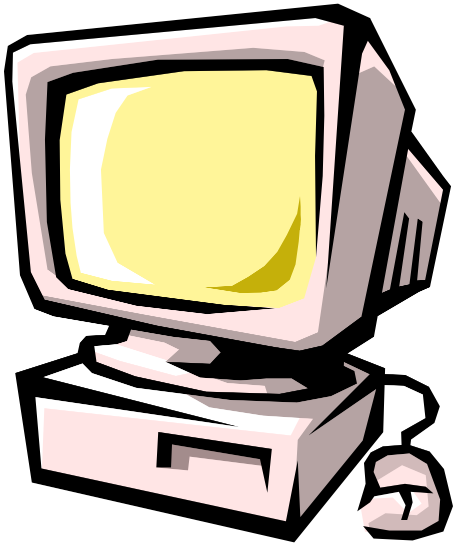
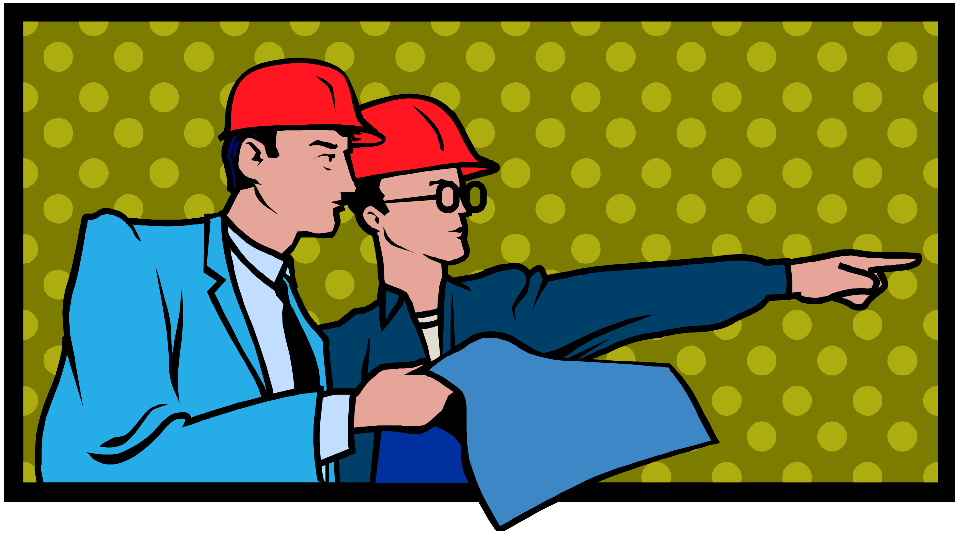


CSE 141-- Introduction to Computer Architecture

Dean Tullsen

What is Computer Architecture?



- Hardware Designer
  - thinks about circuits, components, timing, functionality, ease of debugging

"construction engineer"

- Computer Architect
  - thinks about high-level components, how they fit together, how they work together to deliver performance.

"building architect"

Why do I care?

- **You may actually do computer architecture someday**
- **You may actually care about software performance someday**
  - The ability of application programs, compilers, operating systems, etc. to deliver performance depends critically on an understanding of the underlying computer organization.
  - That becomes more true every year.
    - Up until about 10 years ago, that was primarily due to the increasing complexity of the core microarchitecture
    - Since, it is more about the reliance on hardware parallelism and hardware heterogeneity
- **You may actually care about computer security**
  - Most of the newest and most insidious security attacks have focused on microarchitectural details.

Administration

- Instructor -- Dr. Dean Tullsen
- Who are you?
- TAs:
  - Joey Rudek
  - Jiayan Dong
  - Nishant Ravindra
  - Xuanang (Leon) Li
- OHs, etc.
- Discussion Section

Administration/syllabus

- Lectures, etc.
- Lecture slides
- Textbook
  - Patterson & Hennessy, "Computer Organization and Design -- The Hardware/Software Interface", Morgan Kaufmann, Fifth Edition
  - Other possible sources

Administration/syllabus

- Homeworks
  - Always (?) due on Thursday
  - Turn in via gradescope
  - Typed (recommended)
  - Late policy
- Exams
  - Midterm and final. Final covers entire course

Class Management

- Canvas
  - Announcements
  - Assignments
  - Lecture slides
  - OHs (zoom links when applicable)
- Piazza
  - 141 content q&a
  - Not the place for specific help on hw problems (office hours, direct emails, direct piazza messages…)
  - That's all
- Gradescope
  - All homeworks
- Clickers (still figuring that out)

Grading

- Weekly homework 20%
- Clickers 2%
- Midterm 30%
- Final 48%

- Midterm likely around Feb 14
- Final on Mar 21

Integrity!

- I take this very seriously.
- What is not cheating
- What is cheating
- Penalties
  - Homework
  - Exams

A bit about the learning process

1. Read the text (don’t skim, but don’t need to study, initially). I’ll give you a guide to readings – what to read carefully, what you *can* skim or skip.

   Write down questions

2. Attend and participate in lectures (we’ll focus on more critical, difficult topics, sometimes introduce new topics not in text)

   If questions aren’t cleared up, ask

3. Work problems in class (e.g. clicker questions)

   Have pencil and paper handy

4. Work homework problems

   If steps 1-3 went well, only need to use text and lecture notes as reference

5. Study for tests

   Revisit all of the above, as needed. Work extra problems (eg from book).

What is Computer Architecture?

