

# Design and Manufacturing of Microsystem

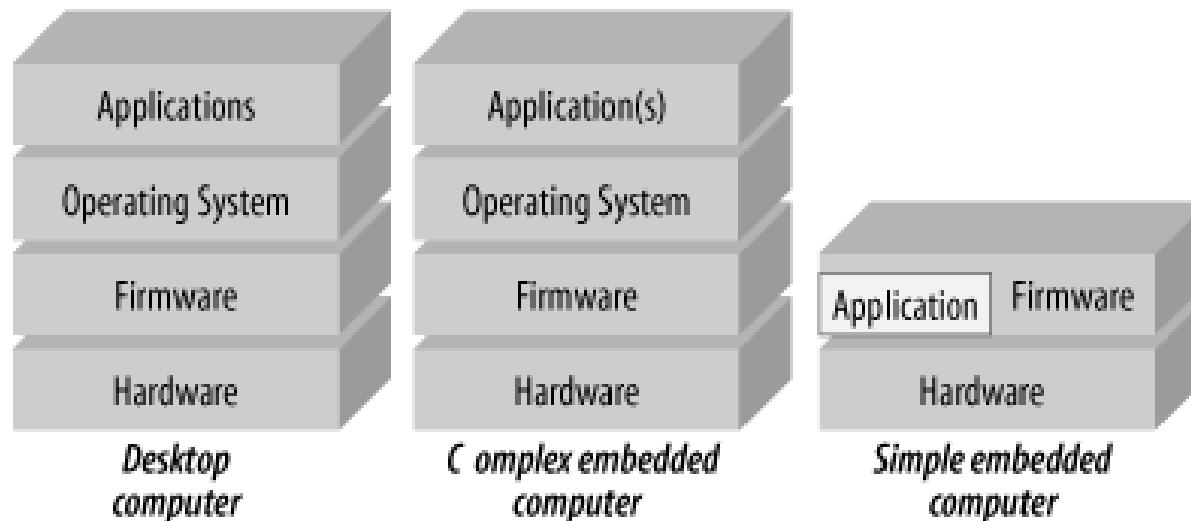
Introduction to Computer Architecture

By:

Engr. Joseph Ronald Cañedo

# Concept: Overview

- Computer is a machine designed to process, store, and retrieve data. Data may be numbers in a spreadsheet, characters of text in a document, dots of color in an image, waveforms of sound, or the state of some system, such as an air conditioner or a CD player. All data is stored in the computer as numbers.

