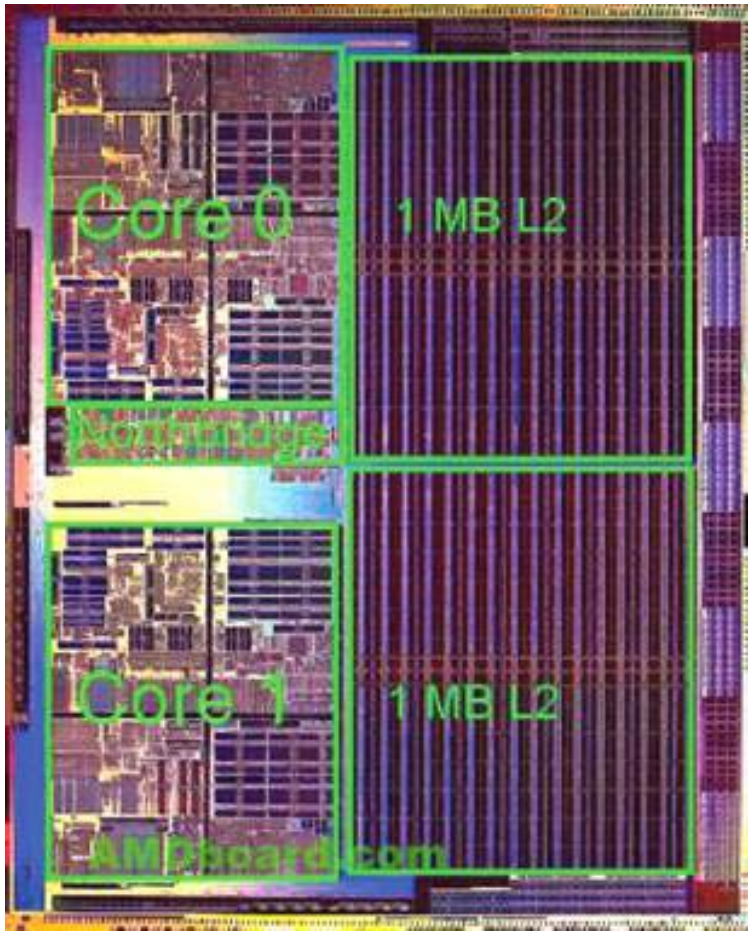
CS232: Computer Architecture II

Spring 2010



AMD dual-core Opteron

Who we are

- *Lecturer*:

  Prof. Viraj Kumar kumar@illinois.edu

  Visiting Lecturer

  Office hours: Friday 4pm to 5pm and by email, 2211 SC

- *Teaching Assistants/Section Instructor*: Room 0212 SC

Ryan Cunningham	rcunnin2@illinois.edu	**Thu. Pm (lab)**
Abner Guzman Rivera	aguzman5@illinois.edu	**Thu. Pm (lab)**
Arushi Aggarwal	aggarwa4@illinois.edu	**Fri. 10-11am**
Pritam Sukumar	sukumar2@illinois.edu	**Thu. 10-11am**

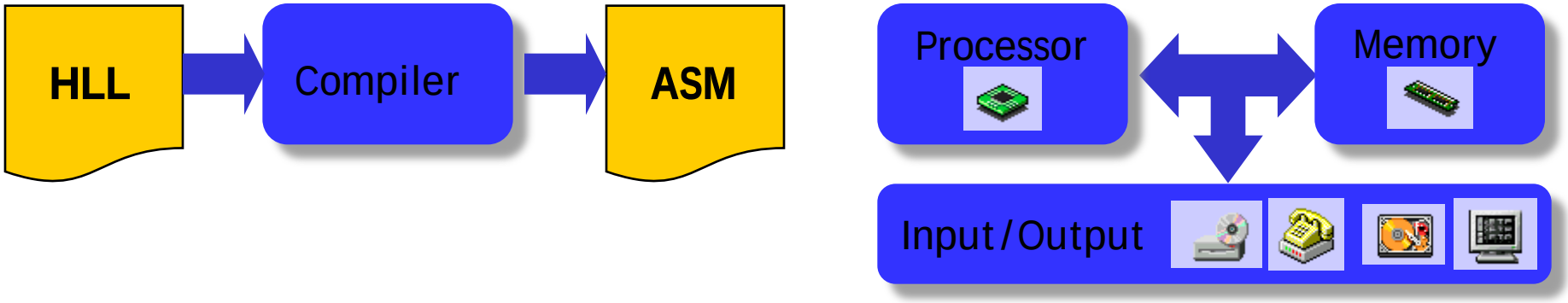
- MPs released on Friday, (usually) due next Friday

