

---

# Introduction to Embedded Systems

Dr. Hassanein Shaban Ahmed

---

---

# What's a System?

**A system is a way of working, organizing or doing one or many tasks according to a fixed plan, program or set of rules.**

**A system is also an arrangement in which all its units assemble and work together according to the plan or program.**

# What's a System?

## SYSTEM EXAMPLES

### WATCH

It is a time display **SYSTEM**

Parts: Hardware, Needles, Battery, Dial,  
Chassis and Strap

#### Rules

- 6. All needles move clockwise only
- 7. A thin needle rotates every second
- 8. A long needle rotates every minute
- 9. A short needle rotates every hour
- 10. All needles return to the original position after 12 hours



# What's a System?

## SYSTEM EXAMPLES

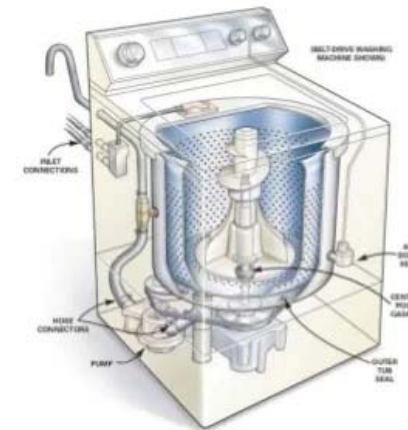
### WASHING MACHINE

It is an automatic clothes washing **SYSTEM**

**Parts:** Status display panel, Switches & Dials, Motor, Power supply & control unit, Inner water level sensor and solenoid valve.

#### Rules

5. Wash by spinning
6. Rinse
7. Drying
8. Wash over by blinking
9. Each step display the process stage
10. In case interruption, execute only the remaining



# What's an Embedded System?

- An embedded system is a computer system embedded in a device with a dedicated function
- This is different from the traditional, general purpose computer systems



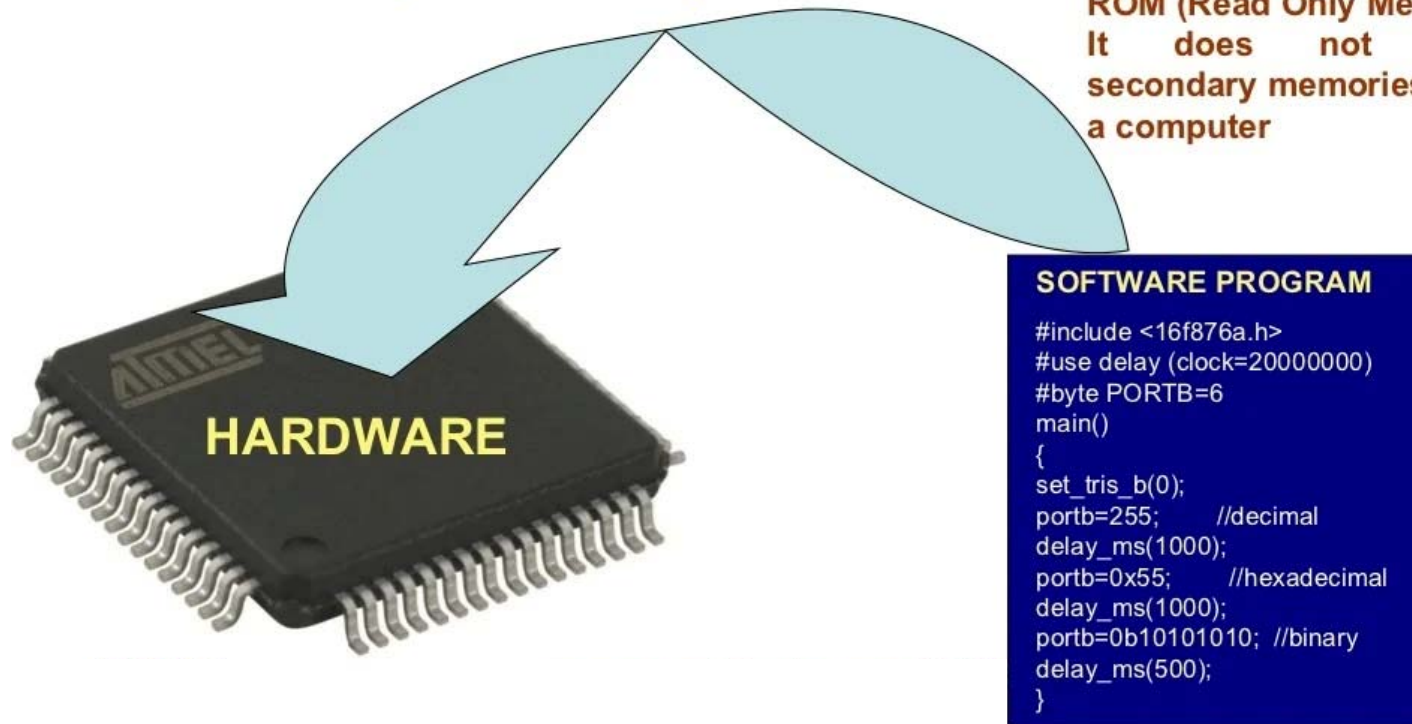


# What's an Embedded System?

## EMBEDDED SYSTEM

**Definition: An Embedded System is one that has computer hardware with software embedded in it as one of its important components.**

Its software embeds in ROM (Read Only Memory). It does not need secondary memories as in a computer



# What's an Embedded System?

- Embedded systems =
  - information processing systems embedded into a larger product
- Two types of computing
  - Desktop – produced millions/year
  - Embedded – billions/year
- Non-Embedded Systems
  - PCs, servers, and notebooks
- The future of computing!
  - Automobiles, communication, aviation, handheld devices, military and medical equipments.



# EXAMPLES OF EMBEDDED SYSTEMS

## Phones and Tablets

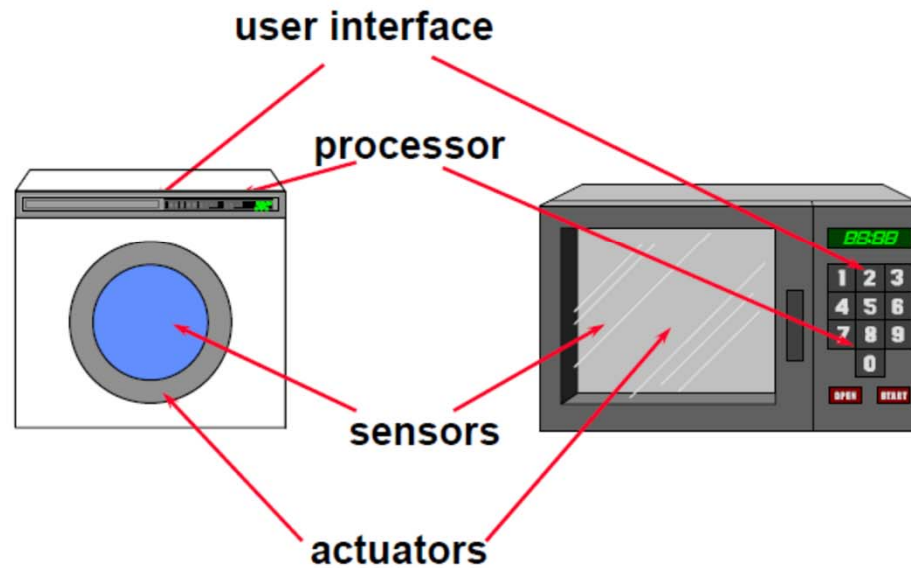


- Communication Processor (s)
  - Wifi
  - GSM/3G/LTE
  - Bluetooth/NFC
- Graphics Processor (s)
  - Graphics and Video Processing
- Application Processor
  - Android / Windows / iOS



# EXAMPLES OF EMBEDDED SYSTEMS

Consumer electronics, for example MP3 Audio, digital camera, home electronics, ... .



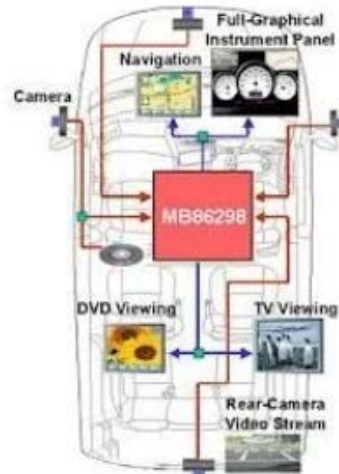
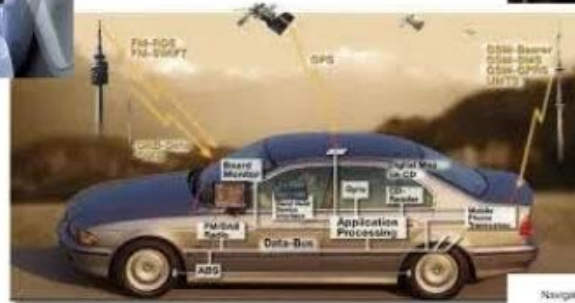
# EXAMPLES OF EMBEDDED SYSTEMS

