CHAPTER VI: INPUT/OUTPUT MANAGEMENT

I/ Definition:

Input/Output refers to any information transfer operation between the processor, memory and external devices.

- From outside → Input (Read)
- To outside → Output (Write)

II/ I/O types:

- **Physical I/O:** managed by the lowest level of the system (hardware).
- **Logical I/O:** an instruction within a program that calls an I/O procedure (e.g. Scanf, Printf).
- **Virtual I/O:** between processor, main memory and virtual memory.
- **Spool I/O:** linked to a slow peripheral, access is via a disk file called "Spool".

III/ Driver (manager):

The driver is a specific program which initiates a physical transfer, controls the operation and handles errors.

Example: the printer driver allows you to :

- Ensure that the printer is plugged in.
- Ensure that it is not busy.
- Make sure there's paper.
- Insert the first sheet.
- Fill the pad with the characters to be printed.
- Start printing and move on to the next set of characters.

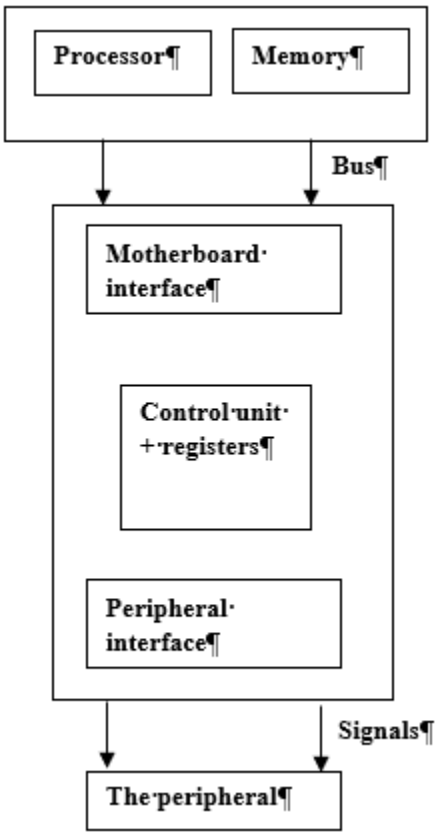
III/ I/O interfaces:

The interface is a device that connects peripherals to the computer, i.e. it adapts the characteristics of each of the two parts, motherboard and peripheral, these characteristics are :

- Type of connection: serial or parallel.
- Transmission mode: synchronous or asynchronous.
- Transmission speed: fast or slow.

IV/ Peripheral controllers:

Used to drive peripherals, each peripheral is linked to a bus by a controller. The role of the controller is to adapt the diversity of peripherals to a common interface. It's an electronic card inserted into the motherboard.



V/ Physical input/output control mode:

Processor control of a peripheral unit requires synchronization between processor actions and the unit being controlled. Several I/O control techniques are used:

a/ Direct physical I/O:

This is to control the flow of physical I/O by the processor itself, where we have :

- *Synchronous mode:*

The processor controls the peripheral from I/O start to finish. To start an I/O, the processor first initializes the appropriate controller registers with the necessary information, then issues a command to the controller to initiate the I/O operation. The controller, in turn, examines the contents of these registers to determine the action to be taken and initiates the transfer.

During the physical transfer of the character, the processor remains mobilized to monitor the I/O operation by consulting a device-specific flag designating its state, when it detects the end of the transfer. (Also known as active waiting).

The major drawback is the loss of CPU time.

- *Interval scan mode:*

In this mode, the processor initiates the I/O operation and then returns to the execution of user processes. At regular short intervals, the processor is interrupted to check whether this operation has been completed.

Same disadvantage of time loss due to handling frequent clock interruptions.

- *Asynchronous mode:*

The processor is freed from transfer completion control, where the controller is provided with an interrupt line that will be used to inform the processor. Once the transfer has been completed, the controller generates an interrupt that informs the processor of the end of the transfer.

The disadvantage of this mode is the time lost in executing these interrupt routines. **The interrupts already seen in Chapter III**

b/ Indirect physical I/O:

This technique is designed to improve processor performance by freeing the processor from tracking I/O until it has finished. An independent device known as a channel is used.

- *Channel mode :*

A channel is a programmable processor that can execute a sequence of I/O operations. It executes a channel program resident in MC.

The processor initiates an I/O operation by sending a special instruction to the channel, telling it where to find the channel program that starts the I/O. The channel then executes all the commands of this program independently of the processor. At the end of execution, the channel sends an interrupt to the processor to warn it of the end of I/O.

- *Direct memory access (DMA):*

DMA is a simplified channel used in small computers, which can be connected between the controller and the memory bus, enabling peripheral devices to access memory without passing through the CPU. If the DMA and CPU simultaneously request memory access, the DMA takes priority.

It consists of :

- A memory address register (contains the start of the block where the information in memory is located).
- A control register RC (indicates the read or write transfer direction).
- Character counter register RCC (number of characters to be read or written).
- The RAP register (the address of the device concerned).

It works as follows:

- The processor initializes the DMA by sending the start address of the block in memory to the address register, the number of characters in the RCC, the address of the device concerned in the RAP and the code of the operation to be executed in the RC register.
- The DMA replaces the processor in the transfer.

Once the operation has been completed, it transfers an interrupt to the processor.

Example:

Explain in detail the working principle of the DMA controller to transfer 500 characters to a display device whose number is 4. These characters are located from address 1400 in main memory.

- Address register = 1400.
- RCC register = 500.
- Register RC = write.
- Register RAP = 4.

The DMA reads the first character from main memory starting at address 1400, transfers it to the screen buffer, decrements the counter and increments the address to move on to the next character.