Graded work

- Five or six MPs, builds towards SPIMbot Tournament, 25% of grade
  - You can work individually, or in groups of 2 or 3
  - Submit something that can be tested by the deadline
  - I'll email you feedback (using an auto-grader) within 24 hours
  - You can resubmit for full credit 48 hours after the deadline
- Three Wednesday evening Midterms, 15% each
  - Exam 1: 2/24 ; Exam 2: 3/17 ; Exam 3: 4/21 (tentative)
- Final, cumulative, date to be decided: 25%
- Section attendance: 5%

What is computer architecture about?

- Computer architecture is about building and analyzing computer systems



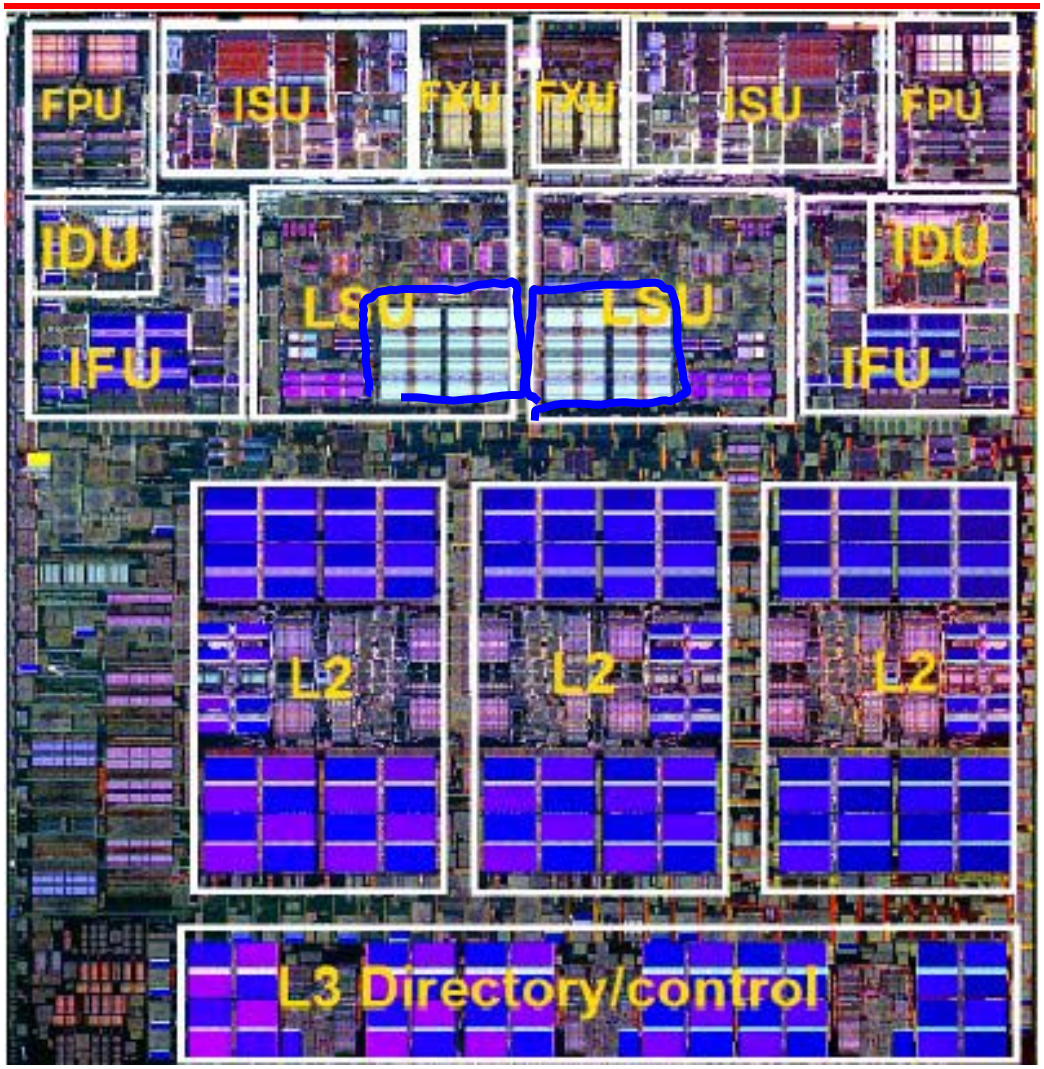
- Instruction Set Architecture is bridge between hardware and software
  - Study the MIPS ISA in detail
  - Learn what compilers do when they translate high-level code into assembly (we won't learn *how* they do it)
  - Learn how HLL program constructs are represented to the machine
- Key techniques: Pipelining, Caching, Virtual Memory
- Tuning complex code for performance (course project)
- Exploiting parallel