Computer Architecture =

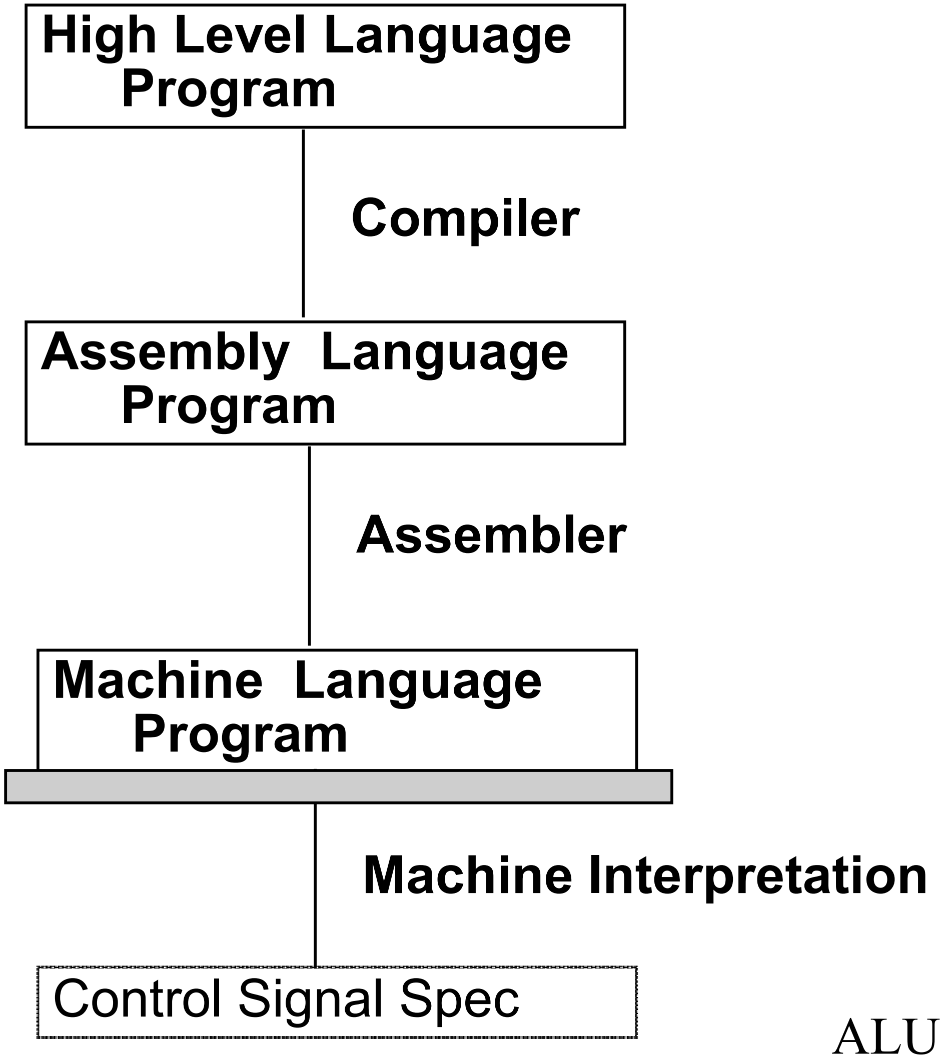
Machine Organization +

Instruction Set Architecture

What the machine hardware looks like

How you talk to the machine

How to Speak Computer



High Level Language Program

Compiler

Assembly Language Program

Assembler

Machine Language Program

Machine Interpretation

Control Signal Spec

```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

```
lw  $15,   0($2)
lw  $16,   4($2)
sw  $16,   0($2)
sw  $15,   4($2)
```

```
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
```

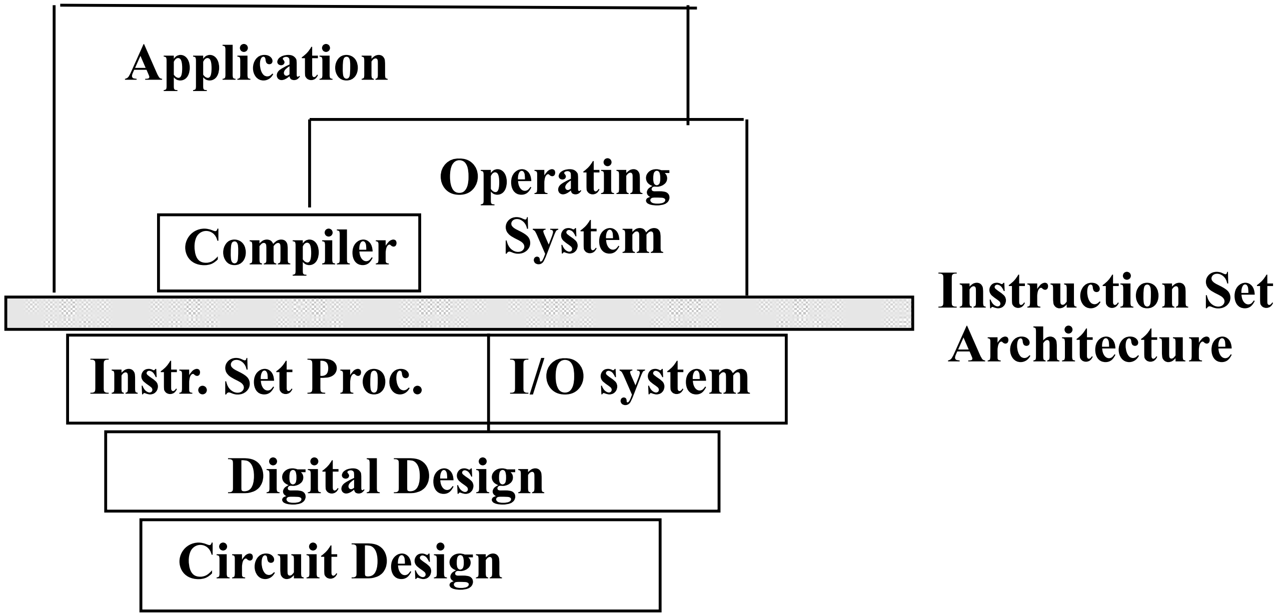
ALUOP[0:3] <= InstReg[9:11] & MASK

The Instruction Set Architecture

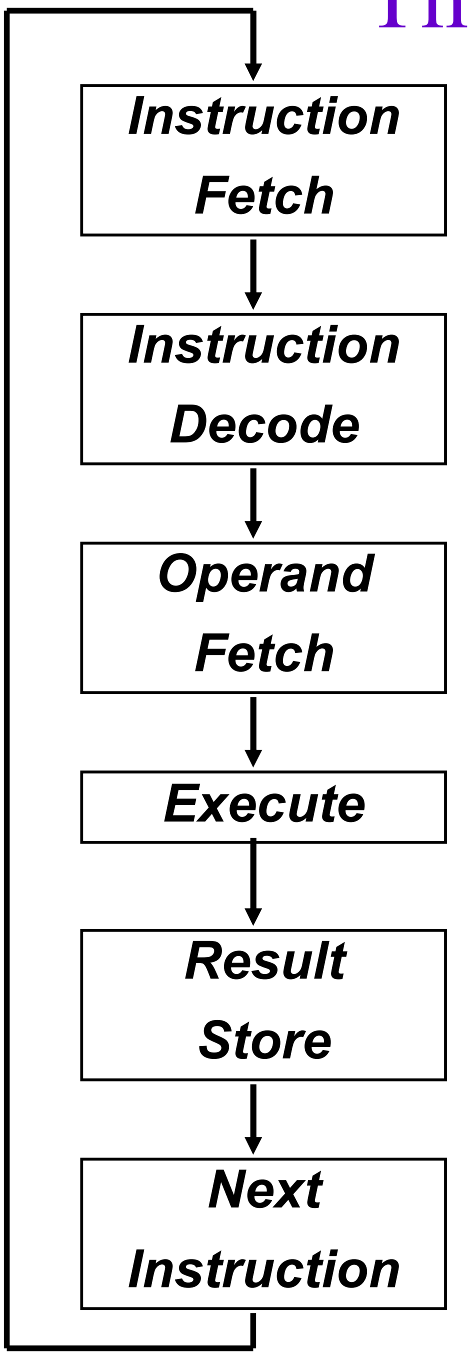
- that part of the architecture that is visible to the programmer
  - opcodes (available instructions)
  - number and types of registers
  - instruction formats
  - storage access, addressing modes
  - exceptional conditions

The Instruction Set Architecture

° is the agreed-upon interface between all the software that runs on the machine and the hardware that executes it.



