Microcomputer Systems

Book: Assembly Language Programming and Organization of the IBM PC - Ytha Yu, Charles Marut

The Components of a Microcomputer System

Integrated-circuit (IC) chips are used in the construction of computer circuits. · each IC chip may consist hundreds or even thousands of transistors. These IC circuits are known as digital circuits as they operate on discrete voltage signal levels, typically, a high voltage and a low voltage. We use the symbols 0 and 1to represent the low- and high-voltage signals, respectively. These symbols are called binary digits or bits. All information processed by the computer Is represented by strings of 0's and 1's; that is, by bit strings. .

The Components of a Microcomputer System

Functionally, the computer circuits consist of three parts:

- The central processing unit (CPU)
- The memory circuits
- The I/O circuits

In a microcomputer, the CPU is a single chip processor called a microprocessor. The CPU is the brain of the computer, and it controls all operations. It uses the memory circuits to store Information, and the I/O circuits to communicate with I/O devices .

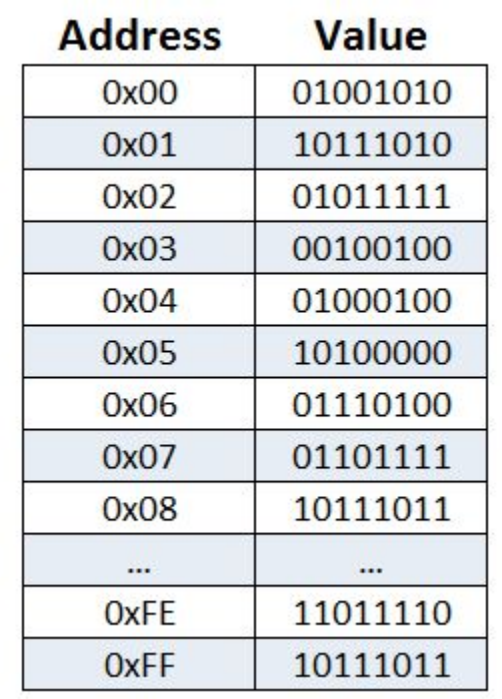
The System Board

Inside the system unit is a main circuit board called the system board, which contains the microprocessor and memory circuits. The system board is also called a motherboard because· it contains expansion slots. which are connectors for additional circuit boards called add-in boards or add-in cards. 1/0 circuits are usually located on add-In cards.

Memory

- Each memory byte is identified by a number that Is called its **address**. like the street address of a house. The first memory byte has address 0.
- The data stored in a memory location is called **content** or also called **value**.
- the difference between address and contents is that the address of a memory byte is fixed and is different from the address of any other memory byte in the computer. Yet the contents of a memory byte are not unique and are subject to change, because they denote the data currently being stored.

Memory

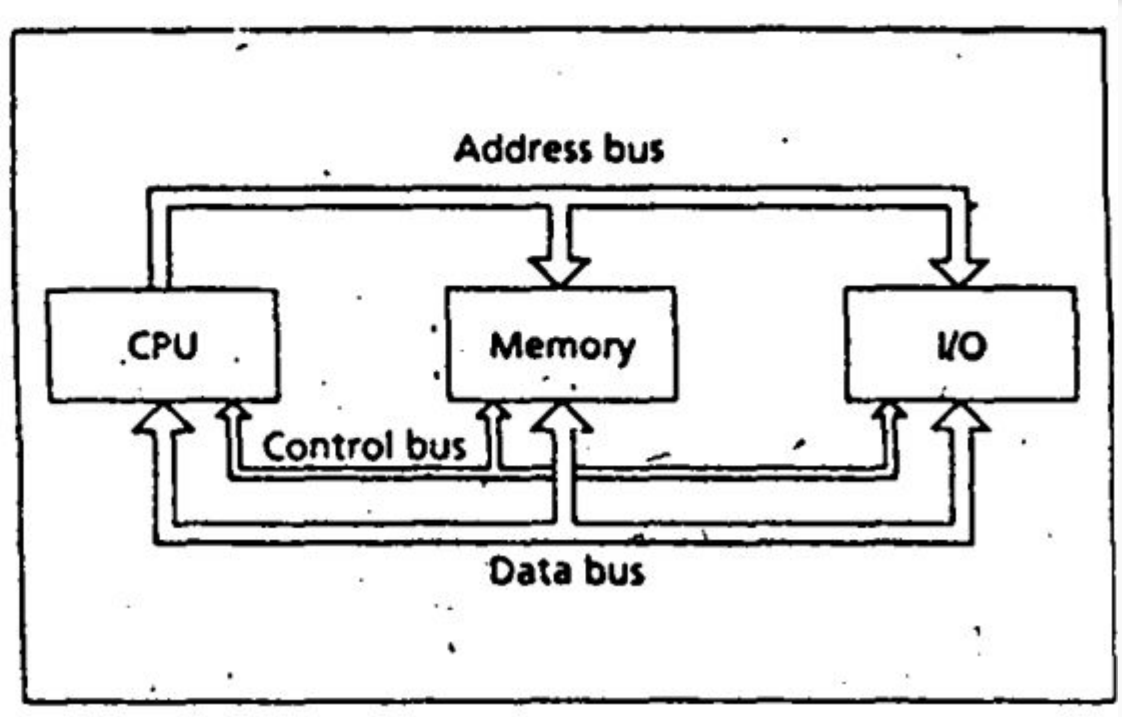


Address	Value
0x00	01001010
0x01	10111010
0x02	01011111
0x03	00100100
0x04	01000100
0x05	10100000
0x06	01110100
0x07	01101111
0x08	10111011
...	...
0xFE	11011110
0xFF	10111011

