Internal organization of computers

The internal working of every computer can be broken down into three parts: CPU (central processing unit), memory , and I/O (input/output) devices (see Figure 0-9). The function of the CPU is to execute (process) information stored in memory. The function of I/O devices such as the keyboard and video monitor is to provide a means of communicating with the CPU. The CPU is connected to memory and I/O through strips of wire called a *bus*. The bus inside a computer carries information from place to place just as a street bus carries people from place to place. In every computer there are three types of buses: address bus, data bus, and control bus.

For a device (memory or I/O) to be recognized by the CPU, it must be assigned an address. The address assigned to a given device must be unique; no two devices are allowed to have the same address. The CPU puts the address (of course, in binary) on the address bus, and the decoding circuitry finds the device. Then the CPU uses the data bus either to get data from that device or to send data to it. The control buses are used to provide read or write signals to the device to indicate if the CPU is asking for information or sending it information. Of the three buses, the address bus and data bus determine the capability of a given CPU.

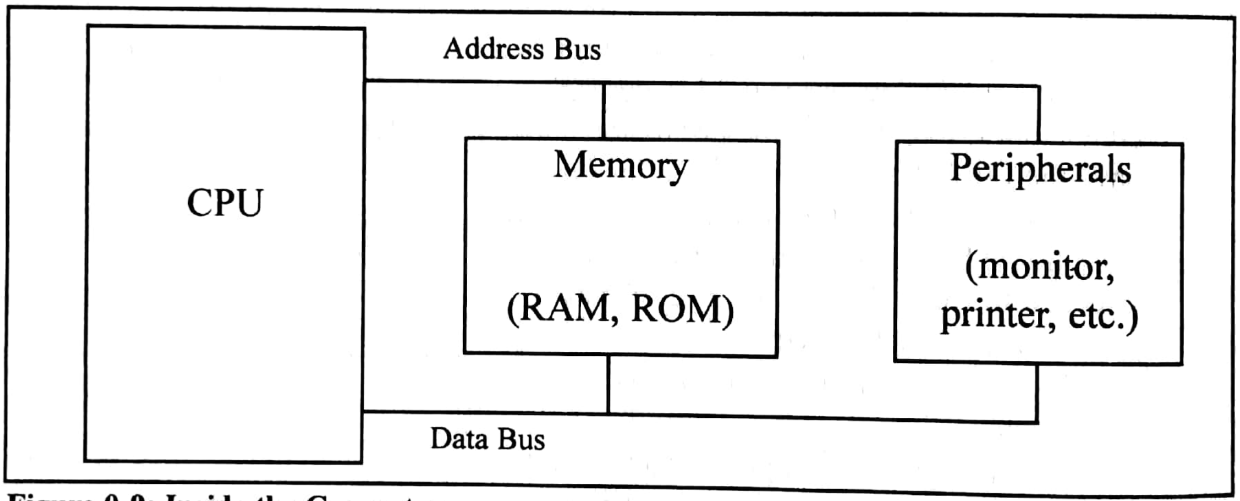

Figure 0-9: Inside the Computer

More about the data bus

Since data buses are used to carry information in and out of a CPU, the more data buses available, the better the CPU. If one thinks of data buses as highway lanes, it is clear that more lanes provide a better pathway between the CPU and its external devices (such as printers, RAM, ROM, etc.; see Figure 0-10). By the same token, that increase in the number of lanes increases the cost of construction. More data buses mean a more expensive CPU and computer. The average size of data buses in CPUs varies between 8 and 64. Early computers such as Apple 2 used an 8-bit data bus, while supercomputers such as Cray use a 64-bit data bus. Data buses are bidirectional, since the CPU must use them either to receive or to send data. The processing power of a computer is related to the size of its buses, since an 8-bit bus can send out 1 byte a time, but a 16-bit bus can send out 2 bytes at a time, which is twice as fast.

More about the address bus

Since the address bus is used to identify the devices and memory connected to the CPU, the more address buses available, the larger the number of devices that can be addressed. In other words, the number of address buses for a CPU determines the number of locations with which it can communicate. The number of locations is always equal to 2^x, where x is the number of address lines, regardless of the size of the data bus. For example, a CPU with 16 address lines can provide a total of 65,536 (2^{16}) or 64K bytes of addressable memory. Each location can have a maximum of 1 byte of data. This is due to the fact that all general-purpose microprocessor CPUs are what is called *byte addressable*. As another example, the IBM PC AT uses a CPU with 24 address lines and 16 data lines. In this case the total accessible memory is 16 megabytes (2^{24} = 16 megabytes). In this example there would be 2^{24} locations, and since each location is one byte, there would be 16 megabytes of memory. The address bus is a *unidirectional* bus, which means that the CPU uses the address bus only to send out addresses. To summarize: The total number of memory locations addressable by a given CPU is always equal to 2^x where x is the number of address bits, regardless of the size of the data bus.

