What is Computer Architecture?

- **Structure**: static arrangements of the parts
- **Organization**: dynamic interaction of the parts and their control
- **Implementation**: design of specific building blocks
- **Performance**: behavioral study of the system or of some of its components.

Alternate definition: Instruction Set Architecture (ISA)

Architecture is an **interface** between layers.

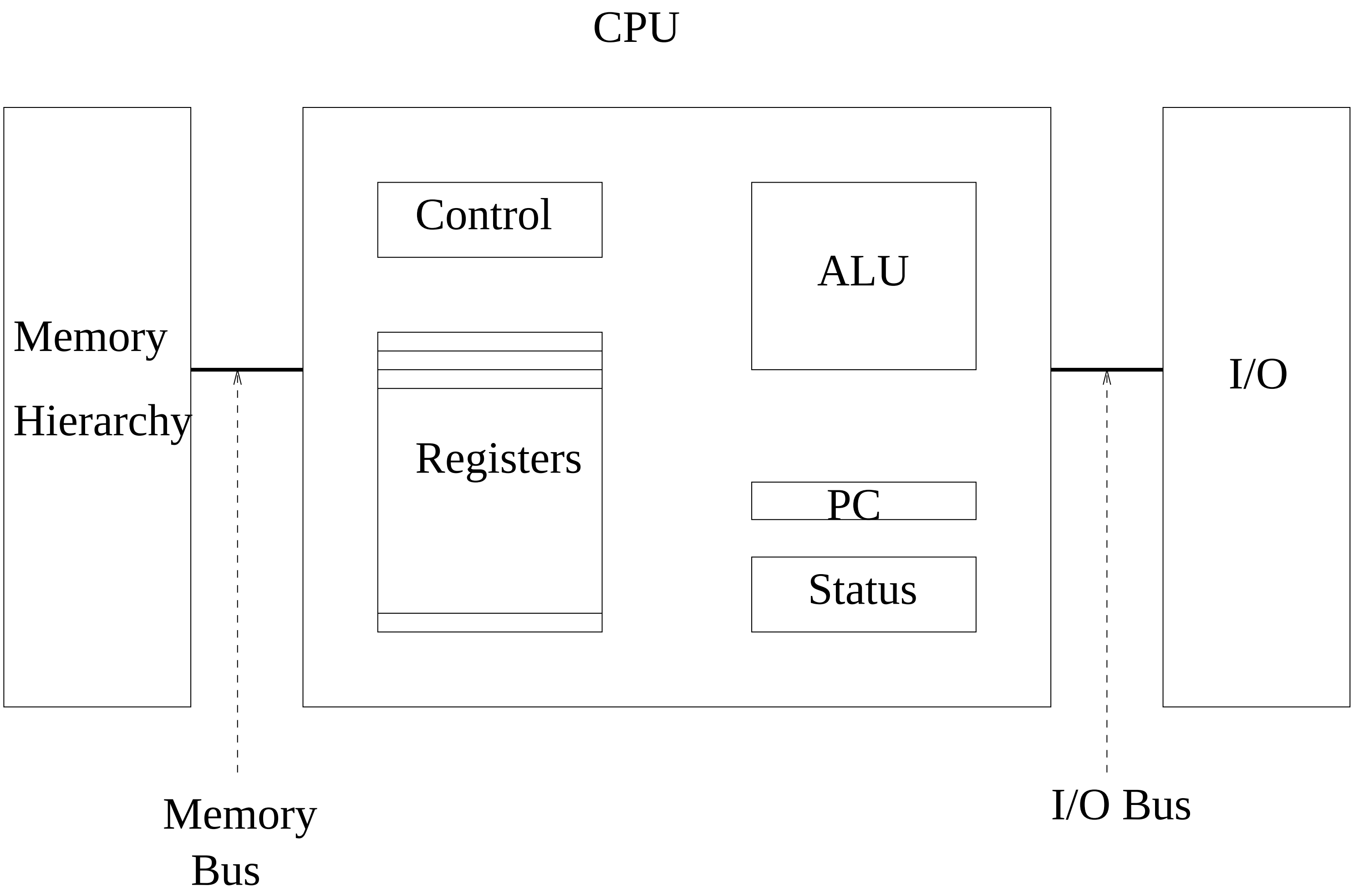
ISA is the interface between hardware and software.

ISA is what is visible to the programmer (and ISA might be different for O.S. and applications)

ISA consists of:

- instructions (operations, how are they encoded?)
- information units (what is their size, how are they addressed)
- registers (general or special-purpose?)
- input-output control

Computer Structure
Von Neumann model



Computer Organization

Organization and architecture often used as synonyms

Organization (in this course) refers to:

- what are the basic blocks of a computer system?
- more specifically
  - what are the basic blocks of the CPU
  - what are the basic blocks of the memory hierarchy
- how are the basic blocks designed, controlled, connected?

Organization used to be transparent to the programmer. This is becoming less true; now the "architecture is exposed" to the user/compiler.

Computer Generations

	1st	2nd	3rd	4th	5th	...
Processor Techn.	Vacuum tubes	transis.	integrated circuits	LSI	VLSI	Very VLSI
Processor structure	single proc.	multiple funct. units	micros and minis	workstat. and PC's	32-bit micros	64-b + MP micros
Memory	Vacuum tubes	core	semi-conductor	semicond. 64KB	semicond. 512KB	semicond. 64MB +

Computer Families

Sequence of machines that have the same ISA (binary compatible). For example:

1. IBM 704, 709, 70xx
2. IBM 360, 370, 43xx, 33xx etc..
3. IBM Power PC
4. DEC PDP-11, VAX
5. DEC Alpha
6. Intel x86 (Pentium, Pentium Pro)
7. MIPS Rx000, SGI
8. SUN SPARC

With portable software, is “binary compatible” important?

