Introduction of Microcontroller & Arduino

Basic Idea about microcontroller; Introduction to Arduino: Brief history of the Arduino; Pin configurations of the board Arduino Uno. Brief idea about Arduino-nano/Arduino R4 Wi-Fi/Arduino MRGA. Sources of constant voltages 5 volt/3.3 volt and ground and corresponding pins of the respective boards. PWM and idea of duty cycle.

Introduction of Microcontroller & Arduino

References:

1. https://www.arduino.cc/en/Tutorial/HomePage
2. Arduino Cookbook, Michael Margolis, O'Reilly Media (2011)
3. Getting Started with Arduino, Massimo Banzi, O'Reilly Media (2009)
4. Programming Arduino: Getting Started with Sketches by Simon Monk

Software:

Arduino IDE (Integrated Development Environment)

Hardware:

1. Arduino UNO
2. Basic electronic components, sensors

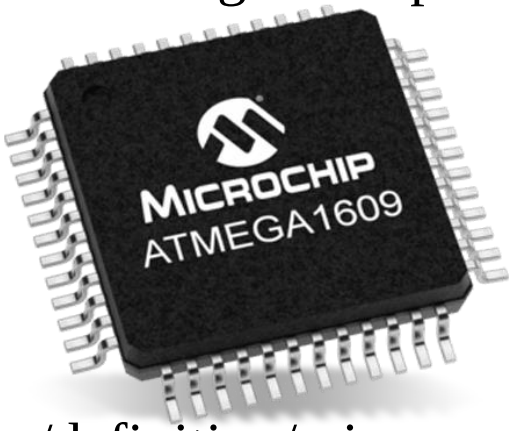
TERMINOLOGIES

A. Open-Source Software (OSS): Type of computer software in which source code is released under a license in which the copyright holder grants users the rights to use, study, change, and distribute the software to anyone and for any purpose. Examples: Linux, Android, Firefox etc.

B. Open-Source hardware (OSH): Physical artifacts of technology designed and offered by the open-design movement. Information about the hardware is easily discerned so that others can make it – coupling it closely to the maker movement. Hardware design (i.e., mechanical drawings, schematics, bills of material, PCB layout data, HDL source code and integrated circuit layout data), in addition to the software that drives the hardware, are all released under free/libre terms. Examples: RepRap (3D printing), Arduino etc.

TERMINOLOGIES

C. Microcontroller: An integrated circuit (IC) device used for controlling other portions of an electronic system, usually via a microprocessor unit (MPU), memory, and some peripherals. These devices are optimized for embedded applications that require both processing functionality and agile, responsive interaction with digital, analog, or electromechanical components. A typical microcontroller includes a processor (CPU), memory and input/output (I/O) peripherals on a single chip. MCUs 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).



https://internetofthingsagenda.techtarget.com/definition/microcontroller

More on Microcontroller

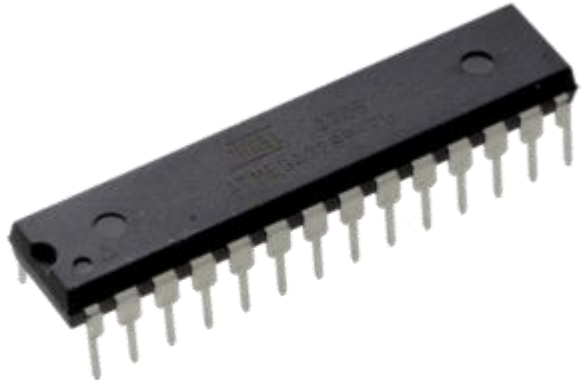
- **Processor (CPU)** is the brain of the microcontroller, which appears in 8-bit, 16-bit, or 32-bit architectures and executes instructions from its memory to perform tasks. Usually, it's a simpler processor compared to general-purpose CPUs found in computers.
- **Memory:** RAM is the temporary memory for storing data while the microcontroller is running. ROM/Flash is permanent storage for the program code or firmware. The memory is often limited, which helps keep the microcontroller's cost and power consumption low.
- **I/O Ports:** Microcontrollers come with I/O pins that allow them to connect to various components (sensors, actuators, displays, etc.). They may have features like analog-to-digital converters (ADCs) for analog sensors, digital I/O pins for controlling LEDs, motors, and more.

More on Microcontroller

- **Timers and Counters:** Help the microcontroller manage tasks that need precise timing, like PWM (Pulse Width Modulation) signals for motor control or event counting.
- **Communication Protocols:** Microcontrollers often include protocols like UART (Universal Asynchronous Receiver Transmitter), SPI (serial peripheral interface), and I2C (Inter-Integrated Circuit) for connecting with other devices or sensors.
- **Common Applications:** Include home appliances (microwaves, washing machines), automotive systems (engine control, airbags), Industrial automation, IoT devices (smart thermostats, wearables, robotics, platforms like Arduino, Raspberry Pi Pico, ESP32 etc.
- Microcontrollers are low power, compact, cost-effective, and specialized for specific tasks but have limited computing power and memory compared to general-purpose computers.

ATmega328P

ATmega328P is 8-bit microcontroller from Atmel (now owned by Microchip Technology). It’s one of the most popular microcontrollers, especially known for powering the Arduino Uno and other Arduino-compatible boards.