## Robotics



# EXAMPLES OF EMBEDDED SYSTEMS

## Automotive



# EXAMPLES OF EMBEDDED SYSTEMS



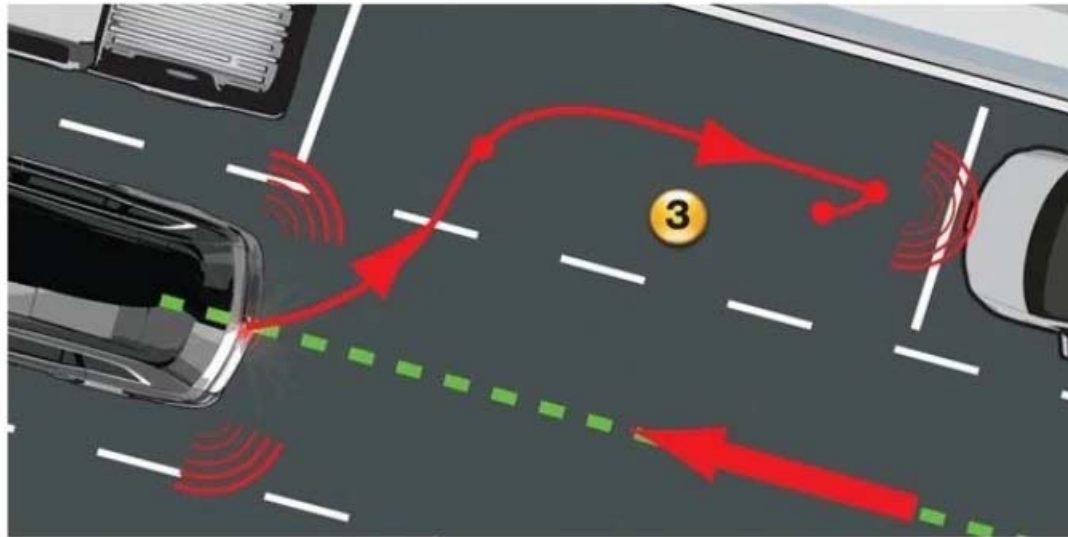
AUTOMOTIVE



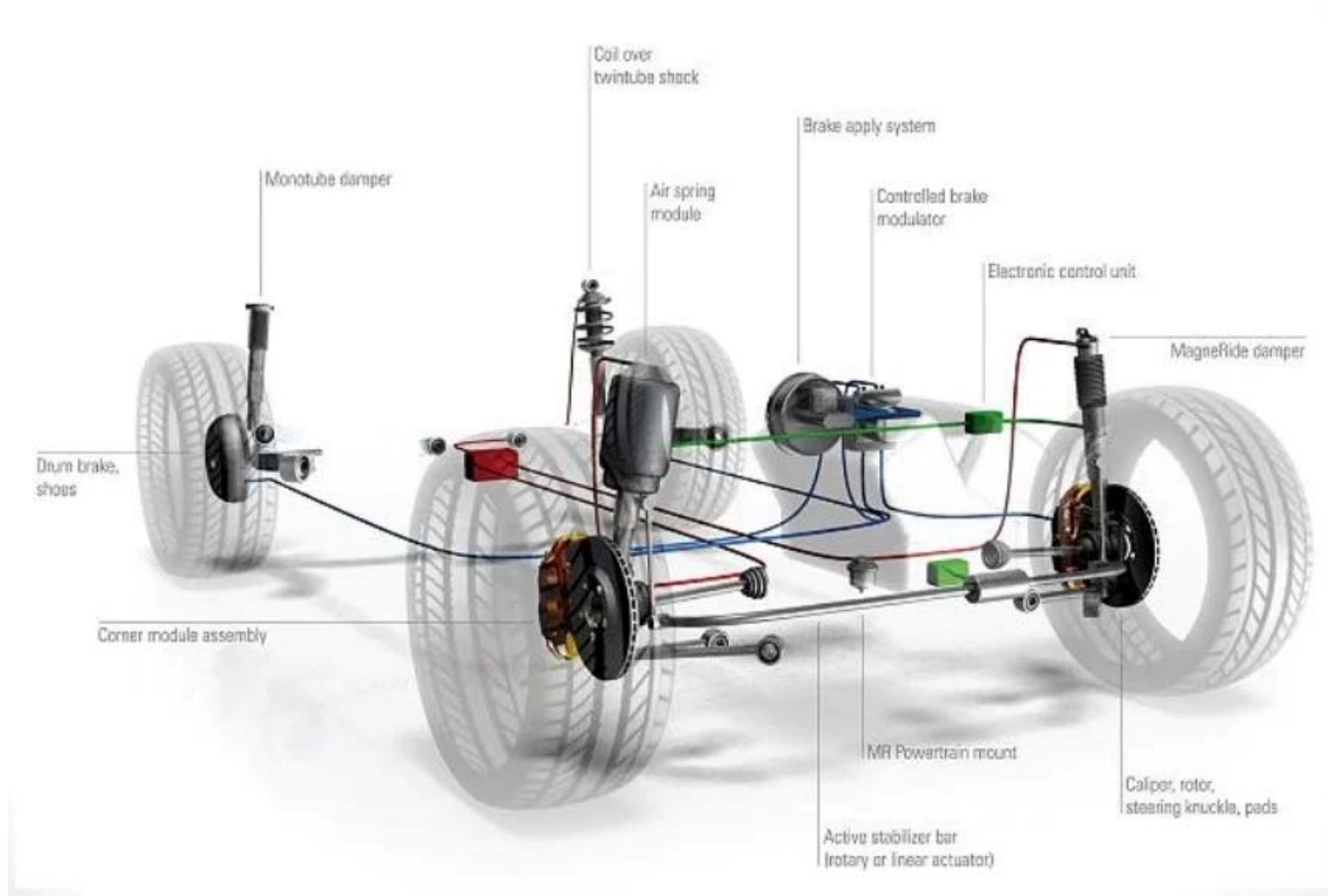
# EXAMPLES OF EMBEDDED SYSTEMS

## Cars Are Getting Smarter...

- Electronics represents 40% of total cost of a car
- 90% of new car features require software



# EXAMPLES OF EMBEDDED SYSTEMS



## Embedded Systems in the Powertrain

# EXAMPLES OF EMBEDDED SYSTEMS



# EXAMPLES OF EMBEDDED SYSTEMS

## Network Devices



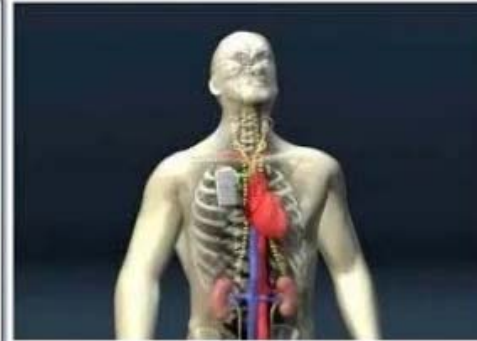
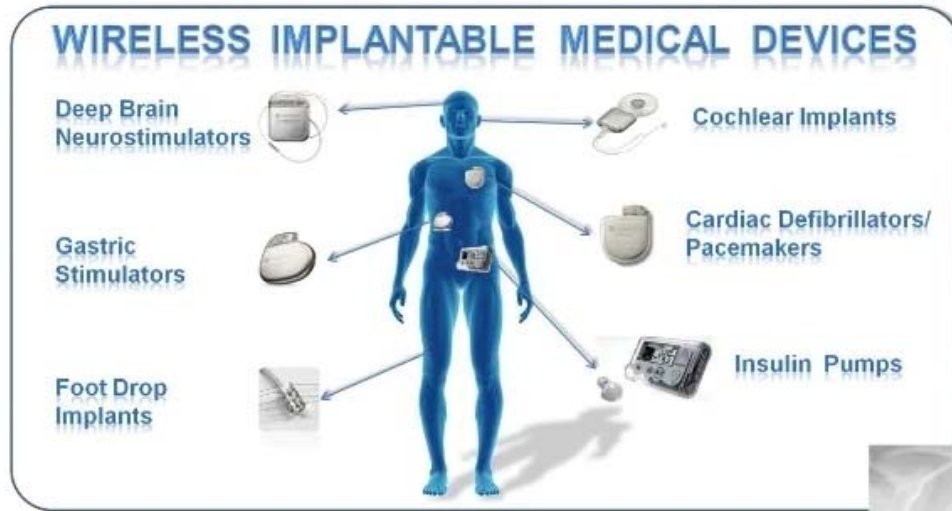


# EXAMPLES OF EMBEDDED SYSTEMS



## Medical Devices

# EXAMPLES OF EMBEDDED SYSTEMS



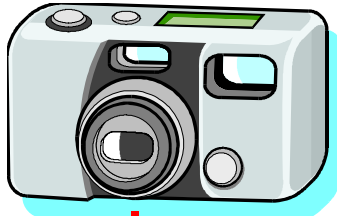
## Implantable Medical Devices

# EXAMPLES OF EMBEDDED SYSTEMS

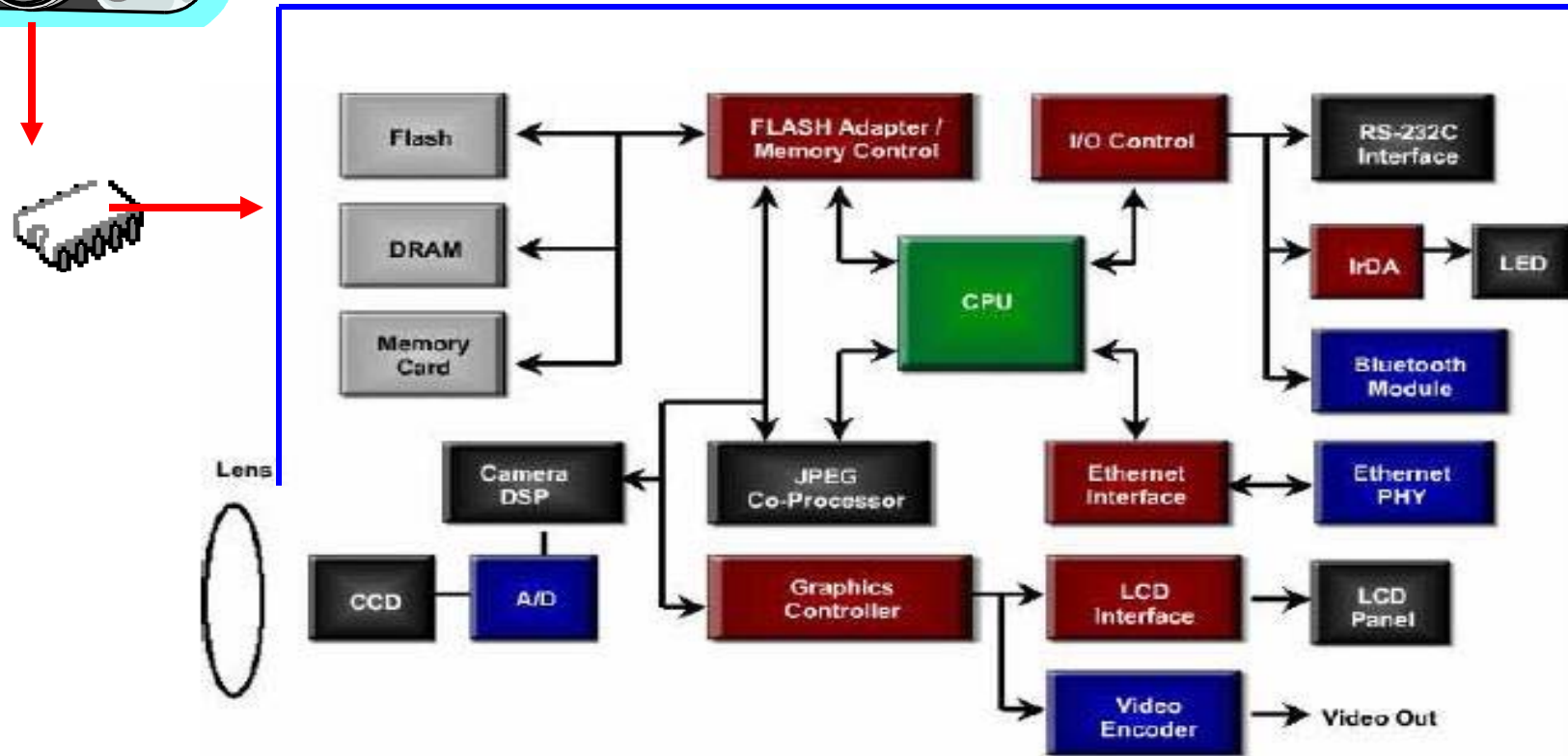
## And Other Gadgets



# An Example Embedded System

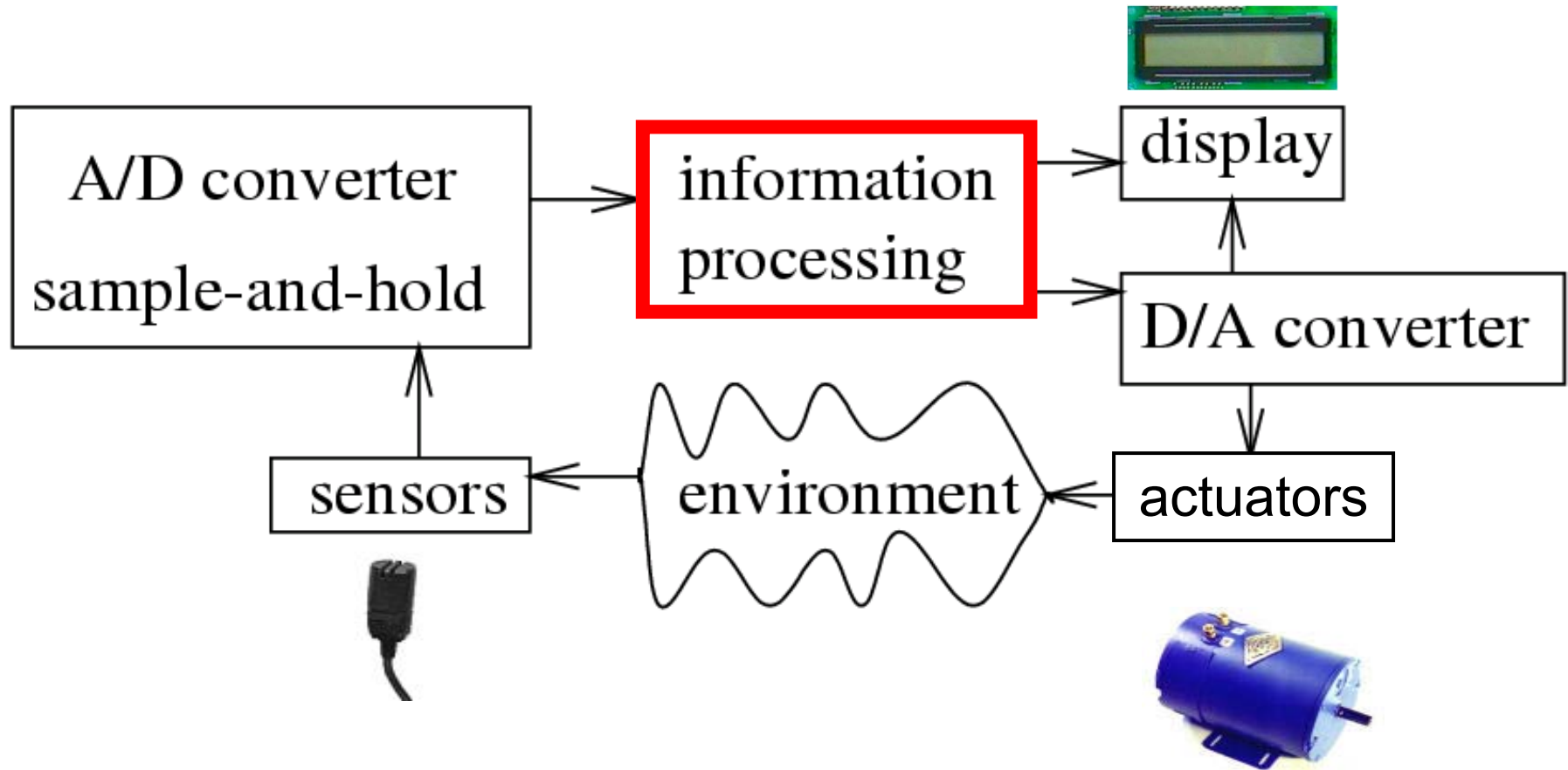


**Digital Camera Block Diagram**





# ES: Simplified Block Diagram



# Components of Embedded Systems

- Analog Components

- Sensors, Actuators, Controllers, ...

