UNIT-I - OVERVIEW OF EMBEDDED SYSTEMS

Embedded System

. An embedded system can be thought of as a computer hardware system having software embedded in it. An embedded system can be an independent system or it can be a part of a large system. An embedded system is a microcontroller or microprocessor based system which is designed to perform a specific task. For example, a fire alarm is an embedded system; it will sense only smoke.

An embedded system has three components −

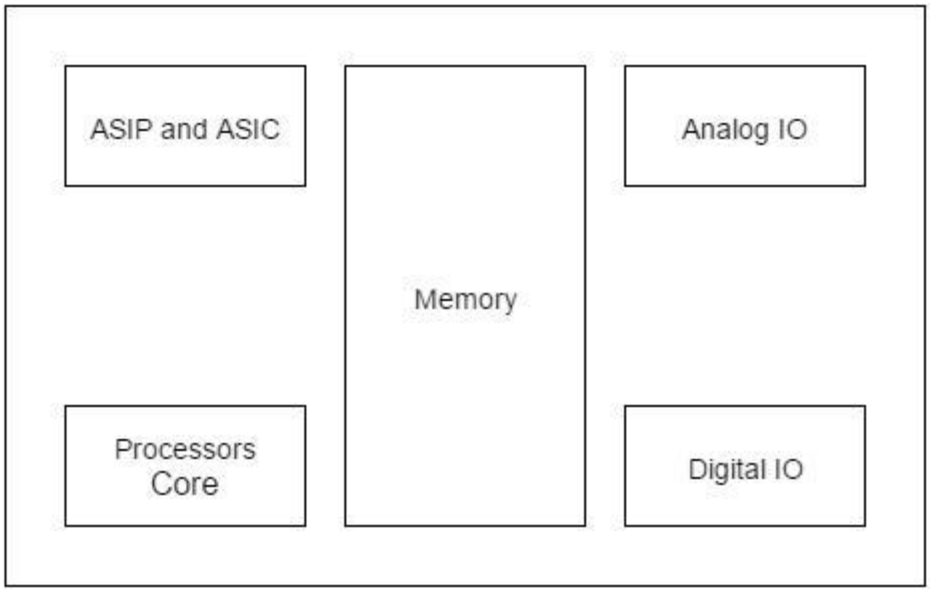
- It has hardware.
- It has application software.
- It has Real Time Operating system (RTOS) that supervises the application software and provide mechanism to let the processor run a process as per scheduling by following a plan to control the latencies. RTOS defines the way the system works. It sets the rules during the execution of application program. A small scale embedded system may not have RTOS.

So we can define an embedded system as a Microcontroller based, software driven, reliable, real-time control system.

Characteristics of an Embedded System

- **Single-functioned** − An embedded system usually performs a specialized operation and does the same repeatedly. For example: A pager always functions as a pager.
- **Tightly constrained** − All computing systems have constraints on design metrics, but those on an embedded system can be especially tight. Design metrics is a measure of an implementation's features such as its cost, size, power, and performance. It must be of a size to fit on a single chip, must perform fast enough to process data in real time and consume minimum power to extend battery life.

- **Reactive and Real time** − Many embedded systems must continually react to changes in the system's environment and must compute certain results in real time without any delay. Consider an example of a car cruise controller; it continually monitors and reacts to speed and brake sensors. It must compute acceleration or de-accelerations repeatedly within a limited time; a delayed computation can result in failure to control of the car.
- **Microprocessors based** − It must be microprocessor or microcontroller based.
- **Memory** − It must have a memory, as its software usually embeds in ROM. It does not need any secondary memories in the computer.
- **Connected** − It must have connected peripherals to connect input and output devices.
- **HW-SW systems** − Software is used for more features and flexibility. Hardware is used for performance and security.



Advantages

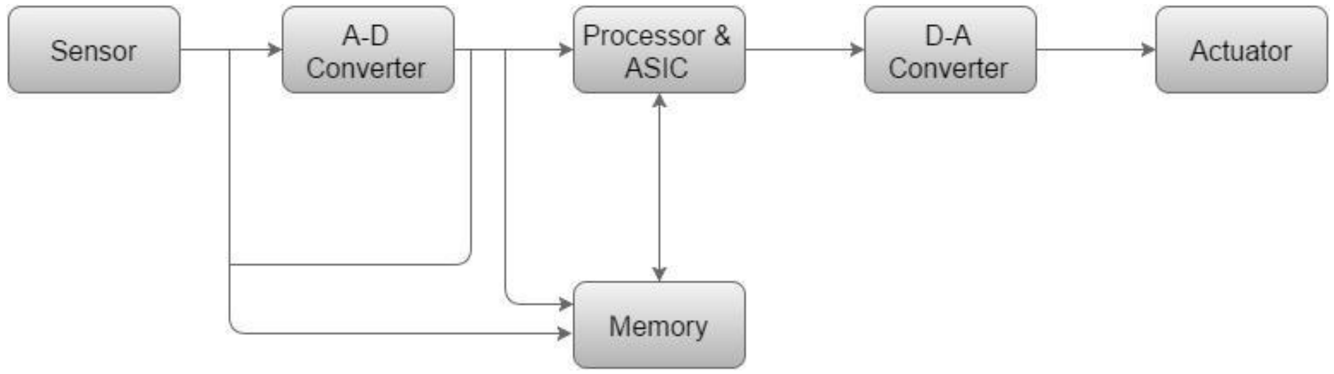
- Easily Customizable
- Low power consumption
- Low cost
- Enhanced performance

Disadvantages

- High development effort
- Larger time to market

Basic Structure of an Embedded System

The following illustration shows the basic structure of an embedded system −



- **Sensor** − It measures the physical quantity and converts it to an electrical signal which can be read by an observer or by any electronic instrument like an A2D converter. A sensor stores the measured quantity to the memory.

- **A-D Converter** − An analog-to-digital converter converts the analog signal sent by the sensor into a digital signal.

- **Processor & ASICs** − Processors process the data to measure the output and store it to the memory.

- **D-A Converter** − A digital-to-analog converter converts the digital data fed by the processor to analog data

- **Actuator** − An actuator compares the output given by the D-A Converter to the actual (expected) output stored in it and stores the approved output.

Processor is the heart of an embedded system. It is the basic unit that takes inputs and produces an output after processing the data. For an embedded system designer, it is necessary to have the knowledge of both microprocessors and microcontrollers.

Processors in a System

A processor has two essential units −

- Program Flow Control Unit (CU)
- Execution Unit (EU)

The CU includes a fetch unit for fetching instructions from the memory. The EU has circuits that implement the instructions pertaining to data transfer operation and data conversion from one form to another.

The EU includes the Arithmetic and Logical Unit (ALU) and also the circuits that execute instructions for a program control task such as interrupt, or jump to another set of instructions.

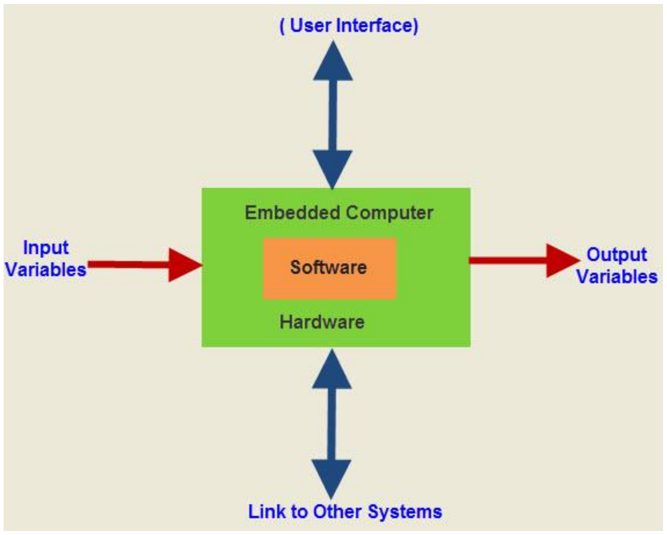
A processor runs the cycles of fetch and executes the instructions in the same sequence as they are fetched from memory.

Types of Processors

Processors can be of the following categories −

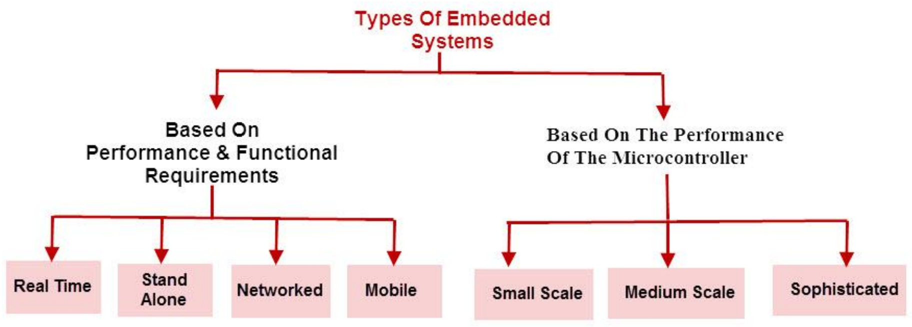
- General Purpose Processor (GPP)
    - Microprocessor
    - Microcontroller
    - Embedded Processor
    - Digital Signal Processor
    - Media Processor
- Application Specific System Processor (ASSP)
- Application Specific Instruction Processors (ASIPs)

.The Embedded system hardware includes elements like user interface, Input/Output interfaces, display and memory, etc.Generally, an embedded system comprises power supply, processor, memory, timers, serial communication ports and system application specific circuits.



Types of Embedded Systems

Embedded systems can be classified into different types based on performance, functional requirements and performance of the microcontroller.


