EE 4720—Computer Architecture

Call Number 8286 (Fall 2001)

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/koppel.html

Tentative office hours: Mon 15:00-16:00; Wed 9:30-10:30; Tue & Thr 14:00-15:30.

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, Second Edition.

EE 4720 Lecture Transparency. Formatted 10:47, 27 August 2001 from lsli01.

Course Content

- Instruction set design.

- Pipelined processor design.

- Multiple-issue processor design.

- Caches and memory.

Course content will follow text.

EE 4720 Lecture Transparency. Formatted 10:47, 27 August 2001 from lsli01.

Graded Material

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.

Course Usefulness

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.

EE 4720 Lecture Transparency. Formatted 10:47, 27 August 2001 from lsli01.

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.
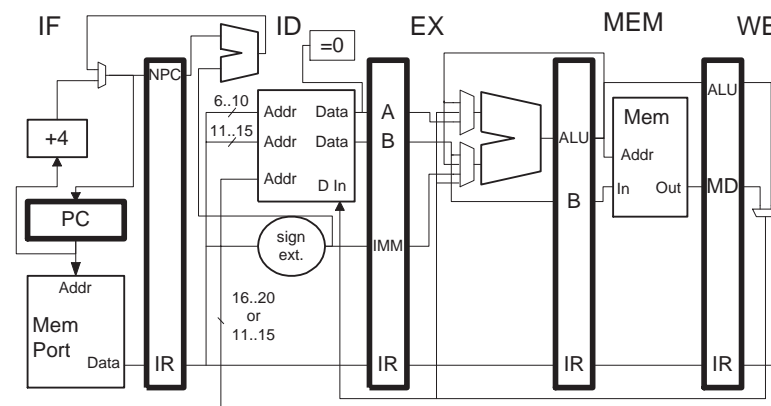
What a computer engineer does:

• 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.

ISA and Implementation Examples

ISA: SPARC International SPARC V9

Impl: Sun Microsystems UltraSPARC-II, UltraSPARC-III

ISA: Intel IA-32

Impl: Pentium III, Pentium 4.

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 . . .

Implementation

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

Organization:

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

Also called *microarchitecture*. (NIB[1]).

This includes memory system, bus, and CPU.

Hardware:

Logic design and packaging.

Course focus: ISA and organization, not hardware.

---

[1] Not in book.

EE 4720 Lecture Transparency. Formatted 10:47, 27 August 2001 from lsli01.

Some Architectures and their Implementations

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.

First Implementations: Model 30, Model 75.

EE 4720 Lecture Transparency. Formatted 10:47, 27 August 2001 from lsli01.

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 forseen.

Includes unpopular features, such as small memory segments.

Nevertheless, implementations have competed well with modern ISAs.

EE 4720 Lecture Transparency. Formatted 10:47, 27 August 2001 from lsli01.

DEC (now Compaq) *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.

Intel IA-64

First general purpose *VLIW* ISA.

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

First implementation: Itanium (2000).

Radically different from other processors.

So far unproven.

EE 4720 Lecture Transparency. Formatted 10:47, 27 August 2001 from lsli01.

Technological Change

Technological Change and Computer Designer

Technology determines "raw materials" for designer.

Raw material: number of gates and their speed.

ISA lifetime can be decades.

Raw materials greatly change over this time.

So, design ISA for now and future.

EE 4720 Lecture Transparency. Formatted 10:47, 27 August 2001 from lsli01.

How technological advancement affects processor.

Logic Speed, Clock Rate

No changes to organization or ISA.

Number of Transistors Available for Logic

Changes to organization and possible changes to ISA.

Memory Size

Change ISA to use larger address space.

Can use ISA having larger instruction codings.

Memory Speed Compared to Processor Speed

Include more sophisticated caching in organization.

Summary

What a computer engineer does:

- Develops an *instruction set* (ISA).

- Designs hardware to execute instruction set.

If instruction set *poorly* designed ...

... many instructions will not be used (wasting silicon) ...

... and instructions will execute slowly.

Why ISA design is surprisingly difficult:

- Hard to predict which instructions useful ...
  ... without writing and running software using instructions.

- Hard to predict which instructions fast ...
  ... in current *and future* technologies.