- 28 pins DIP (dual in-line package)
- **CPU:** 8-bit AVR RISC-based CPU
- **Clock Speed:** Up to 20 MHz
- **Flash Memory:** 32 KB (used to store program code)
- **SRAM:** 2 KB (for variables and temporary data during execution)
- **EEPROM:** 1 KB (for persistent data storage, even after power-off)
- **Operating Voltage:** 1.8V - 5.5V
- **I/O Pins:** 23 I/O pins, with 14 digital and 6 analog input pins (ADC-enabled)
- **Communication Interfaces:** UART, SPI, and I2C for connecting to sensors and other peripherals.

ATmega328P

ATmega328P is 8-bit microcontroller from Atmel (now owned by Microchip Technology). It’s one of the most popular microcontrollers, especially known for powering the Arduino Uno and other Arduino-compatible boards.

- 28 pins DIP (dual in-line package)
- **CPU:** 8-bit AVR RISC-based CPU
- **Clock Speed:** Up to 20 MHz
- **Flash Memory:** 32 KB (used to store program code)
- **SRAM:** 2 KB (for variables and temporary data during execution)
- **EEPROM:** 1 KB (for persistent data storage, even after power-off)
- **Operating Voltage:** 1.8V - 5.5V
- **I/O Pins:** 23 I/O pins, with 14 digital and 6 analog input pins (ADC-enabled).

ATmega328P

- **Timers:** Three timers (two 8-bit and one 16-bit) for timing and event management.
- **PWM:** Six PWM channels for tasks like dimming LEDs and controlling motor speeds.
- **Communication Interfaces:** UART, SPI, and I2C for connecting to sensors and other peripherals.
- **ADC:** 10-bit ADC with 6 channels for reading analog inputs.
- **Power Efficiency:** The "P" in ATmega328P stands for "PicoPower“, which means it has ultra-low power consumption modes, ideal for battery-powered applications.
- **RISC Architecture:** It uses a Reduced Instruction Set Computing (RISC) architecture, which means that the instructions are simple and fast.
- **Limitations:** Limited processing power and memory and hence not suitable for high-performance applications, low clock speed.

What is Arduino?

Arduino is an open-source hardware and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices. Its hardware products are licensed under a CC-BY-SA license, while software is licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially from the official website or through authorized distributors.

https://en.wikipedia.org/wiki/Arduino#:~:text=The%20Arduino%20project%20began%20in,environment%20using%20sensors%20and%20actuators.

What is Arduino?

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards ('shields') or breadboards (for prototyping) and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs. The microcontrollers can be programmed using the C and C++ programming languages, using a standard API which is also known as the "Arduino language". In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) and a command line tool developed in Go.

'This little board has made it possible for people to do things they wouldn't have done otherwise'
- David A. Mellis

https://en.wikipedia.org/wiki/Arduino#:~:text=The%20Arduino%20project%20began%20in,environment%20using%20sensors%20and%20actuators.

Arduino: History

The Arduino project was started at the Interaction Design Institute Ivrea (IDII) in Ivrea, Italy. At that time, the students used a BASIC Stamp microcontroller at a cost of $50. In 2003 Hernando Barragán created the development platform Wiring as a Master's thesis project at IDII, under the supervision of Massimo Banzi and Casey Reas. Casey Reas is known for co-creating, with Ben Fry, the Processing development platform. The project goal was to create simple, low-cost tools for creating digital projects by non-engineers. The Wiring platform consisted of a printed circuit board (PCB) with an ATmega168 microcontroller, an IDE based on Processing and library functions to easily program the microcontroller.

https://en.wikipedia.org/wiki/Arduino#:~:text=The%20Arduino%20project%20began%20in,environment%20using%20sensors%20and%20actuators.

Arduino: History

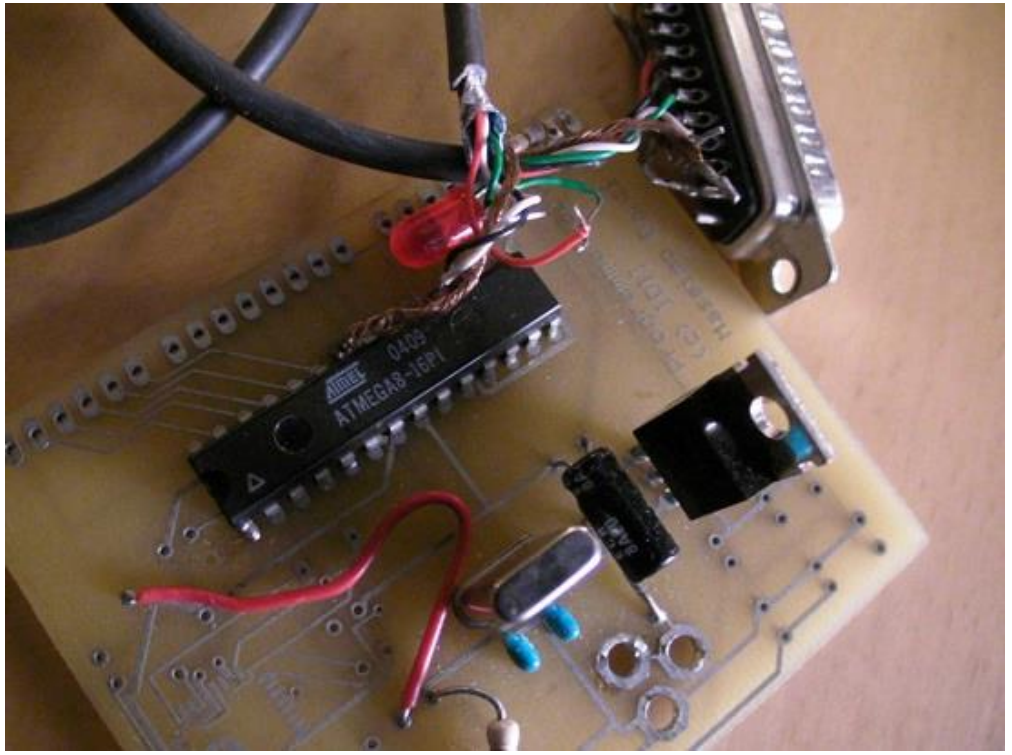
In 2005, Massimo Banzi, with David Mellis, another IDII student, and David Cuartielles, extended Wiring by adding support for the cheaper ATmega8 microcontroller. The new project, forked from Wiring, was called Arduino.