Types Of Embedded Systems
Based On Performance & Functional Requirements
Based On The Performance Of The Microcontroller
Real Time
Stand Alone
Networked
Mobile
Small Scale
Medium Scale
Sophisticated

Types of Embedded systems

Embedded systems are classified into four categories based on their performance and functional requirements:

- Stand alone embedded systems
- Real time embedded systems

- Networked embedded systems
- Mobile embedded systems

Embedded Systems are classified into three types based on the performance of the microcontroller such as

- Small scale embedded systems
- Medium scale embedded systems
- Sophisticated embedded systems

Stand Alone Embedded Systems

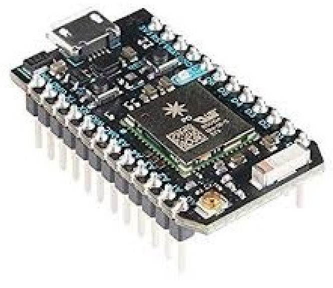
Stand alone embedded systems do not require a host system like a computer, it works by itself. It takes the input from the input ports either analog or digital and processes, calculates and converts the data and gives the resulting data through the connected device-Which either controls, drives and displays the connected devices. Examples for the stand alone embedded systems are mp3 players, digital cameras, video game consoles, microwave ovens and temperature measurement systems.

Real Time Embedded Systems

A real time embedded system is defined as, a system which gives a required o/p in a particular time.These types of embedded systems follow the time deadlines for completion of a task. Real time embedded systems are classified into two types such as soft and hard real time systems.

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. The embedded web server is a type of system wherein all embedded devices are connected to a web server and accessed and controlled by a web browser. Example for the LAN networked embedded system is a home security system wherein all sensors are connected and run on the protocol TCP/IP



- OPEN SOURCE EMBEDDED PLATFORMS

Arduino : Founder is Massimo Banzi
Raspberry Pi: Founder is Eben Upton

MCU (Microcontroller Unit)

- Small computer on a single integrated circuit
- Capable of performing specific task
- Eg: car alarm, washing machine, microwave oven etc.
- Components of MCU: processor, memory, storage, and programmable input and output pins
- Arduino and photon particle are MCU.

Single Board Computer(SBC)

- Complete computer that is built on a single circuit board
- Provides a fanless, low-power computing solution and a low profile architecture
- Components:-
- CPU
- RAM
- Ports and Secondary Storage (both inbuilt and external supported).
- Raspberry PI, Orange pi, Banana pi are SBC.

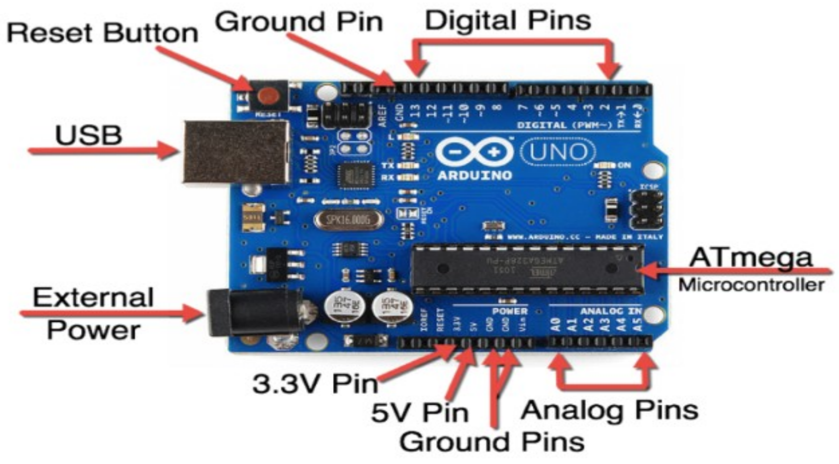
Difference between SBC and MCU

- Hardware
- Memory
- OS
- Ex of MCU : Arduino, Photon Particle
- Ex of SBC: Raspberry pi

Arduino
- Open-source platform used for building electronics prototypes or devices that interact with real world
- Consists of programmable circuit board (H/W) and IDE(S/W)

Why Arduino
- Cross-Platform
- Simple and clear programming environment
- Open source and extensible software

Arduino Uno



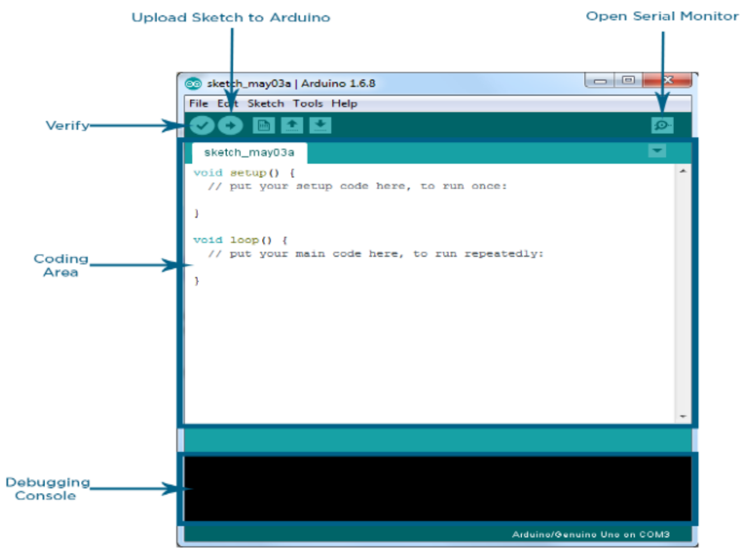
Memory in Arduino

- 32 KB flash memory
  Flash memory (program space), is where the Arduino sketch is stored.
- SRAM-2 KB
  SRAM (static random access memory) is where the sketch creates and manipulates variables when it runs.
- EEPROM-1KB
  EEPROM is memory space that programmers can use to store long-term information. Flash memory and EEPROM memory are non-volatile (the information persists after the power is turned off).
  SRAM is volatile and will be lost when the power is cycled.

Input/ Output pins in Arduino

- 14-digital pins
- Serial pins of an Arduino board are TX (1) and RX (0)
- PWM pins of an Arduino are 3, 5, 6, 9, 10, & 11
- Analog pins A0 – A5

Arduino IDE



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

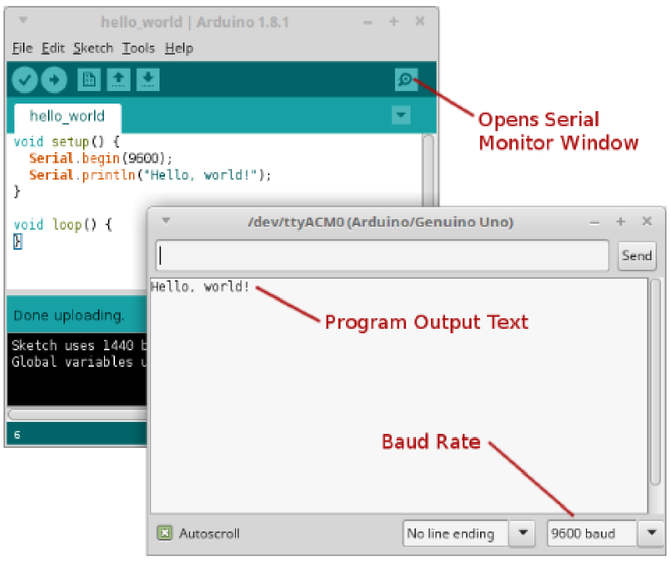
Void Setup()

- Use it to initialize variables, pin modes, start using libraries, etc.
- Function will only run once, after each powerup or reset of the Arduino board.

Void Loop()
- Function executes repeatedly until arduino turned off or restarted
- It contains the main logic of code

Serial.print()
- To display the text and values on the screen
- To put the Serial.begin(9600) statement in setup()

Program Structure


hello_world | Arduino 1.8.1
File Edit Sketch Tools Help
hello_world
Opens Serial Monitor Window
/dev/ttyACM0 (Arduino/Genuino Uno)
Send
Program Output Text
Baud Rate
Autoscroll

```
void setup() {
  Serial.begin(9600);
  Serial.println("Hello, world!");
}

void loop() {
}
```

Done uploading.

```
Sketch uses 1440 b
Global variables u
```

```
Hello, world!
```

No line ending
9600 baud

PINMODE() Function

- Configures the specified pin to behave either as an input or an output.
- Syntax		pinMode(pin, mode)
  pin: the number of the pin whose mode you wish to set
  mode: INPUT, OUTPUT,

Delay() Function
- Pauses the program for the amount of time (in miliseconds)
  Syntax		delay(ms)

