Section: 2nd Year Engineer in Computer Science

Chapter 2

The Devices Related to an Input/Output Operation

Pr. Ch. Bencheriet

2024/2025

Plan

1. Introduction

A computer needs to exchange information with the external environment (e.g., loading programs, communicating, displaying) through three means:

- Peripheral Units
- Input-Output Units
- Bus

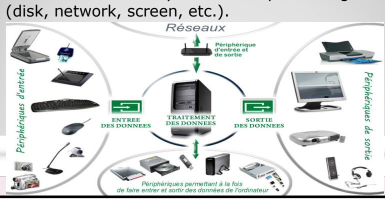
2. The Device

2.1. Definitions

- A **computer device** is a generic term used for hardware components that facilitate communication between the computer's central processing unit and the external world.
- **Input-Output** refers to the exchange of information between the processor and its associated devices:

Inputs are the data sent by a device (disk, network, keyboard, etc.) to the central processing unit;

Outputs are the data sent by the central processing unit to a device (disk, network, screen, etc.).



2. The Device

2.1 Types of Devices

Input Devices: These allow data or commands to be entered, which will be managed by the central processing unit. Examples include: keyboard, mouse, video camera, scanner, etc.

Output Devices: These allow the user to view the results of calculations or data manipulations performed by the computer. Examples include: monitor (screen), printer, projector, speakers, etc.

Input-Output Devices: Some devices allow bidirectional communication between the computer and the external world, and these are input-output devices. Examples include: floppy disk drives, CD-ROM (or DVD) drives, CD (or DVD) burners, USB drives, and modems.

.

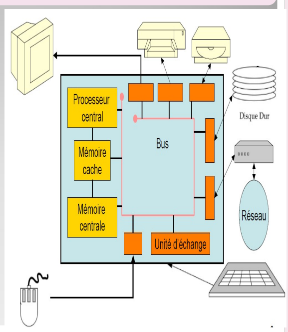
3. Device controller

A controller or device controller: Units that handle data exchange with the external world, often specialized in managing a particular device. Each device is associated with a controller.

Disk controller: A controller typically integrated into the board attached to a hard drive;

Keyboard controller;

USB controller.



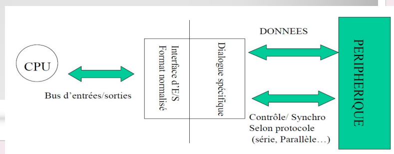
3. Device controller

Problem

The processor must manage devices that:
Serve different roles (input, output, storage),
Operate at different speeds,
Use different "languages."

Solution

Delegate this management to controllers that ensure:
Communication between processor modules and devices,
Adaptation of the diversity of devices (data rate, response time, data format, etc.) to a common interface adhering to standards defined by the manufacturer.



4. Architecture of a Controller

Composed of Three Types of Registers and Control Logic:

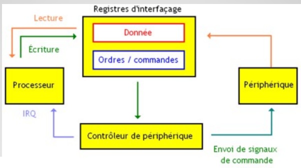
1. Data Register:
One or more registers that enable data exchange between the processor and peripherals.
Typically, there is a read register and a write register, but sometimes both are combined into a single register.

2. Command Register:
Stores the commands sent by the processor.
When the processor wants to send a command to the peripheral, it writes the desired command into this register (e.g., transfer direction, transfer mode).

3. Status Register:
Readable by the processor, it contains information about the state of the peripheral.
Indicates whether the peripheral is available, executing a command, or being used by another processor.

5. Operation of a Controller

➢ A **device controller** reads the commands sent by the processor, interprets them, and controls the peripheral to perform the requested action.
➢ The role of the device controller is to generate control signals that will trigger an action performed by the peripheral.
➢The device controller can be separate from the peripheral it controls, or integrated within the peripheral itself, as is the case for IDE hard drives, for example.

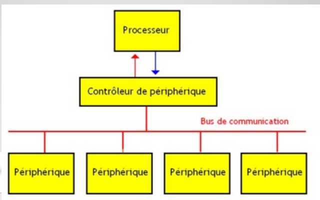


Remarque

- ✓ When a computer uses an operating system, it typically doesn't know how a peripheral or its controller operates.
- ✓ A program needs to be installed that runs when communication with the peripheral is required.
- ✓ This program takes care of everything necessary for data transfer, addressing the peripheral, etc.
- ✓ This small program is called a **driver** or **device driver**.

6. Special Case

- A device controller can be connected to multiple peripherals. For example, with USB: you have several USB ports on your computer, but they are managed by a single USB controller.
- Peripherals connected to the same controller can be different. This is the case for PCI controllers, USB controllers, and others. Through USB, you can connect USB flash drives, printers, scanners, DVD drives, and many others. Their adherence to the USB standard makes them compatible.



7. Interface for communication with the peripheral

Device Addressing

- The device controller must be able to identify each peripheral.
- For example, if a printer, a mouse, and a hard drive are connected via USB to a computer, when I start printing, the device controller must send the data to the printer, not the hard drive.
- To achieve this, it assigns each peripheral one or more addresses, used to identify and select it. Typically, peripherals have several addresses: one for each interface register.
- The address allows the controller to address the peripheral and specify which register to read from or write to.
- A peripheral's address can either be fixed once and for all during its design, or be configured through a register or EEPROM.