Memory

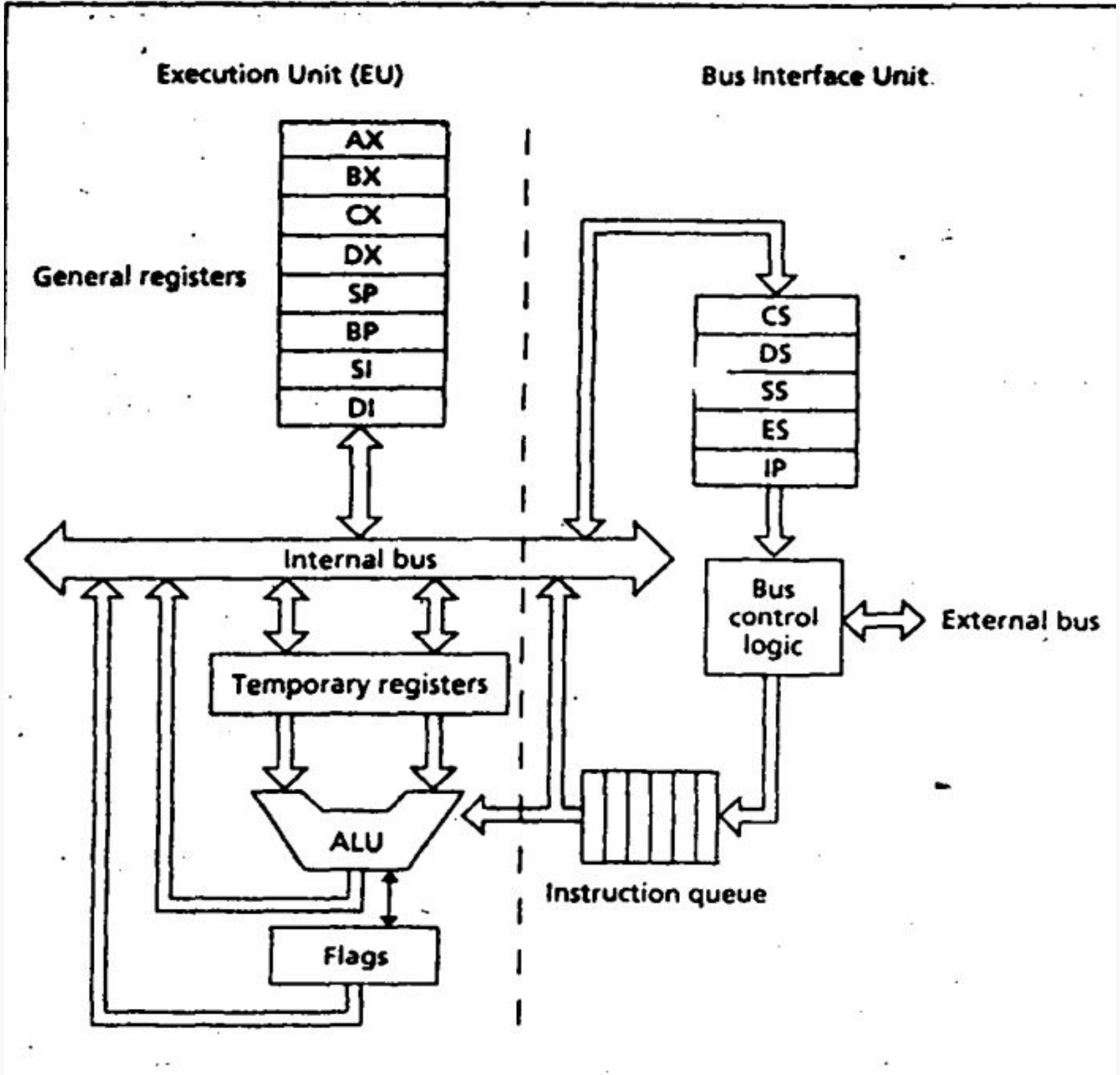
- The number of bits used In the address determines the number of bytes that can be accessed by the processor.
- Suppose a processor uses 20 bits for an address. How many memory bytes can be accessed?

Bus connection of a microcomputer



Intel 8086 Microprocessor Organization

There are two main components: the execution unit and the bus interface unit.



Execution Unit (Eu)

As the name implies, the purpose of the execution unit (EU} is to execute Instructions. It contains a circuit called the arithmetic and logic unit (ALU). The ALU can-perform·arithmetic (+, - , x ,I) and logic (AND, OR, NOT} operations. The data for the operations are stored in circuits called registers.. A register is like a memory location except that we normally refer to it by a name rather than a number. The EU has eight registers for storing data; their names arc AX, BX, CX, DX, SI, DI, BP, and SP. In addition, the EU contains temporary ' registers for holding operands for the ALU, and the FLAGS register whose individual bits reflect the result of a computation.

Bus Interface Unit (BIU)

The bus Interface unit (BIU) facilitates communication between the EU and the memory or I/O circuits. It is responsible for transmitting addresses, data and control signals on the buses. Its registers are named CS, DS, ES, SS and IP; they hold addresses of memory locations. The IP (instruction pointer) contains the address of the next instruction to be executed by the EU.

Instruction Execution

First of all, a machine instruction has two parts: an **opcode** and **operands**. The opcode specifies the type of operation, and the operands are often given as memory addresses to the data to be operated on. The CPU goes through the following steps to execute a machine instruction (the fetch and execute cycle):

Fetch

1. Fetch an instruction from memory. 2. Decode the Instruction to determine the operation. 3. Fetch data from memory if necessary.

Execute

4. Perform the operation on the data. 5. Store the result in memory if needed.