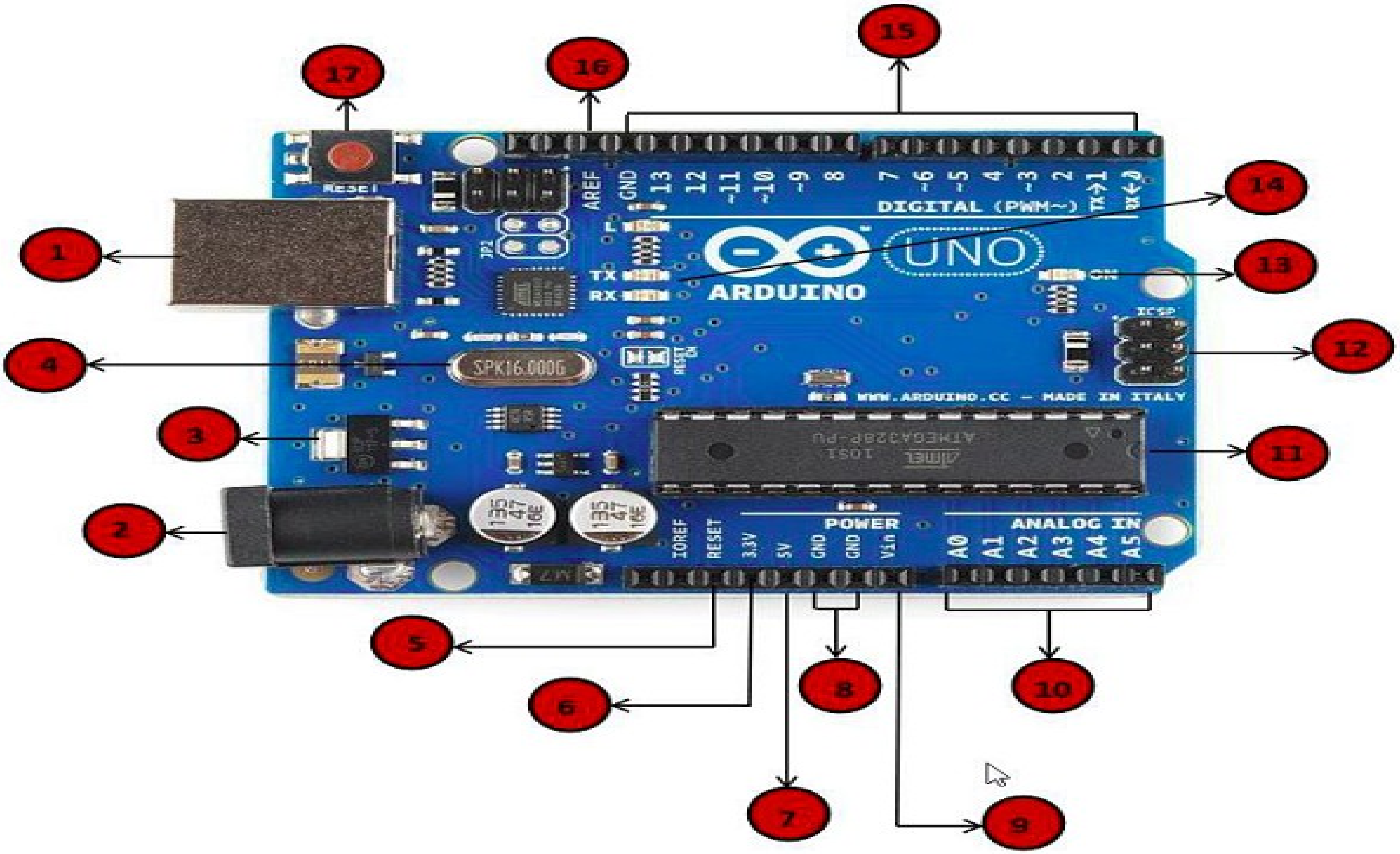
digitalRead() Function
- Reads the value from a specified digital pin, either HIGH or LOW.
- Syntax: digitalRead(pin)
  pin: the Arduino pin number you want to read
  Returns: HIGH or LOW

digitalWrite() Function

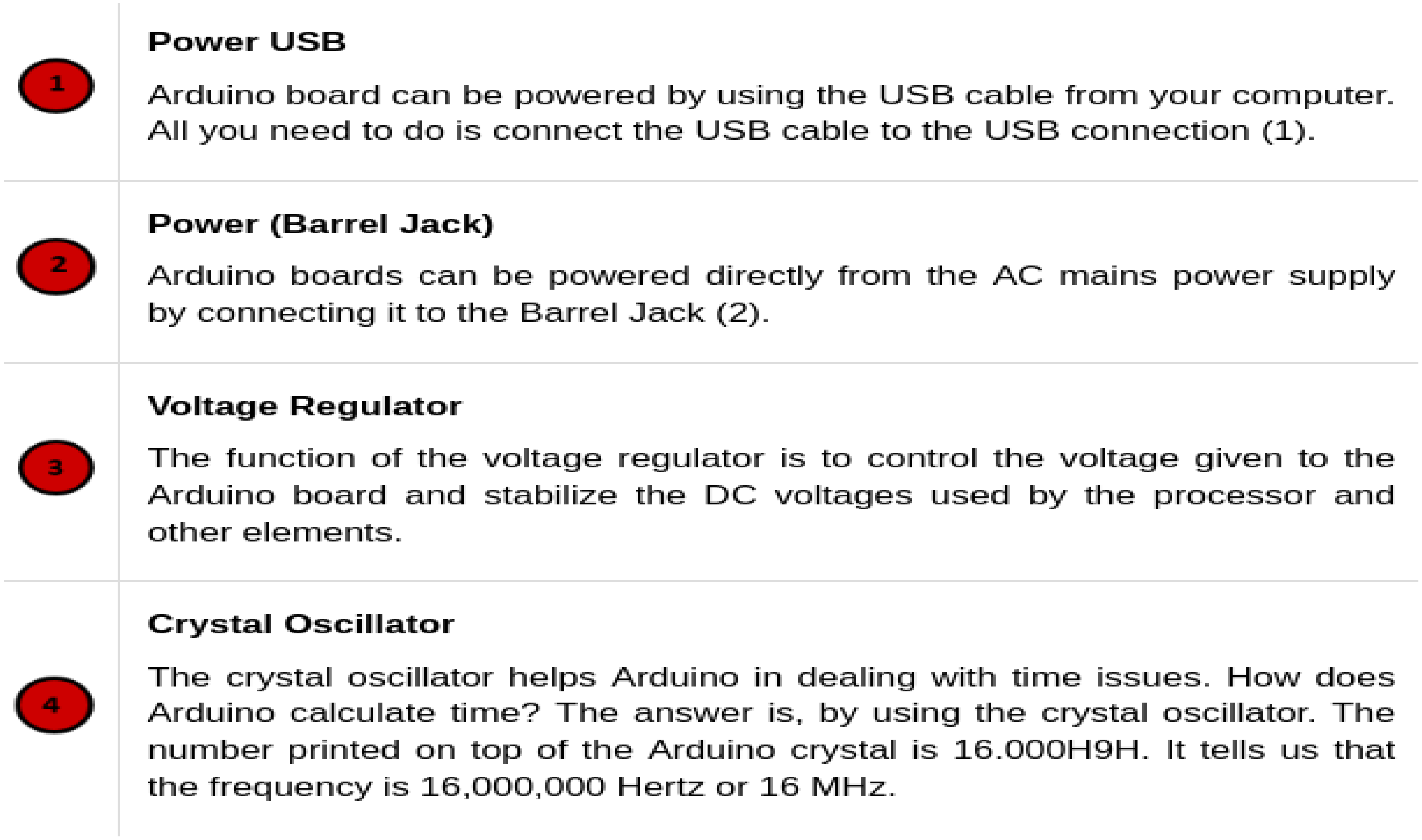
- Write a HIGH or a LOW value to a digital pin.
  Syntax digitalWrite(pin, value)
  pin: the Arduino pin number.
  value: HIGH or LOW

Arduino Uno

Arduino Uno Pin Description



Continue...



1	**Power USB** Arduino board can be powered by using the USB cable from your computer. All you need to do is connect the USB cable to the USB connection (1).
2	**Power (Barrel Jack)** Arduino boards can be powered directly from the AC mains power supply by connecting it to the Barrel Jack (2).
3	**Voltage Regulator** The function of the voltage regulator is to control the voltage given to the Arduino board and stabilize the DC voltages used by the processor and other elements.
4	**Crystal Oscillator** The crystal oscillator helps Arduino in dealing with time issues. How does Arduino calculate time? The answer is, by using the crystal oscillator. The number printed on top of the Arduino crystal is 16.000H9H. It tells us that the frequency is 16,000,000 Hertz or 16 MHz.

Continue...

5,17

Arduino Reset

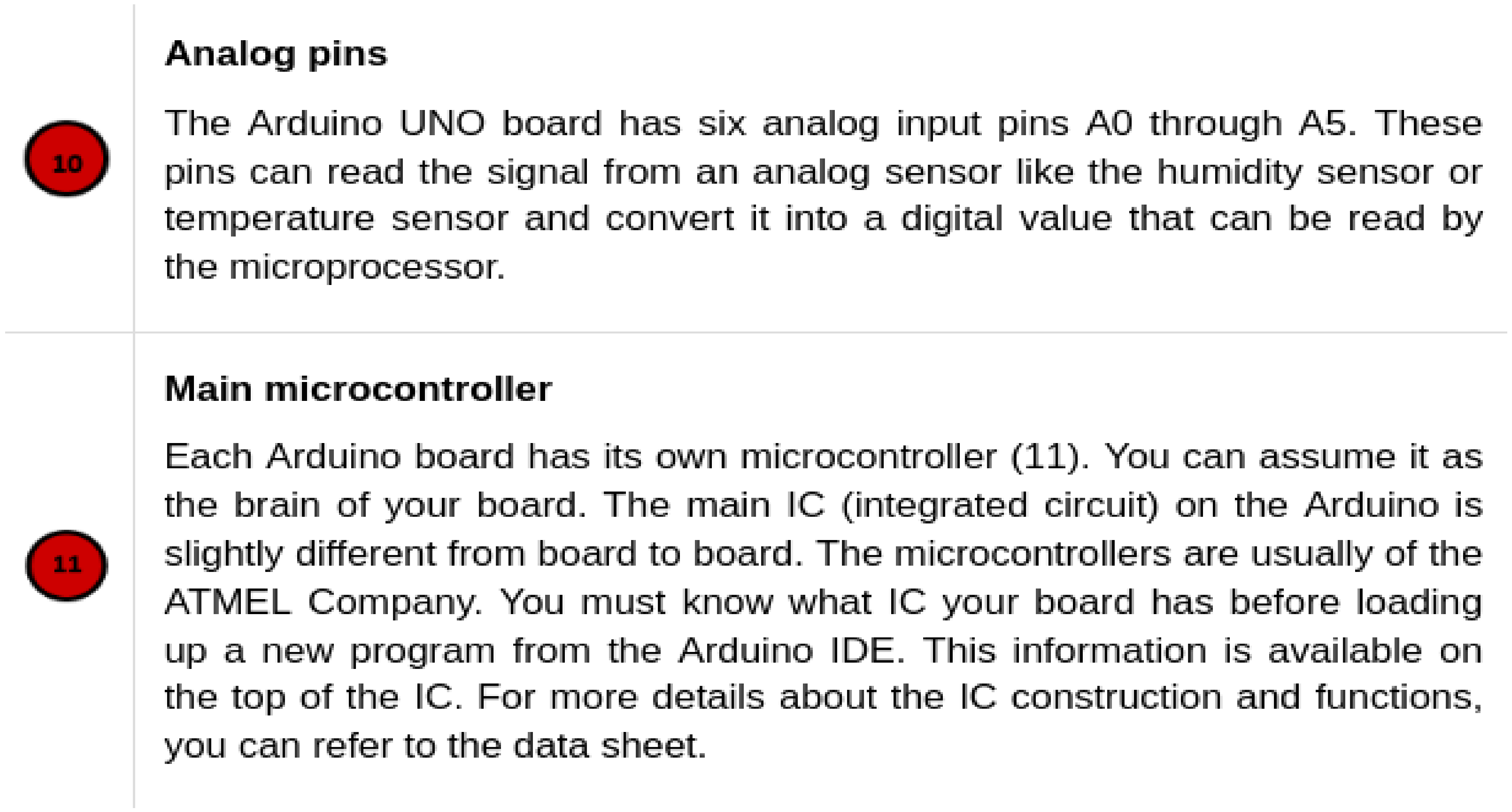
You can reset your Arduino board, i.e., start your program from the beginning. You can reset the UNO board in two ways. First, by using the reset button (17) on the board. Second, you can connect an external reset button to the Arduino pin labelled RESET (5).