- Digital Components

- Processor, Coprocessors
  - Memories
  - Controllers, Buses
  - Application Specific Integrated Circuits (ASIC)

- Converters – A2D, D2A, ...

- Software

- Application Programs
  - Exception Handlers

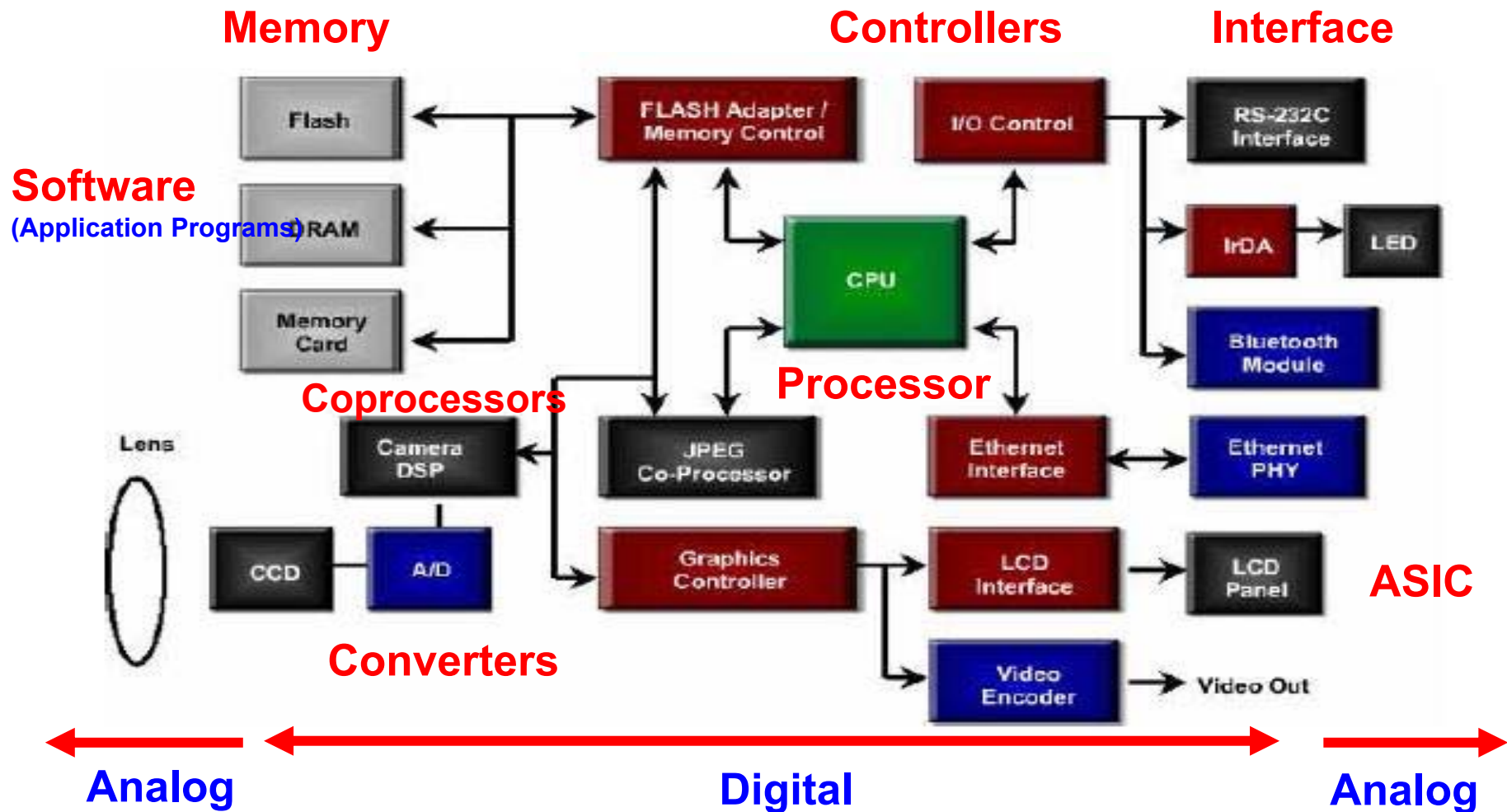
**Hardware**

**Software**

# Hardware Components

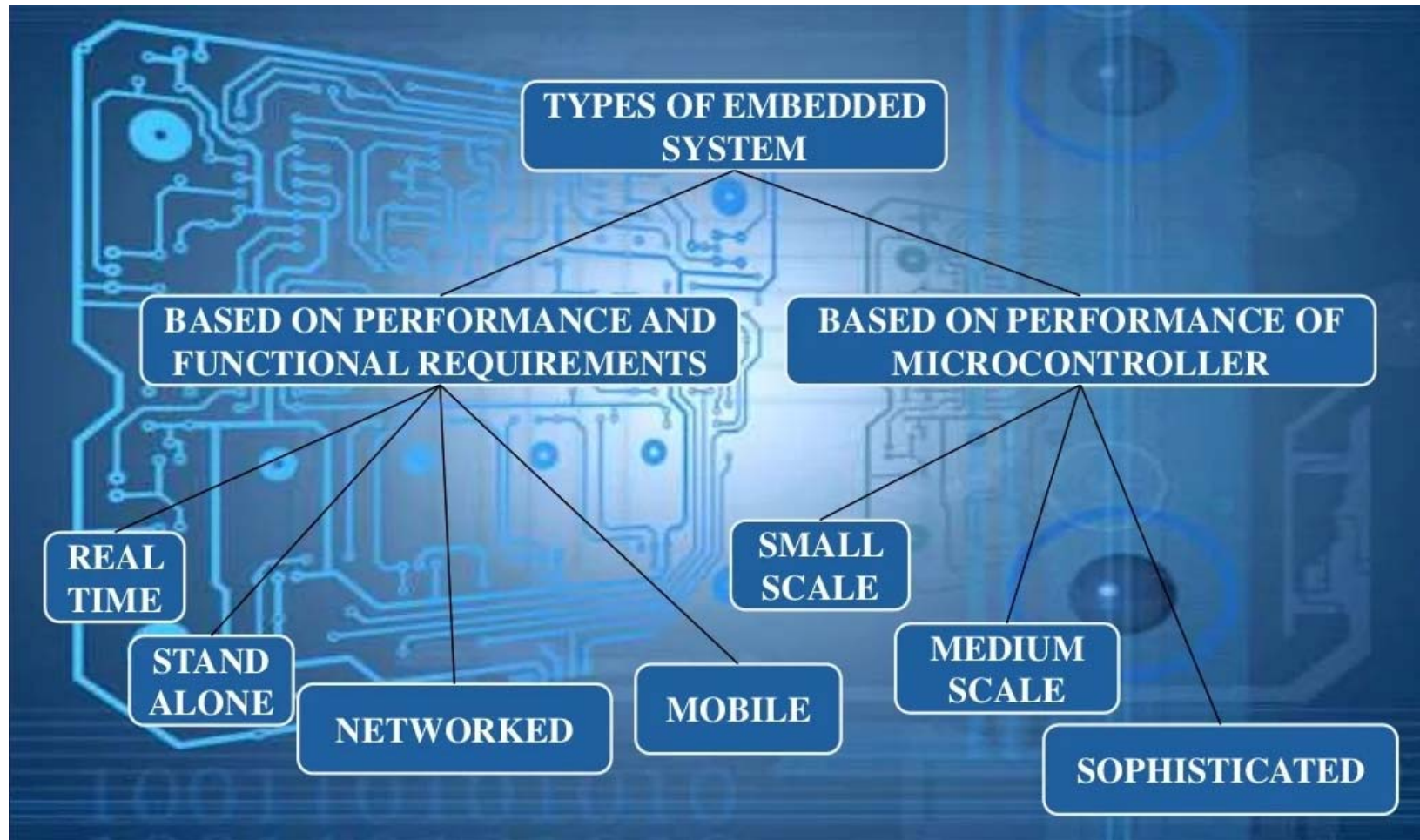
# Hardware Components of Embedded Systems

## an example





# CLASSIFICATION OF EMBEDDED SYSTEMS



<https://www.geeksforgeeks.org/classification-of-embedded-systems>

<https://www.theengineeringprojects.com/2021/05/types-of-embedded-systems.html>

<https://www.electrically4u.com/classification-of-embedded-system>

---

# BASED ON PERFORMANCE AND FUNCTIONAL REQUIREMENTS

# 1. Real Time Embedded Systems

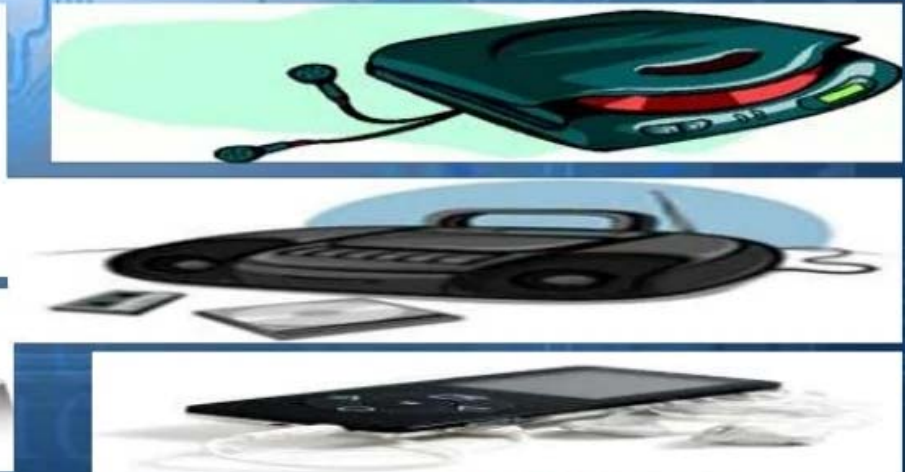
Real-Time embedded systems are defined as those systems in which the correctness of the system depends not only on the logical result of computation, but also on the time at which the results are produced. For example, :

- Hard real-time systems ( e.g., Avionic control ).
- Firm real-time systems ( e.g., Banking ).
- Soft real-time systems ( e.g., Video on demand ).



## 2. Stand Alone Embedded Systems

- A standalone embedded system (device) is able to function independently of other hardware. This means it is not integrated into another device.
- It takes the input from the input ports either analog or digital and processes, calculates, and converts the data and gives the resulting data through the connected device- Which either controls, drives, and displays the connected devices.
- For example, a TiVo box that can record television programs, mp3 players are standalone devices





### 3. Networked Embedded Systems

- *These types of embedded systems are related to a network to access the resources.*
- *The connected network can be LAN, WAN or the internet. The connection can be any wired or wireless. This type of embedded system is the fastest growing area in embedded system applications. .*



## 4. Mobile Embedded Systems

- *Mobile embedded systems are used in portable embedded devices like cell phones, mobiles, digital cameras, mp3 players and personal digital assistants, etc.*
- *The basic limitation of these devices is the other resources and limitation of memory.*



## Based on the Performance of the Microcontroller

### *Small Scale Embedded Systems*

- *These types of embedded systems are designed with a single 8 or 16-bit microcontroller, that may even be activated by a battery.*
- *For developing embedded software for small scale embedded systems, the main programming tools are an editor, assembler, cross assembler and integrated development environment (IDE).*

### *Medium Scale Embedded Systems*

- **Medium Scale Embedded Systems are designed using a single 16-bit or 32-bit microcontroller, RISCs or DSPs.**
- **These medium Scale Embedded Systems are faster than that of small Scale Embedded Systems. Integration of hardware and software is complex in these systems.**
- *For developing embedded software for medium scale embedded systems, the main programming tools are C, C++, JAVA, Visual C++, RTOS, debugger, source code engineering tool, simulator and IDE.*



## Based on the Performance of the Microcontroller

### *Sophisticated Embedded Systems*

- Sophisticated or Complex Embedded Systems are designed using multiple 32-bit or 64-bit micro-controller. These systems are developed to perform large scale complex functions. These systems have high hardware and software complexities. We use both hardware and software components to design final systems or hardware products.





