

Call Number 1911 (Spring 2004)

URL: <http://www.ece.lsu.edu/ee4720>

Offered by:

David M. Koppelman

349 EE Building

578-5482, koppel@ece.lsu.edu, <http://www.ece.lsu.edu/koppel>

Tentative office hours: Monday - Friday, 9:00-10:00.

Teaching Assistant:

To be announced.

Should already know:

How to design a computer.

Will learn:

How to design a *good* computer.

Prerequisites By Course:

EE 3755, Computer Organization.

Prerequisites By Topic:

- Logic design.
- Computer organization.
- Assembly-language programming.

Text

“Computer architecture, a quantitative approach,” John L. Hennessy & David A. Patterson,
or “Computer organization & design,” David A. Patterson & John L. Hennessy.

Course Content

- Instruction set design.
- Pipelined processor design.
- Multiple-issue processor design.
- Caches and memory.

Topics cover modern general-purpose microprocessors.

Midterm Exam, 40%

Fifty minutes, closed book.

Final Exam, 40%

Two hours, closed book.

Yes, it's cumulative.

Homework, 20%

Written and computer assignments.

Lowest grade or unsubmitted assignment dropped.

Material in Course Needed For:

- General-purpose processor designers and testers.
- Special-purpose processor designers and testers.
- Compiler writers.
- Programmers of high-performance systems.
- Answering job interview questions.

Coverage: Sections 1.2 and 1.3.

Slides and other material via <http://www.ece.lsu.edu/ee4720>

Web site also has homework assignments, exams, grades, and other material.

What is a computer?

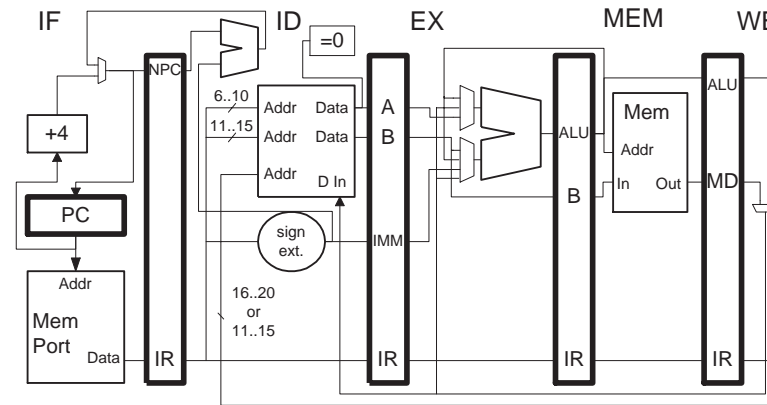
A machine that executes instructions which read and write memory.

What a computer engineer does:

- Develops an *instruction set architecture* (ISA):

```
ld  r1, 0(r2)  ! Load register r1 with contents of mem at r2.
add r3, r1, r4
sw  0(r2), r3
```

- Designs hardware to execute, *implement*, the instruction set:



Definitions

Instruction Set Architecture (ISA):

Precise definition of computer's instructions and their effects.

- It's all programmer needs to program machine.
- It's all hardware designer needs to design machine.

Implementation: [of an ISA] (noun)

Hardware that executes instructions defined by ISA.

Architecture: IBM *System/360*

Developed in 1964 for large business computers.

Designers appreciated and popularized the difference between architecture and implementation.

First planned family of computers.

Very successful, successor machines still in use under name z/Architecture.

First Implementations: Model 30, Model 75.

Architecture: Intel *IA-32*

Initially developed in 1978 for small systems.

First processor: 8086, implements small part of IA-32.

Major improvements in amount of memory addressable by subsequent chips, 80186, 80286 (1982).

The 80386 (1985) could host a modern 32-bit operating system.

Later chips implemented ISA extensions for multimedia and data movement ...
... and continued to incorporate microarchitectural innovations.

80486 (1989), Pentium (1992), Pentium Pro (1995), Pentium II (1997), Pentium III (1999)
Pentium 4 (2000).

Unlike System/360, the way it would be used was not foreseen.

Includes unpopular features, such as small memory segments.

Nevertheless, implementations have competed well with modern ISAs.

Architecture: DEC (then Compaq, now HP) *Alpha*

An example of a *RISC* processor.

Designed for easy programming.

Designed for easy implementation.

RISC programs are larger than others, but run faster.

Developed for a 25-year lifetime.

First implementation: DECchip 21064 (1992).

Later implementations: 21264, 21364.

Implementations are usually the fastest processors.

Alas, Compaq plans to discontinue it.

Architecture: *Itanium* (IA-64)

First general purpose *VLIW* ISA.

ISA helps processor overcome problems in turn-of-the-century processors.

First implementation: Itanium (same name as architecture) (2000).

Radically different from other processors.

So far unproven.

Who ISA Developed For

- Compiler writers.
- Compute-intensive library writers.
E.g., graphics and scientific libraries.

Instruction set requirements don't change very much over time.

Scope of ISA Specification

Describes instruction codings, and what they should do.

Should specify action of all codings, used or not ...

Two aspects of implementation: *organization* and *hardware*.

Organization:

Details of functional units, data paths, control, etc.

Also called *microarchitecture*.

This includes memory system, bus, and CPU.

Technology (Hardware):

Logic design and packaging.

Course focus: ISA and organization, not hardware.

Principles computer designers apply widely.

- Make the common case fast.

Obviously.

(Not covered.)

- *Amdahl's Law*: Don't make common case too fast.

As speed of one part increases...

...impact on total performance drops.

(Not covered.)

- *Locality of Reference*.

Temporal: It might happen again soon.

Spatial: It might happen to your neighbors soon too.

The first two principles are “common sense”.

However, locality is a characteristic of executing programs ...

... which has held and is expected to continue to hold.

Because many designs work only when locality is present ...

... if it all of a sudden programs did not exhibit locality ...

... computers would run them many, many times slower!

Locality usually applied to memory addresses issued by processor.

Temporal: there's a good chance that an address used will be used again soon.

Spatial: once an address is used there's a good chance a nearby address will be used.

For examples, analyze execution of almost any program.

ISA and Implementation

Know the definitions.

Understand the difference, why at one time the distinction was not made, and how making a distinction is useful for a product line of computers.

List examples of ISAs and implementations.

Define organization and technology.

Principles of Computer Design

Understand what “make the common case fast” means, and give examples of who should be reminded of this obvious rule.

Vaguely define Amdahl’s law (don’t need to know the formula).

Define locality, temporal locality, and spatial locality. Understand that these characteristics apply in several different situations, the primary one being memory addresses.