6,7
8,9

Pins (3.3, 5, GND, Vin)

- 3.3V (6) – Supply 3.3 output volt
- 5V (7) – Supply 5 output volt
- Most of the components used with Arduino board works fine with 3.3 volt and 5 volt.
- GND (8)(Ground) – There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- Vin (9) – This pin also can be used to power the Arduino board from an external power source, like AC mains power supply.

Continue...



10	**Analog pins** The Arduino UNO board has six analog input pins A0 through A5. These pins can read the signal from an analog sensor like the humidity sensor or temperature sensor and convert it into a digital value that can be read by the microprocessor.
11	**Main microcontroller** Each Arduino board has its own microcontroller (11). You can assume it as the brain of your board. The main IC (integrated circuit) on the Arduino is slightly different from board to board. The microcontrollers are usually of the ATMEL Company. You must know what IC your board has before loading up a new program from the Arduino IDE. This information is available on the top of the IC. For more details about the IC construction and functions, you can refer to the data sheet.

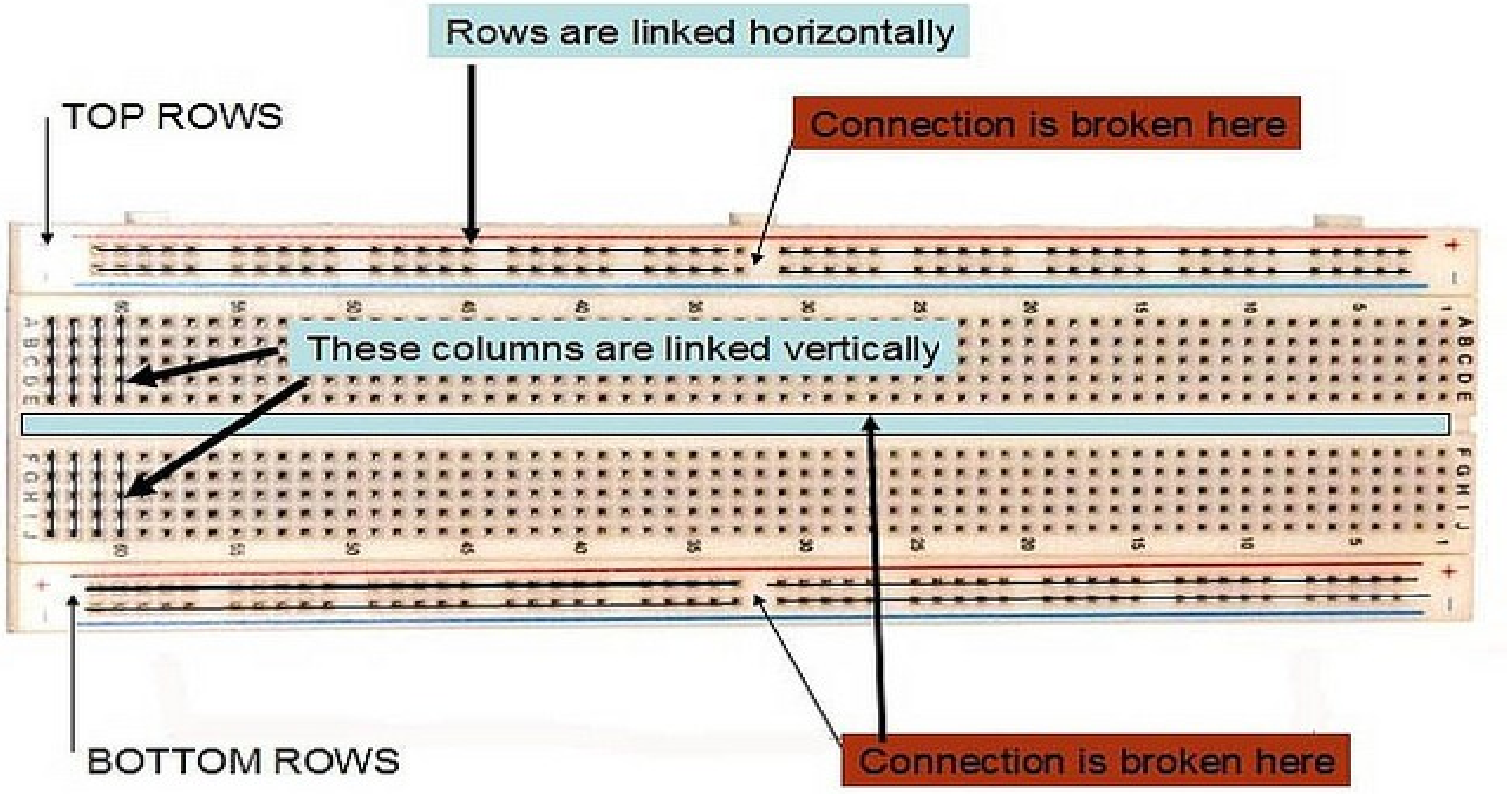
Continue...

12	**ICSP pin** Mostly, ICSP (12) is an AVR, a tiny programming header for the Arduino consisting of MOSI, MISO, SCK, RESET, VCC, and GND. It is often referred to as an SPI (Serial Peripheral Interface), which could be considered as an "expansion" of the output. Actually, you are slaving the output device to the master of the SPI bus.
13	**Power LED indicator** This LED should light up when you plug your Arduino into a power source to indicate that your board is powered up correctly. If this light does not turn on, then there is something wrong with the connection.
14	**TX and RX LEDs** On your board, you will find two labels: TX (transmit) and RX (receive). They appear in two places on the Arduino UNO board. First, at the digital pins 0 and 1, to indicate the pins responsible for serial communication. Second, the TX and RX led (13). The TX led flashes with different speed while sending the serial data. The speed of flashing depends on the baud rate used by the board. RX flashes during the receiving process.

Continue...

15	**Digital I/O** The Arduino UNO board has 14 digital I/O pins (15) (of which 6 provide PWM (Pulse Width Modulation) output. These pins can be configured to work as input digital pins to read logic values (0 or 1) or as digital output pins to drive different modules like LEDs, relays, etc. The pins labeled “~” can be used to generate PWM.
16	**AREF** AREF stands for Analog Reference. It is sometimes, used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins.

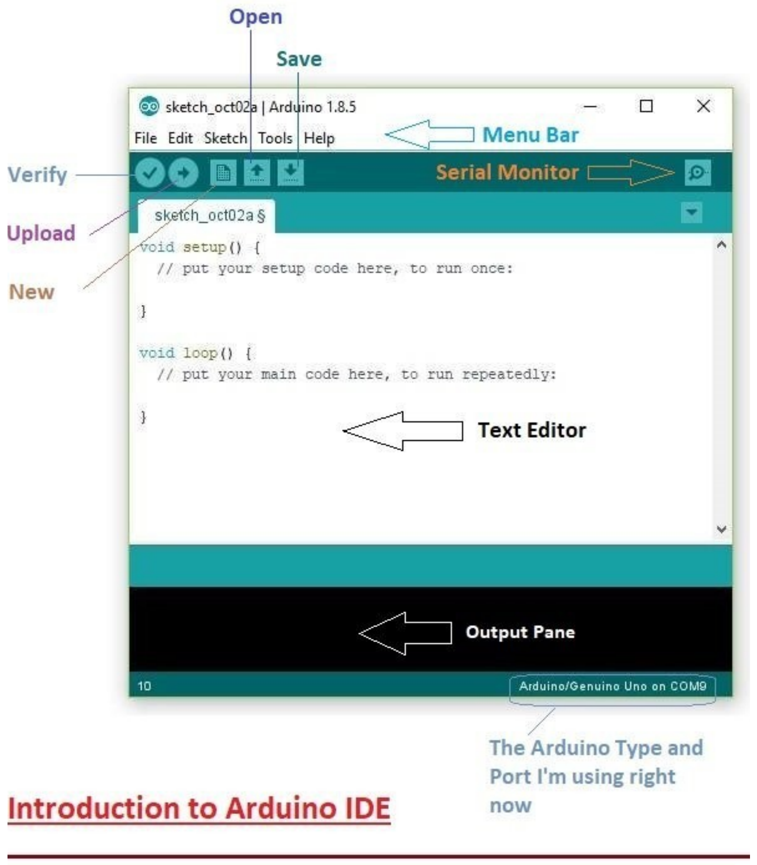
BreadBoard


Rows are linked horizontally
TOP ROWS
Connection is broken here
These columns are linked vertically
BOTTOM ROWS
Connection is broken here

