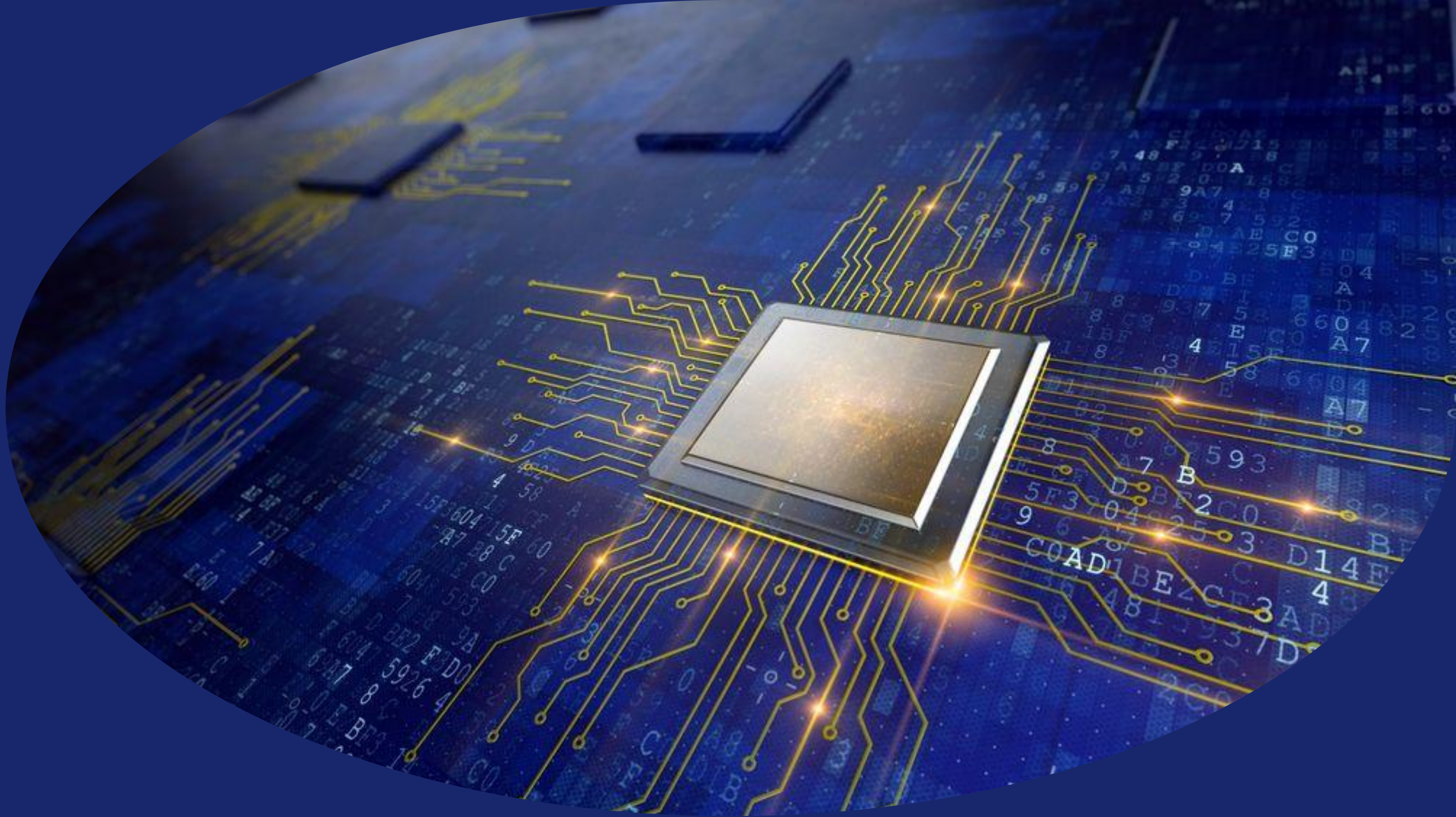
Lecture 3



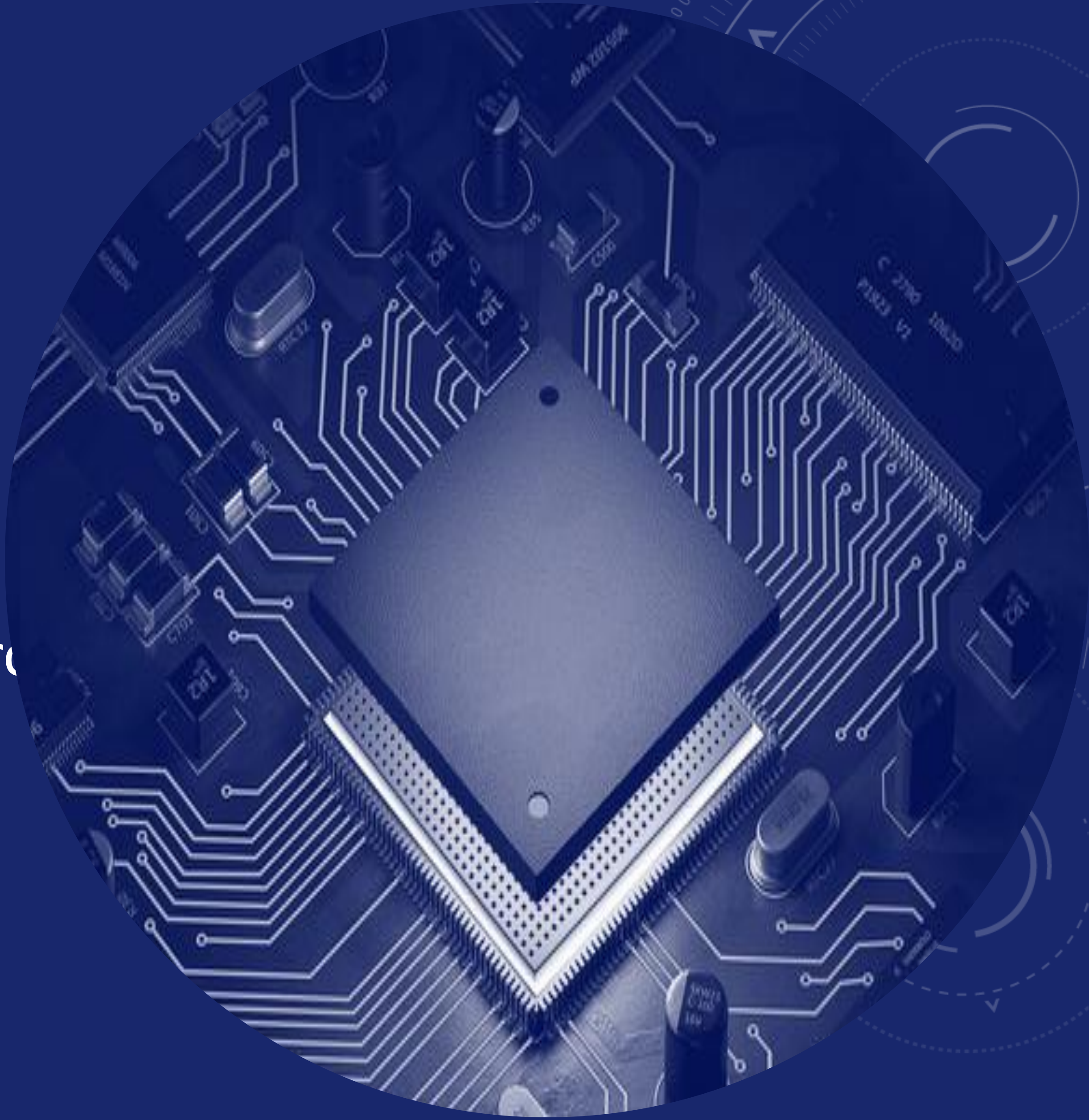
# EMBEDDED SYSTEMS

ASSOCIATED PROF. WAFAA SHALASH

# LECTURE TOPICS

## Microcontroller

1. How do microcontrollers work
2. What are the elements of a microcontr
3. Types of Microcontrolle

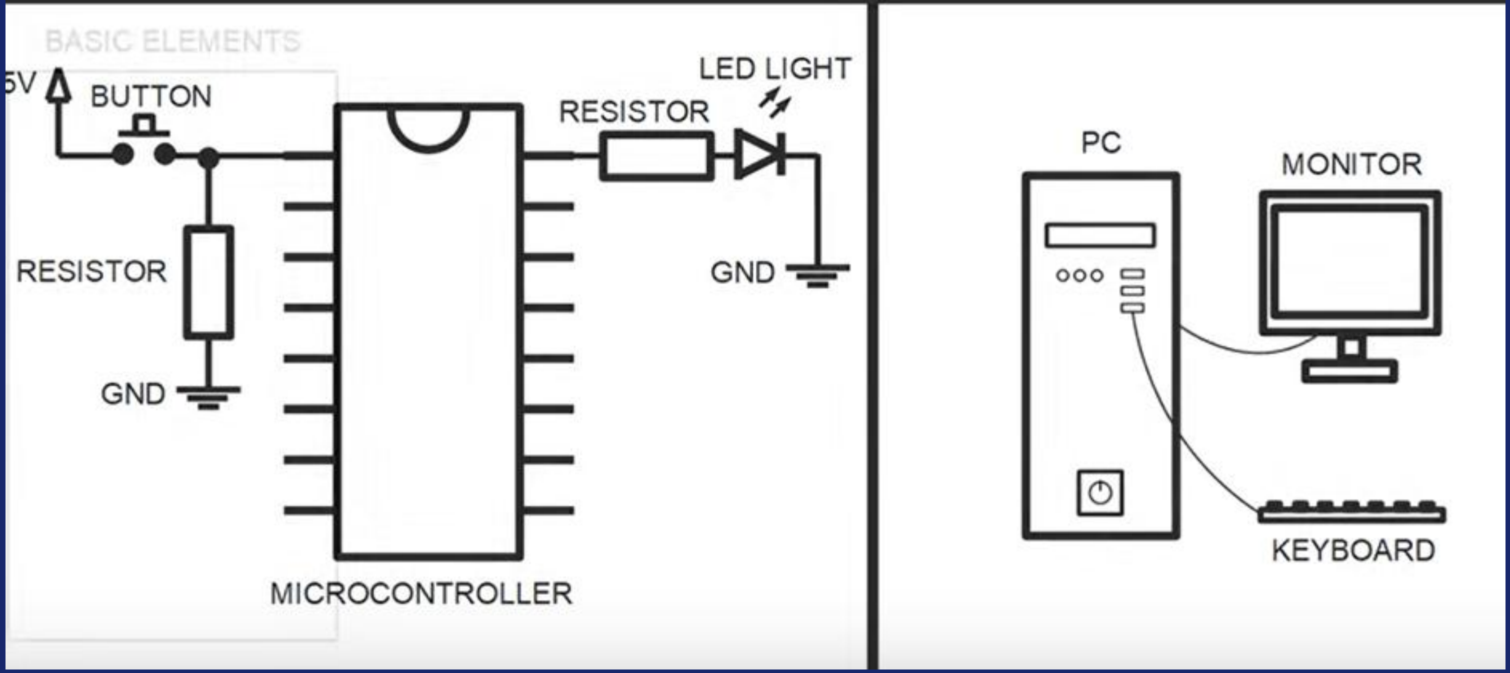


# MICROCONTROLLER

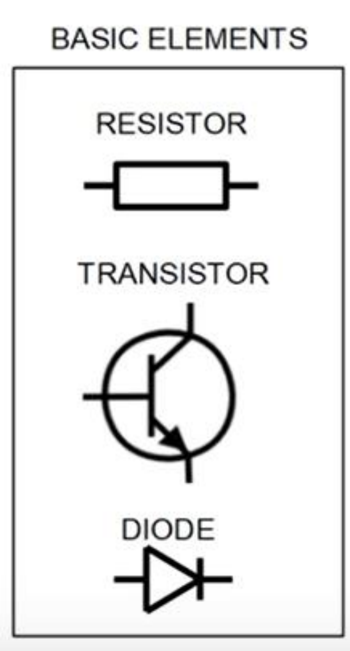
- A microcontroller is a compact integrated circuit designed to govern a specific operation in an embedded system.
- A typical microcontroller includes **a processor, memory and input/output (I/O) peripherals on a single chip.**
- Microcontrollers are found in vehicles, robots, office machines, medical devices, mobile radio transceivers, vending machines and home appliances, among other devices.
- They are essentially simple miniature personal computers (PCs) designed to control small features of a larger component, without a complex front-end operating system (OS).