MIPS is a RISC

RISC = Restricted Instruction Set Computer

R could also stand for Regular. All arithmetic operations are of the form:

$$R_a \leftarrow R_b \; op \; R_c$$

MIPS is a Load-Store architecture. ALU operates only on registers

Registers

Bricks of the CPU. Visible to harware and to programmer.

1. High-speed storage for operands. For example, Assume that variables a, b, c are in registers 8, 9, 10 respectively

   ```
   add $8,$9,$10      #  a = b + c
   ```

2. Easy to name. Most current computers have 32 or 64 registers; So their names are 0, 1, 2 , etc.
3. Also used for addressing memory (see later)

Not all registers are "equal".

- Some are special purpose (e.g., register \$0 in MIPS is hardwired to 0).
- Some are used for integer and some for floating-point
- Some are restricted by convention (see Appendix A page A-22)

Why no more than 32 or 64?

- Well ... sometime there is (SPARC, CRAY, Tera)
- Smaller is faster
- Instruction encoding (the names have to be short)
- In fact there could be more registers but they are hidden from the programmer, hence not part of the ISA.

The Memory System

Memory is a *hierarchy* of devices with faster and more expensive ones closer to CPU

1. Registers
2. Caches (one or more levels)
3. Primary (or main) memory
4. Secondary memory (disks)

Memory level	Capacity(bytes)	Speed	Relative speed
Registers	100's to 1000's	nanoseconds	1
Cache	8-128K on-chip	nanoseconds	1-2
	1-4MB off-chip	10's of nanoseconds	5-10
Primary memory	10MB-1GB	10's to 100,s nano.	10-100
Secondary memory	1G-10G	10's of milliseconds	1,000,000

Information Units

Basic unit is the *bit* (stores a 0 or a 1)

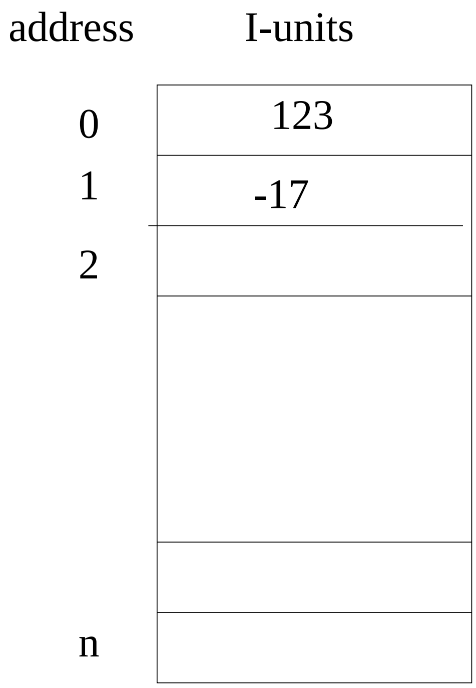
Bits are grouped together in information units:

- bytes = 8 bits
- words = 4 bytes
- double words = 2 words etc...

Memory

Memory is an array of information units

- Each unit has the same size
- Each unit has a *unique* address
- Address and contents are different (obvious!)



Addressing

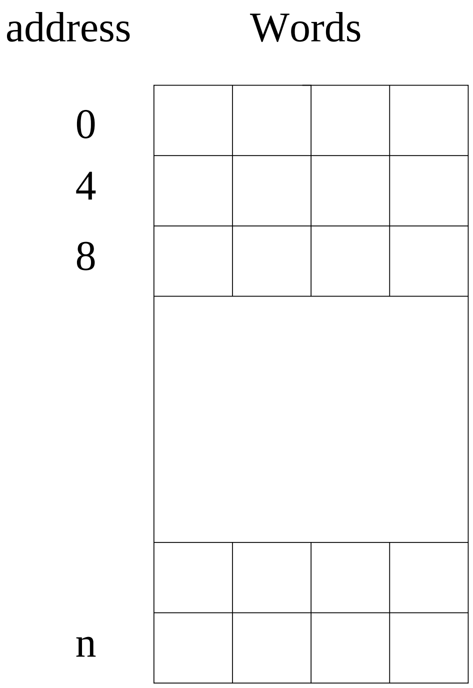
The *address space* is the set of all I-units that a program can reference.

Most machines today are *byte addressable*

- on a 16-bit processor, the address space is 64K bytes (too small)
- on a 32-bit processor, the address space is 4G bytes (beginning to be too small)
- on a 64-bit processor, the address space is enormous (should last for a while!)

Addressing words

Every word starts at an address divisible by 4.



Which byte is byte 0 within a word? Big Endian vs. Little Endian
(The assembly language we will use – SPIM – is Little Endian)

The CPU – Instruction Execution Cycle

The CPU executes the program by repeatedly following this cycle:

1. Fetch the next instruction
2. Execute next instruction
3. Compute address of next instruction
4. Back to item 1

Instructions

An instruction tells the CPU

- the operation to be performed – OPCODE
- The operands (zero or more) on which to perform the operation

For a given instruction, the ISA specifies

- what the opcode means
- how many operands it requires (and the type of the operands)

Operand is either

- a register
- a memory address
- a constant