# RISC vs. CISC Instruction Sets

- Two fundamentally different approaches for instructions sets:

Reduced Instruction Set Computers (RISC)

Complex Instruction Set Computers (CISC)

RISC	CISC
<ol style="list-style-type: none"><li>1. Simple addressing modes</li><li>2. All instructions fitting in a single word</li><li>3. Single operation/instruction</li><li>4. Arithmetic/logic operations on registers</li><li>5. Load/Store architecture for data transfers</li><li>6. More instructions executed per program</li></ol>	<ol style="list-style-type: none"><li>1. More complex addressing modes</li><li>2. Instructions can span more than one word</li><li>3. More operations/instructions</li><li>4. Arithmetic/logic can operate on memory</li><li>5. Memory-to-memory data transfers are allowed</li><li>6. Fewer instructions executed per program</li></ol>
Simpler instructions make it easier to design faster hardware (e.g., use of pipelining)	Complexity makes it somewhat more difficult to design fast hardware, but still possible

# Embedded Systems Platforms Classification

- There are two main families of embedded system platforms:
  - Microcontroller Family
  - Microprocessor Family

# Embedded Systems Classification

## Microcontrollers

- Examples: PIC (MicroChip), AVR (Atmel), ...
- Used for example in Arduino Boards
- Originally 8/16 bit but recently there are 32 bit chips
- Simple instruction set
- No or simple OS Support
- Limited performance (clock speed up to 10s MHz)
- Programming in assembly, or C
- Useful in small systems with lower Cost
- Typical usage:
  - Interfacing to sensors
  - Control of motors in simple robotics systems
  - Simple home automation
  - etc...

# Embedded Systems Classification

## Microprocessors

- Examples: ARM, Intel ATOM, MIPS
- Used for example in Raspberry Pi, BeagleBone Black, ...
- 32 bit (and sometimes 64 bits)
- Support Linux and other RTOSs
- Higher performance (clock speed in 100s MHz to few GHz)
- Programming in C/C++ (sometimes with little assembly), Java, Python, ...
- Strong library support (act as a small computer)
- Useful in more complicated systems but with higher cost
- Typical Usage:
  - All what the microcontroller can do
  - Sophisticated control systems
  - Audio Processing
  - Image Processing
  - Video Processing
  - Communication Systems
  - Advanced guidance and navigation systems

---

## Explain microcontroller as the heart of embedded system

- Microcontroller is an IC chip that takes input process data according to program written in its memory and gives output as control signal for controlling other machines and devices.



---

## Explain microcontroller as the heart of embedded system

- A microcontroller (sometimes abbreviated  $\mu C$ ,  $uC$  or MCU) is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals.

---

## Explain microcontroller as the heart of embedded system

- A microcontroller (sometimes abbreviated  $\mu C$ ,  $uC$  or MCU) is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals.

---

# Differentiate Microcontroller and microprocessor

## Microprocessor :

- ❑ Microprocessor is the heart of any processing device. Its a basic building block of modern processors and controllers.
- ❑ Its a register based multi-purpose electronics device which takes input from us, process that input data according to the program written in external memory and gives us useful results.

---

## Differentiate Microcontroller and microprocessor

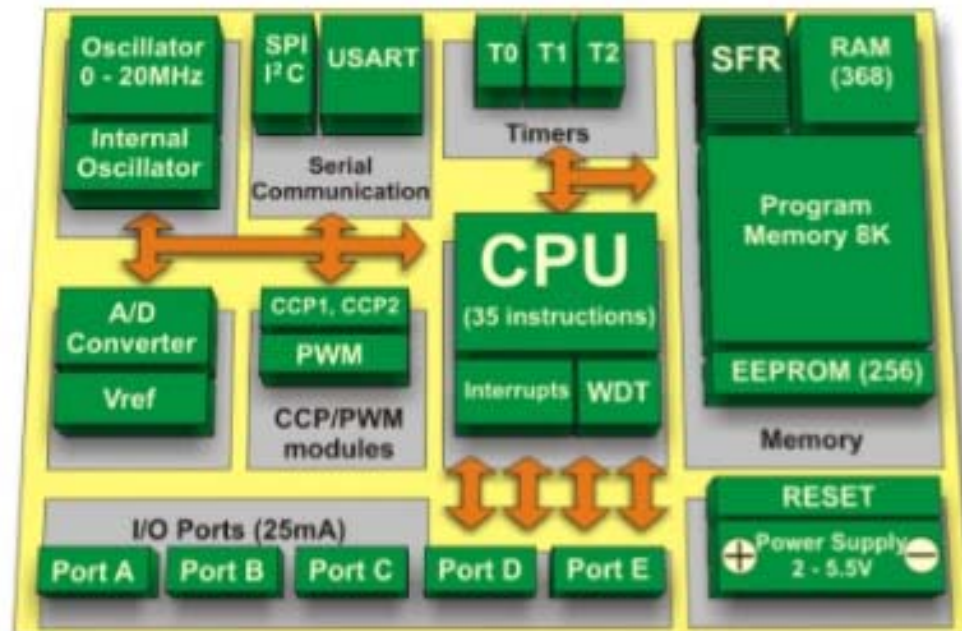
### Microprocessor :

- ❑ This device is only consists of processing unit, that is Memory and I/O devices are need to be connected externally.
- ❑ As it requires external memory and I/O devices so it requires large space and is larger in size. It is of no use without interfacing with external memory and I/O ports.

# Differentiate Microcontroller and microprocessor

## Microcontroller :

- ❑ Microcontroller is also like a Microprocessor except that a Microcontroller made by Integrating Memory and I/O ports on a single chip.
- ❑ It doesn't require external ROM, I/O ports for its operation. As memory such as ROM/RAM is integrated on a single IC chip, thus it is small in size.





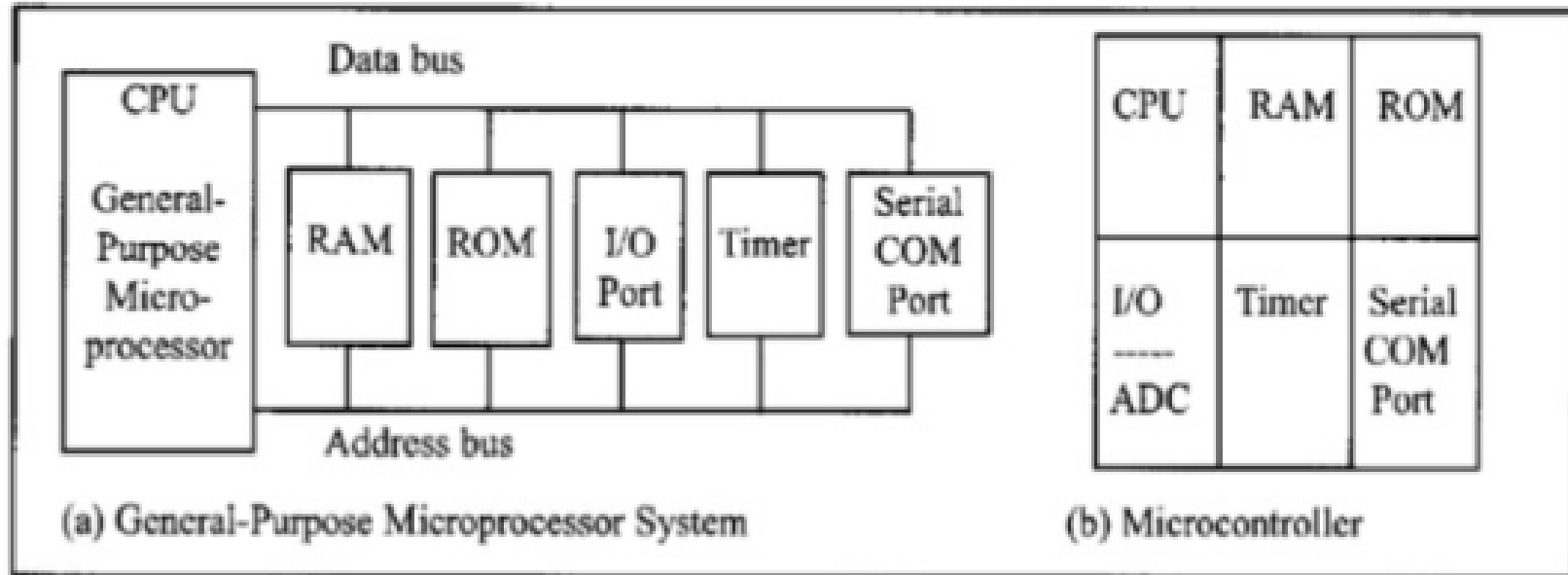
---

# Differentiate Microcontroller and microprocessor

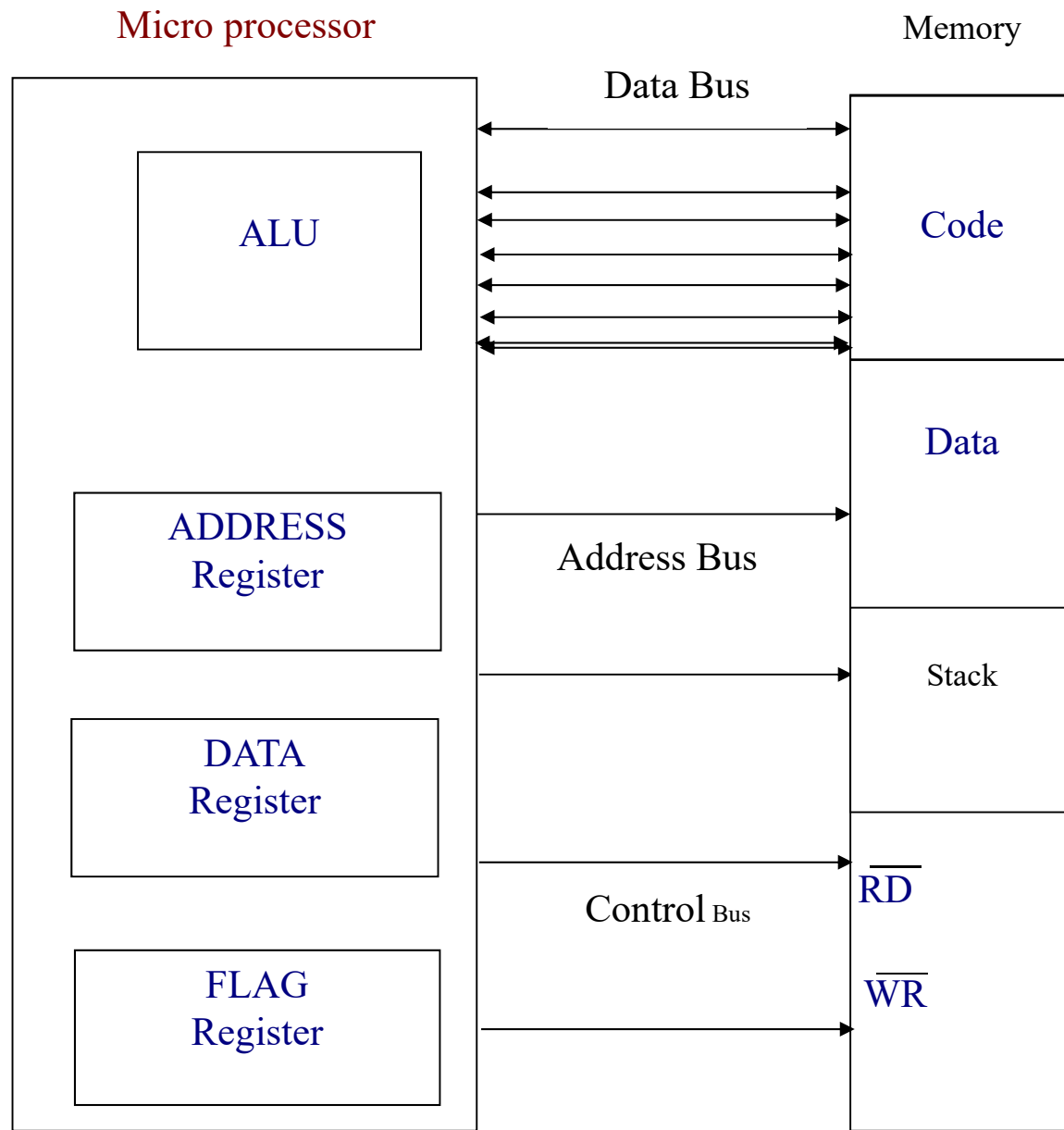
Microcontroller :

- ❑ It is basically used for controlling various machines. Programming of both Microcontroller And Microprocessor is almost similar.