Hey Prof. Kumar, Today I interviewed at Microsoft. I referenced spimbot and used concepts learned in class multiple times. I just wanted to say THANKS!

Multi-Core Processors

- Two (or more) complete processors, fabricated on the same silicon chip
- Execute instructions from two (or more) programs/threads at same time

#1



IBM Power5

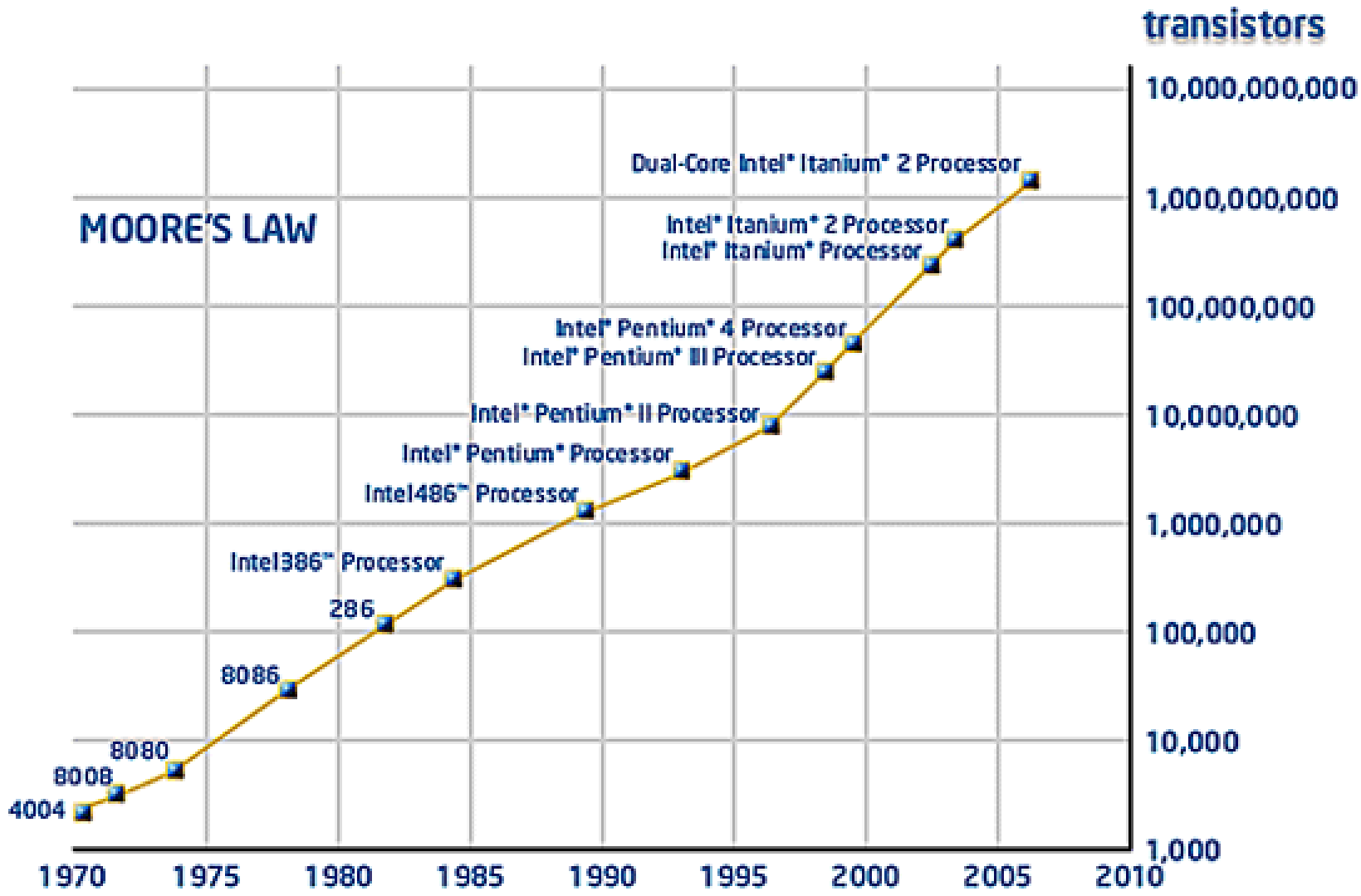
#2

XBox360: 3 PowerPC cores

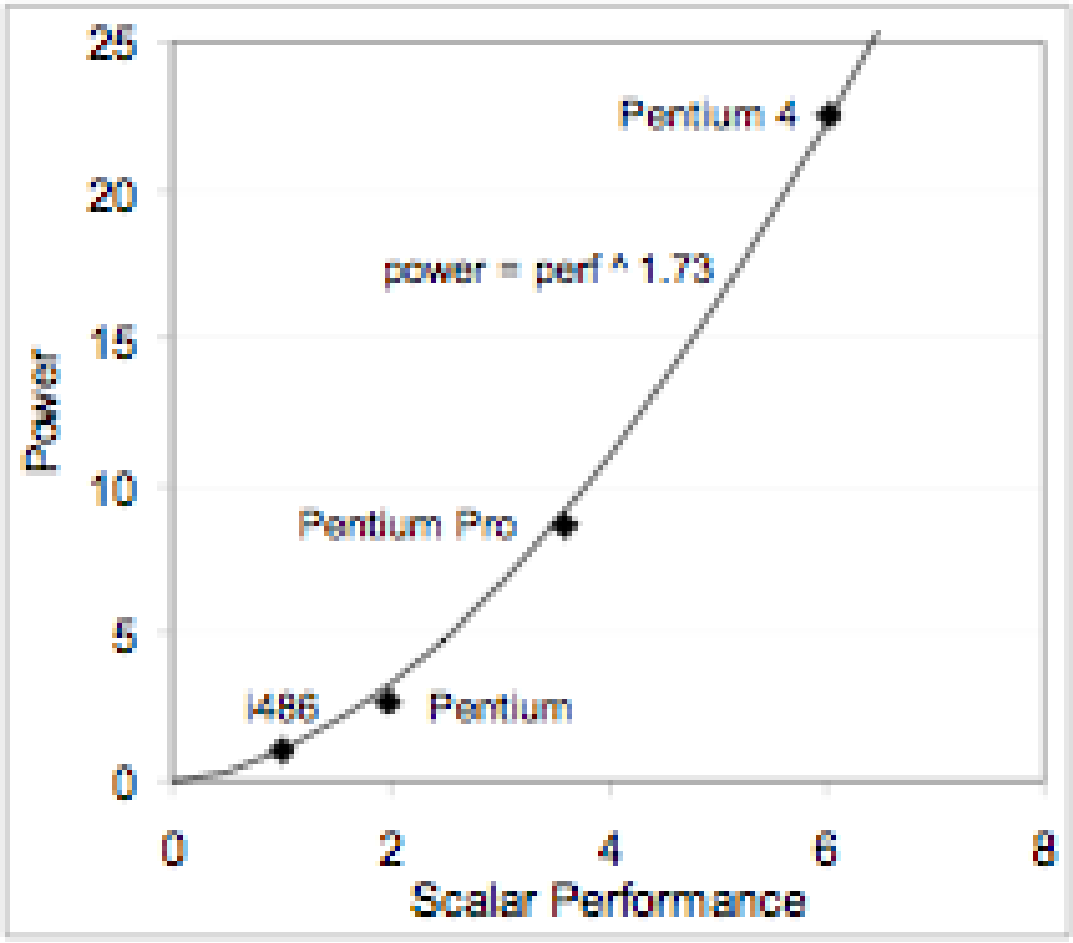
Sony PS 3: asymmetric 9 cores

Why Multi-cores Now?