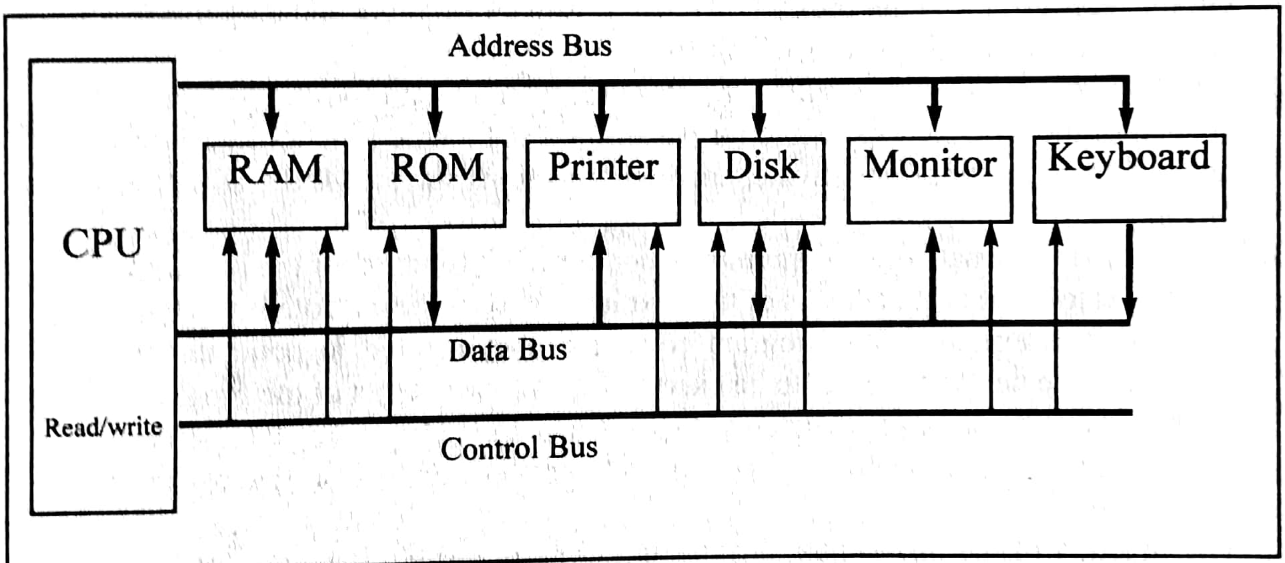

Figure 0-10: Internal Organization of Computers

CPU and its relation to RAM and ROM

For the CPU to process information, the data must be stored in RAM or ROM. The function of ROM in computers is to provide information that is fixed and permanent. This is information such as tables for character patterns to be displayed on the video monitor, or programs that are essential to the working of the computer, such as programs for testing and finding the total amount of RAM installed on the system, or programs to display information on the video monitor. In contrast, RAM is used to store information that is not permanent and can change with time, such as various versions of the operating system and application packages such as word processing or tax calculation packages. These programs are loaded into RAM to be processed by the CPU. The CPU cannot get the information from the disk directly since the disk is too slow. In other words, the CPU gets the information to be processed, first from RAM (or ROM). Only if it is not there does the CPU seek it from a mass storage device such as a disk, and then it transfers the information to RAM. For this reason, RAM and ROM are sometimes referred to as *primary memory* and disks are called *secondary memory*. Figure 0-11 shows a block diagram of the internal organization of the PC.

Inside CPUs

A program stored in memory provides instructions to the CPU to perform an action. The action can simply be adding data such as payroll data or controlling a machine such as a robot. It is the function of the CPU to fetch these instructions from memory and execute them. To perform the actions of fetch and execute, all CPUs are equipped with resources such as the following:

1. Foremost among the resources at the disposal of the CPU are a number of *registers*. The CPU uses registers to store information temporarily. The information could be two values to be processed, or the address of the value needed to be fetched from memory. Registers inside the CPU can be 8-bit, 16-bit, 32-bit, or even 64-bit registers, depending on the CPU. In general, the more and bigger the registers, the better the CPU. The disadvantage of more and bigger registers is the increased cost of such a CPU.
2. The CPU also has what is called the *ALU* (arithmetic/logic unit). The ALU section of the CPU is responsible for performing arithmetic functions such as add, subtract, multiply, and divide, and logic functions such as AND, OR, and NOT.
3. Every CPU has what is called a *program counter*. The function of the program counter is to point to the address of the next instruction to be executed. As each instruction is executed, the program counter is incremented to point to the address of the next instruction to be executed. It is the contents of the program counter that are placed on the address bus to find and fetch the desired instruction. In the IBM PC, the program counter is a register called IP, or the instruction pointer.
4. The function of the *instruction decoder* is to interpret the instruction fetched into the CPU. One can think of the instruction decoder as a kind of dictionary, storing the meaning of each instruction and what steps the CPU should take upon receiving a given instruction. Just as a dictionary requires more pages the more words it defines, a CPU capable of understanding more instructions requires more transistors to design.