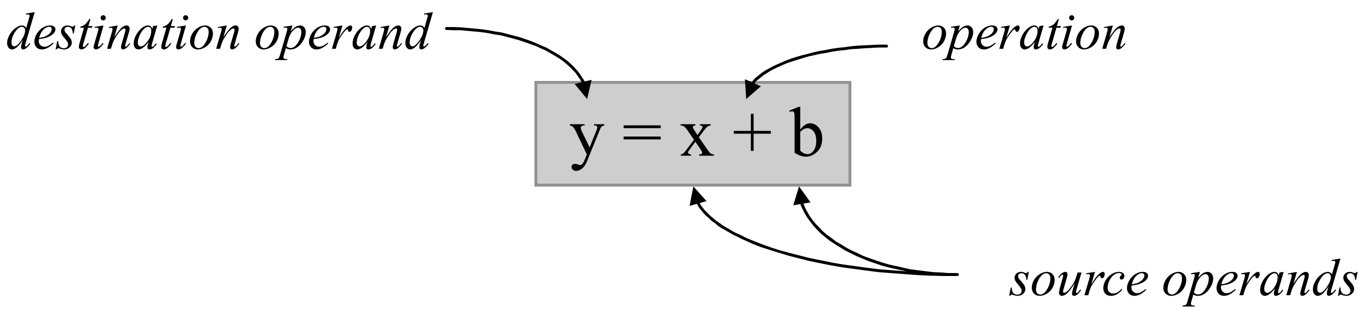
The Instruction Execution Cycle



Instruction Fetch	**Obtain instruction from program storage**
Instruction Decode	**Determine required actions and instruction size**
Operand Fetch	**Locate and obtain operand data**
Execute	**Compute result value or status**
Result Store	**Deposit results in storage for later use**
Next Instruction	**Determine successor instruction**

Key ISA decisions

- operations
  - how many?
  - which ones
- operands
  - how many?
  - location
  - types
  - how to specify?
- instruction format
  - size
  - how many formats?



destination operand

operation

y = x + b

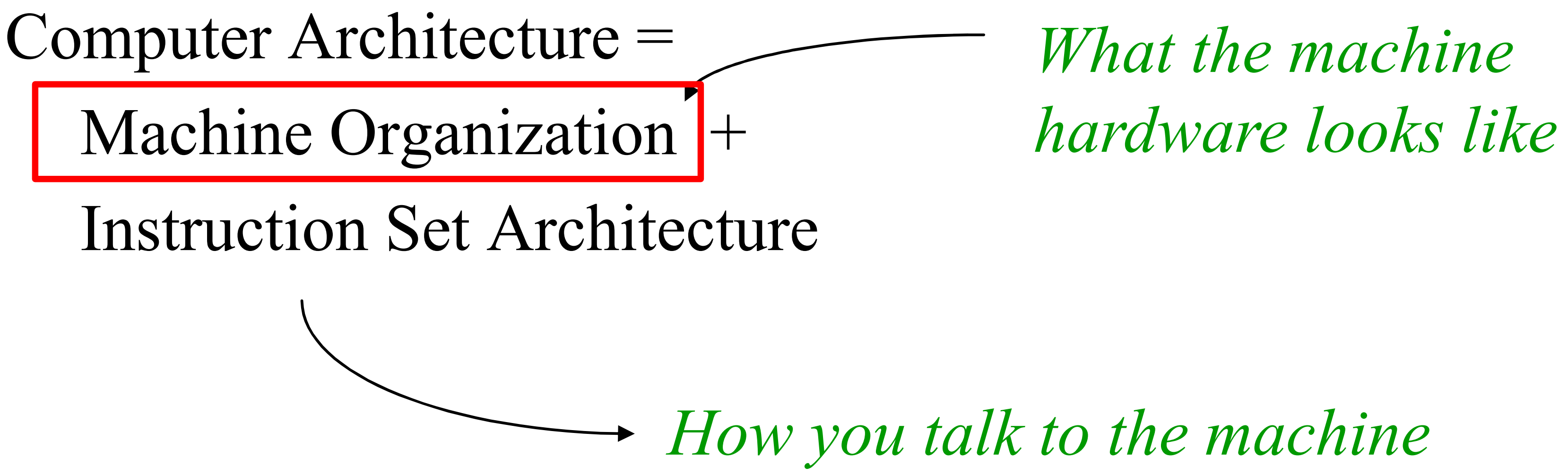
source operands

(add r1, r2, r5)

Examples of ISAs

- Intel 80x86
- VAX
- MIPS
- SPARC
- Alpha AXP
- IBM 360
- Intel IA-64 (Itanium)
- PowerPC
- IBM Cell SPE
- ARM
- Thumb
- RISC-V

What is Computer Architecture?



Computer Architecture =

Machine Organization +

Instruction Set Architecture

What the machine hardware looks like

How you talk to the machine