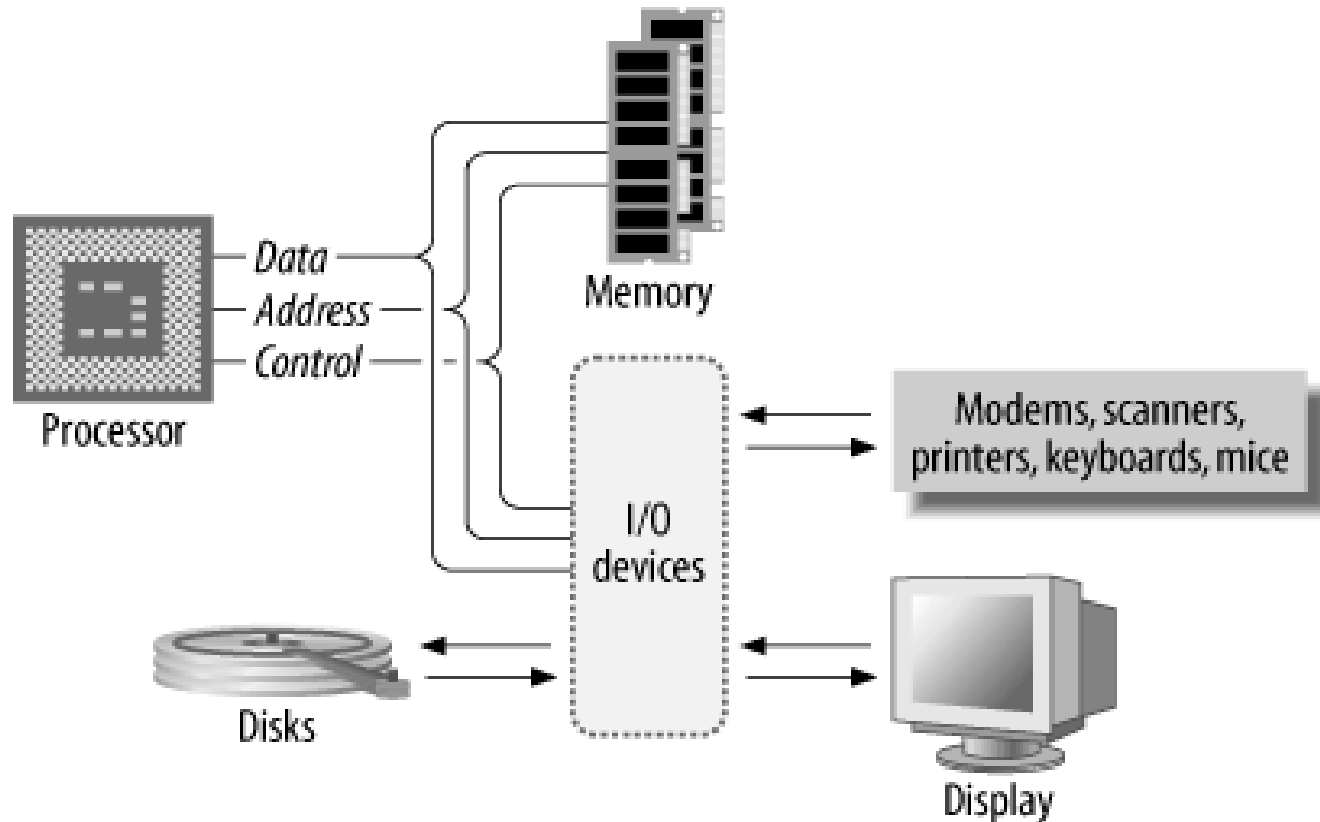


# Concept: Processor

- The processor is the most important part of a computer, the component around which everything else is centered.
- In essence, the processor is the computing part of the computer.
- A processor is an electronic device capable of manipulating data (information) in a way specified by a sequence of instructions.
  - The instructions are also known as **opcodes** or machine code. This sequence of instructions may be altered to suit the application, and, hence, computers are programmable. A sequence of instructions is what constitutes a program.

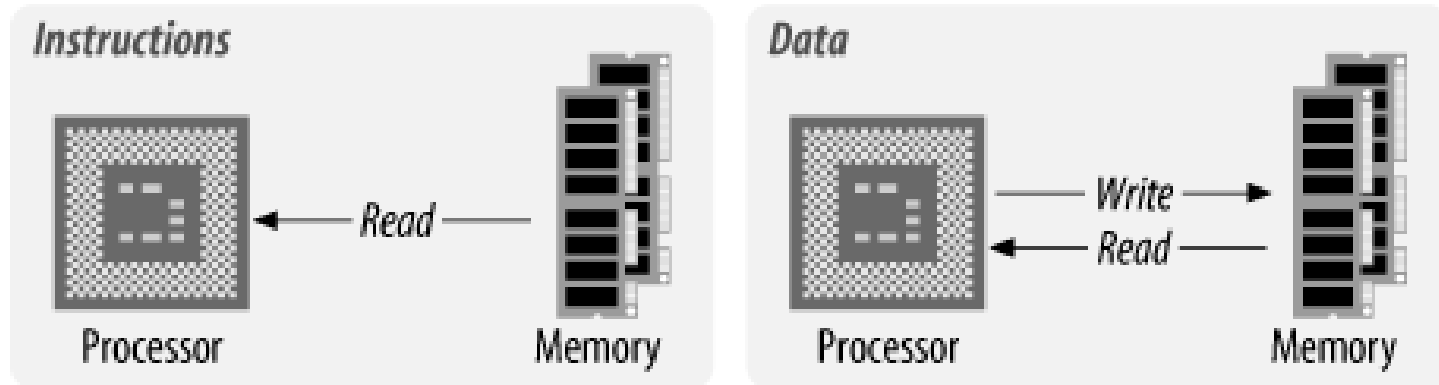
# Concept: Basic System Architecture

- The processor alone is incapable of successfully performing any tasks. It requires memory (for program and data storage), support logic, and at least one I/O device ("input/output device") used to transfer data between the computer and the outside world.