Internal working of computers

To demonstrate some of the concepts discussed above, a step-by-step analysis of the process a CPU would go through to add three numbers is given next. Assume that an imaginary CPU has registers called A, B, C, and D. It has an 8-bit data bus and a 16-bit address bus. Therefore, the CPU can access memory from addresses 0000 to FFFFH (for a total of 10000H locations). The action to be performed by the CPU is to put hexadecimal value 21 into register A, and then add to register A values 42H and 12H. Assume that the code for the CPU to move a puts it in register A. Meanwhile the program counter becomes 1404, the address of the next instruction.

5. Address 1404H is put on the address bus and the code is fetched into the CPU, decoded, and executed. This code is again adding a value to register A. The program counter is updated to 1406H.
6. Finally, the contents of address 1406 are fetched in and executed. This HALT instruction tells the CPU to stop incrementing the program counter and asking for the next instruction. In the absence of the HALT, the CPU would continue updating the program counter and fetching instructions.

Now suppose that address 1403H contained value 04 instead of 42H. How would the CPU distinguish between data 04 to be added and code 04? Remember that code 04 for this CPU means move the next value into register A. Therefore, the CPU will not try to decode the next value. It simply moves the contents of the following memory location into register A, regardless of its value.

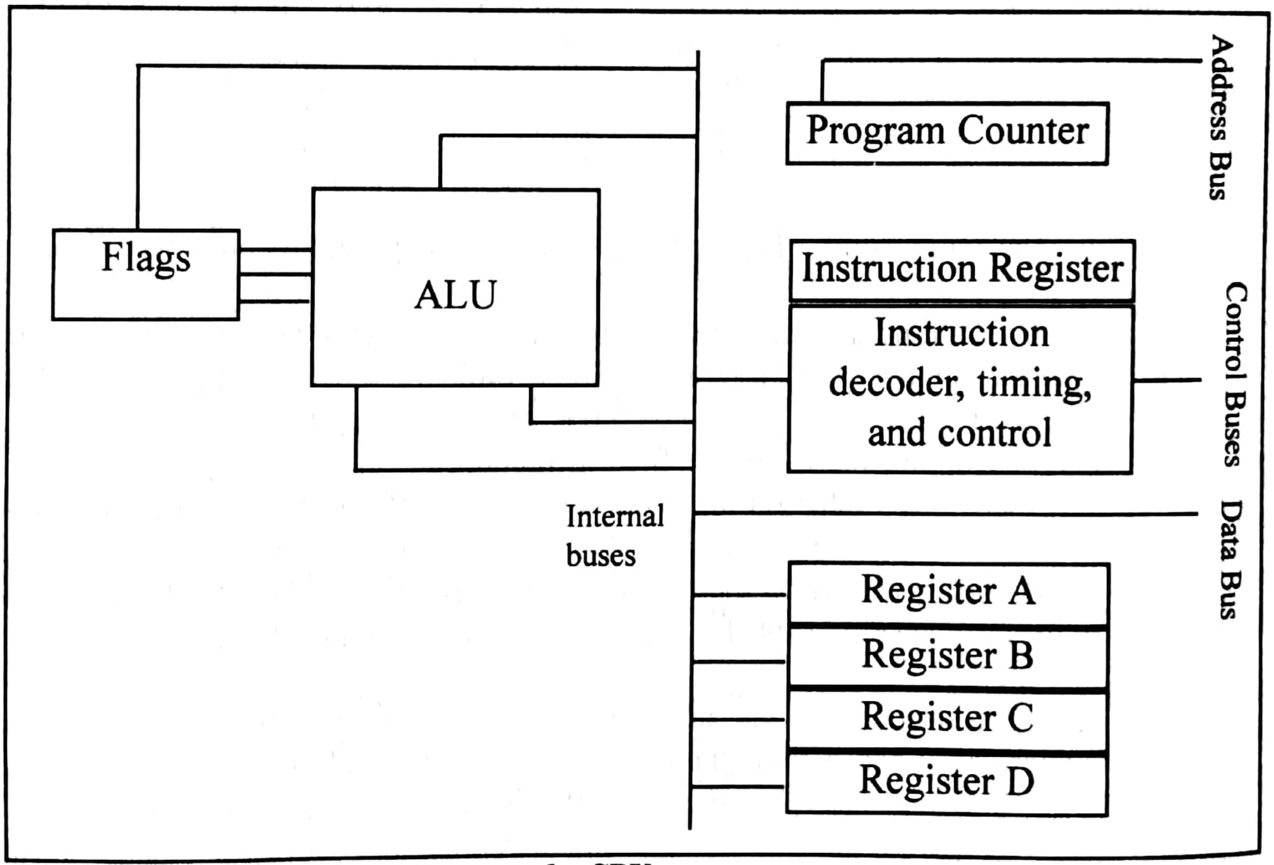

Figure 0-11: Internal Block Diagram of a CPU

Review Questions

1. How many bytes is 24 kilobytes?
2. What does "RAM" stand for? How is it used in computer systems?
3. What does "ROM" stand for? How is it used in computer systems?
4. Why is RAM called volatile memory?
5. List the three major components of a computer system.
6. What does "CPU" stand for? Explain its function in a computer.