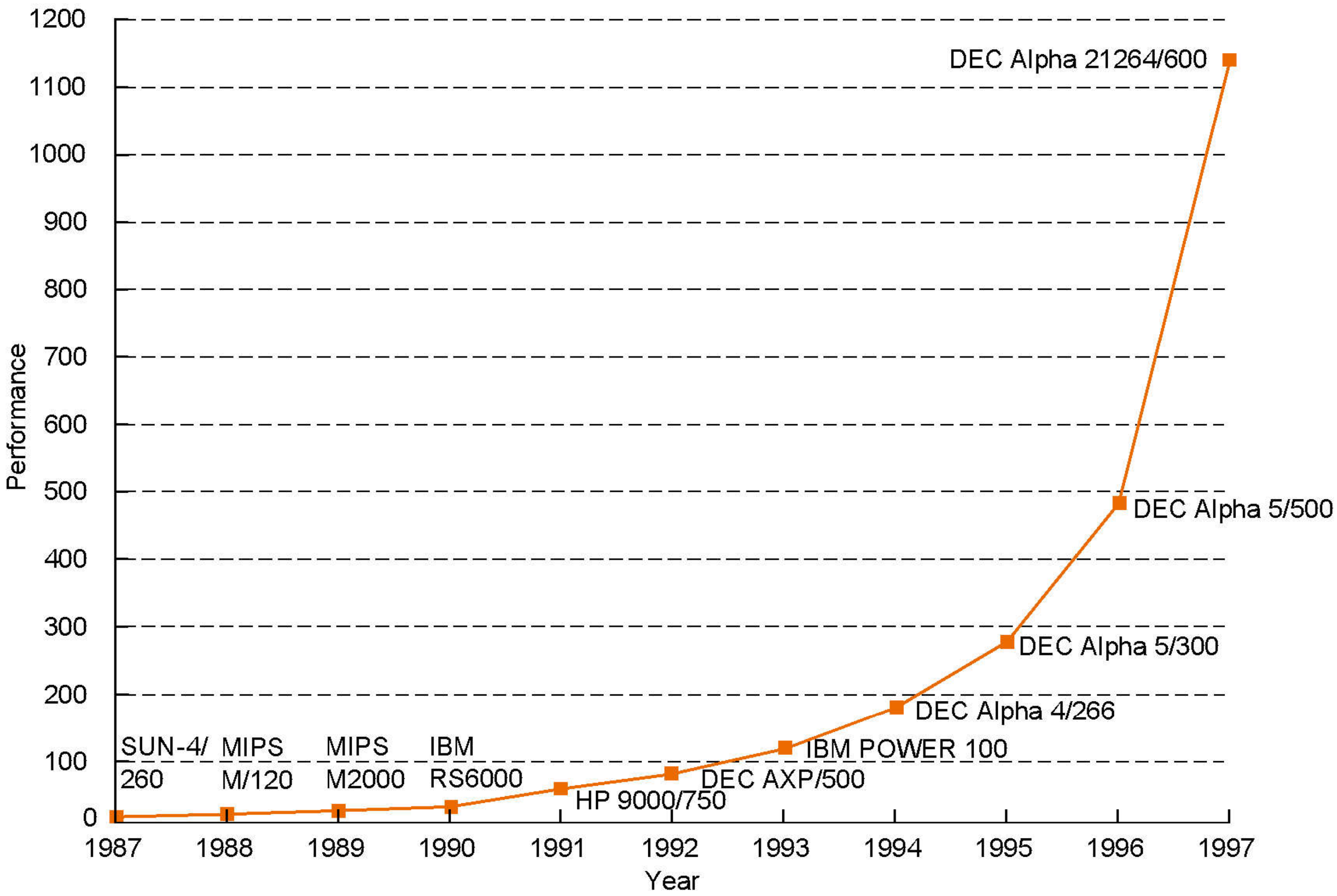
Computer Organization

- Once you have decided on an ISA, you must decide how to design the hardware to execute those programs written in the ISA as fast as possible (or as cheaply as possible, or using as little power as possible, …).
- This must be done every time a new implementation of the architecture is released, with typically very different technological constraints.

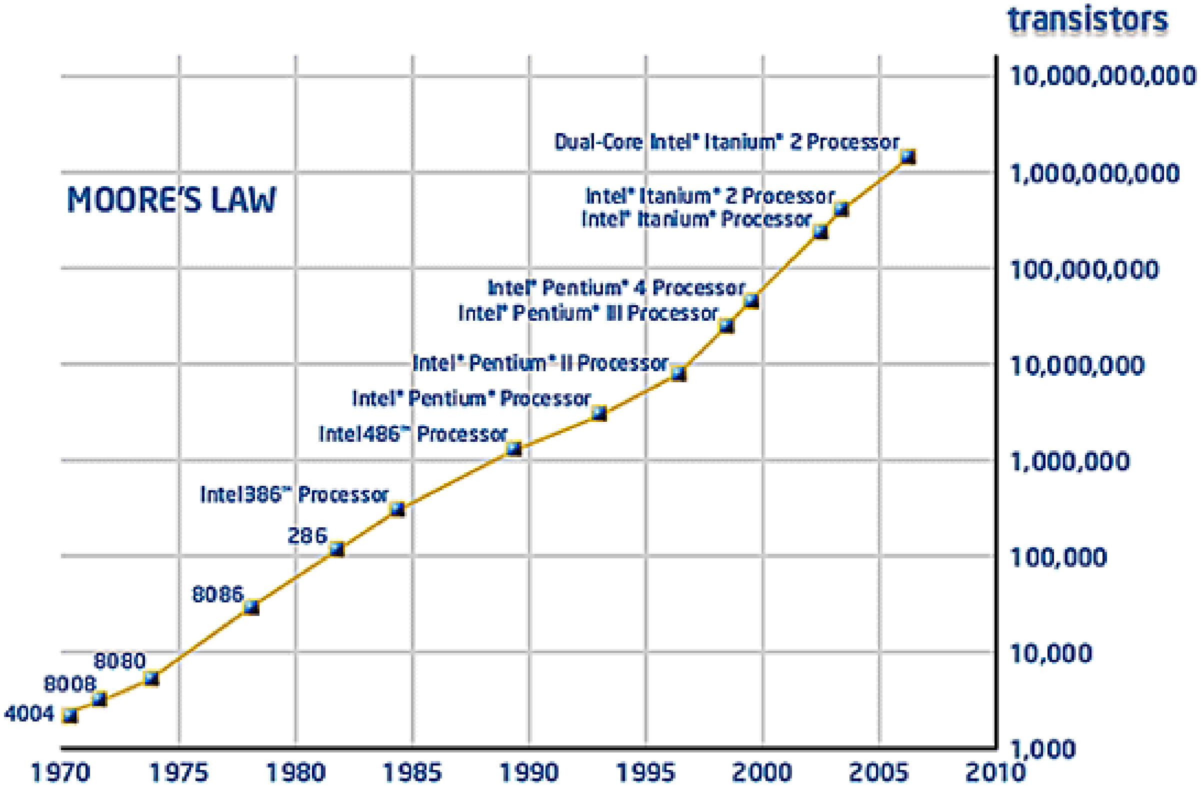
The Challenge of Computer Architecture

- The industry changes faster than just about any other.
- The ground rules change every year.
  - new problems
  - new opportunities
  - different tradeoffs
- It's "all" about making programs run faster than the next person's machine. Or more efficiently. Or more secure.

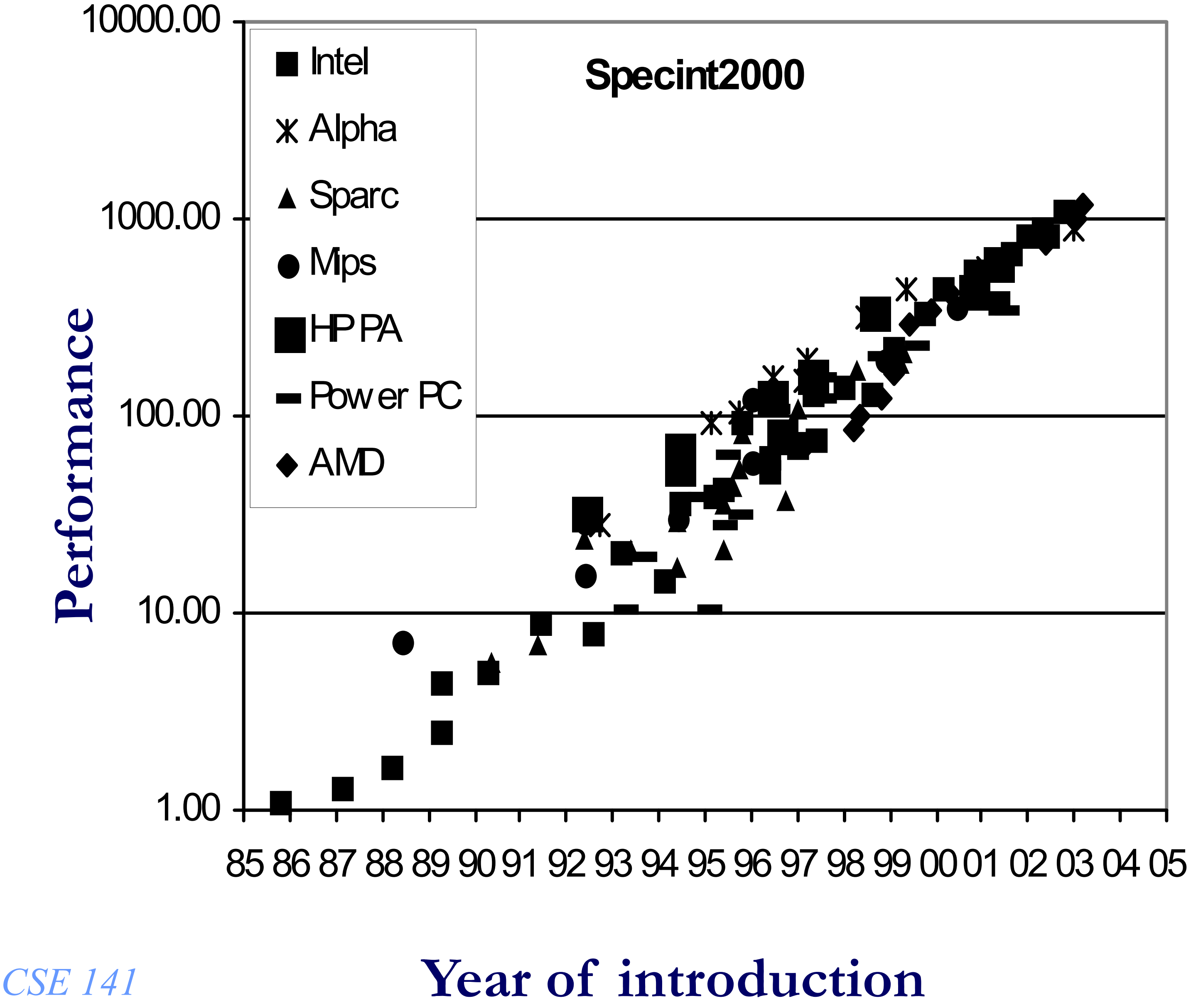
Performance Trends



Transistor Counts



Processor Performance with Time



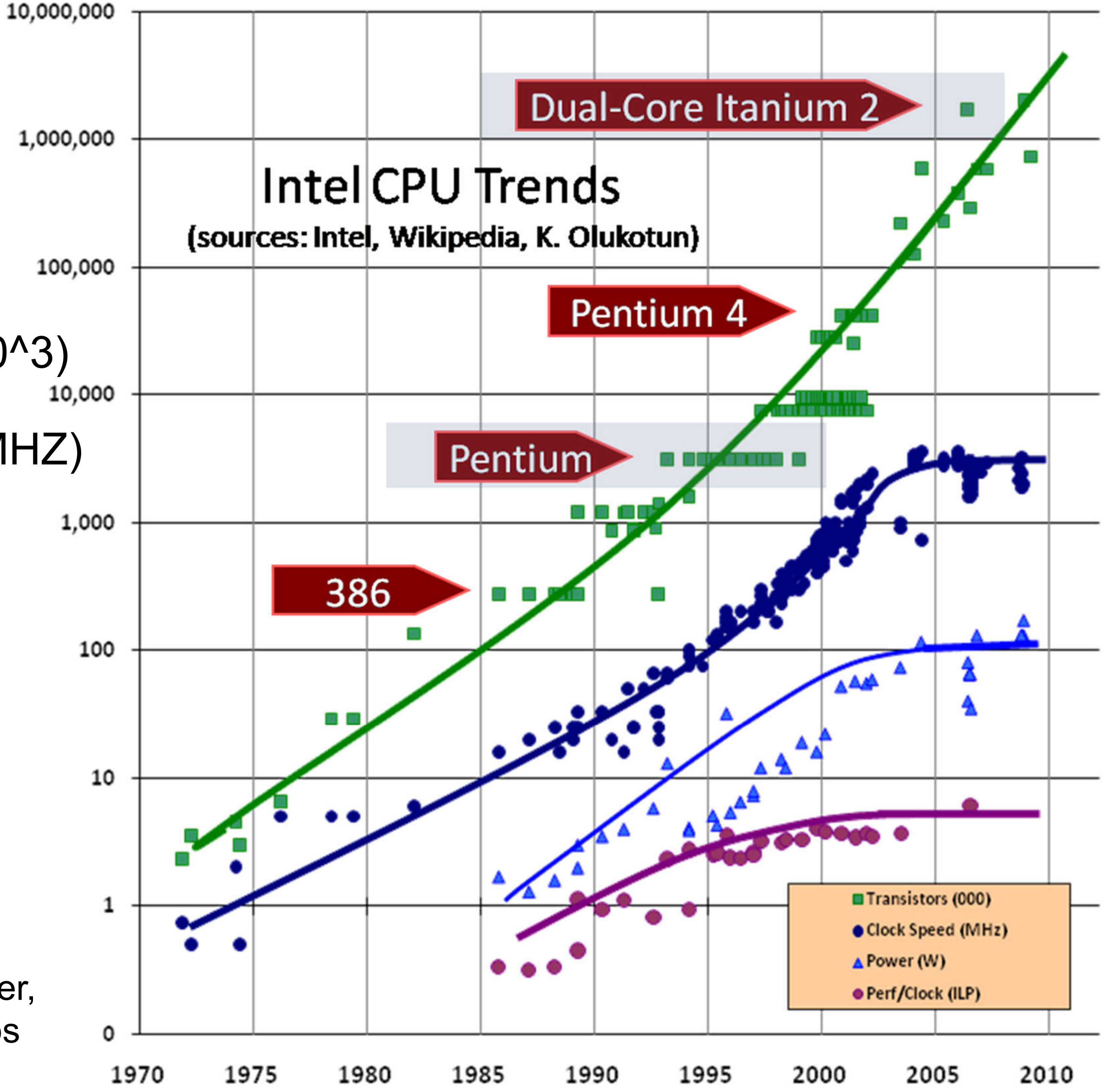
Processor Design Trends

🟩 Transistors (*10^3)

■ Clock Speed (MHZ)

■ Power (W)

■ ILP (IPC)



*From Herb Sutter, Dr. Dobbs Journal

What went wrong (2005)