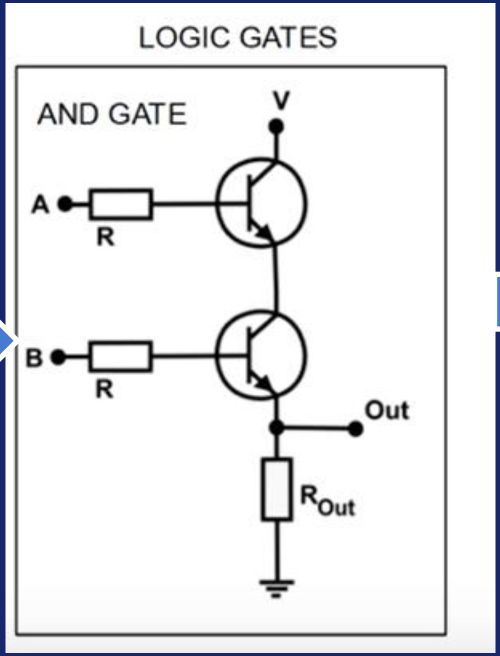
# HOW DO MICROCONTROLLERS WORK?

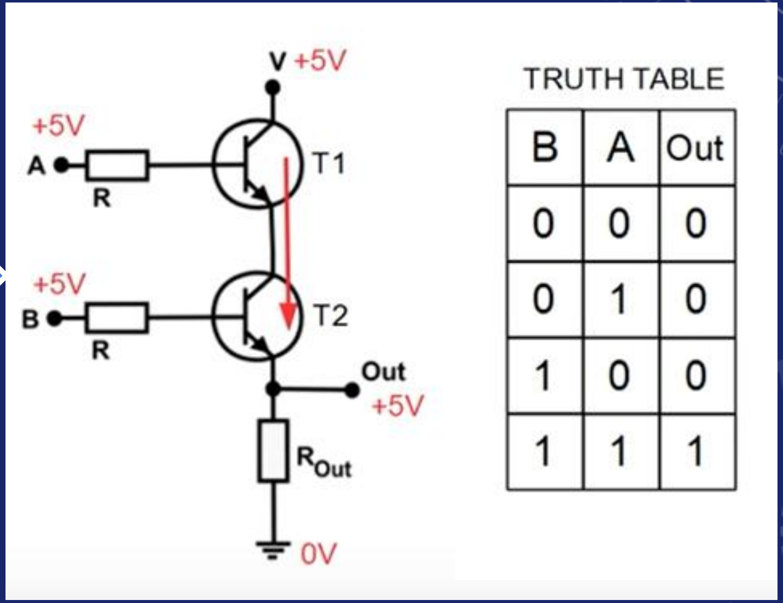
- A microcontroller is embedded inside of a system to control a singular function in a device.
- It does this by interpreting data it receives from its I/O peripherals using its central processor.
- The temporary information that the microcontroller receives is stored in its data memory, where the processor accesses it and uses instructions stored in its program memory to decipher and apply the incoming data.
- It then uses its I/O peripherals to communicate and enact the appropriate action