This chapter begins with a discussion of the role and importance of microcontrollers in everyday life. In Section 1.1 we also discuss criteria to consider in choosing a microcontroller, as well as the use of microcontrollers in the embedded market. Section 1.2 covers various members of the 8051 family such as the 8052 and 8031, and their features. In addition, we discuss various versions of the 8051 such as the 8751, AT89C51, and DS5000.

SECTION 1.1: MICROCONTROLLERS AND EMBEDDED PROCESSORS

In this section we discuss the need for microcontrollers and contrast them with general-purpose microprocessors such as the Pentium and other x86 microprocessors. We also look at the role of microcontrollers in the embedded market. In addition, we provide some criteria on how to choose a microcontroller.

Microcontroller versus general-purpose microprocessor

What is the difference between a microprocessor and microcontroller? By microprocessor is meant the general-purpose microprocessors such as Intel's x86 family (8086, 80286, 80386, 80486, and the Pentium) or Motorola's 680x0 family (68000, 68010, 68020, 68030, 68040, etc.). These microprocessors contain no RAM, no ROM, and no I/O ports on the chip itself. For this reason, they are commonly referred to as *general-purpose microprocessors*.

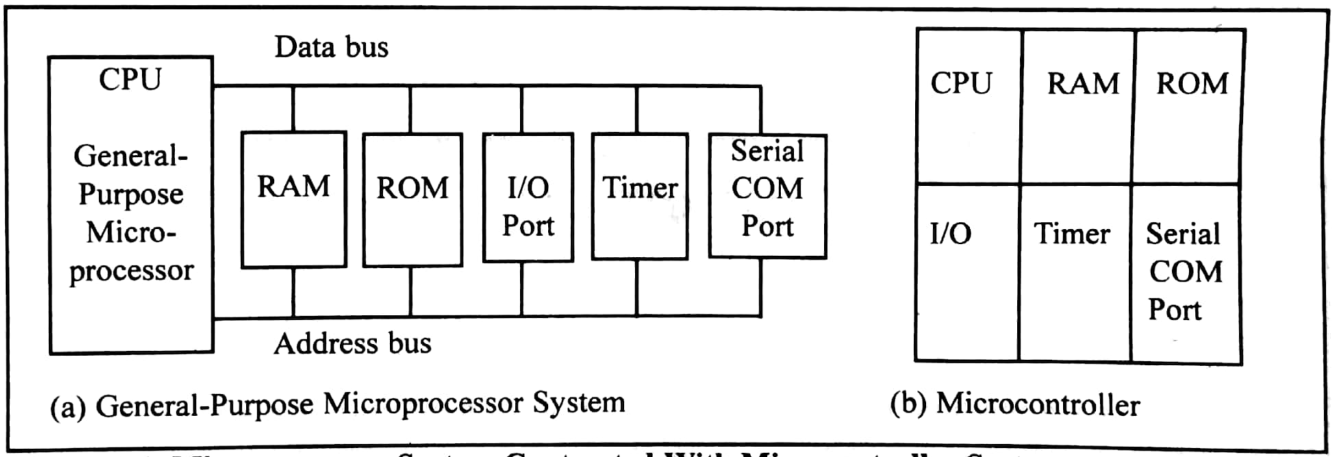

Figure 1-1. Microprocessor System Contrasted With Microcontroller System

A system designer using a general-purpose microprocessor such as the Pentium or the 68040 must add RAM, ROM, I/O ports, and timers externally to make them functional. Although the addition of external RAM, ROM, and I/O ports makes these systems bulkier and much more expensive, they have the advantage of versatility such that the designer can decide on the amount of RAM, ROM, and I/O ports needed to fit the task at hand. This is not the case with microcontrollers. A microcontroller has a CPU (a microprocessor) in addition to a fixed amount of RAM, ROM, I/O ports, and a timer all on a single chip. In other words, the processor, the RAM, ROM, I/O ports, and timer are all embedded together on one chip; therefore, the designer cannot add any external memory, I/O, or timer to it. The fixed amount of on-chip ROM, RAM, and number of I/O ports in microcontrollers makes them ideal for many applications in which cost and space are