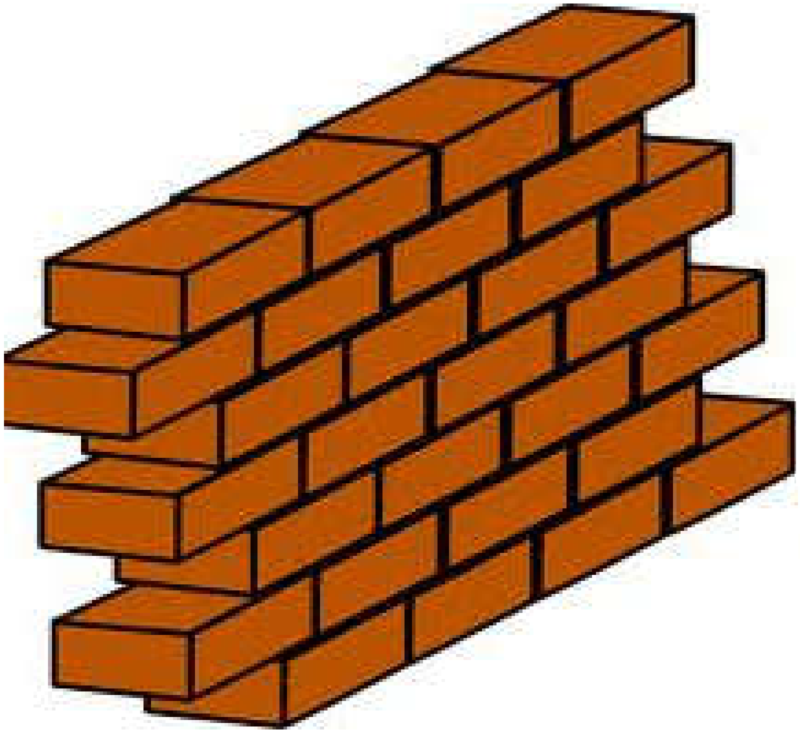
Power wall

Thermal wall

ILP wall

+ Memory wall

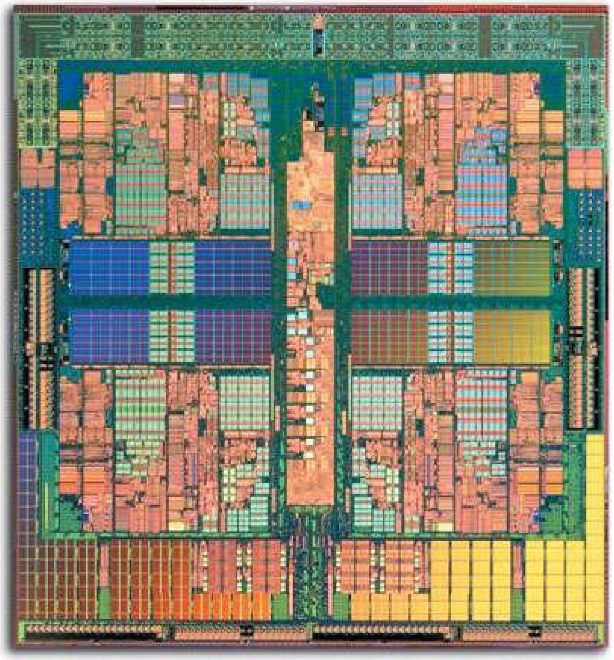
= Brick Wall



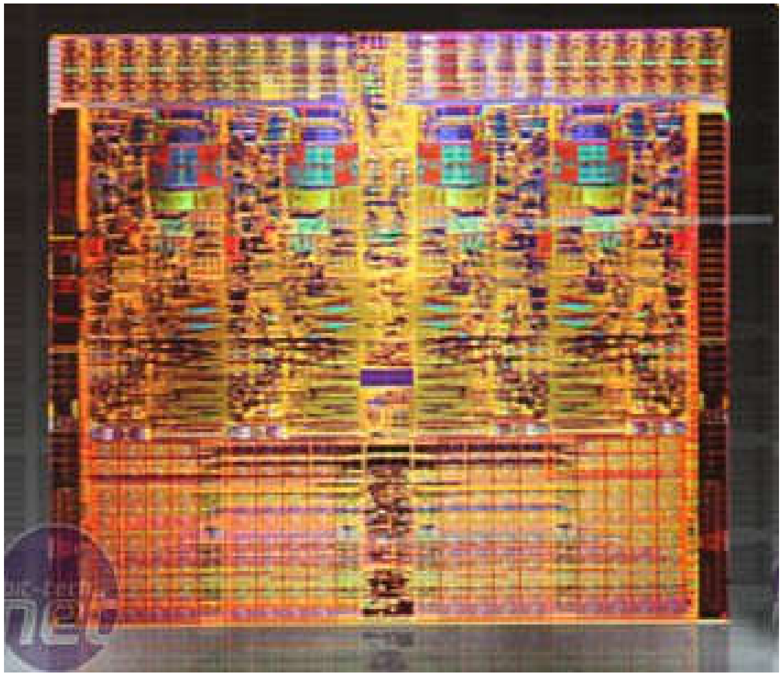
*The Landscape of Parallel Computing Research: A View from Berkeley

All is not lost

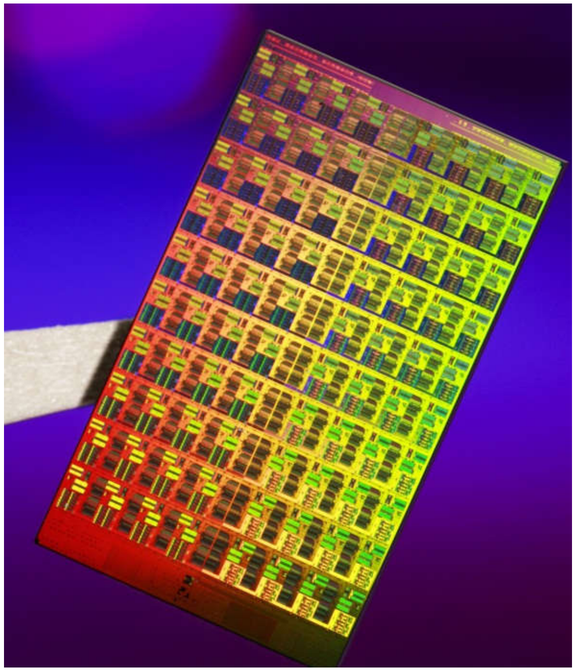
- Refocus on thread throughput over latency
  - Simultaneous Multi-Threading (SMT)
  - Single Chip Multi-Processor (CMP)



Intel Quad Core



Intel Nehalem



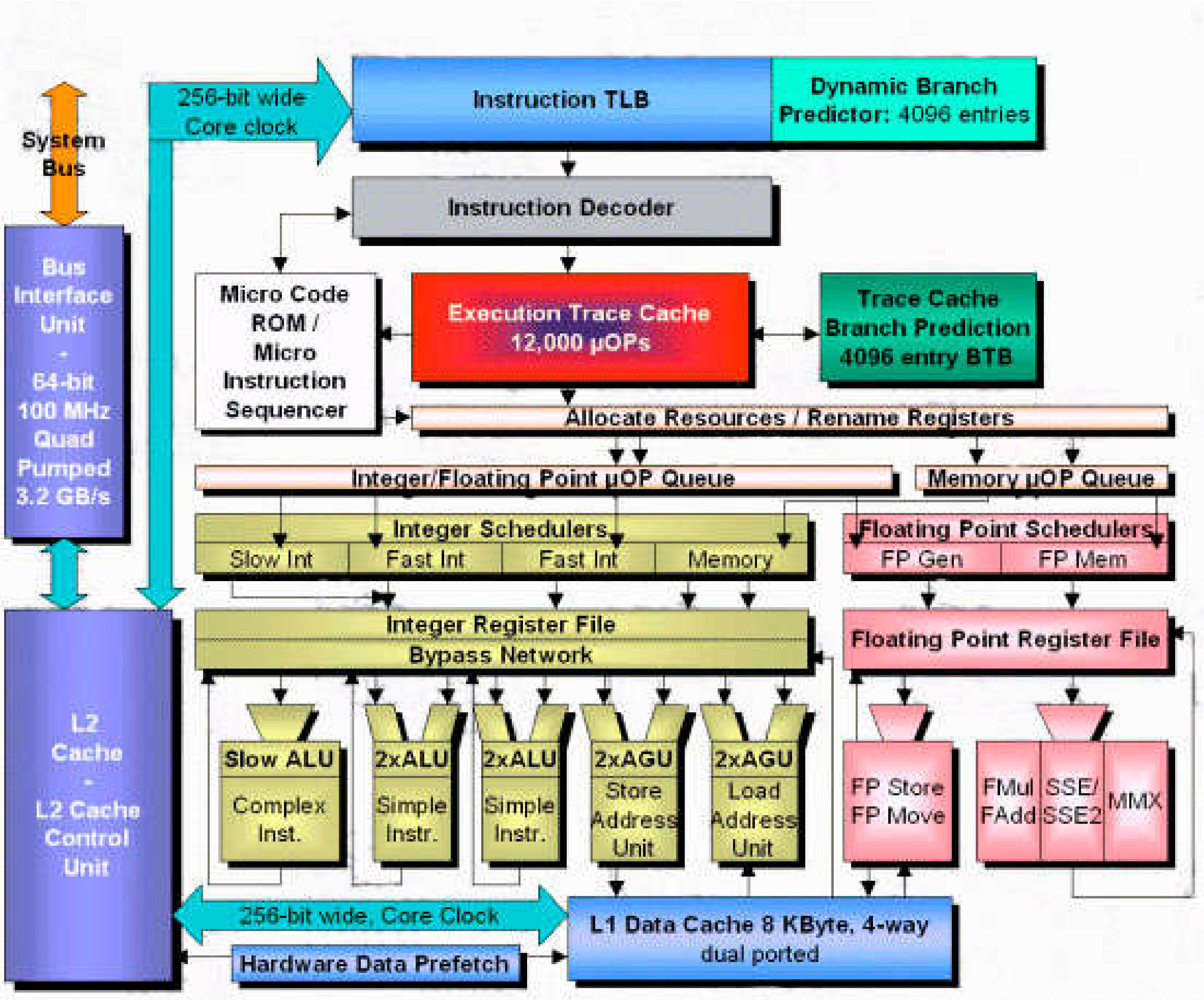
Intel: 80-core prototype

All is not lost

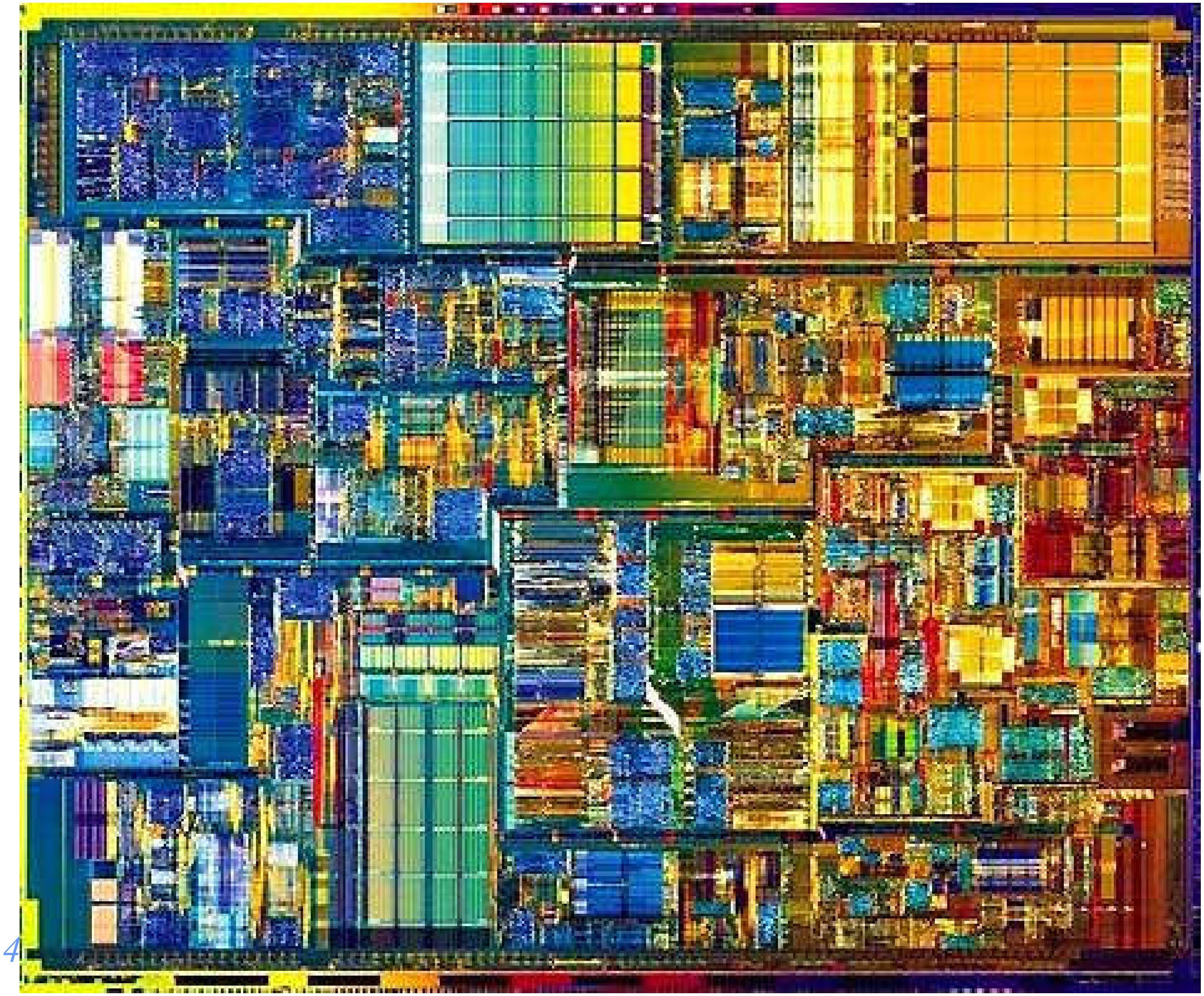
- Refocus on thread throughput over latency
  - Simultaneous Multi-Threading (SMT)
  - Single Chip Multi-Processor (CMP)

- But also
  - More focus on architectures for specific problems, rather than general architectures.
  - E.g., graphics, ML, security, networking…

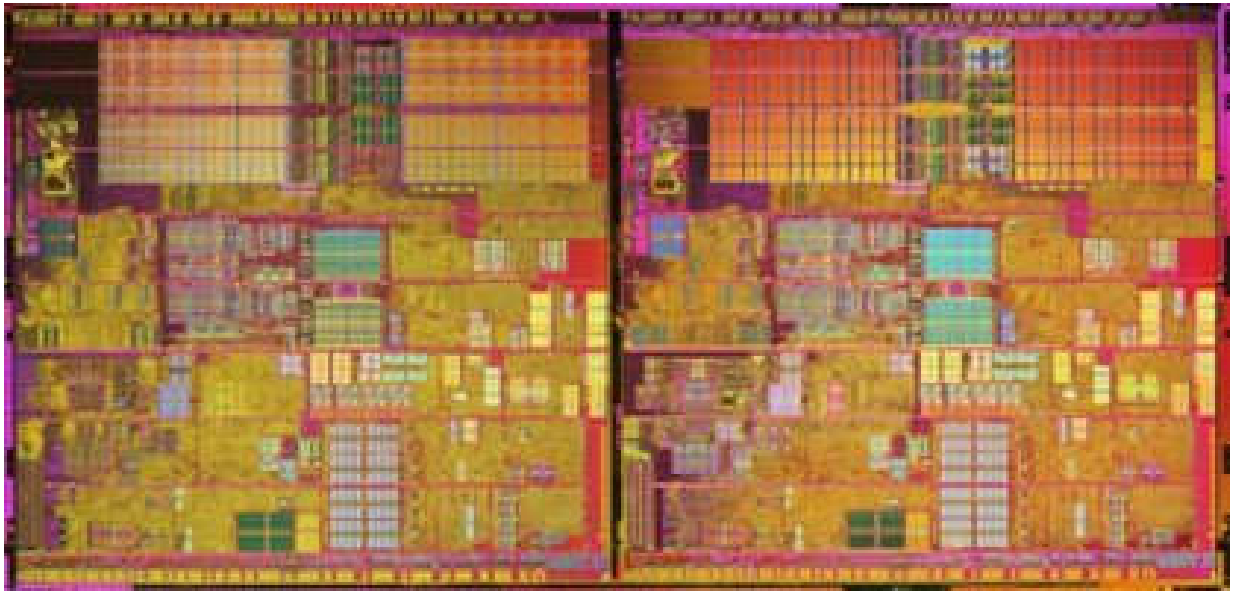
Pentium 4


System Bus
Bus Interface Unit - 64-bit 100 MHz Quad Pumped 3.2 GB/s
256-bit wide Core clock
Instruction TLB
Dynamic Branch Predictor: 4096 entries