- Number of transistors we can put on a chip growing exponentially



Performance vs. Power trade-off

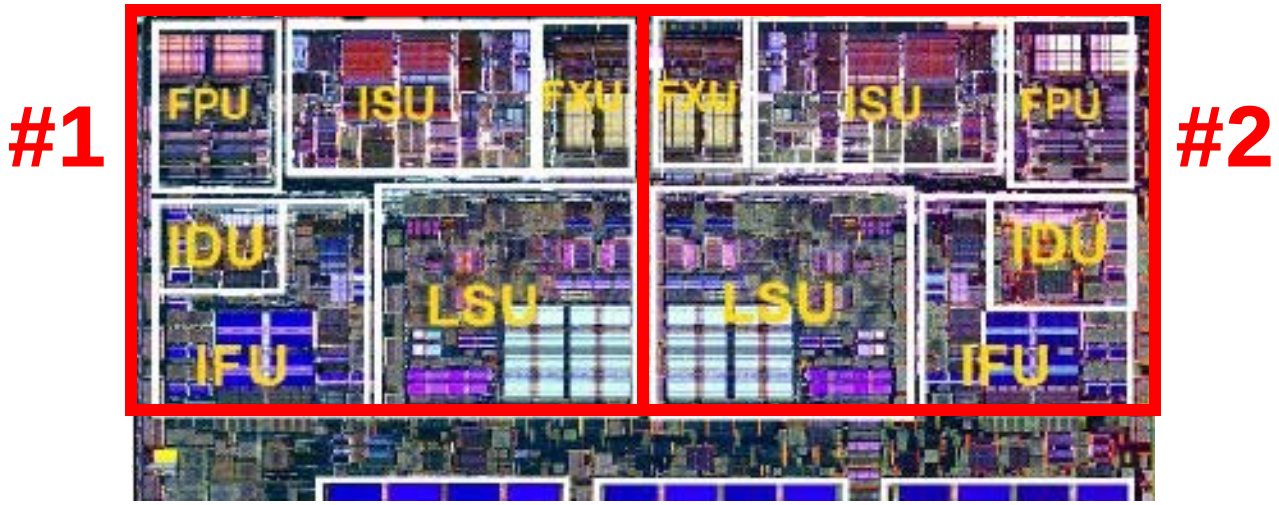


- Power has become a limiting factor for single cores
  — hence multi-cores

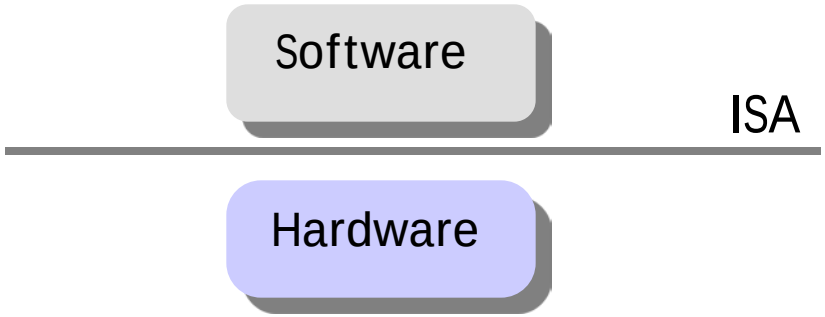
As programmers, do we care?

- What happens if we run a program on a multi-core?

```
void array_add(int A[], int B[], int C[], int length) {
  int i;
  for (i = 0 ; i < length ; ++i) {
    C[i] = A[i] + B[i];
  }
}
```



Instruction set architectures

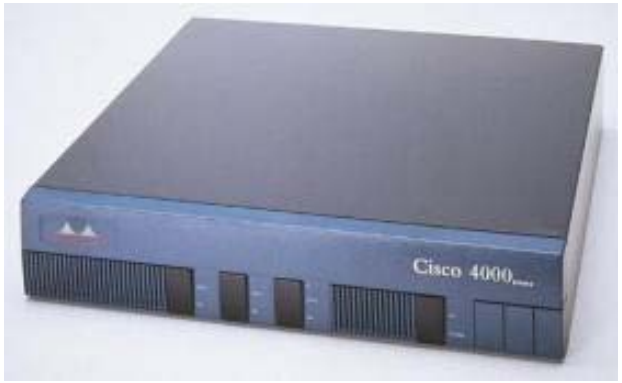


- The ISA is an interface between software and hardware
  - the hardware "promises" to implement all ISA instructions
  - the software uses ISA primitives to build complex programs
- The instruction set architecture affects the hardware design
  - simple ISAs require simpler, cheaper processors
- Also affects software design
  - simple ISAs result in longer programs

Why MIPS?

- We study the MIPS instruction set architecture to illustrate concepts in assembly language and machine organization
  - concepts are not MIPS-specific
  - MIPS is just convenient because it is real, yet simple (unlike x86)

- MIPS ISA is used in many places, primarily in embedded systems
  - routers from Cisco
  - game machines like the Nintendo 64 and Sony Playstation 2



What you will need to learn for Exam 1

- You must become "fluent" in MIPS assembly:
  - Translate from C++ to MIPS and MIPS to C++

- Example: Translate the following recursive C++ function into MIPS

```
int pow(int n, int m) {
  if (m == 1)
    return n;
  return n * pow(n, m-1);
}
```

How are arguments passed?

How are values returned?

How are complex expressions broken into simple instructions?

How is recursion done?

MP 1: Gray codes

- A binary representation of integers, where successive integers differ in exactly one bit
  - the standard representation does not have this property

- For an integer n, let (n) denote the binary representation of n

- The gray-code representation of n is : (n) $\oplus$ ($\lfloor n/2 \rfloor$)
  - here, "$\oplus$" is bit-wise XOR and $\lfloor\ \rfloor$ is the floor function

- *Example*: gray-code(6) =

MP 1: Gray codes contd.

- The gray-code representation of n is : $(n) \oplus (\lfloor n/2 \rfloor)$
- Note that $\lfloor n/2 \rfloor = n >> 1$ (right-shift)
- *Example*: gray-code(6) =