# Differentiate Microcontroller and microprocessor

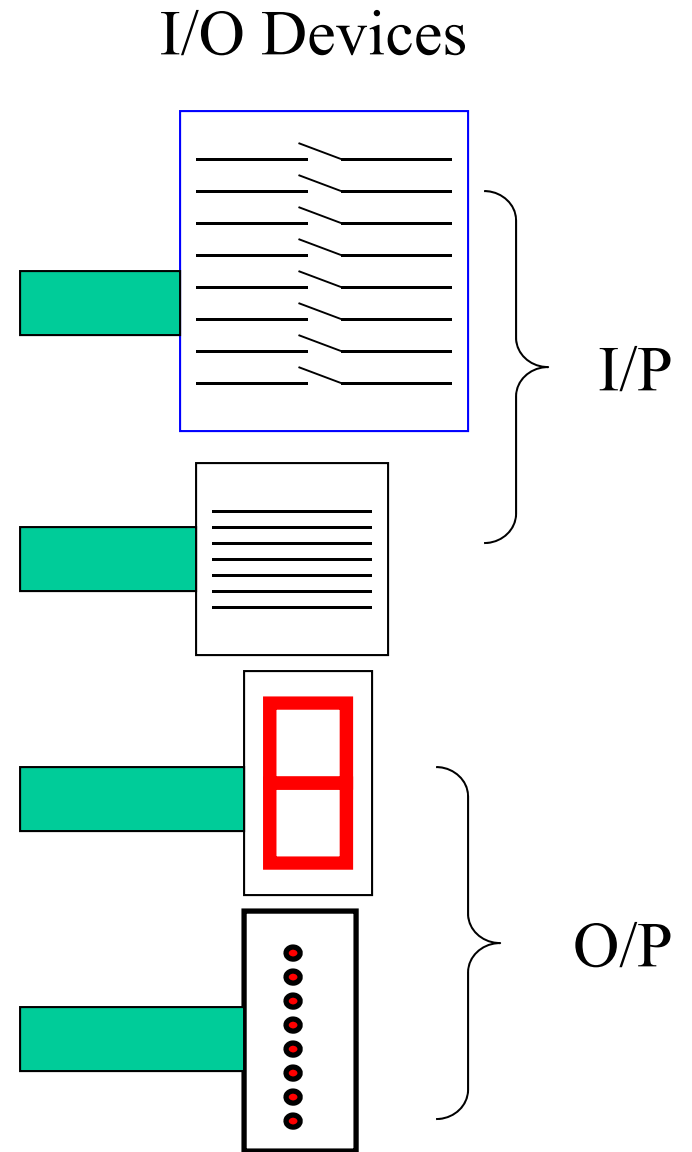


# Micro processor



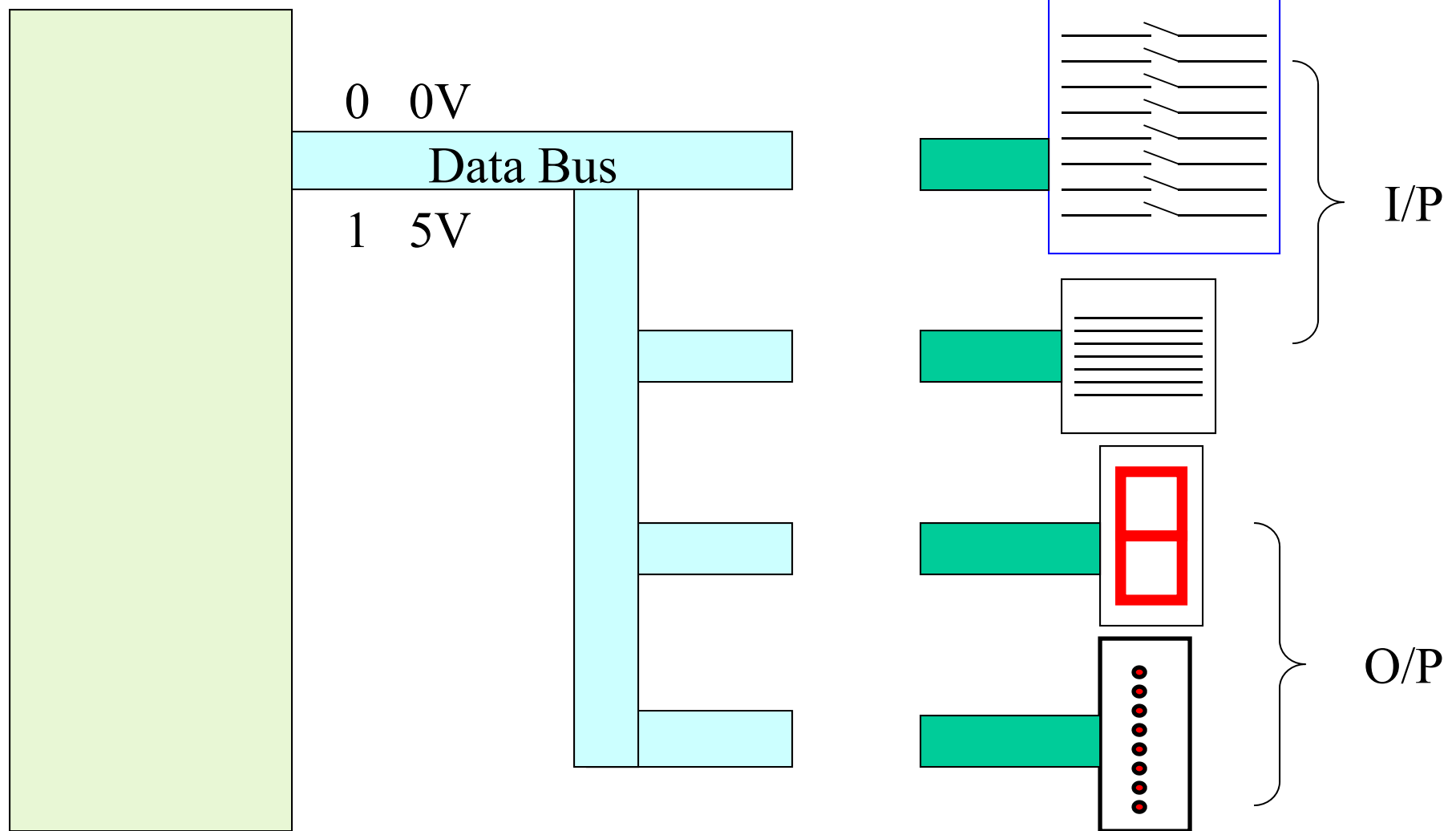
# Input and Output Ports

---



## Microprocessor

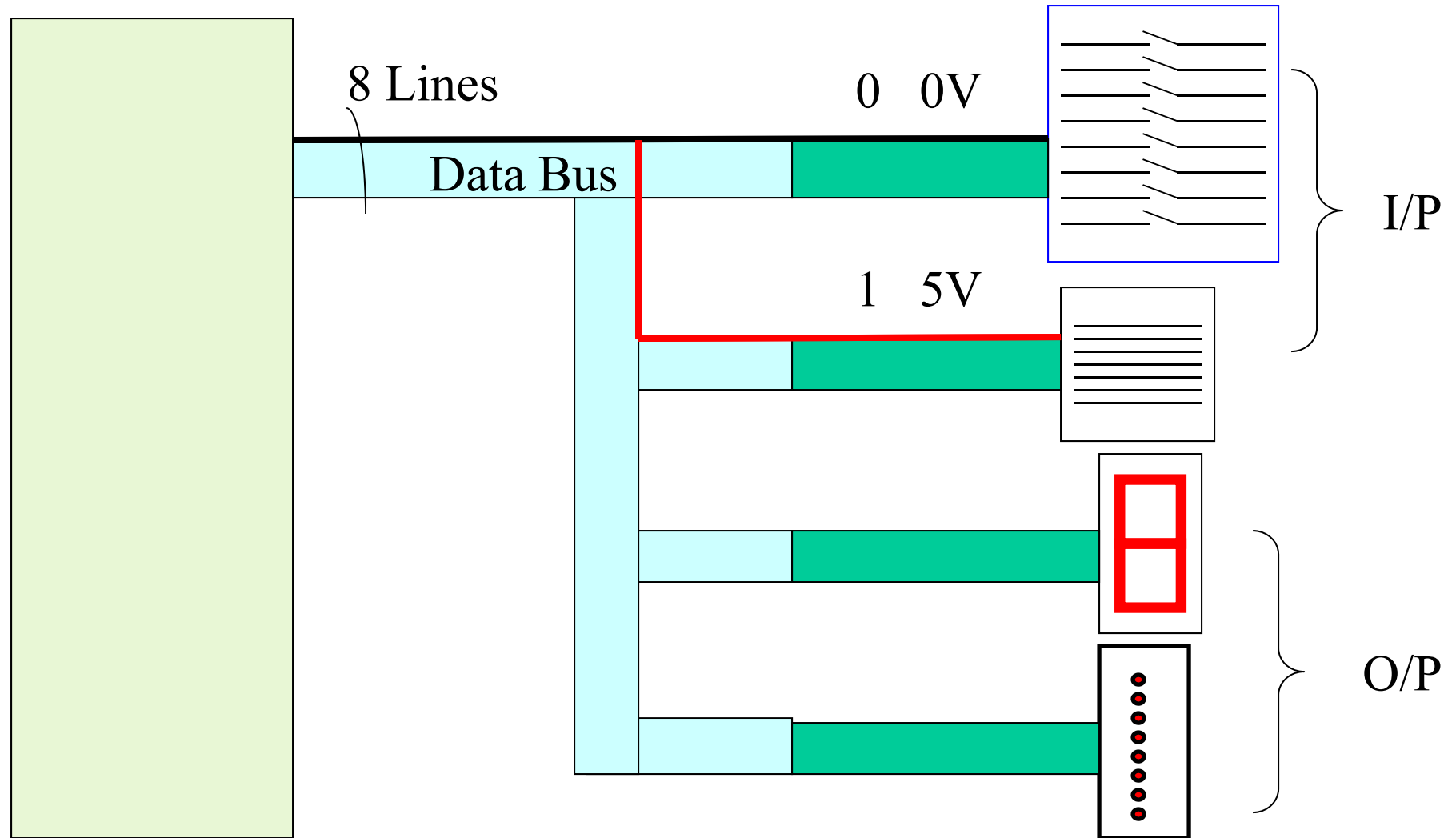
## I/O Devices





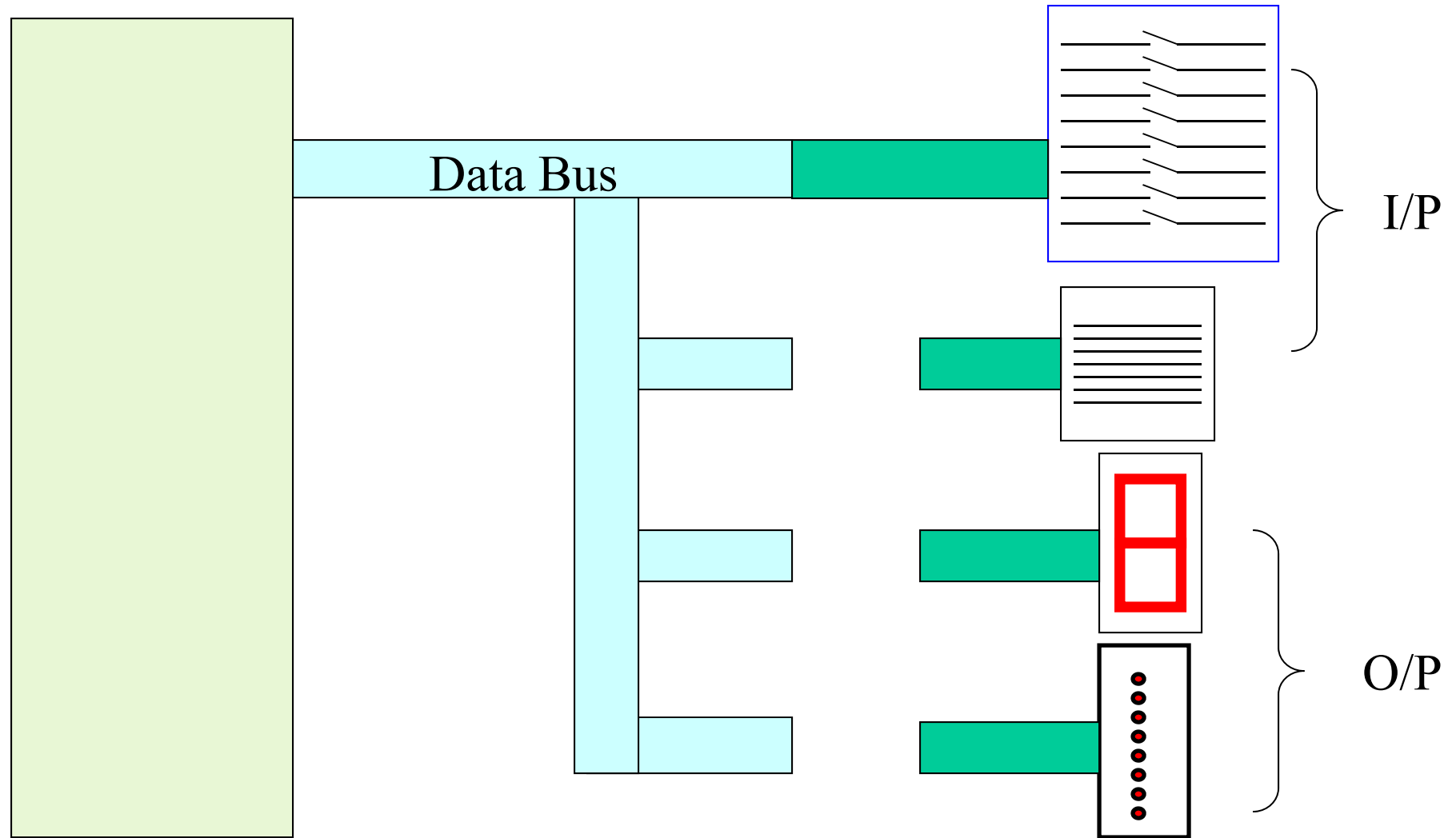
# Microprocessor

# I/O Devices



# Microprocessor

# I/O Devices

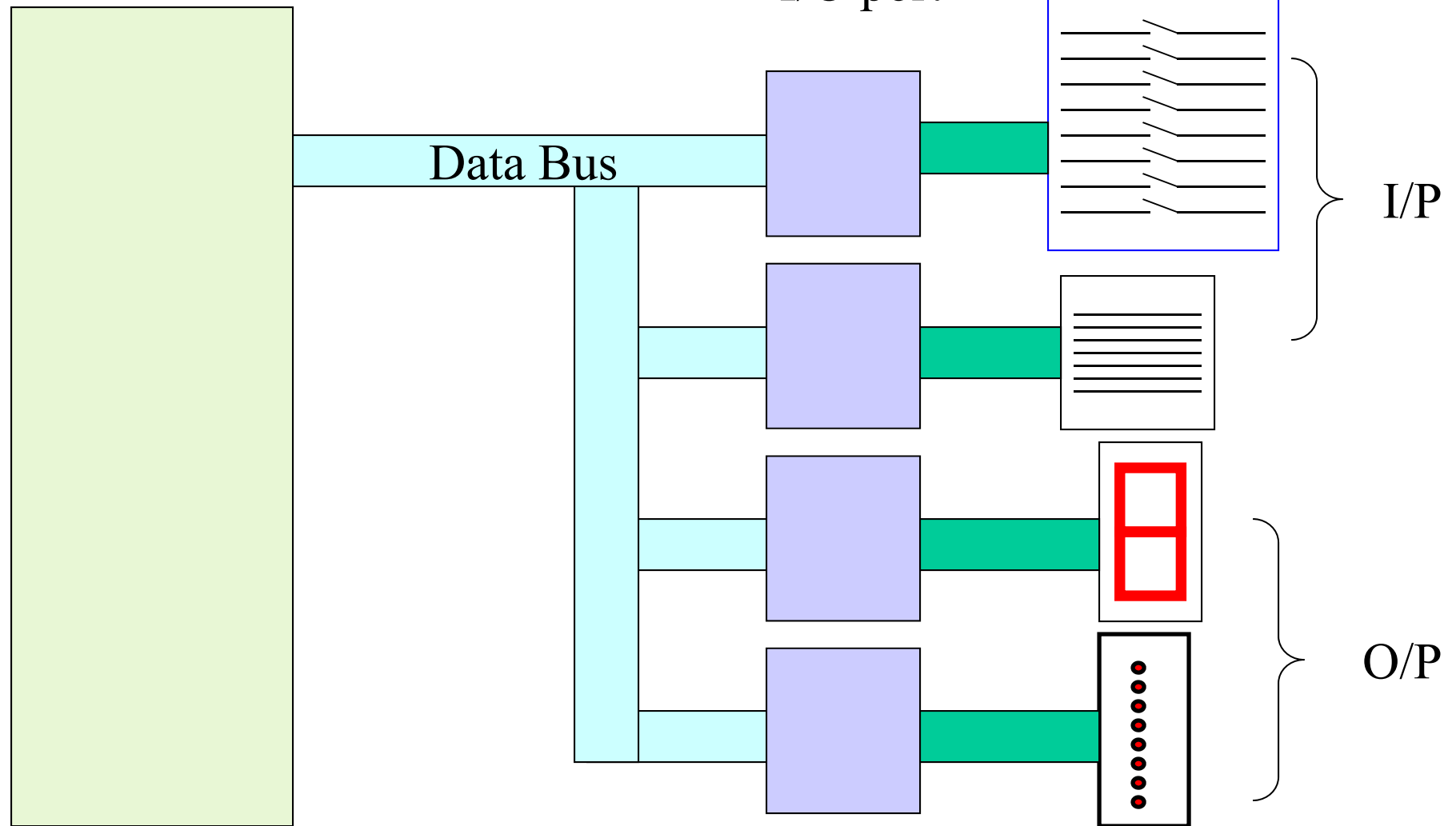


---

Microprocessor

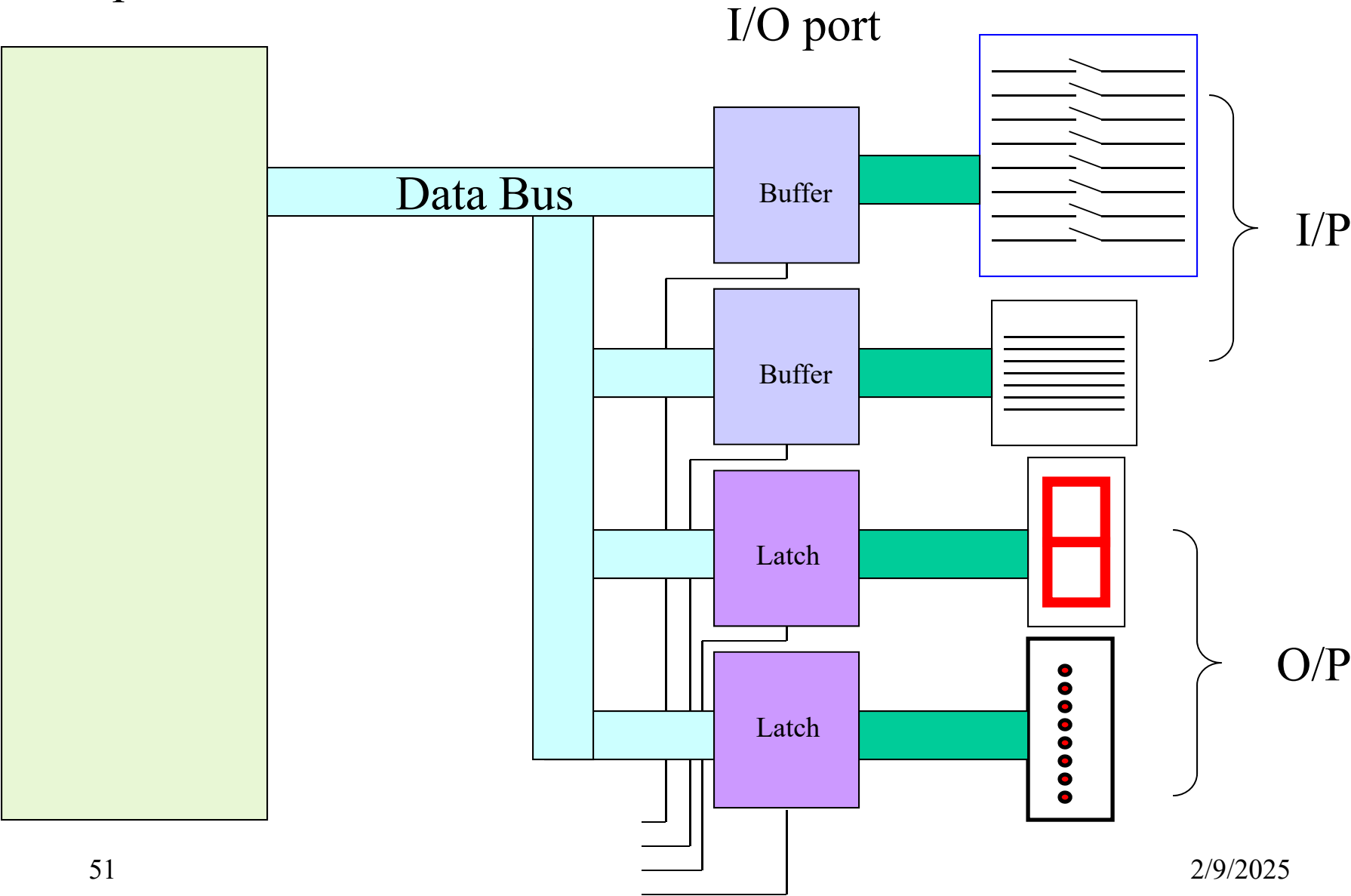
I/O Devices

I/O port



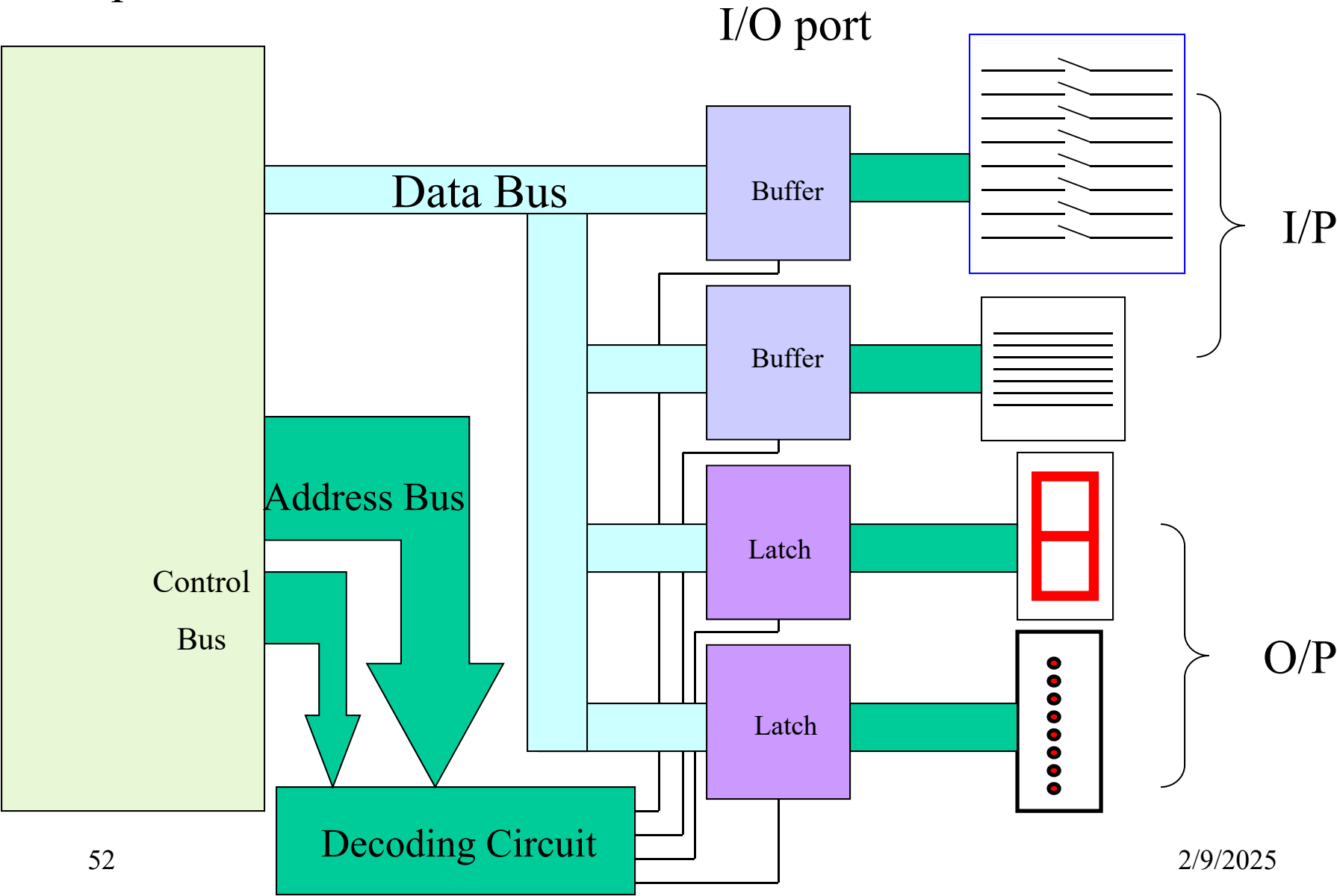
# Microprocessor

# I/O Devices



# Microprocessor

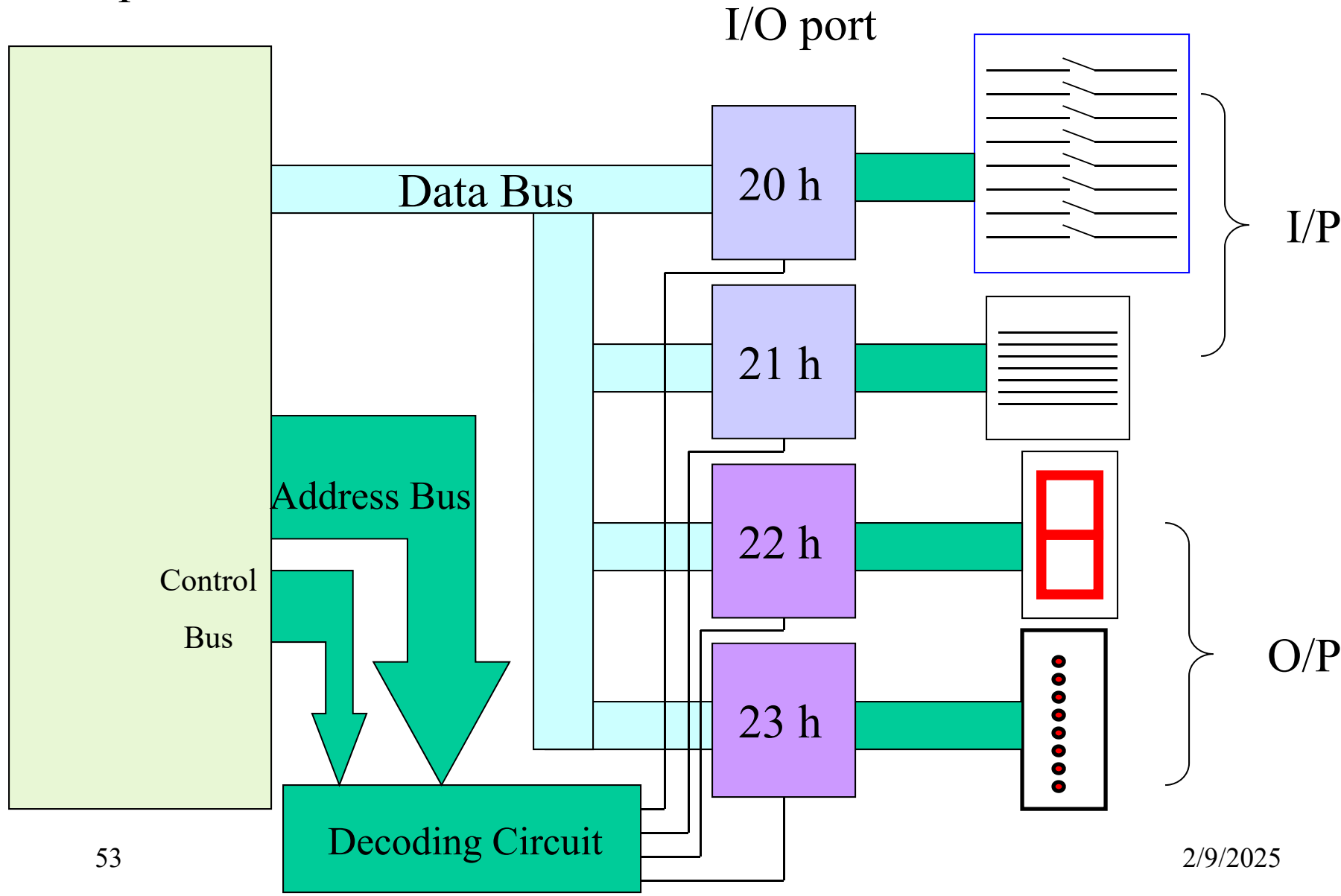
# I/O Devices





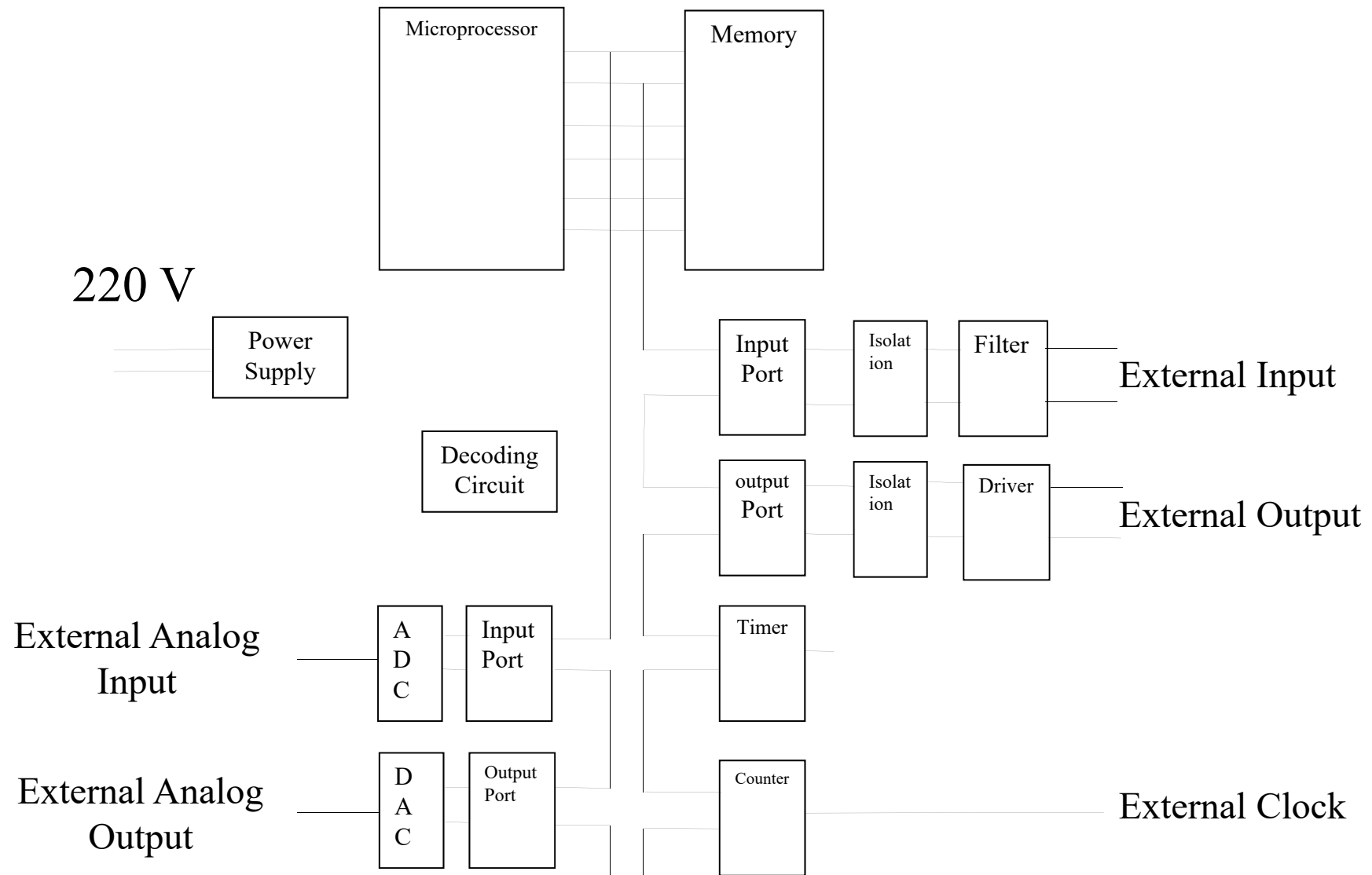
# Microprocessor

# I/O Devices

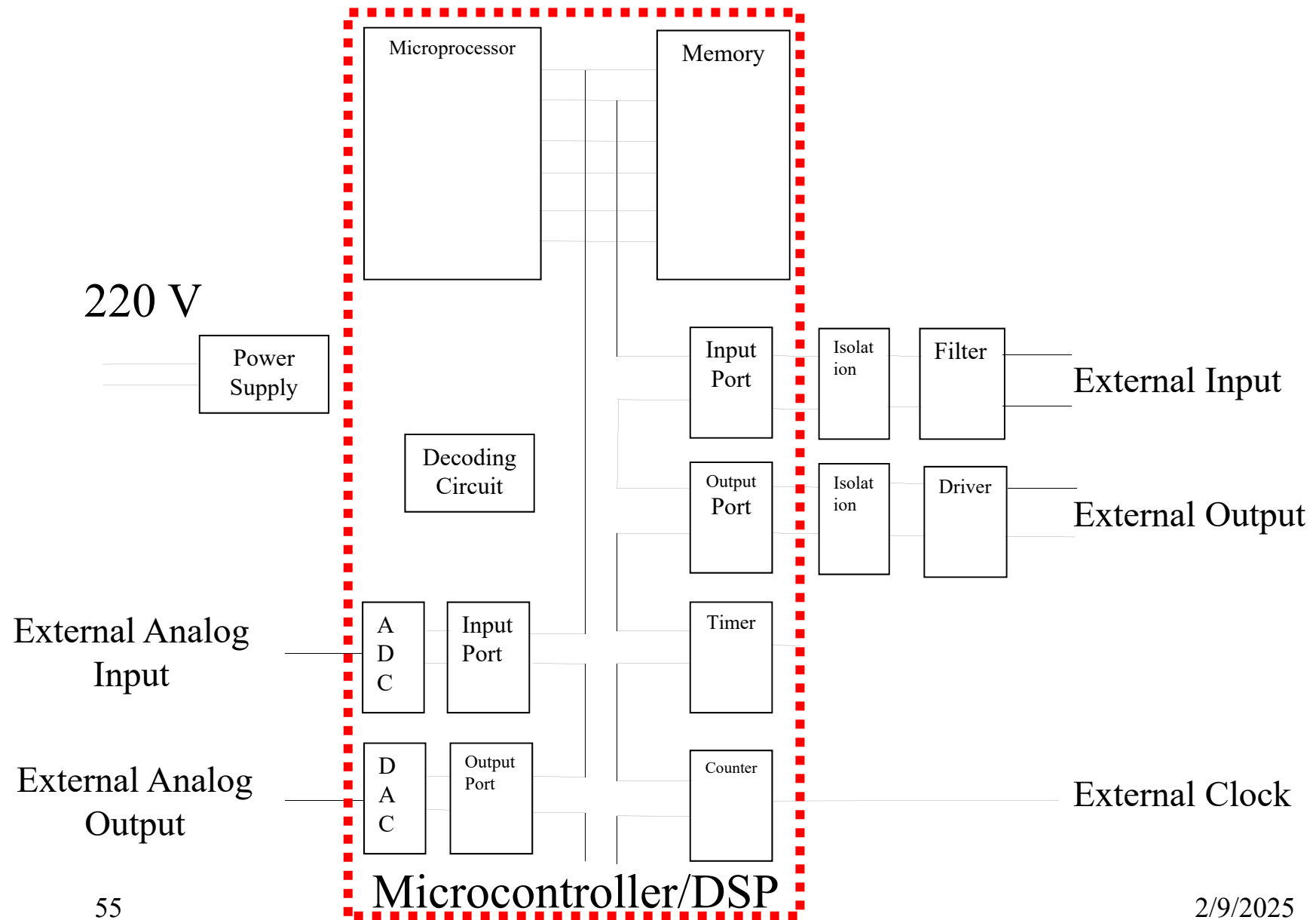


# Microprocessor Based Control System

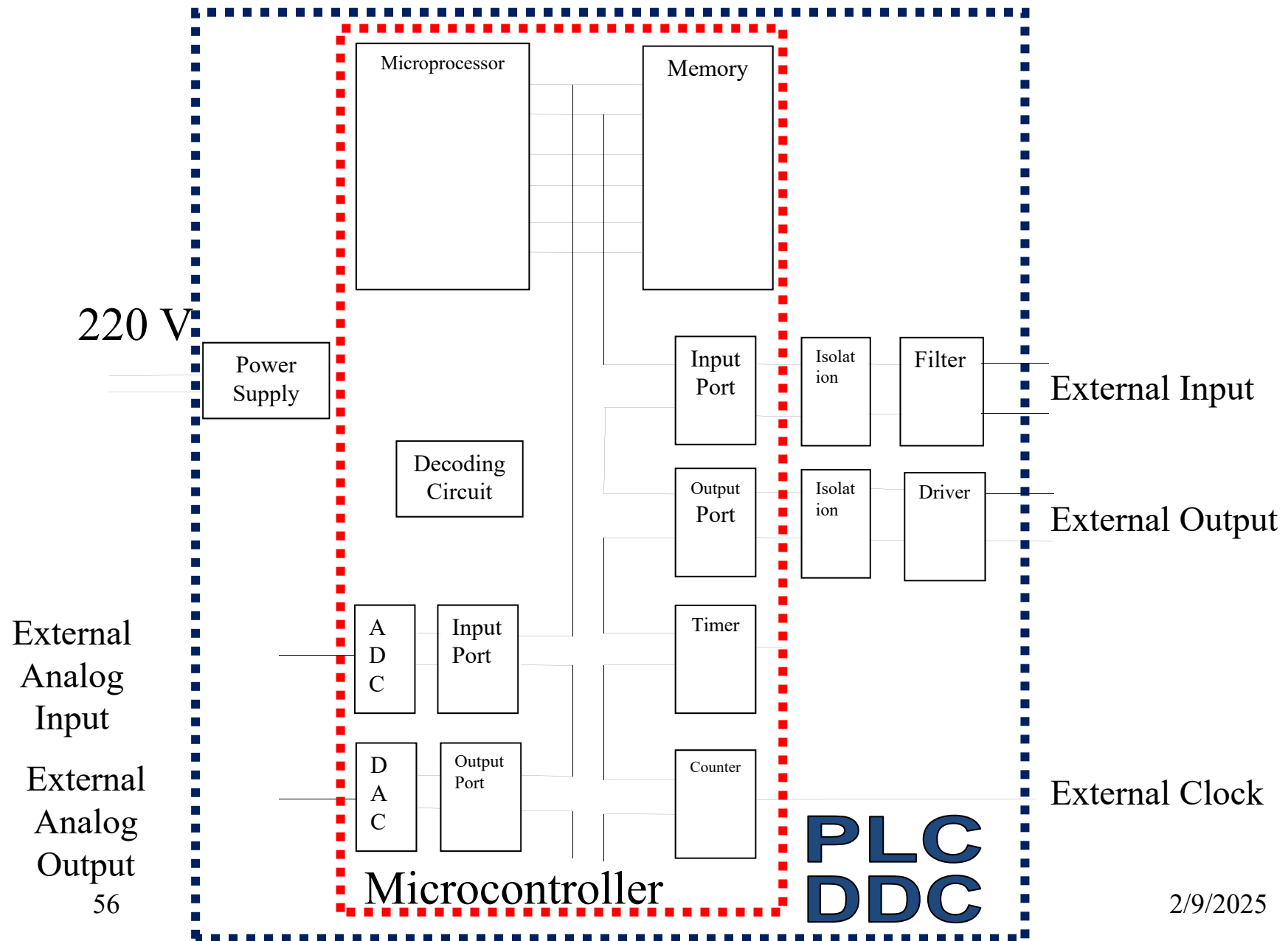
---