Instruction Decoder
Micro Code ROM / Micro Instruction Sequencer
Execution Trace Cache 12,000 µOPs
Trace Cache Branch Prediction 4096 entry BTB
Allocate Resources / Rename Registers
Integer/Floating Point µOP Queue
Memory µOP Queue
Integer Schedulers
Slow Int
Fast Int
Fast Int
Memory
Floating Point Schedulers
FP Gen
FP Mem
Integer Register File
Bypass Network
Floating Point Register File
Slow ALU Complex Inst.
2xALU Simple Instr.
2xALU Simple Instr.
2xAGU Store Address Unit
2xAGU Load Address Unit
FP Store FP Move
FMul FAdd
SSE/ SSE2
MMX
L2 Cache - L2 Cache Control Unit
256-bit wide, Core Clock
Hardware Data Prefetch
L1 Data Cache 8 KByte, 4-way dual ported

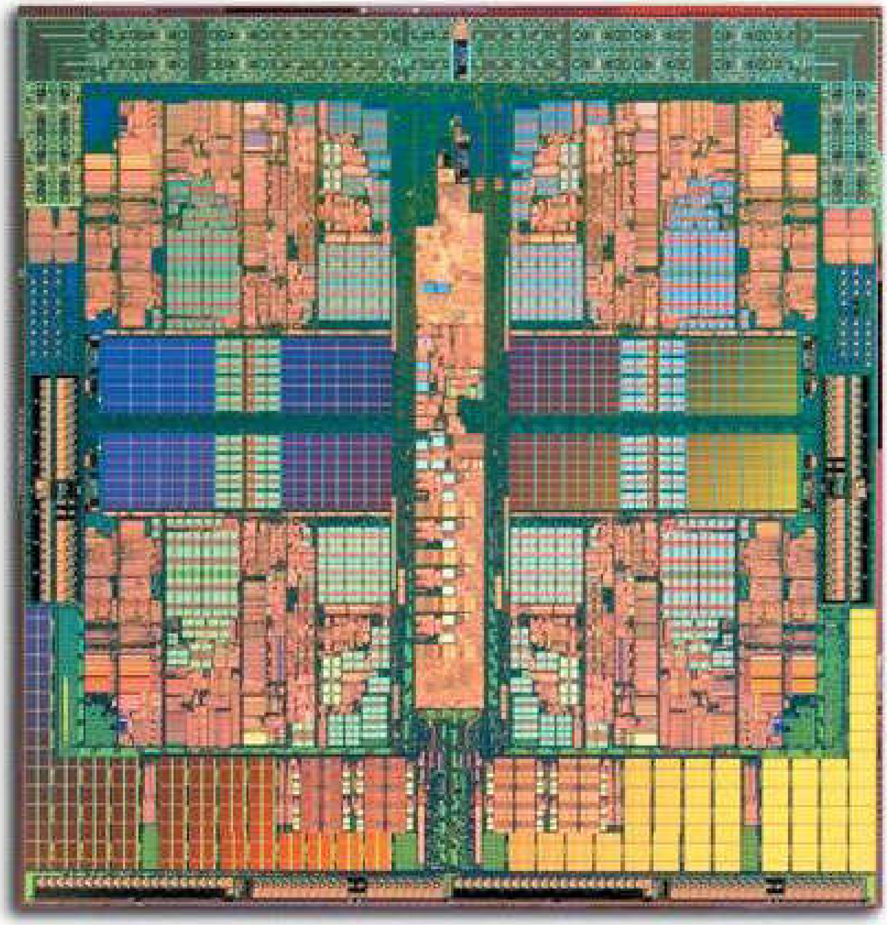
Pentium 4



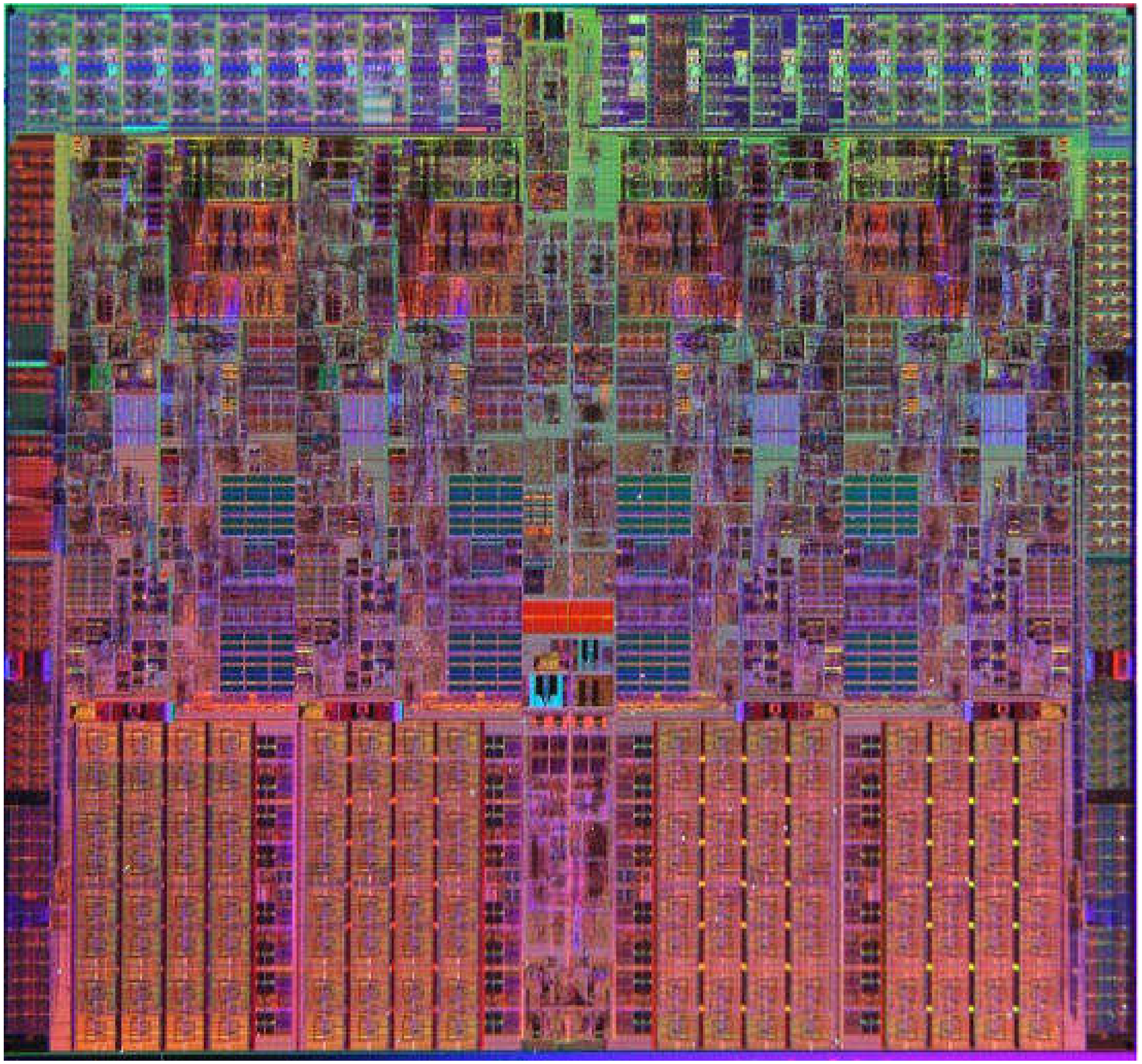
Intel Core Duo



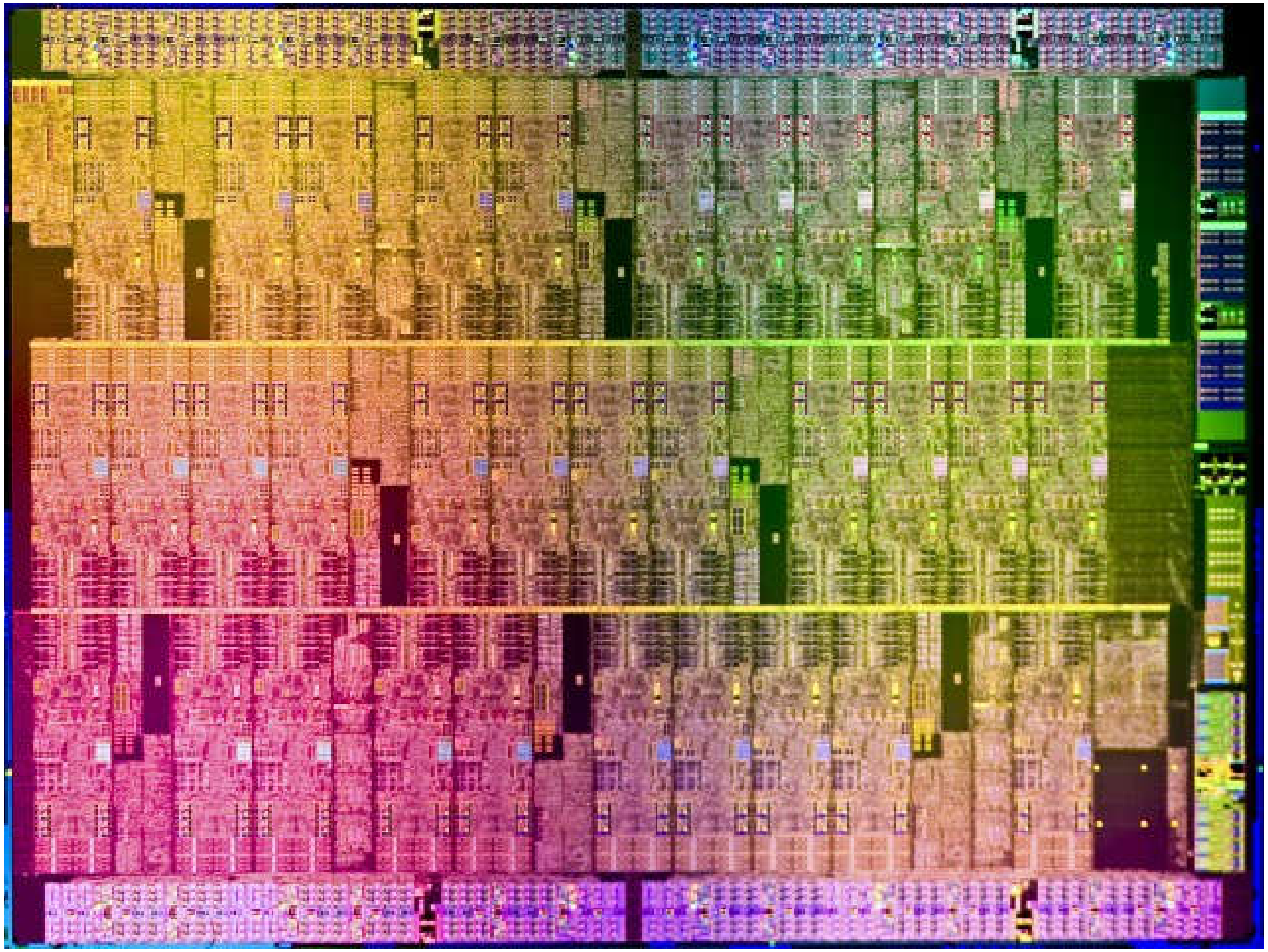
Quad-Core Opteron



Intel Nehalem
4 core, simultaneous multithreading



Intel Knight’s Landing (72 core)



Course Outline

I. Instruction Set Architecture

II. Computer System Performance and Performance Metrics

III. Computer Arithmetic and Number Systems