# BASIC ELECTRONIC ELEMENTS

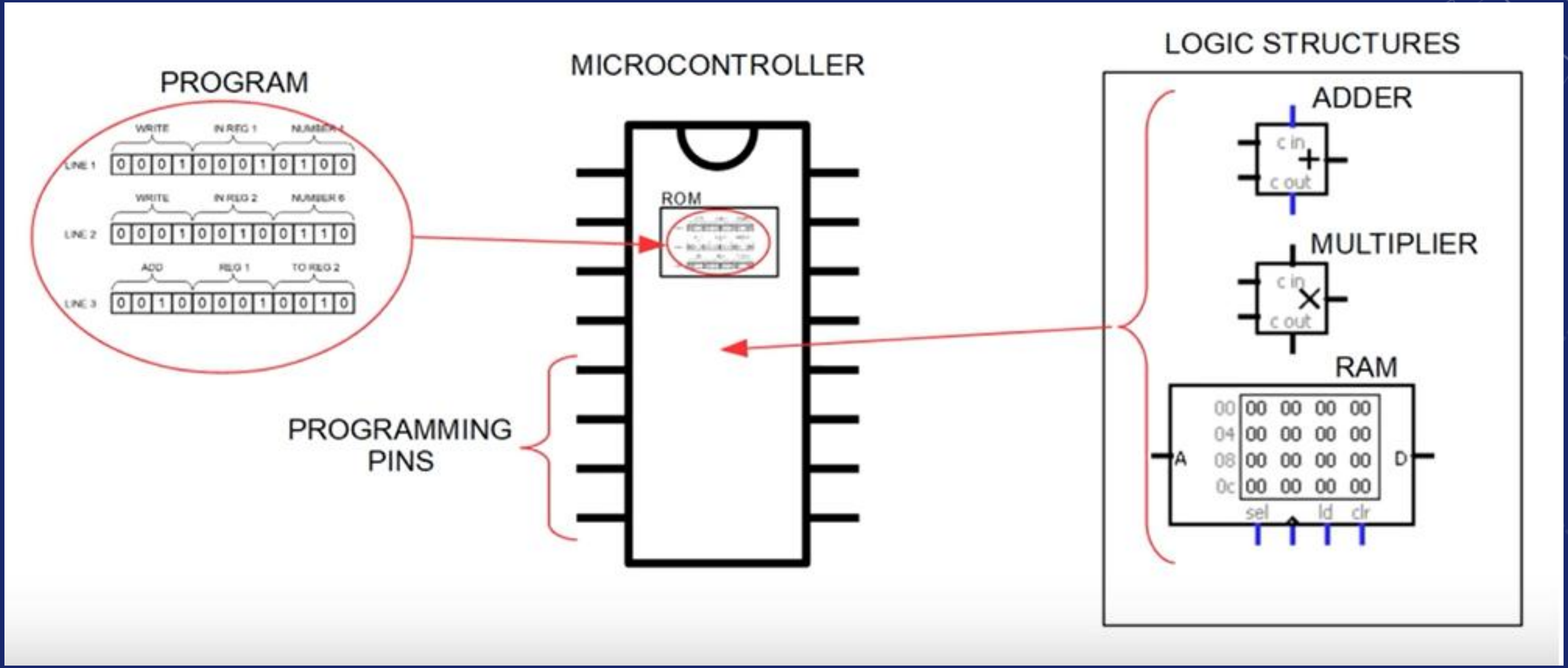




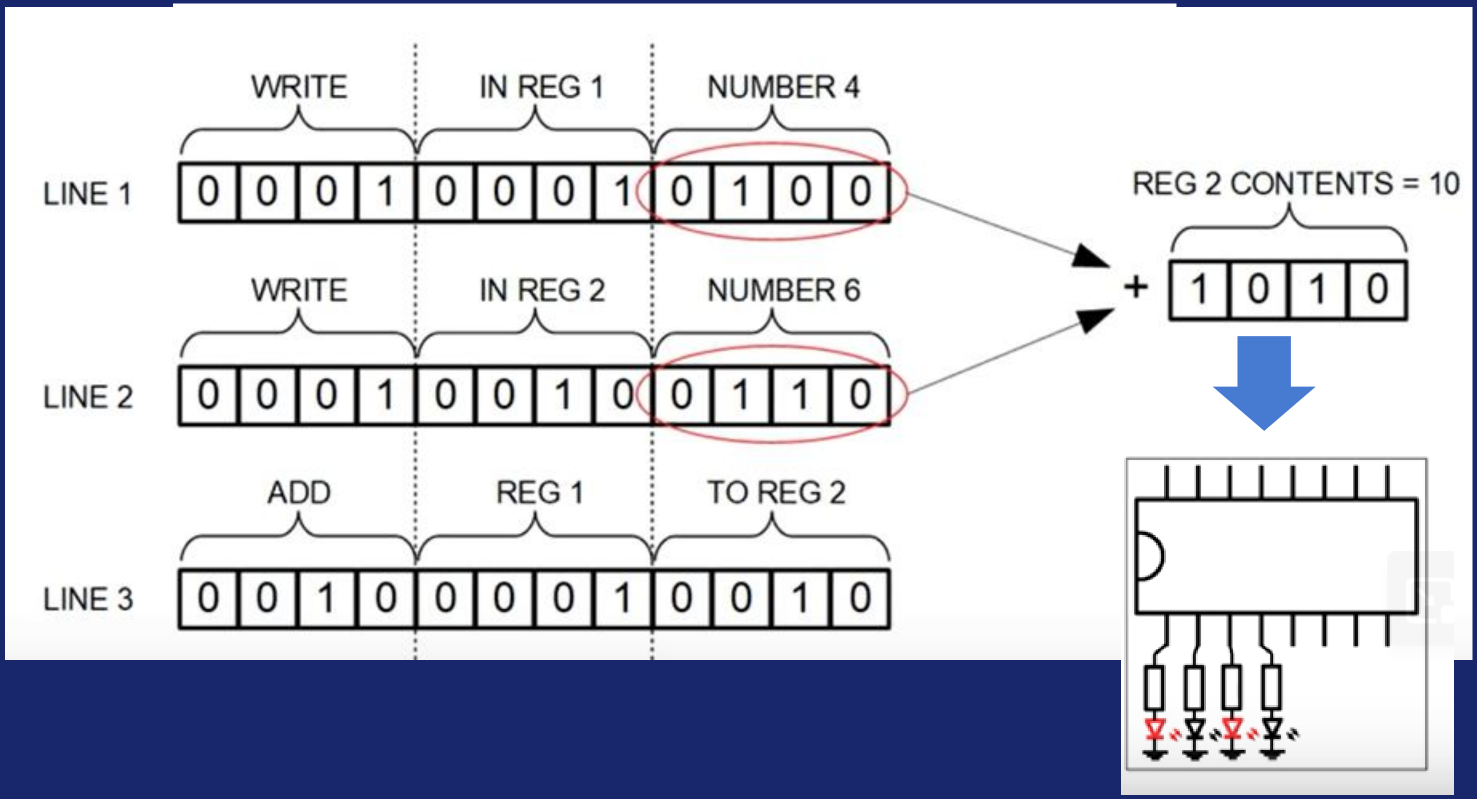


TRUTH TABLE

| B | A | Out |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |


PROGRAM
WRITE
IN REG 1
NUMBER 4
LINE 1 0 0 0 1 0 0 0 1 0 1 0 0
WRITE
IN REG 2
NUMBER 6
LINE 2 0 0 0 1 0 0 1 0 0 1 1 0
ADD
REG 1
TO REG 2
LINE 3 0 0 1 0 0 0 0 1 0 0 1 0
MICROCONTROLLER
ROM
PROGRAMMING PINS
LOGIC STRUCTURES
ADDER
c in
+
c out
MULTIPLIER
c in
×
c out
RAM
00 00 00 00 00
04 00 00 00 00
08 00 00 00 00
0c 00 00 00 00
A
D
sel
ld
clr

# HOW CODE EXECUTED ON MCU?



WRITE | IN REG 1 | NUMBER 4

LINE 1 0 0 0 1 0 0 0 1 0 1 0 0

WRITE | IN REG 2 | NUMBER 6

LINE 2 0 0 0 1 0 0 1 0 0 1 1 0

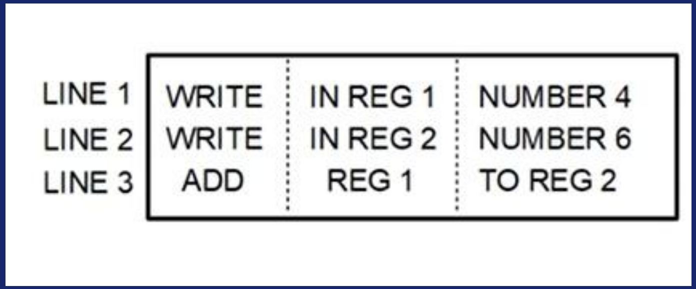
ADD | REG 1 | TO REG 2

LINE 3 0 0 1 0 0 0 0 1 0 0 1 0

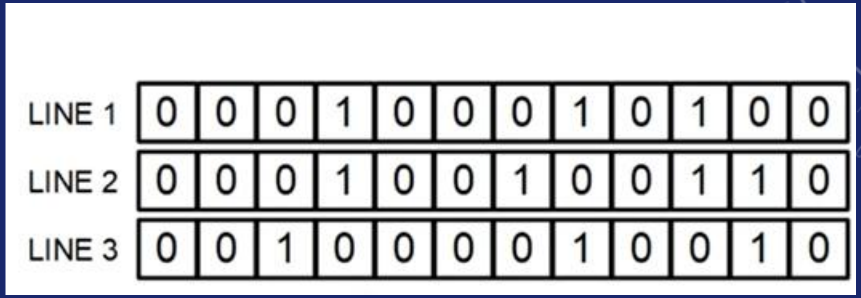
REG 2 CONTENTS = 10

+ 1 0 1 0

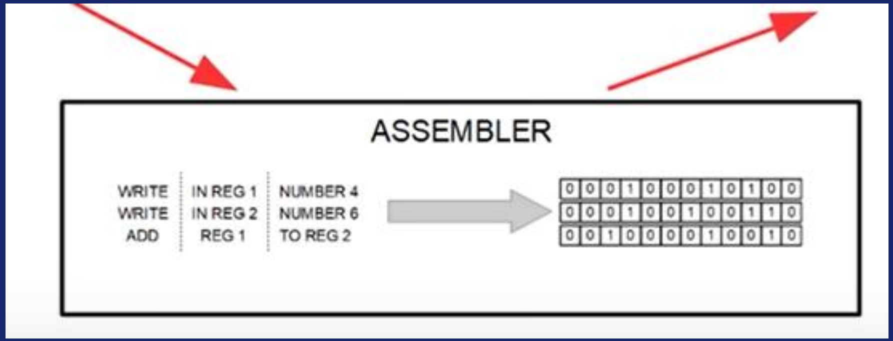
# HOW CODE REPRESENTED?