7. Interface for communication with the peripheral

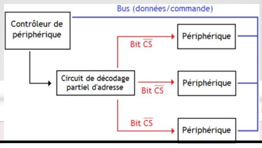
Device Addressing

- When the device controller sends a transmission over the bus, it must ensure that the data reaches its intended destination. Two solutions are possible:

Solution 1: Delegates this responsibility to the peripherals and memory. Each component connected to the bus checks whether the address sent by the processor matches its own. If it does, it connects to the bus, while the other components remain disconnected.

Solution 2: This involves **address decoding**. It uses a circuit that determines, based on the address, which component is being addressed. Only this component will be activated/connected to the bus, while the others will be deactivated/disconnected.

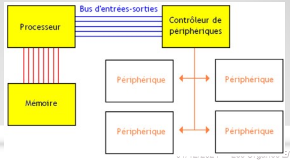
- To implement this last solution, each peripheral has a **CS (Chip Select)** input, which activates or deactivates the peripheral based on its value. To avoid conflicts on the bus, only one device controller should have its **CS bit** set to 1 at a time.



7. Interface for communication with the peripheral

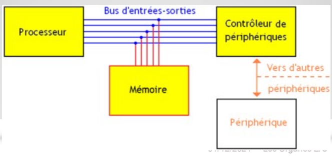
Buses Connected to the Device Controller

- The device controller acts as an intermediary between one or more peripherals and the rest of the computer.
- It is connected to the processor through a specialized bus called the **input/output bus**, which is typically a high-performance, fast, high-bandwidth bus.
- It is connected to the peripherals through another separate bus called the **secondary bus**, which is slower and has a lower data transfer rate.



7. Interface for communication with the peripheral

- On other computers, the bus that connects the processor to the device controller is the same bus that connects the processor to the RAM.
- **Advantage**: This saves wires, pins on the processor, and other resources. The wiring is simpler, and the manufacturing process is also easier.
- The bus that connects the processor, memory, and device controller is also called a **system bus**.



8. Communication Interface with the CPU

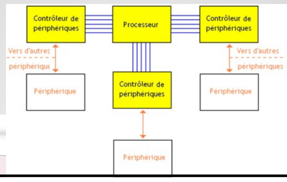
Problem?

A computer contains multiple device controllers. In practice, any modern computer contains dozens, if not hundreds, of device controllers. How can they be connected to the processor?

Solution

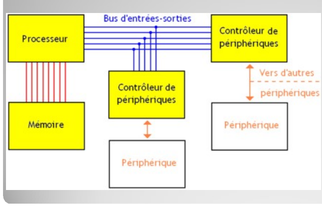
The simplest solution would be to connect each device controller to the processor with a point-to-point connection.

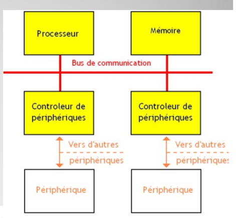
This solution requires a lot of wires and connections, making it impractical when there are many controllers to connect. It is mainly used in simpler computers.



8. Communication Interface with the CPU

✓ Connect multiple device controllers to the processor using a single input/output bus. However, this is rarely sufficient.
✓ Merge the input/output bus with the memory bus.

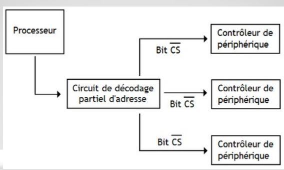




8. Communication Interface with the CPU

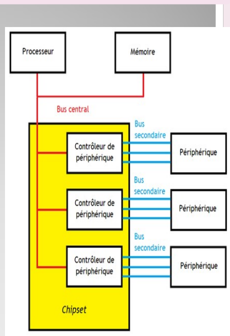
Device Controller Addressing

- Using a single bus to connect the processor to multiple device controllers brings us back to the same problem as before.
- The solution is the same: each device controller has its own memory address.
- The device controllers are either placed within the peripherals themselves or on the motherboard.



8. Communication Interface with the CPU

- Many peripherals that integrate their own controller are common, such as hard drives, sound cards, or video cards.
- However, there are still many controllers placed on the motherboard.
- These are typically grouped into a single circuit called the **chipset**, with a few exceptions.
- Thanks to the **chipset**, each peripheral has its own dedicated bus, which is generally tailored to the specific needs of the peripheral.
- The **chipset** integrates the address decoding and/or controller selection mechanisms mentioned earlier.
- It is responsible for redirecting commands/data to the appropriate controller.



9. Orders Executed by a Controller

The orders executed by a device controller depend on the type of peripheral it controls and the specific instructions needed to perform its functions. These can be summarized as follows:

Data Reading: Read data from the peripheral and transfer it to the main memory or processor.

Data Writing: Write data from the main memory or processor to the peripheral.

Configuration Commands: Configure the peripheral's operation, such as transmission speed settings, operating modes, etc.

Interrupt Management: Manage interrupts, such as enabling or disabling interrupts from the peripheral.

State Control: Query or configure the peripheral's state, such as checking data availability, error detection, etc.

9. Ordres exécutés par un controleur

- **Movement Commands (if applicable)**: For peripherals such as motors or actuators, movement commands can be issued to move objects or perform physical operations.
- **Timing Commands**: To manage time-synchronized operations, such as delays or timeouts.
- **Reset and Diagnostics**: To reset the peripheral, perform self-tests, or diagnose potential issues.
- **Power Management**: To activate or deactivate power-saving modes or adjust the peripheral's power consumption.
- **Flow Control**: To manage the data flow between the peripheral and main memory, for example, enabling or disabling data transfers.