The initial Arduino core team consisted of Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis.

Following the completion of the platform, lighter and less expensive versions were distributed in the open-source community. It was estimated in mid-2011 that over 300,000 official Arduinos had been commercially produced and in 2013 that 700,000 official boards were in users' hands.

https://en.wikipedia.org/wiki/Arduino#:~:text=The%20Arduino%20project%20began%20in,environment%20using%20sensors%20and%20actuators.

ARDUINO: History



David Cuartielles, Gianluca Martino, Tom Igoe, David Mellis, and Massimo Banzi

https://www.circuitstoday.com/story-and-history-of-development-of-arduino

https://en.wikipedia.org/wiki/Arduino#:~:text=The%20Arduino%20project%20began%20in,environment%20using%20sensors%20and%20actuators.

Why Arduino?

- **Simple, clear programming environment -** The Arduino Software (IDE) is easy-to-use for beginners, yet flexible enough for advanced users to take advantage of as well. For teachers, it's conveniently based on the Processing programming environment, so students learning to program in that environment will be familiar with how the Arduino IDE works.
- **Open source and extensible software -** The Arduino software is published as open source tools, available for extension by experienced programmers. The language can be expanded through C++ libraries, and people wanting to understand the technical details can make the leap from Arduino to the AVR C programming language on which it's based. Similarly, you can add AVR-C code directly into your Arduino programs if you want to.

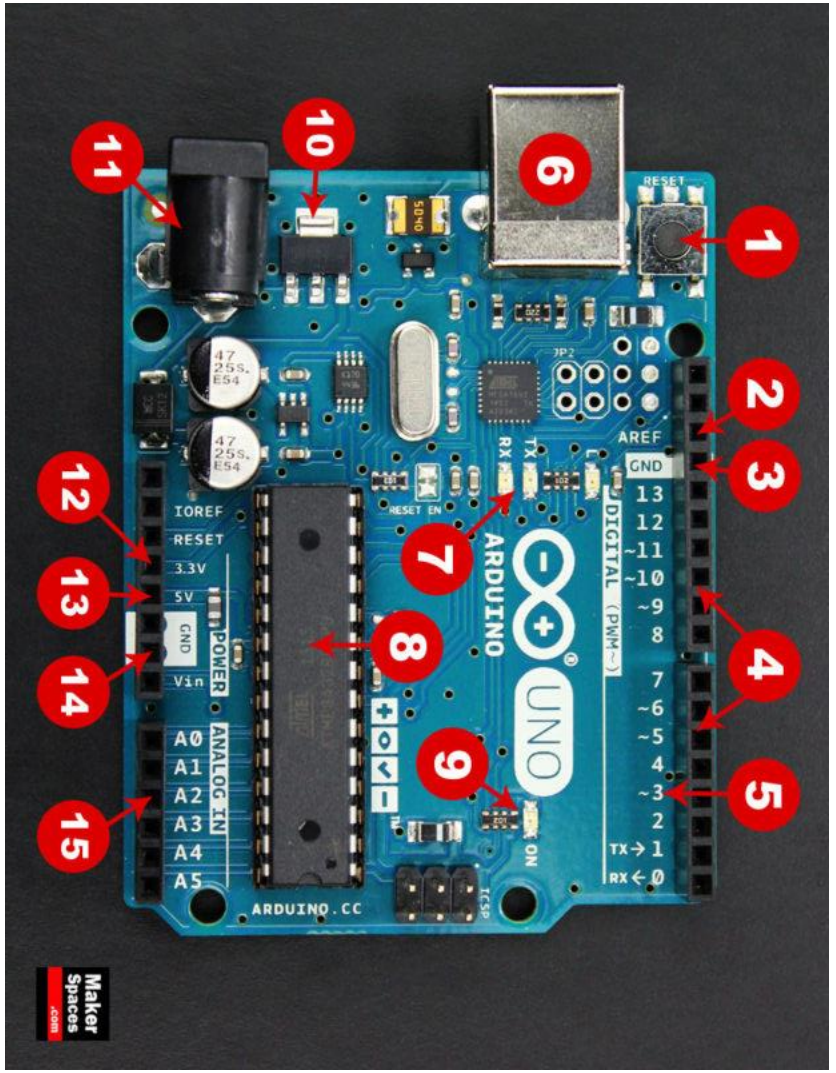
https://www.arduino.cc/en/Guide/Introduction

Why Arduino?