| | | | |
|---|---|---|---|
| LINE 1 | WRITE | IN REG 1 | NUMBER 4 |
| LINE 2 | WRITE | IN REG 2 | NUMBER 6 |
| LINE 3 | ADD | REG 1 | TO REG 2 |



| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LINE 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| LINE 2 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| LINE 3 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |

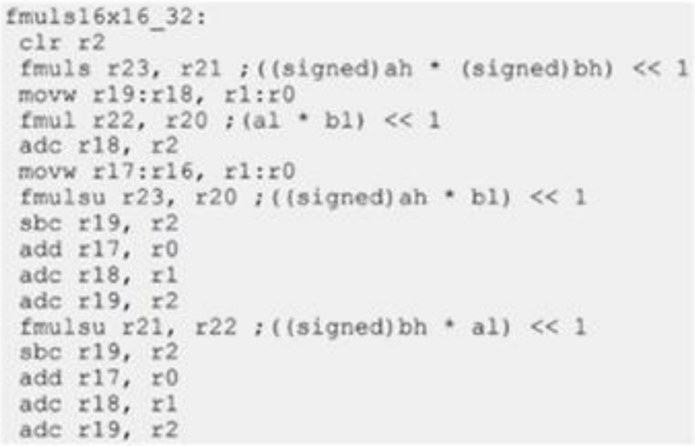


ASSEMBLER

| | | |
|---|---|---|
| WRITE | IN REG 1 | NUMBER 4 |
| WRITE | IN REG 2 | NUMBER 6 |
| ADD | REG 1 | TO REG 2 |

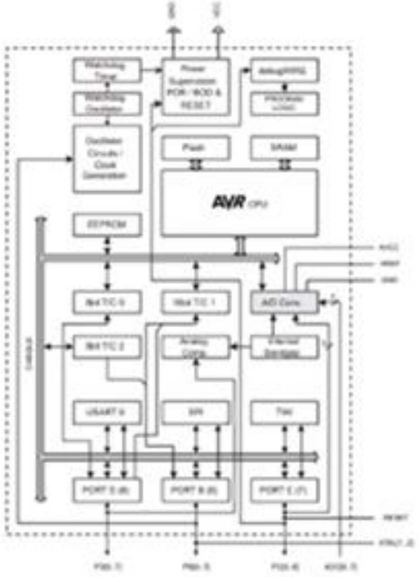
| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |

## Assembly code example

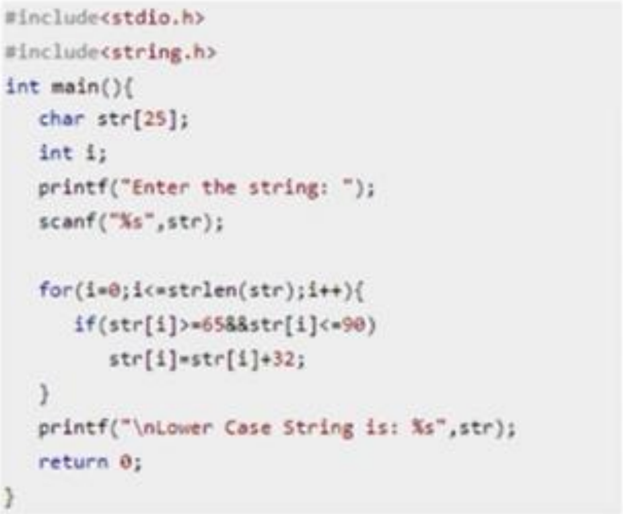


```
fmuls16x16_32:
 clr r2
 fmuls r23, r21 ;((signed)ah * (signed)bh) << 1
 movw r19:r18, r1:r0
 fmul r22, r20 ;(al * bl) << 1
 adc r18, r2
 movw r17:r16, r1:r0
 fmulsu r23, r20 ;((signed)ah * bl) << 1
 sbc r19, r2
 add r17, r0
 adc r18, r1
 adc r19, r2
 fmulsu r21, r22 ;((signed)bh * al) << 1
 sbc r19, r2
 add r17, r0
 adc r18, r1
 adc r19, r2
```