# Concept: Basic System Architecture

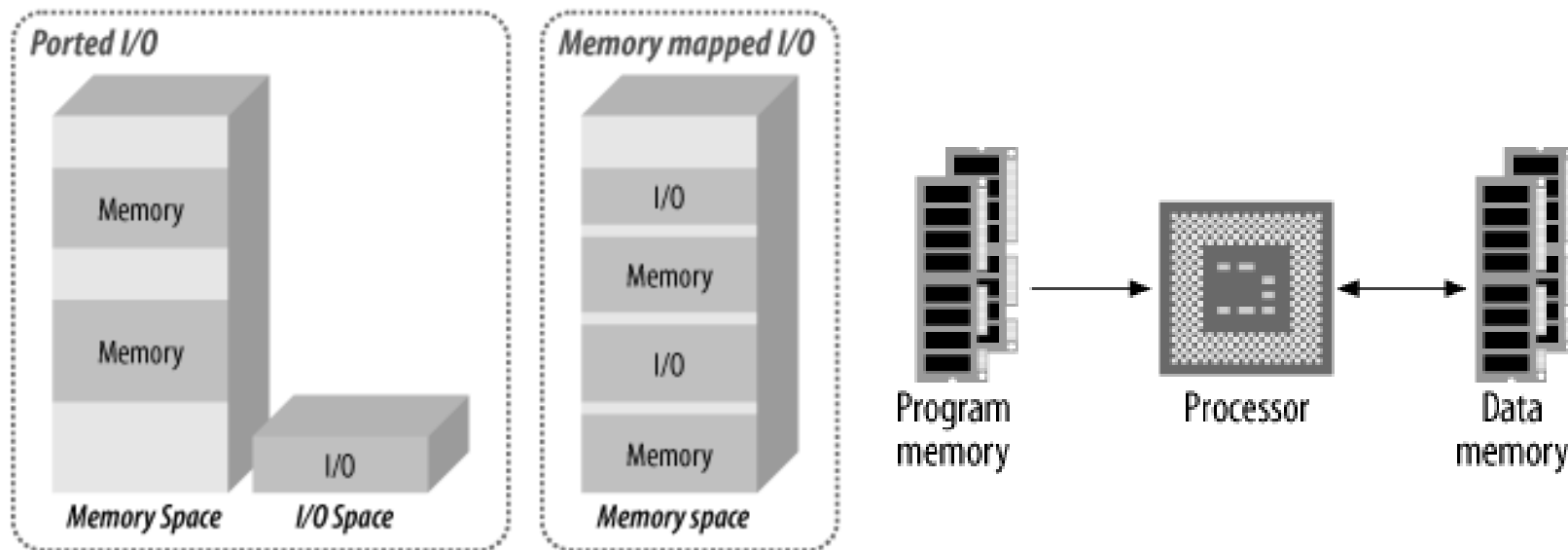
- Instructions are read (fetched) from memory, while data is both read from and written to memory.



- This form of computer architecture is known as a **Von Neumann** machine, named after John Von Neumann, one of the originators of the concept.

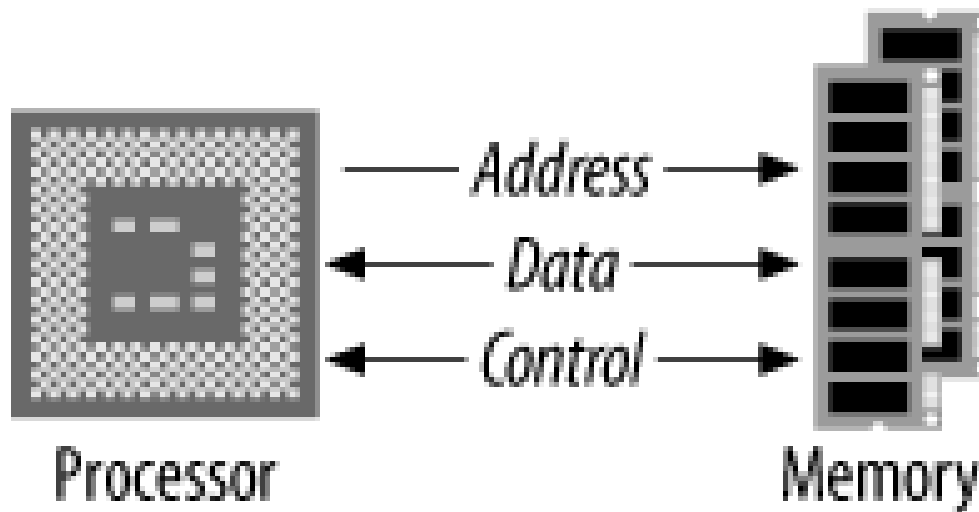
# Concept: Basic System Architecture

- Most microprocessors available are standard Von Neumann machines. The main deviation from this is the **Harvard architecture**, in which instructions and data have different memory spaces with separate address, data, and control buses for each memory space. This has a number of advantages in that instruction and data fetches can occur concurrently, and the size of an instruction is not set by the size of the standard data unit (word).