Arduino IDE

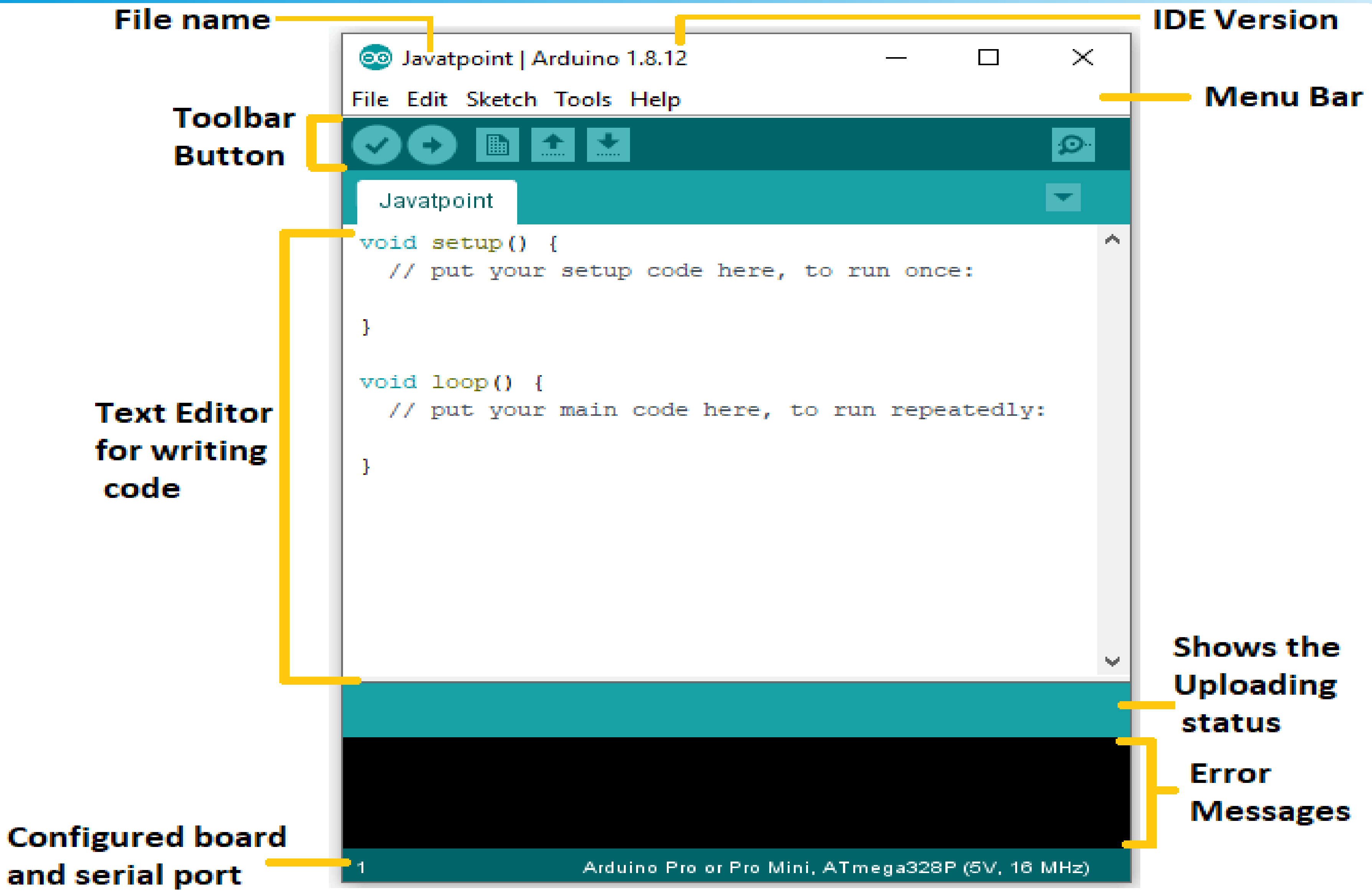
- The IDE environment is mainly distributed into three sections

1. Menu Bar
2. Text Editor
3. Output Pane

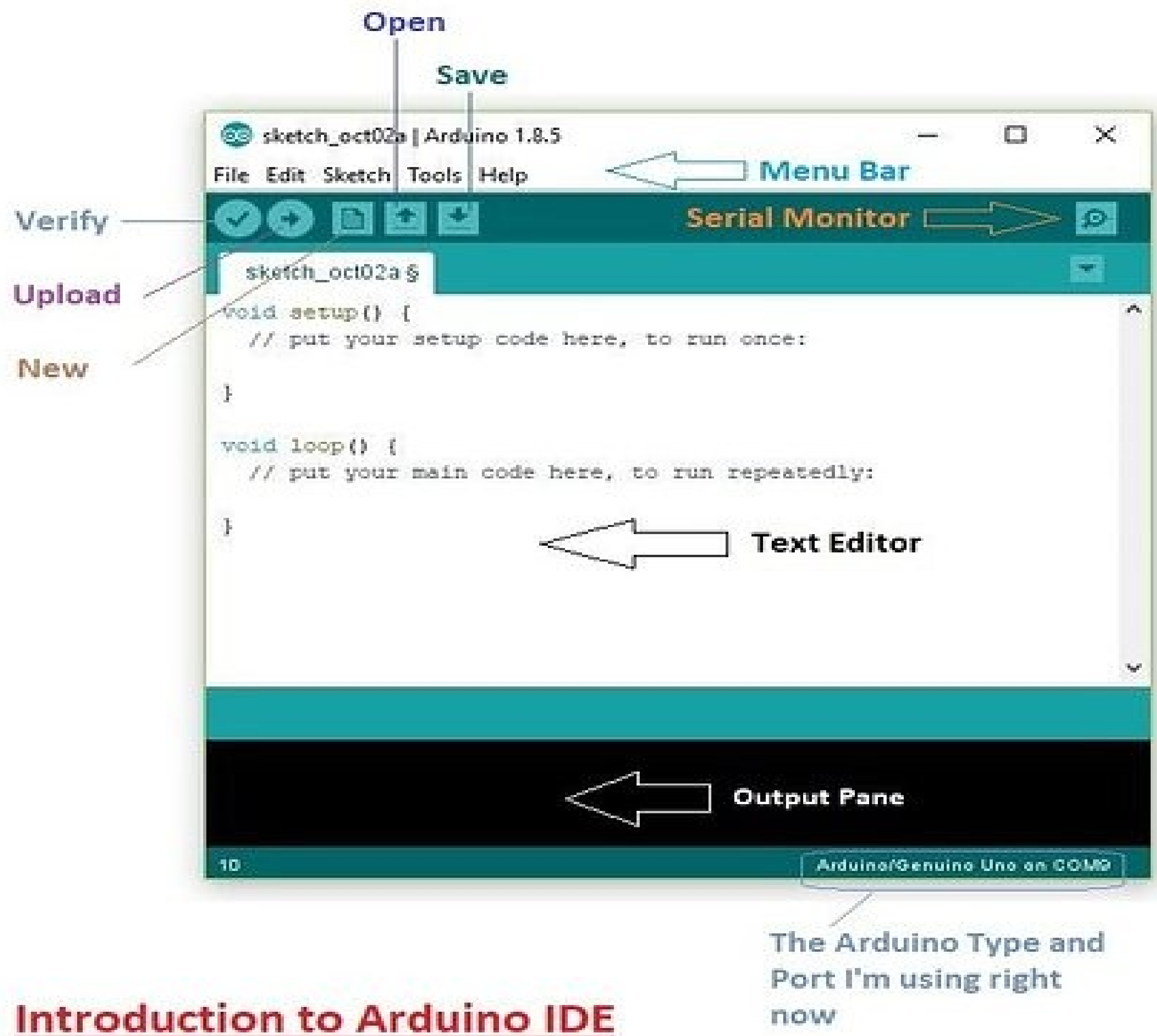

Open
Save
sketch_oct02a | Arduino 1.8.5
File Edit Sketch Tools Help
Menu Bar
Verify
Upload
New
Serial Monitor
sketch_oct02a §
Text Editor
Output Pane
10
Arduino/Genuino Uno on COM9
The Arduino Type and Port I'm using right now
Introduction to Arduino IDE