## Block diagram of a microcontroller

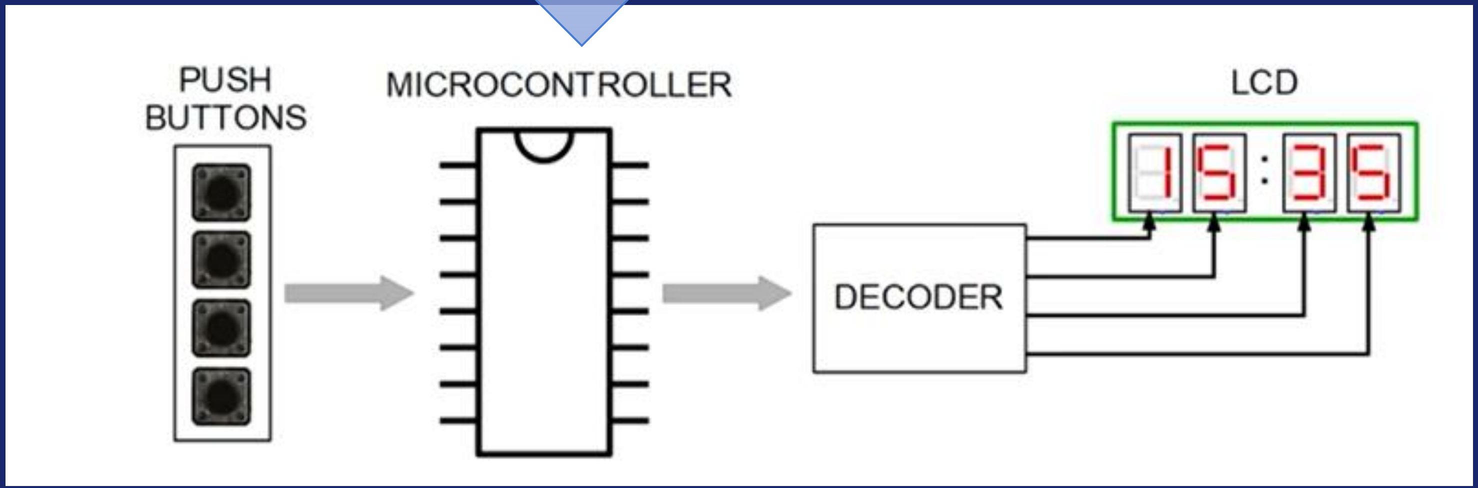


## "C" code example



```
#include<stdio.h>
#include<string.h>
int main(){
   char str[25];
   int i;
   printf("Enter the string: ");
   scanf("%s",str);

   for(i=0;i<=strlen(str);i++){
      if(str[i]>=65&&str[i]<=90)
         str[i]=str[i]+32;
   }
   printf("\nLower Case String is: %s",str);
   return 0;
}
```



# WHAT ARE THE ELEMENTS OF A MICROCONTROLLER?

- **The processor (CPU) :**A processor can be thought of as the brain of the device.
- It processes and responds to various instructions that direct the microcontroller's function.
- This involves performing basic arithmetic, logic and I/O operations.
- It also performs data transfer operations, which communicate commands to other components in the larger embedded system.

# MEMORY :

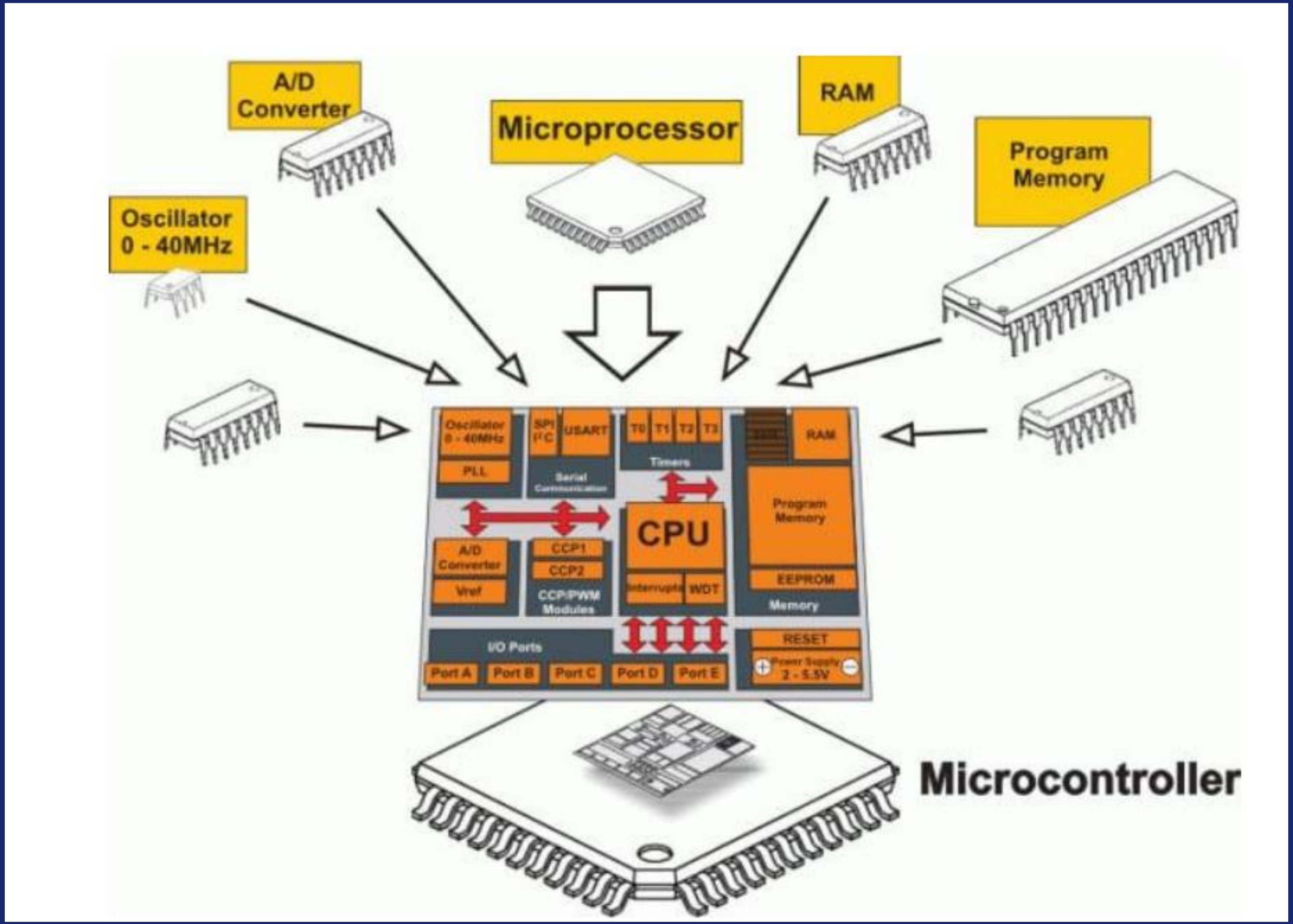
- A microcontroller's memory is used to store the data that the processor receives and uses to respond to instructions that it's been programmed to carry out.
- A microcontroller has two main memory types:
- • **Program memory** :which stores long-term information about the instructions that the CPU carries out.
- Program memory is non-volatile memory, meaning it holds information over time without needing a power source.
- • **Data memory:** which is required for temporary data storage while the instructions are being executed. Data memory is volatile, meaning the data it holds is temporary and is only maintained if the device is connected to a power source.

# I/O PERIPHERALS :

- The input and output devices are the interface for the processor to the outside world.
- The input ports receive information and send it to the processor in the form of binary data.
- The processor receives that data and sends the necessary instructions to output devices that execute tasks external to the microcontroller.
- While the processor, memory and I/O peripherals are the defining elements of the microprocessor, there are other elements that are frequently included.
- The term I/O peripherals itself simply refers to supporting components that interface with the memory and processor.
- There are many supporting components that can be classified as peripherals.
- Having some manifestation of an I/O peripheral is elemental to a microprocessor, because they are the mechanism through which the processor is applied.