- **Open source and extensible hardware -** The plans of the Arduino boards are published under a Creative Commons license, so experienced circuit designers can make their own version of the module, extending it and improving it. Even relatively inexperienced users can build the breadboard version of the module in order to understand how it works and save money.
- **Cross-platform -** The Arduino Software (IDE) runs on Windows, Macintosh OSX, and Linux operating systems. Most microcontroller systems are limited to Windows.
- **Inexpensive -** Arduino boards are relatively inexpensive compared to other microcontroller platforms. The least expensive version of the Arduino module can be assembled by hand, and even the pre-assembled Arduino modules cost less than $50.

https://www.arduino.cc/en/Guide/Introduction

Arduino UNO R3



1. Reset Button – This will restart any code that is loaded to the Arduino board

2. AREF – Stands for "Analog Reference" and is used to set an external reference voltage

3. Ground Pin – There are a few ground pins on the Arduino and they all work the same

4. Digital Input/Output – Pins 0-13 can be used for digital input or output

5. PWM – (Pulse Width Modulation) The pins marked with the (~) symbol can simulate analog output

https://www.makerspaces.com/arduino-uno-tutorial-beginners/

Arduino UNO R3

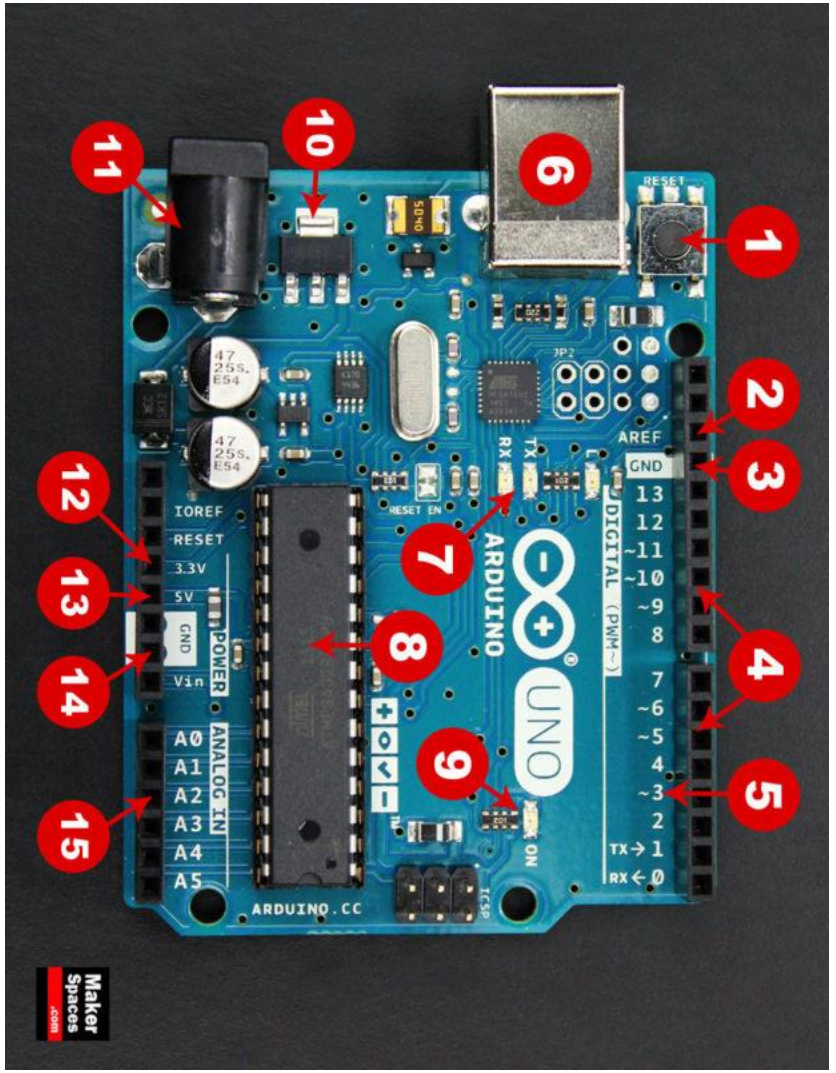
6. USB Connection – Used for powering up your Arduino and uploading sketches