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Arduino IDE

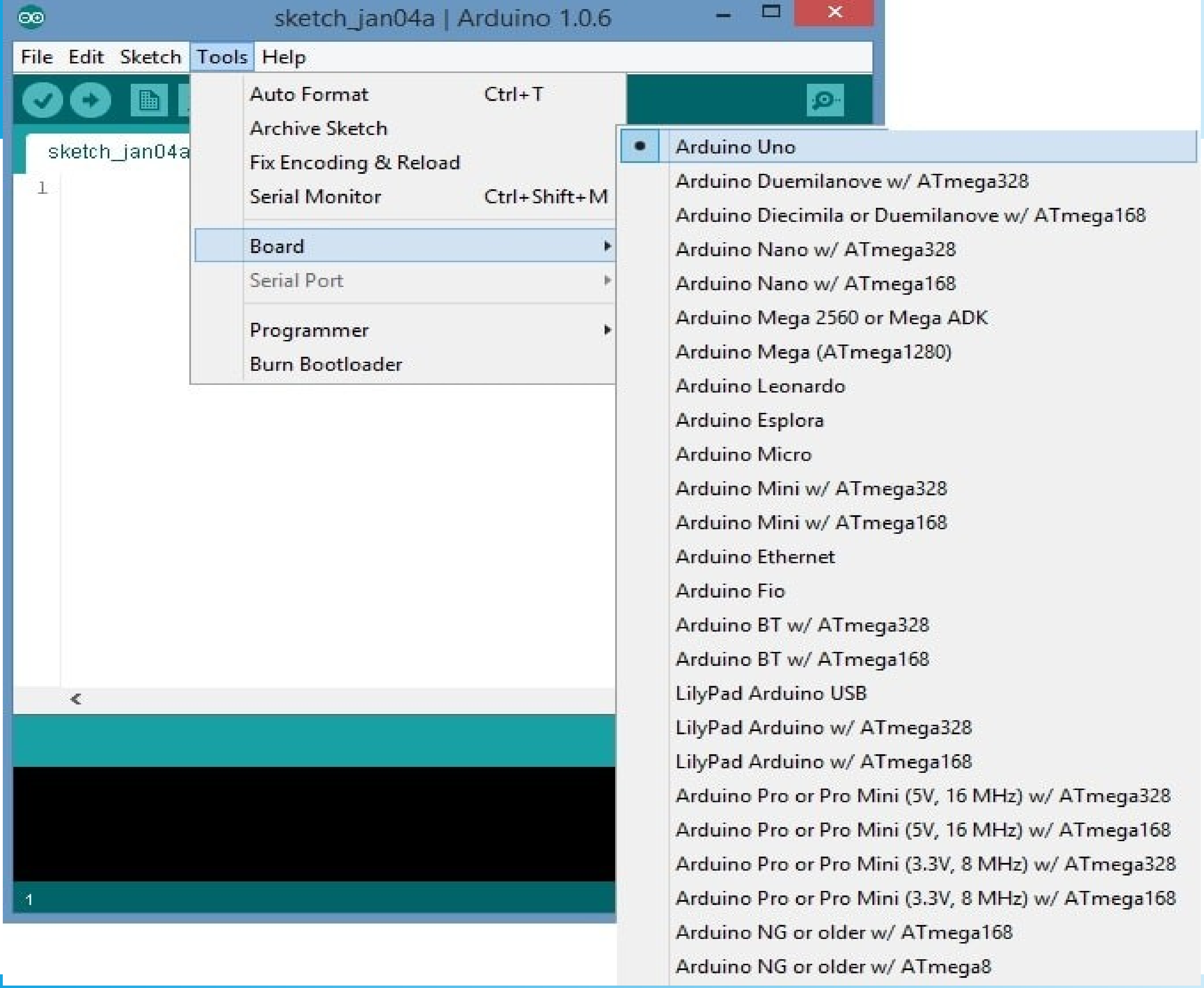

File name
IDE Version
Menu Bar
Toolbar Button
Text Editor for writing code
Shows the Uploading status
Error Messages
Configured board and serial port
Javatpoint | Arduino 1.8.12
File Edit Sketch Tools Help
Javatpoint
1
Arduino Pro or Pro Mini, ATmega328P (5V, 16 MHz)

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```


Open
Save
sketch_oct02a | Arduino 1.8.5
File Edit Sketch Tools Help
Menu Bar
Verify
Serial Monitor
Upload
sketch_oct02a §
New
Text Editor
Output Pane
10
Arduino/Genuino Uno on COM9
The Arduino Type and Port I'm using right now