# Concept: Buses

- A bus is a physical group of signal lines that have a related function. Buses allow for the transfer of electrical signals between different parts of the computer system and thereby transfer information from one device to another.
- The majority of microprocessors available today (with some exceptions) use the three-bus system architecture. The three buses are the address bus, the data bus, and the control bus.

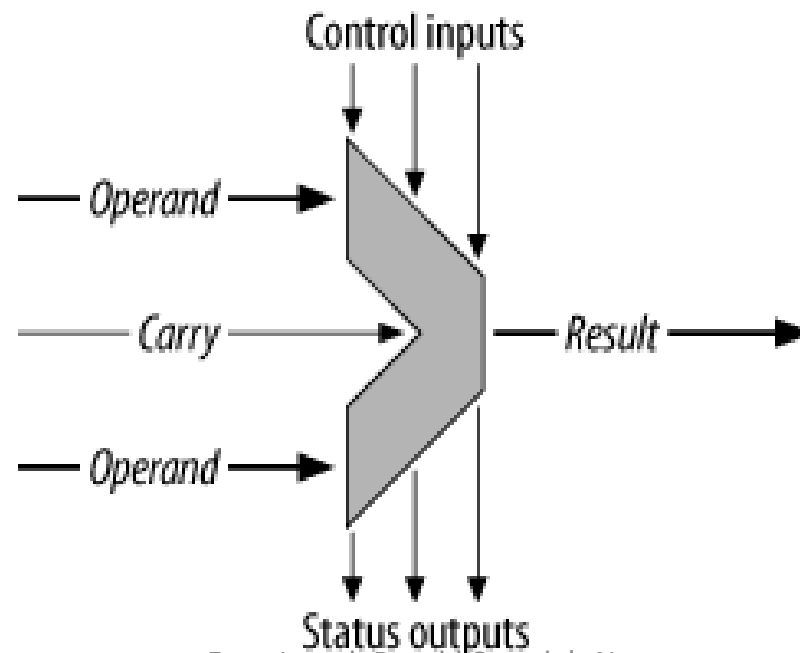


# Concept: Processor Operation

- There are six basic types of access that a processor can perform with external chips.
  1. write data to memory or
  2. write data to an I/O device
  3. read data from memory or
  4. read data from an I/O device,
  5. read instructions from memory, and
  6. perform internal manipulation of data within the processor.

# Concept: ALU

- The **Arithmetic Logic Unit (ALU)** performs the internal arithmetic manipulation of data in the processor. The instructions that are read and executed by the processor control the data flow between the registers and the ALU. The instructions also control the arithmetic operations performed by the ALU via the ALU's control inputs.



# Concept: Interrupts

- **Interrupts** (also known as traps or exceptions in some processors) are a technique of diverting the processor from the execution of the current program so that it may deal with some event that has occurred. Such an event may be an error from a peripheral, or simply that an I/O device has finished the last task it was given and is now ready for another. An interrupt is generated in your computer every time you type a key or move the mouse.
- Two types of Interrupts:
  1. Hardware Interrupts
  2. Software Interrupts

# Concept: CISC and RISC

- There are two major approaches to processor architecture:
  1. Complex Instruction Set Computer (CISC, pronounced "Sisk") processors and
    - Classic CISC processors are the Intel x86, Motorola 68xxx, and National Semiconductor 32xxx processors, and, to a lesser degree, the Intel Pentium.
  2. Reduced Instruction Set Computer (RISC) processors.
    - Common RISC architectures are the Freescale/IBM PowerPC, the MIPS architecture, Sun's SPARC, the ARM, the Atmel AVR, and the Microchip PIC.

# Concept: CISC and RISC

- Instructions Sets can be indentified by it assembly pseudo-code.

## 1. Complex Instruction Set Computer

```
clear 0x1000 ; clear memory location 0x1000
```

```
load r1,#5 ; load register 1 with the value 5
```

## 2. Reduced Instruction Set Computer

```
xor r1,r1 ; clear register 1
```

```
store r1,0x1000 ; clear memory location 0x1000
```

```
add r1,#5 ; load register 1 with the value 5
```

# Concept: DSP

- A special type of processor architecture is that of the **Digital Signal Processor (DSP)**. These processors have instruction sets and architectures optimized for numerical processing of array data. They often extend the Harvard architecture concept further by not only having separate data and code spaces, but also by splitting the data spaces into two or more banks. This allows concurrent instruction fetch and data accesses for multiple operands. As such, DSPs can have very high throughput and can outperform both CISC and RISC processors in certain applications.