7. TX/RX – Transmit and receive data indication LEDs

8. ATmega Microcontroller – This is the brains and is where the programs are stored (ATmega328P, 16 MHz)

9. Power LED Indicator – This LED lights up anytime the board is plugged in a power source

10. Voltage Regulator – This controls the amount of voltage going into the Arduino board



https://www.makerspaces.com/arduino-uno-tutorial-beginners/

Arduino UNO R3

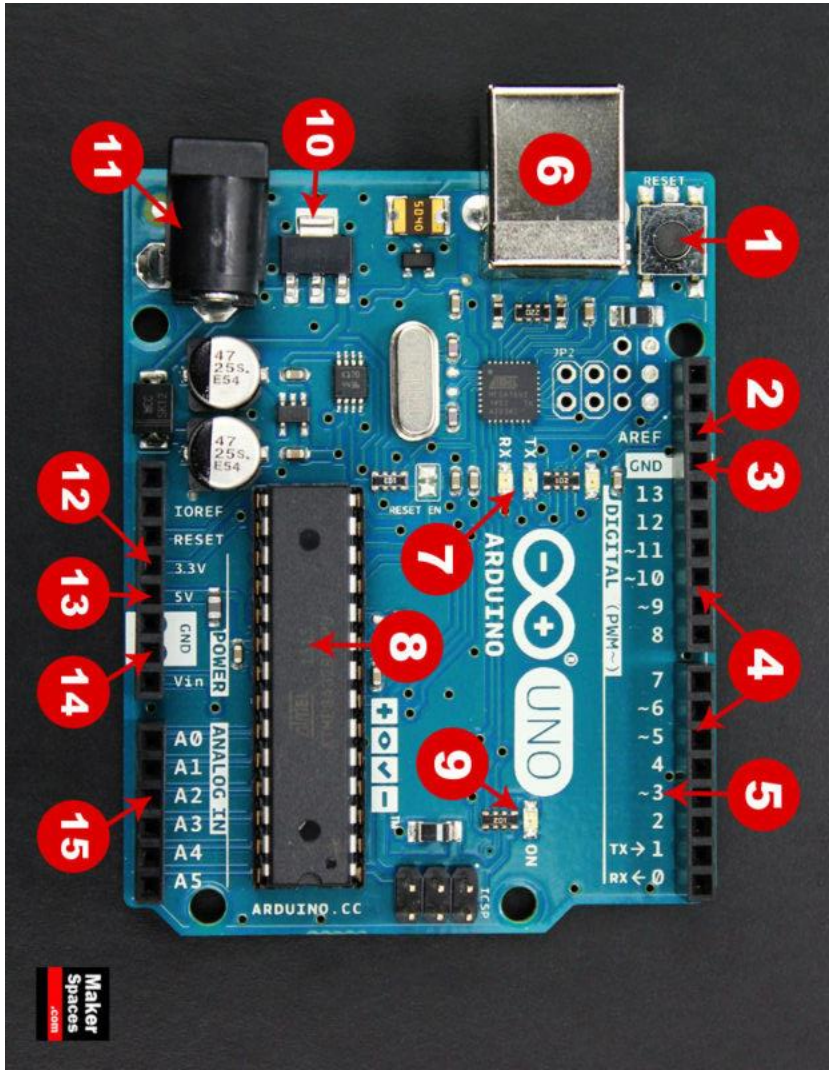
11. DC Power Barrel Jack – This is used for powering your Arduino with a power supply

12. 3.3V Pin – This pin supplies 3.3 volts of power to your projects

13. 5V Pin – This pin supplies 5 volts of power to your projects

14. Ground Pins – There are a few ground pins on the Arduino and they all work the same

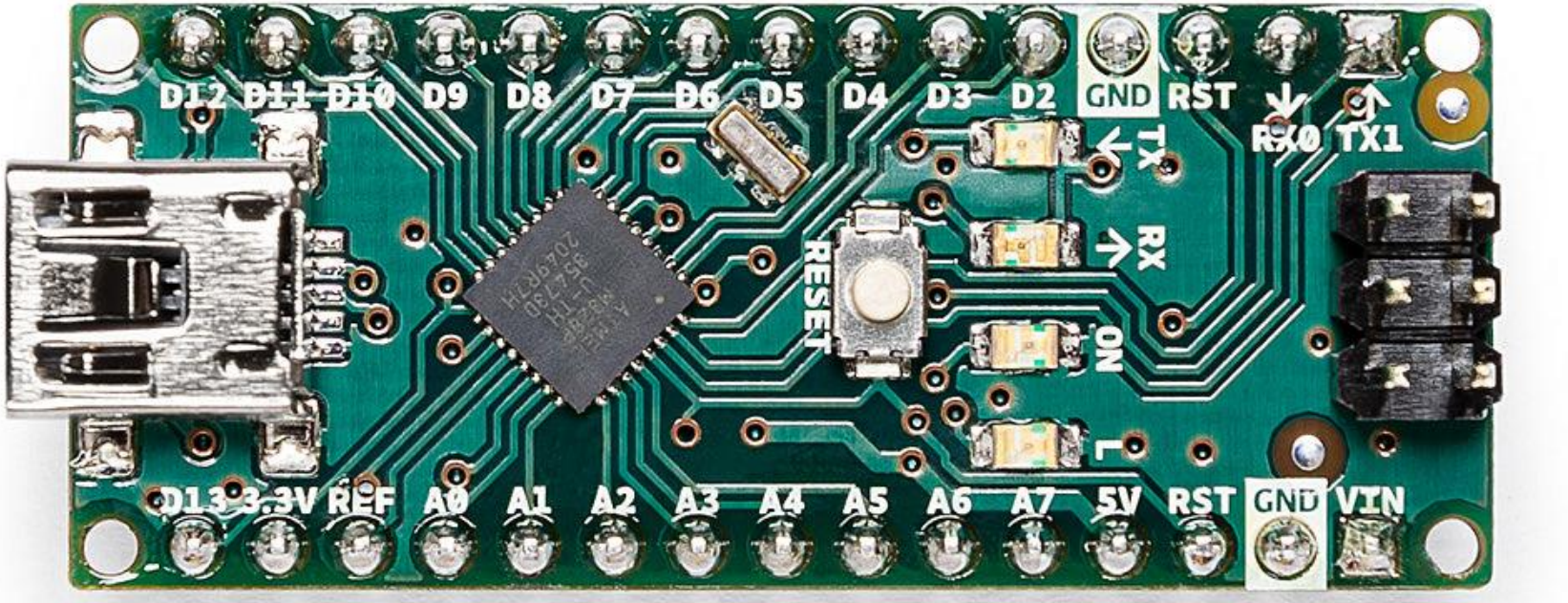
15. Analog Pins – These pins can read the signal from an analog sensor and convert it to digital