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

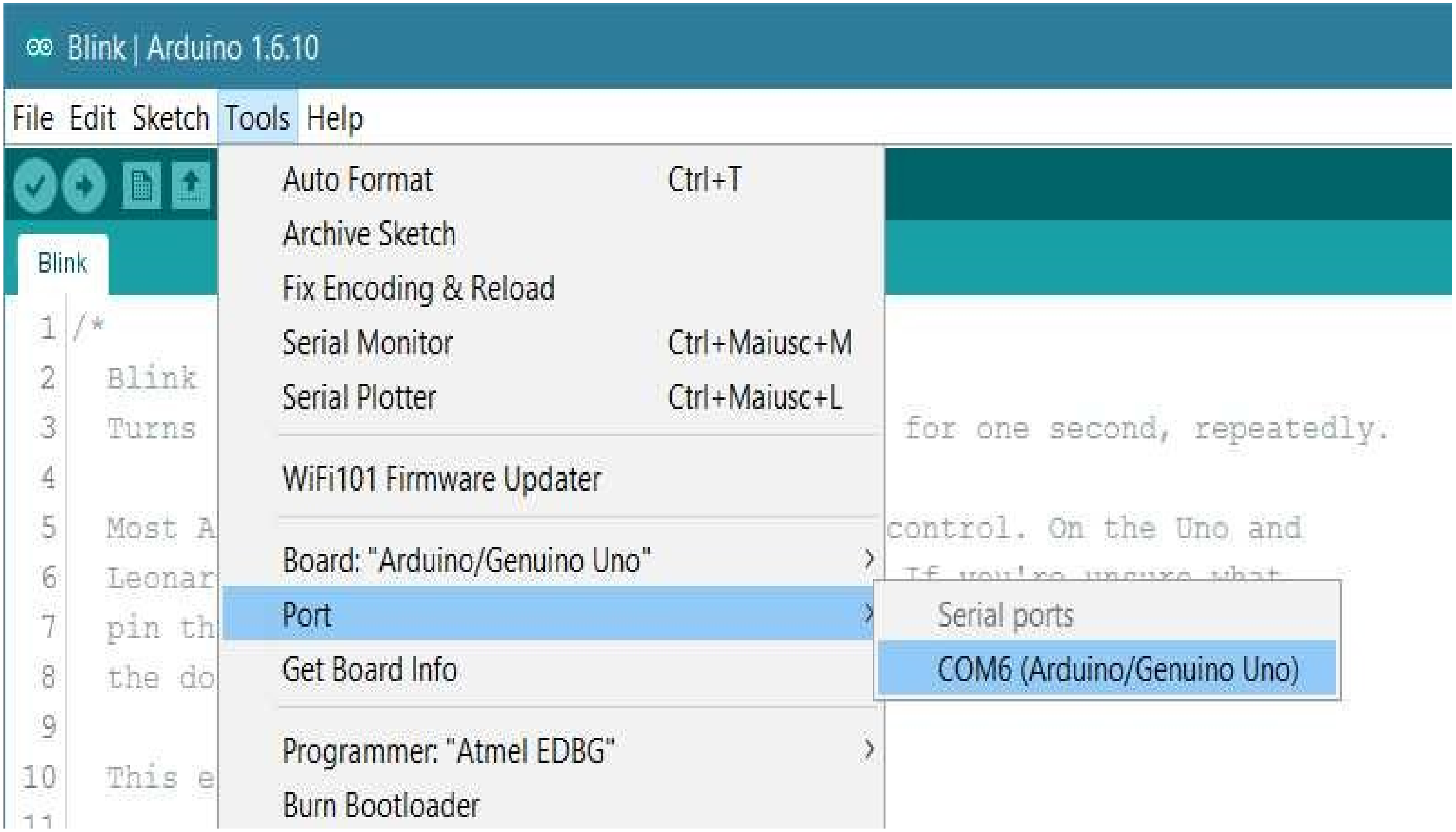
Introduction to Arduino IDE


sketch_jan04a | Arduino 1.0.6

File Edit Sketch Tools Help

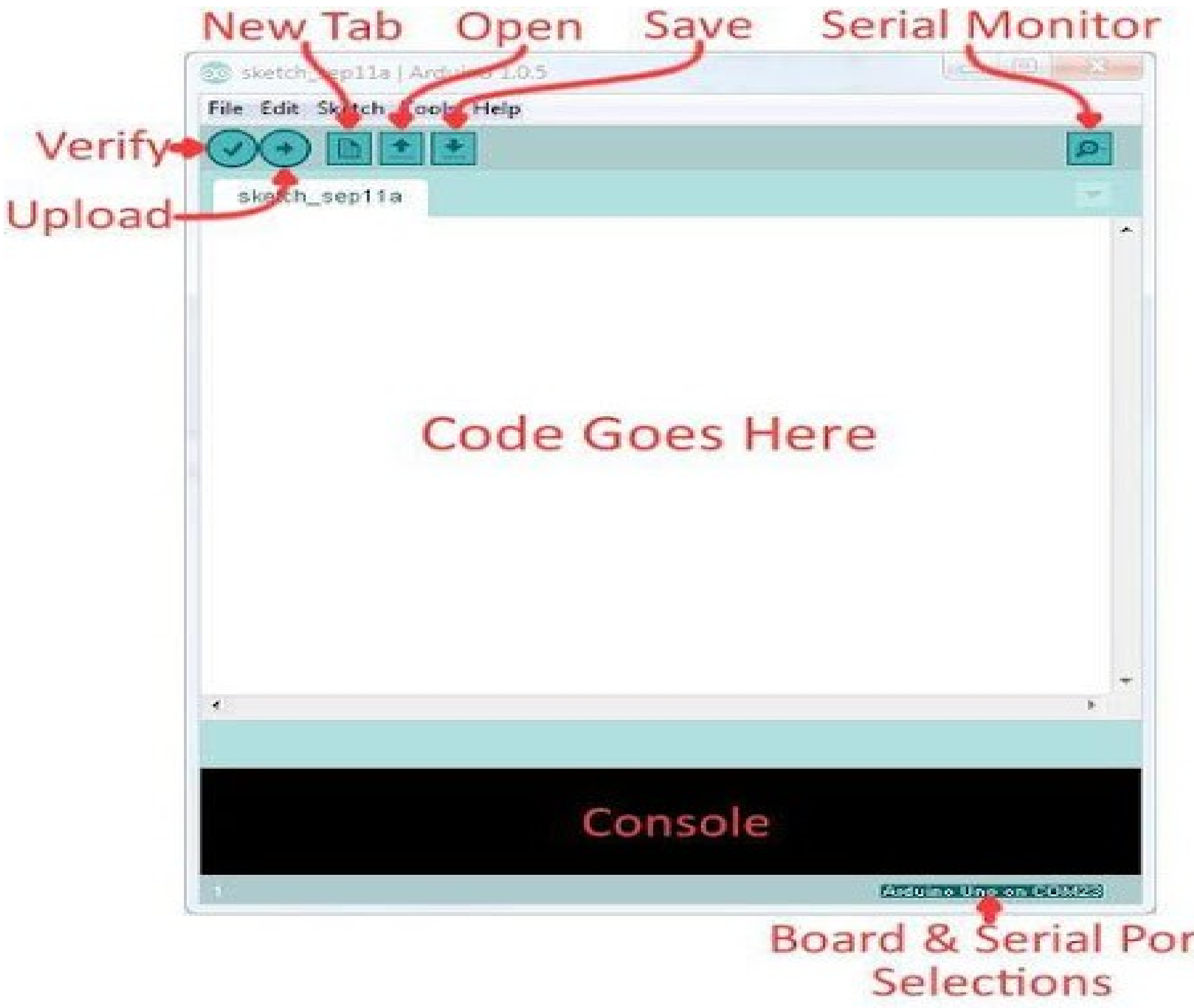
sketch_jan04a

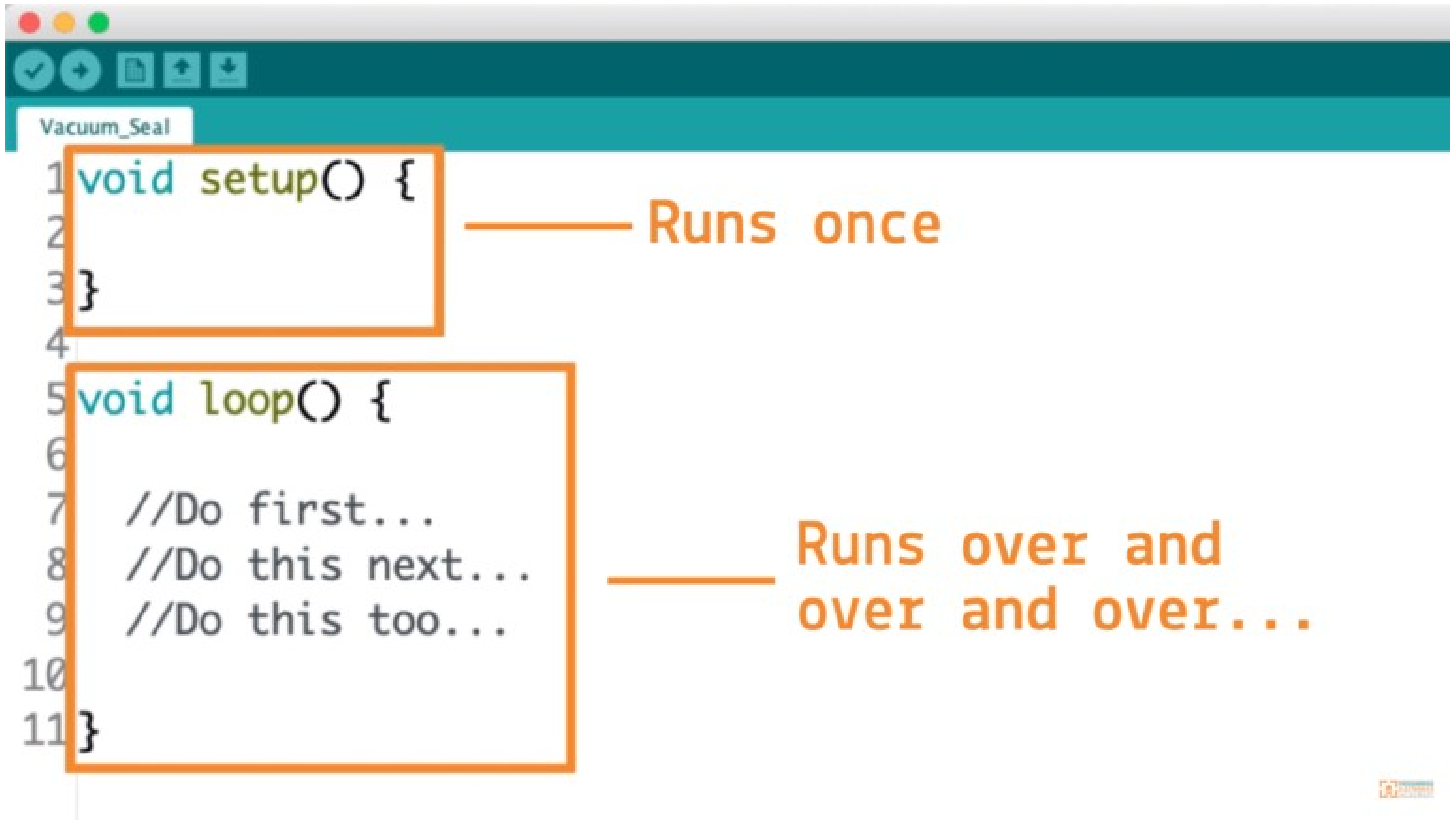
- ● Arduino Uno
- Arduino Duemilanove w/ ATmega328
- Arduino Diecimila or Duemilanove w/ ATmega168
- Arduino Nano w/ ATmega328
- Arduino Nano w/ ATmega168
- Arduino Mega 2560 or Mega ADK
- Arduino Mega (ATmega1280)
- Arduino Leonardo
- Arduino Esplora
- Arduino Micro
- Arduino Mini w/ ATmega328
- Arduino Mini w/ ATmega168
- Arduino Ethernet
- Arduino Fio
- Arduino BT w/ ATmega328
- Arduino BT w/ ATmega168
- LilyPad Arduino USB
- LilyPad Arduino w/ ATmega328
- LilyPad Arduino w/ ATmega168
- Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
- Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
- Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
- Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
- Arduino NG or older w/ ATmega168
- Arduino NG or older w/ ATmega8



```
/*
  Blink
  Turns                                    for one second, repeatedly.

  Most A                                 control. On the Uno and
  Leonar                                 If you're unsure what
  pin th
  the do

  This e
```

- Then press verify (if no errors found then)
- Then press upload





```
void setup() {

}

void loop() {

  //Do first...
  //Do this next...
  //Do this too...

}
```

Arduino Menu Bar

- Edit - Used for copying and pasting the code with further modification for font
- Sketch - For compiling and programming
- Tools - Mainly used for testing projects. The Programmer section in this panel is used for burning a bootloader to the new microcontroller.
- Help - In case you are feeling skeptical about software, complete help is available from getting started to troubleshooting.

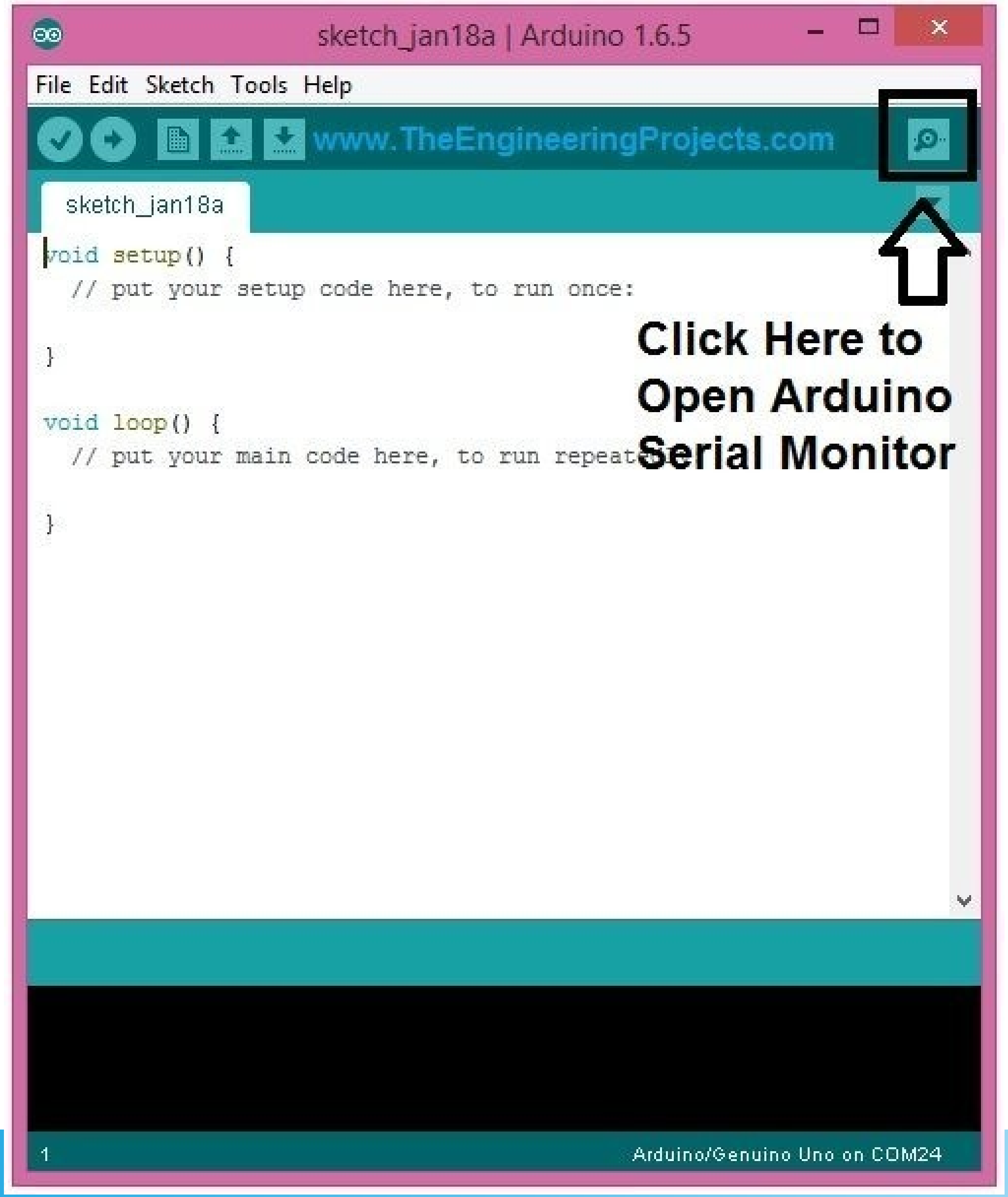
Continue

- Serial Monitor - A separate pop-up window that acts as an independent terminal and plays a vital role for sending and receiving the Serial Data. You can also go to the Tools panel and select Serial Monitor, or pressing Ctrl+Shift+M all at once will open it instantly. The Serial Monitor will actually help to debug the written Sketches where you can get a hold of how your program is operating.