# CONVERTERS

- Analog to Digital Converter (ADC) : An ADC is a circuit that converts analog signals to digital signals. It allows the processor at the center of the microcontroller to interface with external analog devices, such as sensors.
- ▪ Digital to Analog Converter (DAC) :A DAC performs the inverse function of an ADC and allows the processor at the center of the microcontroller to communicate its outgoing signals to external analog components.
- System bus : The system bus is the connective wire that links all components of the microcontroller together.
- ▪ Serial port : The serial port is one example of an I/O port that allows the microcontroller to connect to external components. It has a similar function to a USB or a parallel port but differs in the way it exchanges bits


A/D Converter
Microprocessor
RAM
Program Memory
Oscillator 0 - 40MHz
Oscillator 0 - 40MHz
PLL
SPI I²C
USART
Serial Communication
T0
T1
T2
T3
Timers
RAM
Program Memory
CPU
A/D Converter
Vref
CCP1
CCP2
CCP/PWM Modules
Interrupts
WDT
EEPROM
Memory
I/O Ports
Port A
Port B
Port C
Port D
Port E
RESET
Power Supply 2 - 5.5V
Microcontroller

# TYPES OF MICROCONTROLLER

- **1) PIC Microcontroller**
- PIC Stands for Peripheral Interface Controller is a kind of microcontroller components was used in the development of electronics, computer robotics, and similar devices.
- Even though the PIC was produced by Microchip technology and based on hardware computing architecture, here the code and data are placed in separate registers to increase the input and output. Pic has a built-in data memory, data bus and dedicated microprocessor for preparing all I/O purposes and methods.

# TYPES OF MICROCONTROLLER

- **2) ARM Microcontroller**
- ARM stands for Advanced RISC Machine. It’s the most popular Microcontrollers Programming in the digital embedded system world, and most of the industries prefer only ARM microcontrollers since it consists of significant features to implement products with an excellent appearance.
- It is cost sensitive and high-performance device which has been used in a wide range of application such as Industrial Instrument control systems, wireless networking and sensors, and automotive body systems, etc

# TYPES OF MICROCONTROLLER

- **3) AVR Microcontroller**
- AVR stands for Alf and Vegard's RISC Processor. It was the modified Harvard architecture machine, where program and data were stored in the separate physical memory system that appears in different address spaces, but having the ability to browse information things from program memory victimization particular directions.
- AVR isn't associate degree signifier and doesn't symbolize something specially

# TYPES OF MICROCONTROLLER

- **4) MSP Microcontroller**
- MSP stands for Mixed Signal Processor. It’s the family from Texas Instruments. Built around a 16 -bit CPU, the MSP is designed for low cost and respectively, low power dissipation embedded statements.
- It’s the controller's appearance is directly related to the 16-bit data bus, and seven addressing modes and the decreased instructions set, which allows a shorter, denser programming code for fast performance

# 1. Overview

| Feature | PIC (Microchip) | AVR (Atmel/Microchip) | ARM (Various Vendors) | MSP430 (TI) |
|---|---|---|---|---|
| **Built-in Peripherals** | ADC, PWM, UART, SPI, I2C | ADC, PWM, UART, SPI, I2C | **Rich peripherals (USB, Ethernet, CAN, ADC, etc.)** | ADC, PWM, UART, SPI, I2C |
| **Ease of Programming** | Moderate (PIC Assembly, C) | **Easy (Arduino, C)** | **Easy (Keil, STM32Cube, PlatformIO, Arduino)** | Moderate (Code Composer Studio, C) |
| **Development Tools** | MPLAB X, XC8, XC16, XC32 | Arduino IDE, Atmel Studio | Keil, STM32Cube, PlatformIO, MBED | Code Composer Studio |
| **Price Range** | Low to Moderate | Low | Moderate to High | Low to Moderate |
| **Common Uses** | Industrial, automotive, embedded systems | DIY, consumer electronics, embedded control | **IoT, AI, high-performance, mobile, embedded systems** | **Ultra-low power IoT, medical, wearables, battery-powered applications** |

# 1. Overview

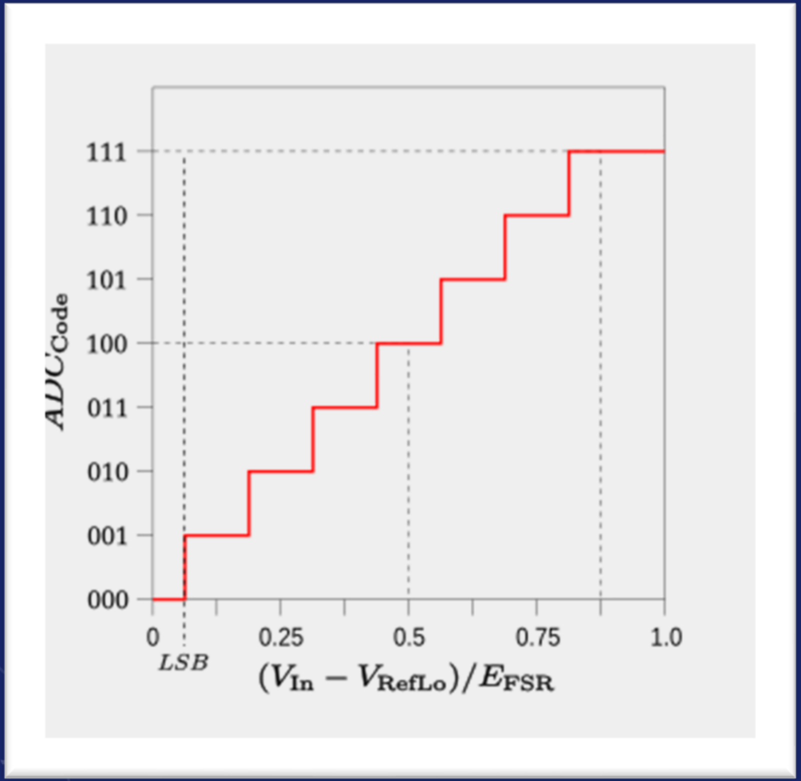
| Feature | PIC (Microchip) | AVR (Atmel/Microchip) | ARM (Various Vendors) | MSP430 (TI) |
|---|---|---|---|---|
| **Architecture** | Harvard (8-bit, 16-bit, 32-bit) | Harvard (8-bit, some 32-bit) | Von Neumann & Harvard (32-bit, 64-bit) | Von Neumann (16-bit) |
| **Bit Width** | 8-bit, 16-bit, 32-bit | 8-bit, some 32-bit | 32-bit, 64-bit | 16-bit |
| **Processing Power** | Moderate | Moderate | High (Cortex-A, Cortex-M, Cortex-R) | Low to Moderate |
| **Clock Speed** | 10 MHz – 200 MHz | 1 MHz – 20 MHz (for 8-bit), higher for 32-bit | 48 MHz – 3+ GHz | 1 MHz – 25 MHz |
| **Power Efficiency** | Low to moderate | Moderate | High (especially Cortex-M series) | **Very High (optimized for ultra-low power)** |
| **Memory** (RAM/Flash) | 32B – 2 MB | 32B – 512 KB | **256B – GBs (Scalable)** | 128B – 512 KB |
| **Operating Voltage** | 2V – 5.5V | 2.7V – 5.5V | **1.8V – 3.6V (typically 3.3V)** | **1.8V – 3.6V** |

