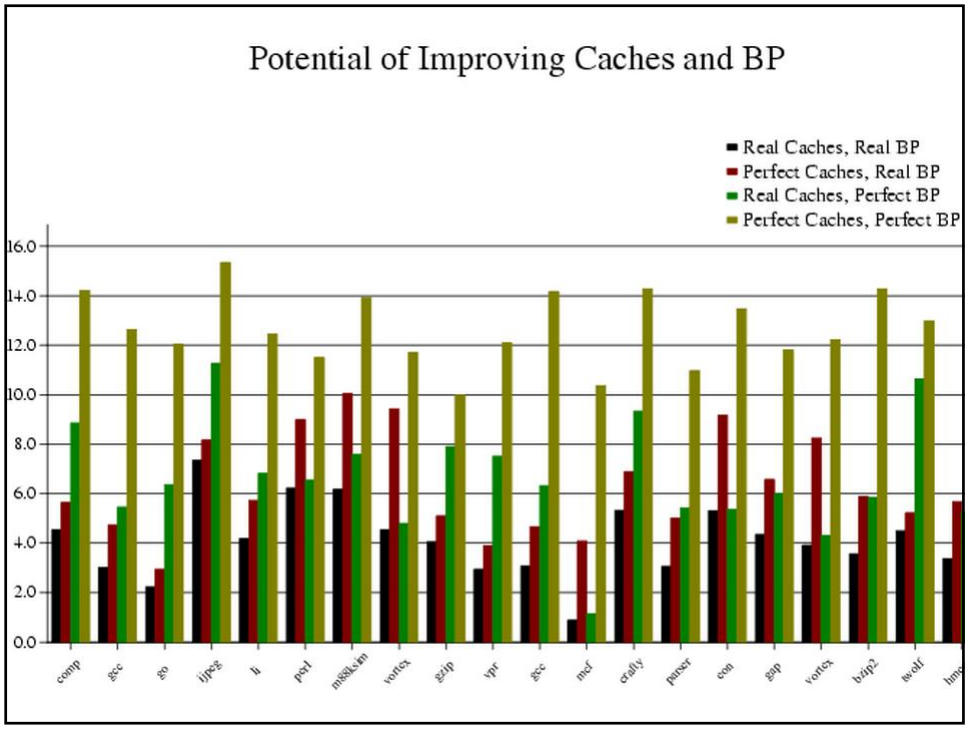
Multi-nonsense

- ***Multi-core was a solution to a performance problem***
- ***Hardware works sequentially***
- ***Make the hardware simple – thousands of cores***
- ***Do in parallel at a slower clock and save power***
- ***ILP is dead***

ILP is dead

- ***We double the number of transistors on the chip***
  - ***Pentium M: 77 Million transistors (50M for the L2 cache)***
  - ***2nd Generation: 140 Million (110M for the L2 cache)***
- ***We see 5% improvement in IPC***
- ***Ergo: ILP is dead!***
- ***Perhaps we have blamed the wrong culprit.***

- ***The EV4,5,6,7,8 data: from EV4 to EV8:***
  - ***Performance improvement: 55X***
  - ***Performance from frequency: 7X***
  - ***Ergo: 55/7 > 7 -- more than half due to microarchitecture***



Moore's Law

- ***A law of physics***
- ***A law of process technology***
- ***A law of microarchitecture***
- ***A law of psychology***

Multi-nonsense

- ***Multi-core was a solution to a performance problem***
- ***Hardware works sequentially***
- ***Make the hardware simple – thousands of cores***
- ***Do in parallel at a slower clock and save power***
- ***ILP is dead***
- ***Examine what is (rather than what can be)***

Examine what is (rather than what can be)

Should sample benchmarks drive future designs?

Another bridge over the East River?

Multi-nonsense

- ***Multi-core was a solution to a performance problem***
- ***Hardware works sequentially***
- ***Make the hardware simple – thousands of cores***
- ***Do in parallel at a slower clock and save power***
- ***ILP is dead***
- ***Examine what is (rather than what can be)***
- ***Communication: off-chip hard, on-chip easy***
- ***Abstraction is a pure good***
- ***Programmers are all dumb and need to be protected***
- ***Thinking in parallel is hard***

Outline

- ***Multi-core: how we got here***
- ***Mis-information***
- ***Where do we go from here***

In the next few years:

- ***Process technology: 50 billion transistors***
  - ***Gelsinger says we are can go down to 10 nanometers (I like to say 100 angstroms just to keep us focused)***
- ***Dreamers will use whatever we come up with***
- ***What should we put on the chip? How should software interface to it?***

How will we use 50 billion transistors?

How have we used the transistors up to now?

The Good News: Lots of cores on the chip

The Bad News: Not much benefit.