https://www.makerspaces.com/arduino-uno-tutorial-beginners/

Arduino Nano

The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328 (Arduino Nano 3.x). It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one.



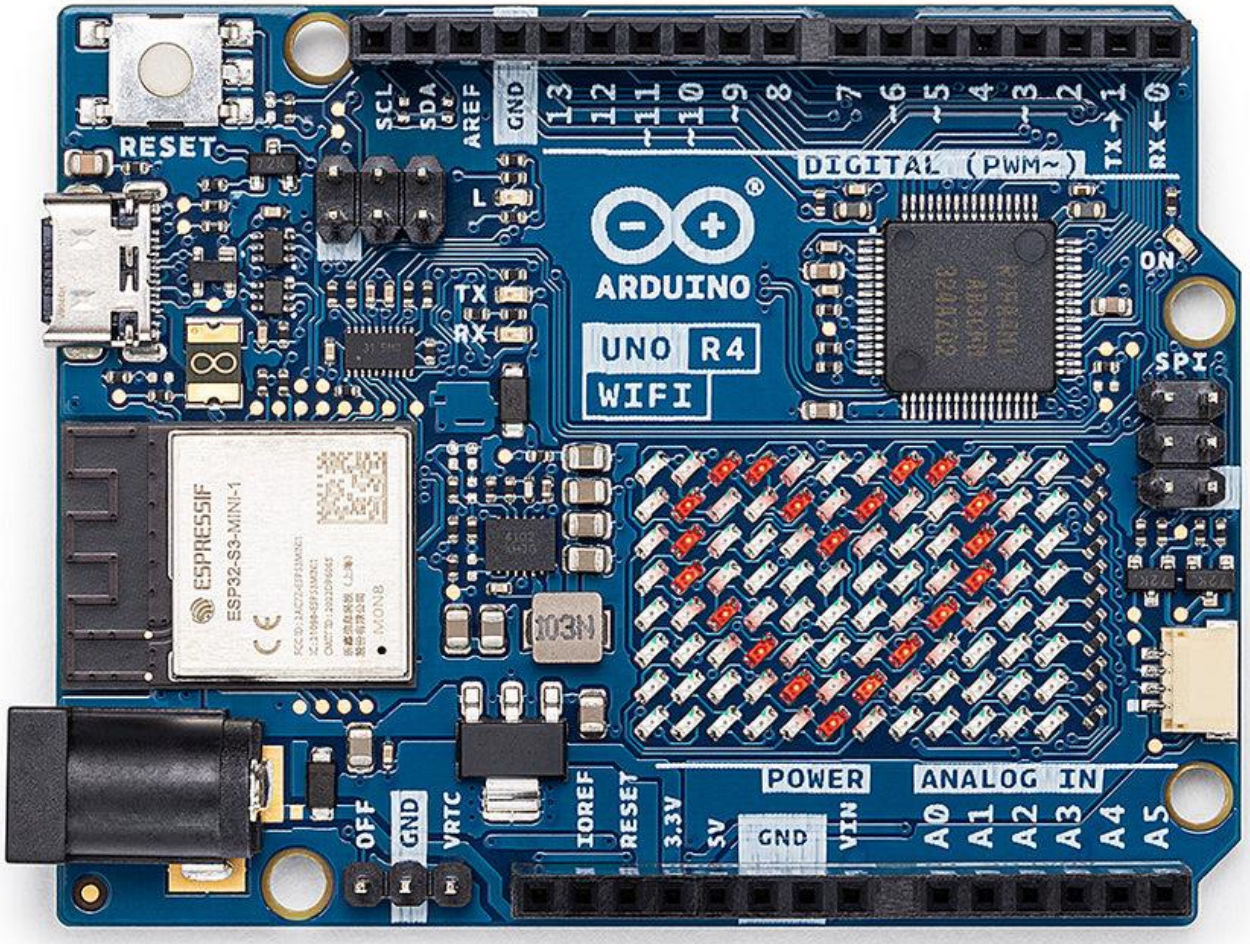
https://store.arduino.cc/products/arduinonano#:~:text=The%20Arduino%20Nano%20is%20a,instead%20of%20a%20standard%20one