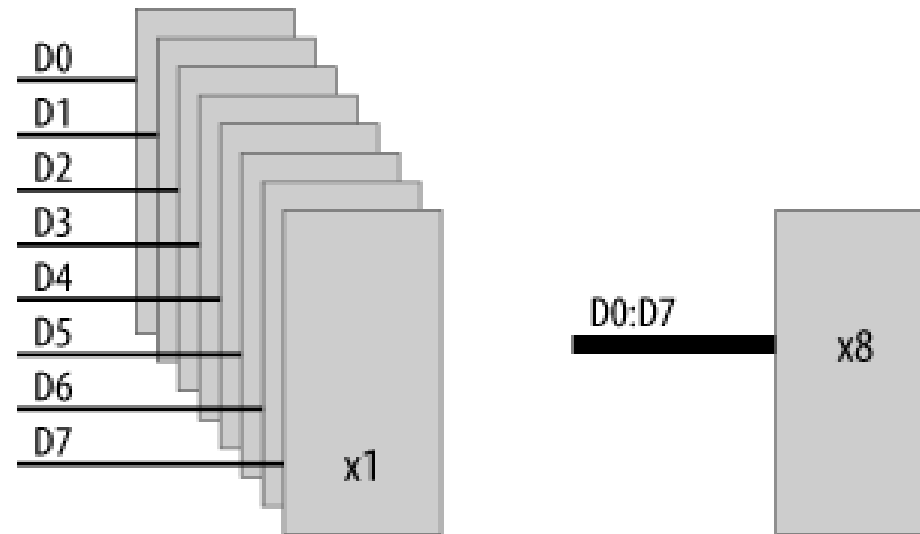
# Memory

# Memory: Overview

- Memory is used to hold data and software for the processor.
- There is a variety of memory types, and often a mix is used within a single system.
  - Some memory will retain its contents while there is no power, yet will be slow to access.
  - Other memory devices will be high-capacity, yet will require additional support circuitry and will be slower to access.
  - Still other memory devices will trade capacity for speed, yielding relatively small devices, yet will be capable of keeping up with the fastest of processors.

# Memory

- Memory chips can be organized in two ways, either in
  1. word-organized scheme, a complete nybbles, bytes, or words are stored within a single component, or
  2. bit-organized memory, each bit of a byte or word is allocated to a separate component



bit-organized memory

word-organized

# Memory: RAM

- RAM stands for **Random Access Memory**.
- RAM is the "working memory" in the computer system. It is where the processor may easily write data for temporary storage.
- RAM is generally volatile, losing its contents when the system loses power. Any information stored in RAM that must be retained must be written to some form of permanent storage before the system powers down.
- There are special nonvolatile RAMs that integrate a battery-backup system, such that the RAM remains powered even when the rest of the computer system has shut down.
- RAMs generally fall into two categories:
  1. static RAM (also known as SRAM) and
  2. dynamic RAM (also known as DRAM).

# Memory: ROM

- ROM stands for Read-Only Memory. This is also a bit of a misnomer, since many (modern) ROMs can also be written to. ROMs are nonvolatile memory, requiring no power to retain their contents. They are generally slower than RAM, and considerably slower than fast static RAM.
- The primary purpose of ROM within a system is to hold the code (and sometimes data) that needs to be present at power-up. Such software is generally known as firmware and contains software to initialize the computer by placing I/O devices into a known state. It may contain either a bootloader program to load an operating system off disk or network or, in the case of an embedded system, it may contain the application itself.
- Many microcontrollers contain on-chip ROM, thereby reducing component count and simplifying system design.

# Memory: ROM

- Types of ROM
  1. Erasable Programmable Read-Only Memory, or EPROM
  2. EEROM is Electrically Erasable Read-Only Memory, also known as EEPROM (Electrically Erasable Programmable Read-Only Memory)
  3. Flash is the newest ROM technology and is now dominant.

# Input/Output

# Input/Output

- The address space of the processor can contain devices other than memory. These are input/output devices (I/O devices, also known as **peripherals**) and are used by the processor to communicate with the external world.
  - Some examples are serial controllers that communicate with keyboards, mice, modems, etc.; parallel I/O devices that control some external subsystem; or disk-drive controllers, video and audio controllers, or network interfaces.