In my opinion the reason is:

Our inability to effectively exploit:

-- The transformation hierarchy
-- Parallel programming

Problem

Algorithm

Program

ISA (Instruction Set Arch)

Microarchitecture

Circuits

Electrons

Up to now

- ***Maintain the artificial walls between the layers***
- ***Keep the abstraction layers secure***
  - ***Makes for a better comfort zone***
- ***(Mostly) Improving the Microarchitecture***
  - ***Pipelining, Caches***
  - ***Branch Prediction, Speculative Execution***
  - ***Out-of-order Execution, Trace Cache***
- ***Today, we have too many transistors***
  - ***Bandwidth, power considerations too great***
  - ***We MUST change the paradigm***

We Must Break the Layers

- ***(We already have in limited cases)***
- ***Pragmas in the Language***
- ***The Refrigerator***
- ***X + Superscalar***
- ***The algorithm, the language, the compiler, & the microarchitecture all working together***

IF we break the layers:

- ***Compiler, Microarchitecture***
  - ***Multiple levels of cache***
  - ***Block-structured ISA***
  - ***Part by compiler, part by uarch***
  - ***Fast track, slow track***
- ***Algorithm, Compiler, Microarchitecture***
  - ***X + superscalar – the Refrigerator***
  - ***Niagara X / Pentium Y***
- ***Microarchitecture, Circuits***
  - ***Verification Hooks***
  - ***Internal fault tolerance***

Unfortunately:

- ***We train computer people to work within their layer***
- ***Too few understand anything outside their layer***

and, as to multiple cores:

- ***People think sequential***

Conventional Wisdom Problem 1: "Abstraction" is Misunderstood

- *Taxi to the airport*
- *The Scheme Chip (Deeper understanding)*
- *Sorting (choices)*
- *Microsoft developers (Deeper understanding)*

Conventional Wisdom Problem 2: Thinking in Parallel is Hard

- *Perhaps: Thinking is Hard*
- *How do we get people to believe:*
  Thinking in parallel is natural

Parallel Programming is Hard?

- ***What if we start teaching parallel thinking in the first course to freshmen***
- ***For example:***
  - ***Factorial***
  - ***Parallel search***
  - ***Streaming***

~~*We have an Education Problem*~~ *We have an Opportunity*

- ***Too many computer professionals don't get it***
- ***Applications can drive Microarchitecture***
  - ***IF we can understand each other's job***
- ***Thousands of cores, Special function units***
  - ***Ability to power on/off under program control***
- ***Algorithms, Compiler, Microarchitecture, Circuits all talking to each other …***
- ***IF we can specify the right interfaces,***
- ***IF we can specify the language constructs that can use the underlying microarchitecture structures***

IF we understand:

- ***50 billion transistors means we can have:***
  - ***A large number of simple processors, AND***
  - ***A few very heavyweight processors, AND***
  - ***Enough "refrigerators" for handling special tasks***
- ***Some programmers can take advantage of all this***
- ***Those who can't need support***
- ***We need software that can enable all of the above***

that is:

- ***IF we are willing to continue to pursue ILP***
- ***IF we are willing to break the layers***
- ***IF we are willing to embrace parallel programming***
- ***IF we are willing to provide more than one interface***
- ***IF we are willing to understand more than our own layer of the abstraction hierarchy so we really can talk to each other***

Then maybe we can really harness the resources of the multi-core and many-core chips

It WILL BE a Multi-core chip

- ***But it will be PentiumX/Niagara Y***
- ***With multiple interfaces to the software***
- ***It will tackle off-chip bandwidth***
- ***It will tackle power consumption (ON/OFF switches)***
- ***It will tackle soft errors (internal fault tolerance)***
- ***It will tackle seurity***
- ***And it WILL CONTAIN a heavyweight ILP processor***
  - ***With lots of Refrigerators***
  - ***And with the levels of transformation integrated***

A Glimpse of the Heavyweight Processor:

- ***Compiler/Microarchitecture Symbiosis***
  - ***Multiple levels of cache***
  - ***Fast track / Slow track***
  - ***Part by compiler, part by microarchitecture***
  - ***Block-structured ISA***
- ***Better Branch Prediction (e.g., indirect jumps)***
- ***Ample sprinkling of Refrigerators***
- ***SSMT (Also known as helper threads)***
- ***Power Awareness (more than ON/OFF switches)***
- ***Verification hooks (CAD a first class citizen)***
- ***Internal Fault tolerance (for soft errors)***
- ***Better security***

The Serious ILP Processor (continued):

- ***Most importantly: two interfaces***
  - ***One for programmers who understand***
  - ***One for programmers who don't understand***
- ***And, layers of software for those who don't.***