# Microprocessor Based Control System



# Microprocessor Based Control System



# Differentiate Microcontroller and microprocessor

## COMPARISON

### **Microcontroller**

Depend mainly on its peripherals like:  
Program memory, I/O ports, timers, interrupt circuitry, ADC...Etc.

Used for a few dedicated functions determined by the system designer.

Usually used as a part of a larger system

### **General purpose microprocessor**

Depend mainly on other devices like:  
I/O devices, memory, DMA controllers ..Etc.

Used in many applications, according to the program running on it

It's in the heart of our PC's.



# MICROPROCESSOR VS. MICROCONTROLLER

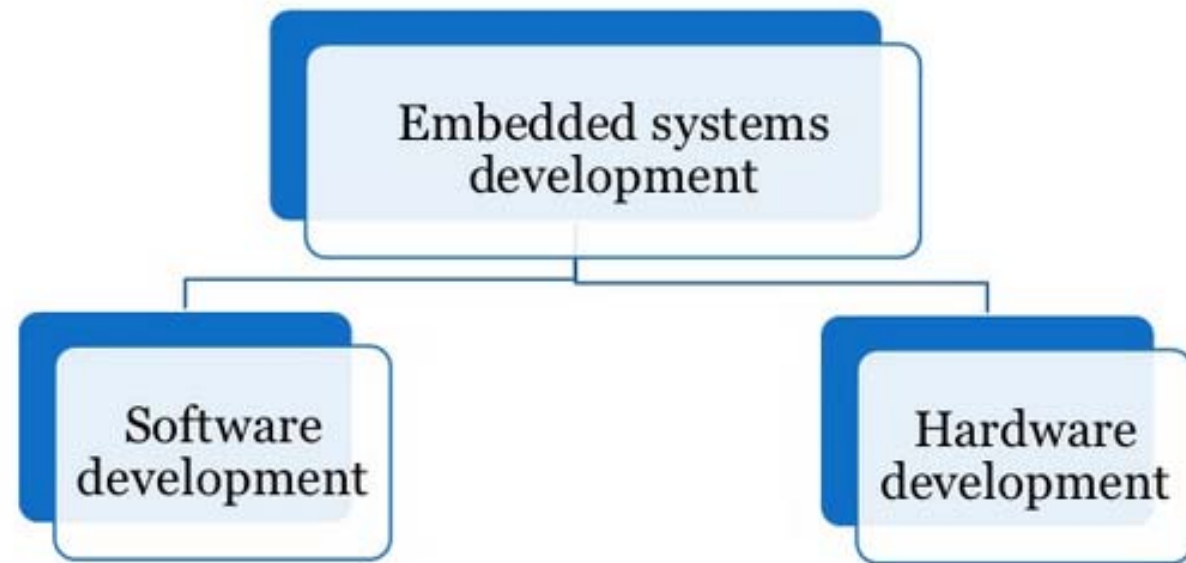
Microprocessor	Microcontroller
Higher Clock speed	Slower clock speed
CPU is stand-alone, RAM, ROM, I/O, timer are separate	CPU, RAM, ROM, I/O and timer are all on a single chip
Designer can decide on the amount of ROM, RAM and I/O ports	Fix amount of on-chip ROM, RAM, I/O ports