# Input/Output

There are three main ways in which data may be exchanged with the external world:

1. Programmed I/O
  - The processor accepts or delivers data at times convenient to it (the processor).
2. Interrupt-driven I/O
  - External events control the processor by requesting the current program be suspended and the external event be serviced. An external device will interrupt the processor (assert an interrupt control line into the processor), at which time the processor will suspend the current task (program) and begin executing an interrupt service routine. The service of an interrupt may involve transferring data from input to memory, or from memory to output.
3. Direct Memory Access (DMA)
  - DMA allows data to be transferred from I/O devices to memory directly without the continuous involvement of the processor. DMA is used in high-speed systems, where the rate of data transfer is important. Not all processors support DMA.

# DMA

# DMA: Introduction

- DMA is a way of streamlining transfers of large blocks of data between two sections of memory, or between memory and an I/O device. Let's say you want to read in 100M from disk and store it in memory.
- You have two options.
  1. One option is for the processor to read one byte at a time from the disk controller into a register and then store the contents of the register to the appropriate memory location. For each byte transferred, the processor must read an instruction, decode the instruction, read the data, read the next instruction, decode the instruction, and then store the data. Then the process starts over again for the next byte.
  2. The second option in moving large amounts of data around the system is DMA. A special device, called a DMA Controller (DMAC), performs high-speed transfers between memory and I/O devices. Using DMA bypasses the processor by setting up a channel between the I/O device and the memory. Thus, data is read from the I/O device and written into memory without the need to execute code to perform the transfer on a byte-by-byte (or word-by-word) basis.

# DMA: Introduction

## **There are four basic types of DMA:**

1. Standard block transfer
  - Accomplished by the DMA controller performing a sequence of memory transfers. The transfers involve a load operation from a source address followed by a store operation to a destination address. Standard block transfers are initiated under software control and are used for moving data structures from one region of memory to another.
2. Demand-mode transfers
  - Similar to standard mode except that the transfer is controlled by an external device. Demand-mode transfers are used to move data between memory and I/O or vice versa. The I/O device requests and synchronizes the movement of data.

# DMA: Introduction

**There are four basic types of DMA:**

## 3. Fly-by transfer

- Provides high-speed data movement in the system. Instead of using multiple bus accesses as with conventional DMA transfers, fly-by transfers move data from source to destination in a single access. The data is not read into the DMAC before going to its destination. During a fly-by transfer, memory and I/O are given different bus control signals. For example, an I/O device is given a read request at the same time that memory is given a write request. Data moves from the I/O device straight into the memory device.

## 4. Data-chaining transfers

- Allow DMA transfers to be performed as specified by a linked-list in memory. Data chaining is started by specifying a pointer to a descriptor in memory. The descriptor is a table specifying byte count, source address, destination address, and a pointer to the next descriptor. The DMAC loads the relevant information about the transfer from this table and begins moving data. The transfer continues until the number of bytes transferred is equal to the entry in the byte-count field. On completion, the pointer to the next descriptor is loaded. This continues until a null pointer is found.

# DMA: Parallel and Distributed Computers

- **SIMD computers**

- **Single-Instruction Multiple-Data (SIMD)** computers are highly parallel machines, employing large arrays of simple processing elements. In an SIMD machine, each processing element has a small amount of local memory. The instructions executed by the SIMD computer are broadcast from a central instruction server to every processing element within the machine. In this way, each processor executes the same instruction as all other processing elements within the machine. Since each processor executes the instruction on its local data, all elements within the data structure are worked upon simultaneously.