Arduino Nano

Microcontroller	ATmega328
Architecture	AVR
Operating Voltage	5 V
Flash Memory	32 KB of which 2 KB used by bootloader
SRAM	2 KB
Clock Speed	16 MHz
Analog IN Pins	8
EEPROM	1 KB
DC Current per I/O Pins	20 mA (I/O Pins)
Input Voltage	7-12V
Digital I/O Pins	22 (6 of which are PWM)
PWM Output	6

UNO R4 WiFi

The Arduino UNO R4 WiFi is designed around the 32-bit microcontroller RA4M1 from Renesas while also featuring a ESP32 module for Wi-Fi® and Bluetooth® connectivity. Its distinctive 12x8 LED matrix makes it possible to prototype visuals directly on the board, and with a Qwiic connector, you can create projects plug-and-play style.



https://store.arduino.cc/products/uno-r4-wifi

UNO R4 WiFi

Board	Name	Arduino® UNO R4 WiFi
	SKU	ABX00087
Microcontroller	Renesas RA4M1 (Arm® Cortex®-M4)	
USB	USB-C®	Programming Port
Pins	Digital I/O Pins	14
Pins	Analog input pins	6
	DAC	1
	PWM pins	6
Communication	UART	Yes, 1x
	I2C	Yes, 1x
	SPI	Yes, 1x
	CAN	Yes 1 CAN Bus

Clock speed	Main core	48 MHz
	ESP32-S3	up to 240 MHz
Memory	RA4M1	256 kB Flash, 32 kB RAM
	ESP32-S3	384 kB ROM, 512 kB SRAM
Dimensions	Width	68.85 mm
	Length	53.34 mm

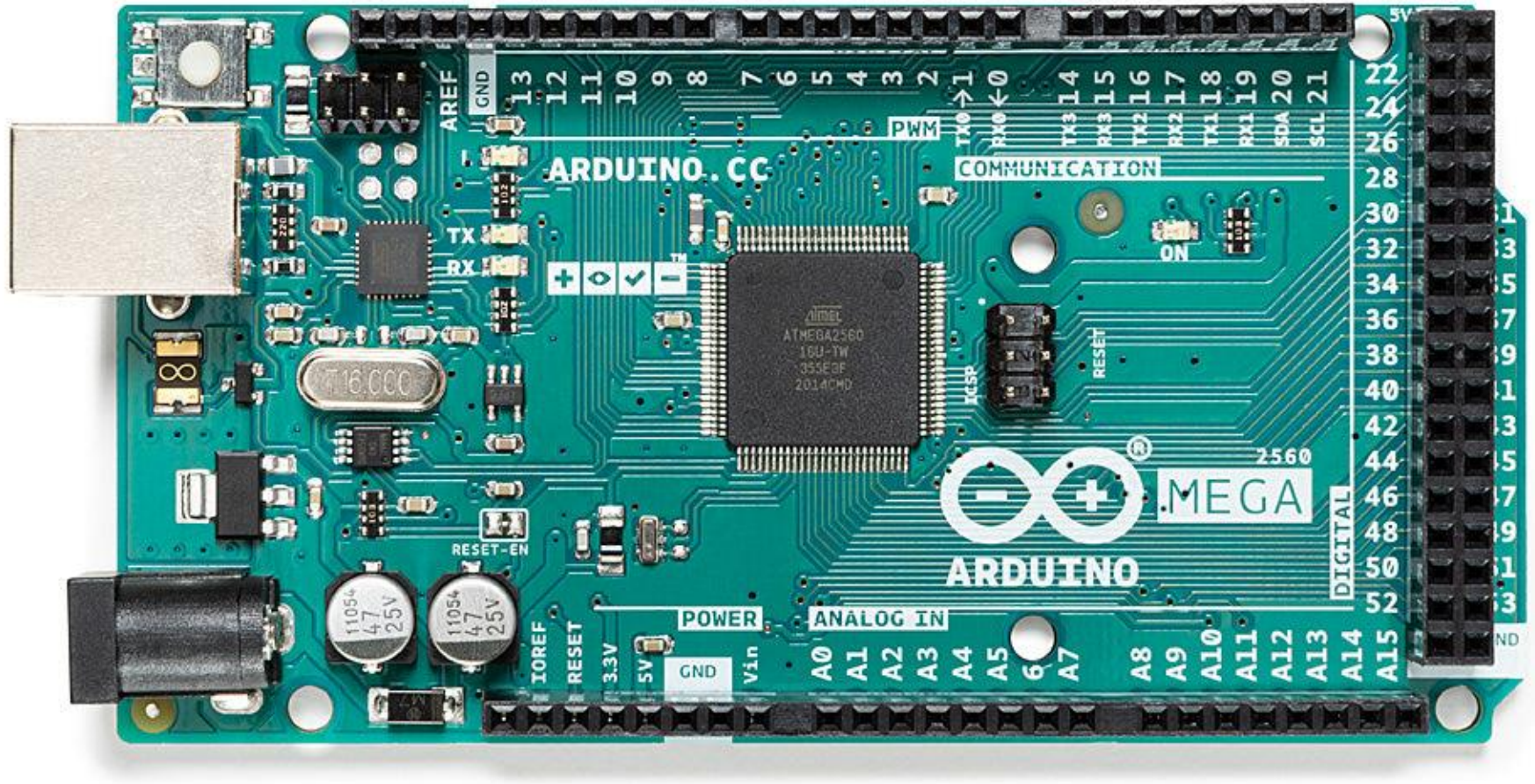
Arduino Mega

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins (of which 15 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega 2560 board is compatible with most shields designed for the Uno.