IV. CPU Architecture

V. Pipelining

VI. Superscalars

VII. The Memory/Cache Hierarchy

VIII. Parallel Machines

A little more about computer architecture at UCSD.

- UCSD has an amazing team of architecture faculty

Brief (and oversimplified) overview of the UCSD Architecture Research Groups

Steven Swanson

- Non-volatile memory technologies (persistent storage)
  - Architectures for
  - System software for
  - Problems with real hardware…

Jishen Zhao

- New memory technologies, and how to architect for them and exploit them
- Specialized architectures for machine learning

Hadi Esmaeilzadeh

- Approximate Computing
- Architectures and systems for machine learning
  - Languages for
  - Systems for automating design/architecture/software for…

Pat Pannuto

- Really small systems
  - Energy harvesting
  - Intermittent power
- Carbon footprint based designs

Yiying Zhang

- Architectures for big disaggregated systems (eg data-center level)

Dean Tullsen

- Parallel architectures
  - Simultaneous multithreading (virtually any hi-perf processor today)
  - Heterogeneous architectures (ARM big.Little, Intel Alder Lake, Apple M1, M2, …)
- More recently
  - Secure computer architectures
    - New attacks
    - New attack-tolerant architectures

What you can expect to get out of this class

- to become conversant with computer architecture terms and concepts.
- to understand fundamental concepts in computer architecture and how they impact computer and application performance.
- to be able to read and evaluate architectural descriptions of even today’s most complex processors.
- to gain experience designing a working CPU completely from scratch.

Key Points

- High-performance software requires a deep understanding of the underlying machine organization.
- The instruction set architecture defines how software is allowed to use the processor. Multiple computers with vastly different organizations and performance can share an ISA.
- The era of single-threaded computing is over. We have now entered the world of multi-core architectures. But fundamental core design principles remain largely unchanged.