Expansive	Cheap
General-purpose	Single-purpose
High Access time for memory	Low Access time for memory
Very High power	Low power

# Advantages of Microcontroller Application

Some of the benefits are based tools microcontroller:

- ❑ High reliability and high degree of integration;
- ❑ Reduction in size;
- ❑ Reduced component count and manufacturing cost owner;
- ❑ Shorter development time;
- ❑ Lower power consumptions.


# Developing Embedded Systems



# Hardware Development

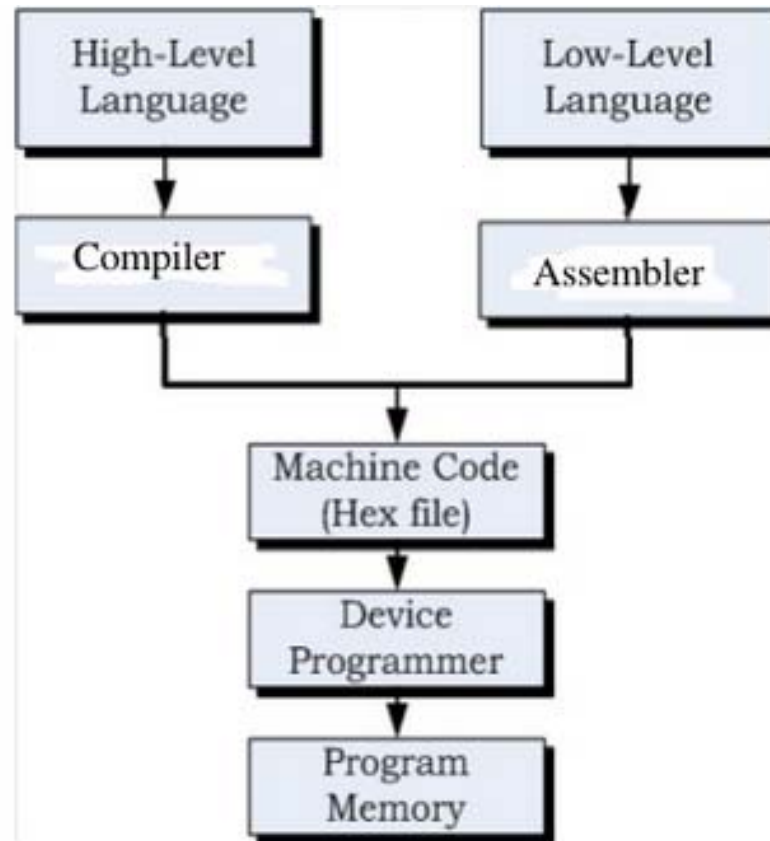
- This includes choosing the right MCU for your application, so that it can satisfy the requirements of your project.
- The criteria for choosing a microcontroller is:
  - 1- Number of I/O ports.
  - 2- Serial communication modules.
  - 3- Peripherals like (Timer, ADC, PWM ..Etc.)
  - 4- Memory requirements.