Arduino Mega

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
LED_BUILTIN	13

Arduino Mega



Pulse Width Modulation (PWM)

Pulse width modulation (PWM) is a technique that controls a signal by repeatedly switching it between high and low states in a consistent pattern. The width of the digital pulses is adjusted to create different average direct current (DC) voltages. PWM is used to control the average power or amplitude of an electrical signal.

(a) Modulating signal (sinusoidal)

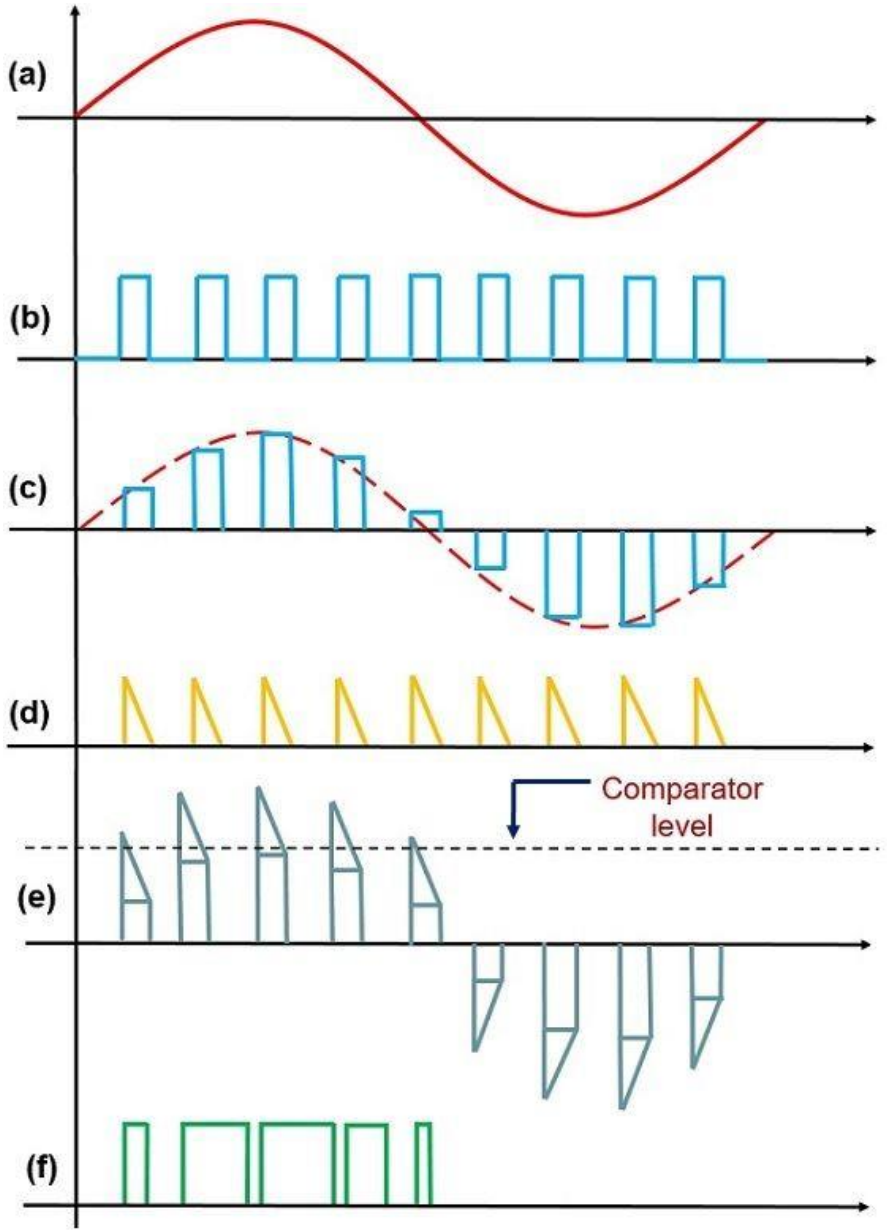
(b) Pulsed carrier

(c) Modulated signal (Pulse Amplitude Modulated – PAM)

(d) Ramp signal

(e) PAM + Ramp and compared with reference voltage

(f) PWM waveform



Waveform representation of PWM signal generation

Electronics Coach

Pulse Width Modulation (PWM)

Duty Cycle: Percentage of time the signal remains "on" (high) during one cycle which directly controls the average power delivered by the PWM signal.

$$\text{Duty Cycle (\%)} = \frac{\text{on time}}{\text{total cycle time}} \times 100$$

Example: PWM signal is on for 2 ms and the total cycle time is 10 ms. Duty cycle = (2/10)×100 = 20%

- 0% Duty Cycle: Signal is always off; no power delivered.
- 50% Duty Cycle: Signal is on half the time; delivers half the maximum power.
- 100% Duty Cycle: Signal is always on; delivers full power.