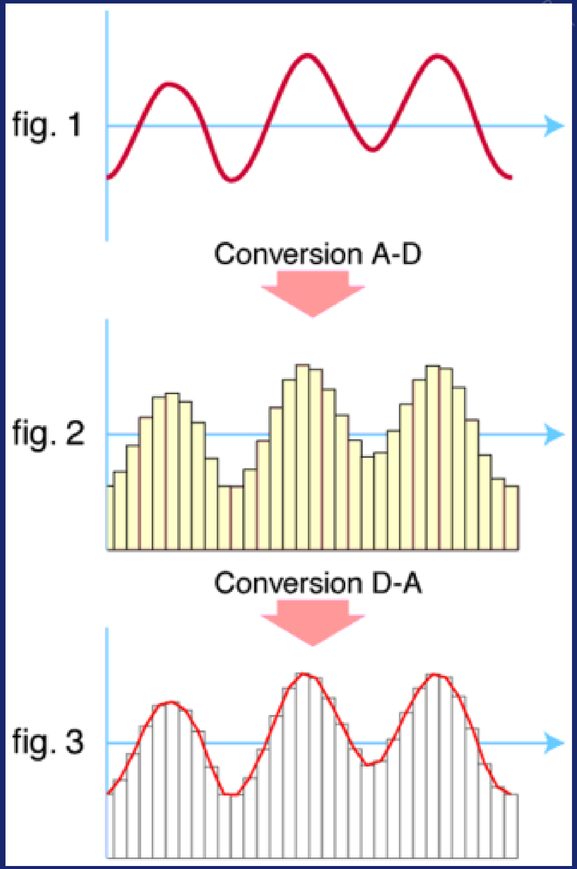
# THESE ARE COMMON COMMUNICATION AND CONTROL INTERFACES USED IN MICROCONTROLLERS (MCUS) AND EMBEDDED SYSTEMS:

1. ADC (Analog-to-Digital Converter)
    1. Converts **analog signals (e.g., sensor voltage)** into **digital values**.
    2. Used in applications like **temperature sensors, light sensors, and audio processing**.
2. PWM (Pulse Width Modulation)
    1. A technique for **controlling power delivery** to devices like motors, LEDs, and heaters.
    2. Adjusts the **duty cycle** of a digital signal to vary brightness or speed.
3. UART (Universal Asynchronous Receiver-Transmitter)
    1. A **serial communication protocol** for sending data between devices using **TX (transmit) and RX (receive) lines**.
    2. Used in devices like **GPS modules, Bluetooth, and serial monitors**.
4. SPI (Serial Peripheral Interface)
    1. A **high-speed, synchronous** communication protocol with **Master-Slave architecture**.
    2. Uses **MOSI (Master Out Slave In), MISO (Master In Slave Out), SCK (Clock), and SS (Slave Select)**.
    3. Used in **SD cards, displays, sensors, and flash memory**.
5. I2C (Inter-Integrated Circuit)
    1. A **two-wire communication** protocol using **SDA (Data) and SCL (Clock)**.
    2. Supports **multiple devices (multi-master, multi-slave)** on the same bus.
    3. Used in **sensors, EEPROM, OLED displays, and RTC (Real-Time Clocks)**.

# 1. ADC (ANALOG-TO-DIGITAL CONVERTER)

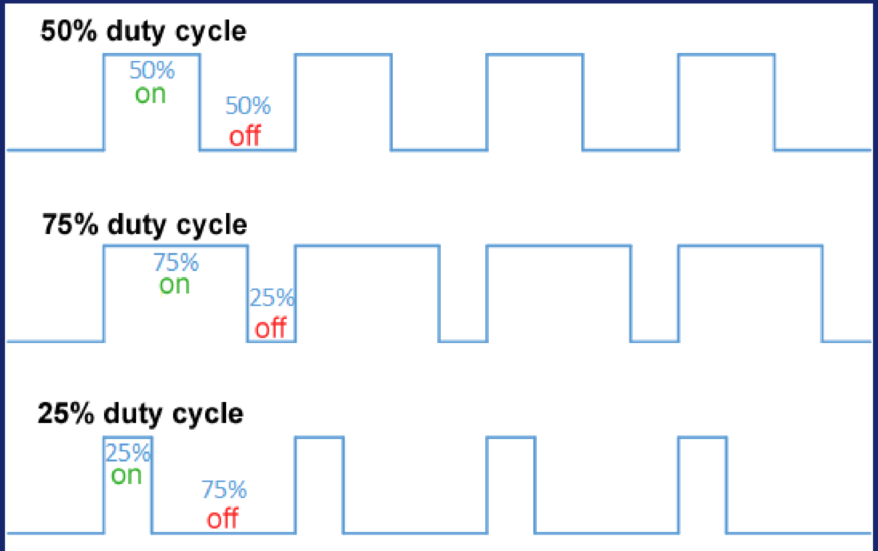
1. Converts **analog signals (e.g., sensor voltage)** into **digital values**.
2. Used in applications like **temperature sensors, light sensors, and audio processing**.





# 2. PWM (PULSE WIDTH MODULATION)

1. A technique for **controlling power delivery** to devices like motors, LEDs, and heaters.
2. Adjusts the **duty cycle** of a digital signal to vary brightness or speed.


50% duty cycle
50% on
50% off
75% duty cycle
75% on
25% off
25% duty cycle
25% on
75% off