  - 5- Processing speed required.
  - 6- Power requirements.

# Software Development

- 
- Writing the required algorithm using assembly or a high level language (almost C).
  - Using a compiler or assembler and a linker.
  - Debugging your code.



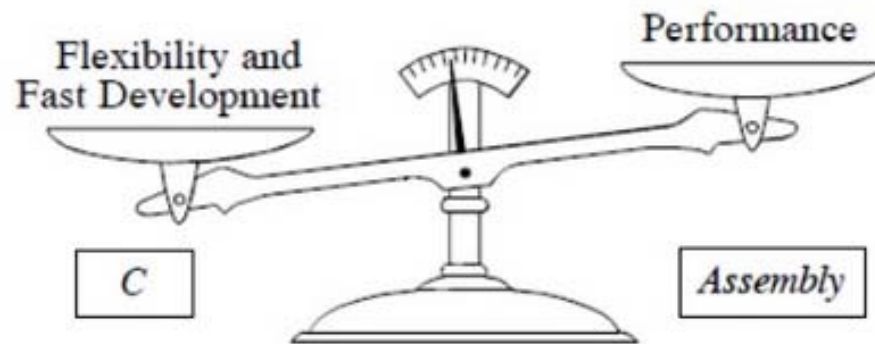
# Software Development



## SOFTWARE DEVELOPMENT

- Using **assembly** involves **learning the used microcontroller's specific instruction set** but results in the most compact and fastest code.
- Using **C programming language** makes your code **portable**, which means that you can use it for another target microcontroller without learning its instruction set, this eases the process of software development (short time to market) with acceptable quality.

## C VS ASSEMBLY



Assembly programs are optimized more than C programs, but to develop more complicated programs, using C is more practical and also efficient.