# DMA: Parallel and Distributed Computers

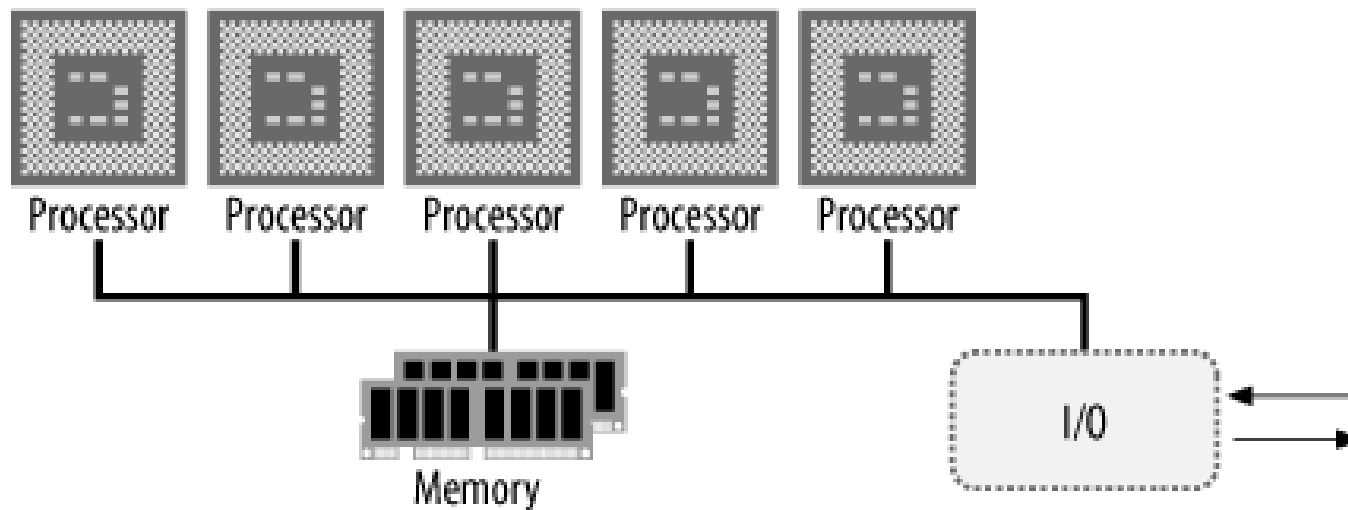
- **MIMD computers**

- The other major form of parallel machine is the **Multiple-Instruction Multiple-Data** (MIMD) computer. These machines are typically coarsely grained collections of semi-autonomous processors, each with their own local memory and local programs. An algorithm being executed on an MIMD computer is typically broken up into a series of smaller sub-problems, each executed on a processor of the MIMD machine. By giving each processing element in the MIMD machine identical programs to execute, the MIMD machine may be treated as an SIMD computer. The grain of an MIMD computer is much less than that of an SIMD machine. MIMD computers tend to use a smaller number of very powerful processors, rather than a large number of less powerful ones.

# DMA: Parallel and Distributed Computers

MIMD computers can be of one of two types:

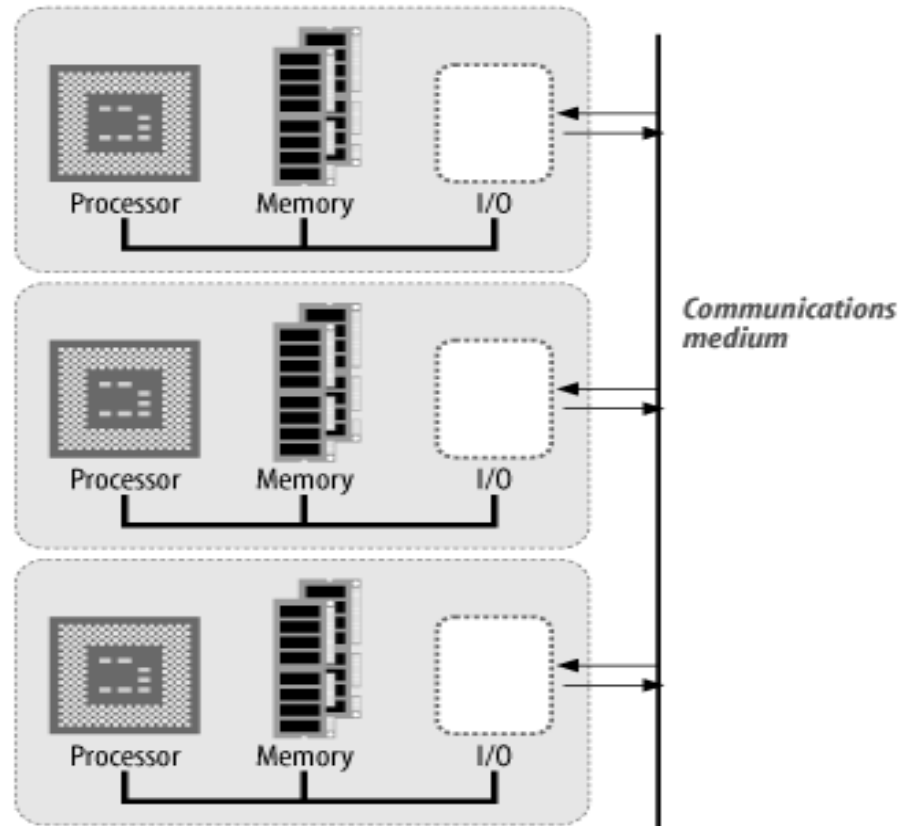
1. shared-memory MIMD



# DMA: Parallel and Distributed Computers

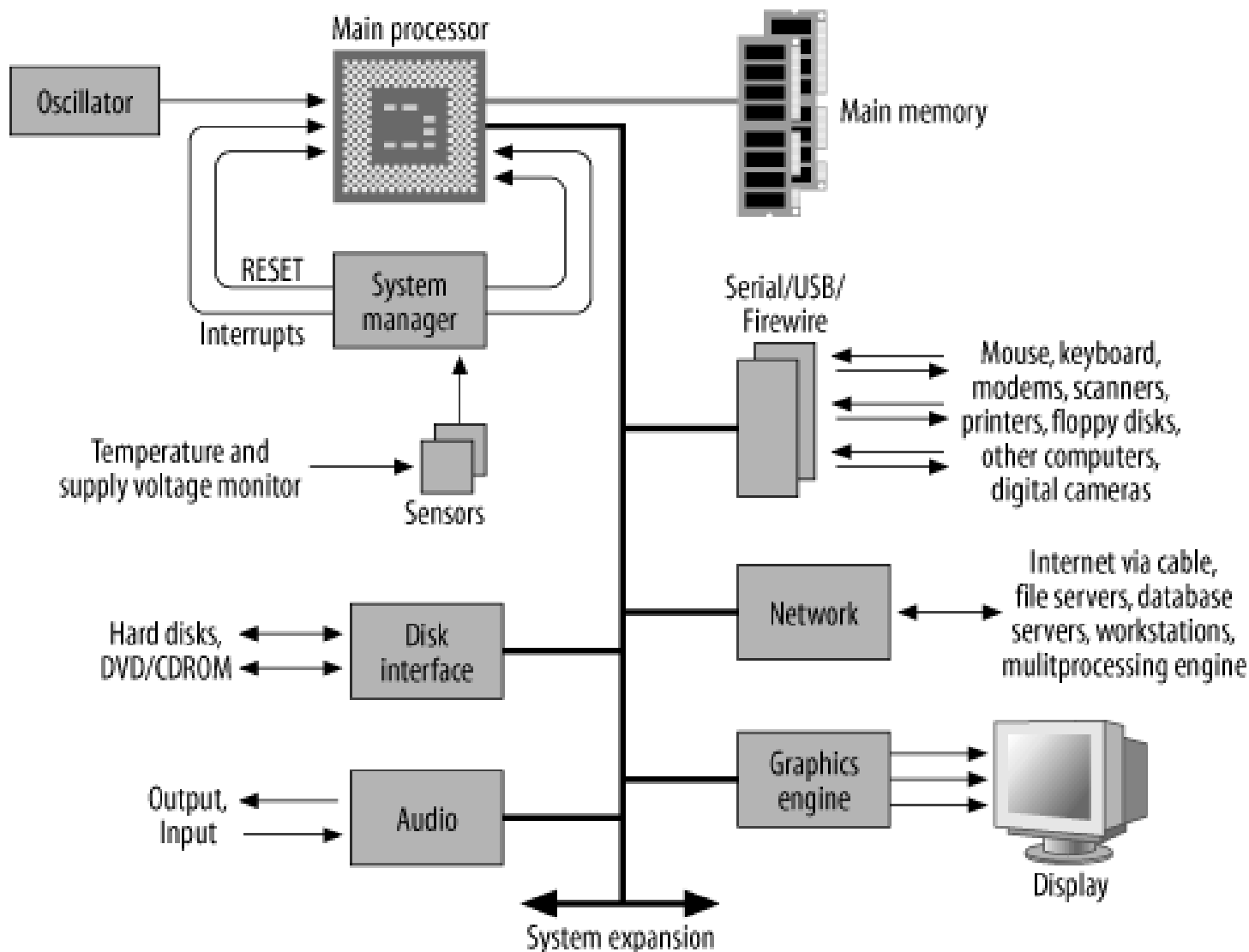
MIMD computers can be of one of two types:

2. message-passing MIMD.



# **Embedded Computer Architecture**

# A Generic Computer



# An Embedded Computer