Continue

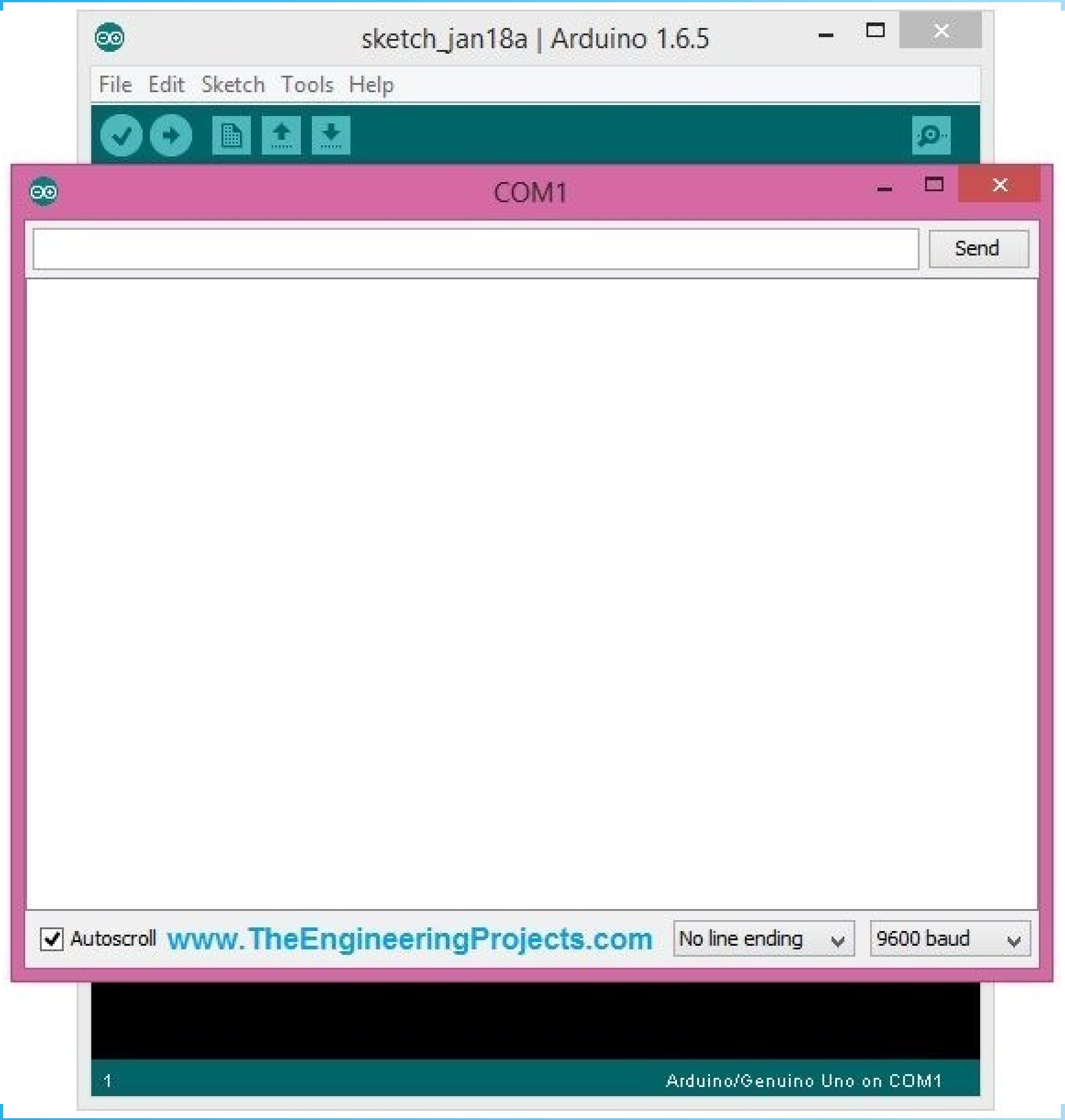
- Your Arduino Module should be connected to your computer by USB cable in order to activate the Serial Monitor.
- You need to select the baud rate of the Arduino Board you are using right now. For my Arduino Uno Baud Rate is 9600, as you write the following code and click the Serial Monitor, the output will show as the image below
- Open your Arduino Software and then click on the small box in the right top corner as shown in the figure:

Continue

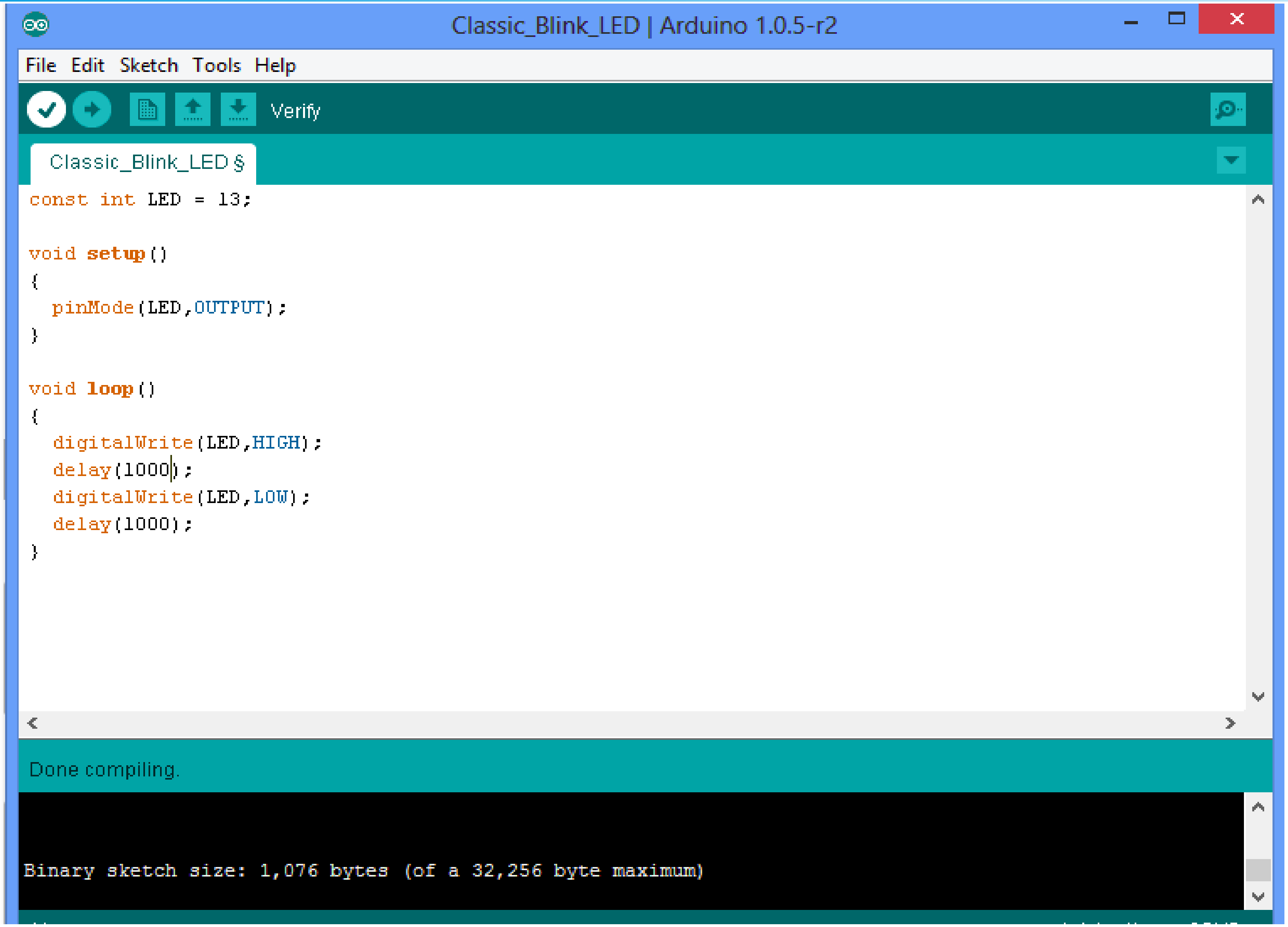

sketch_jan18a | Arduino 1.6.5
File Edit Sketch Tools Help
www.TheEngineeringProjects.com
sketch_jan18a
Click Here to Open Arduino Serial Monitor
1
Arduino/Genuino Uno on COM24

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Serial Monitor


sketch_jan18a | Arduino 1.6.5
File Edit Sketch Tools Help
COM1
Send
Autoscroll
www.TheEngineeringProjects.com
No line ending
9600 baud
1
Arduino/Genuino Uno on COM1

LED blinking code

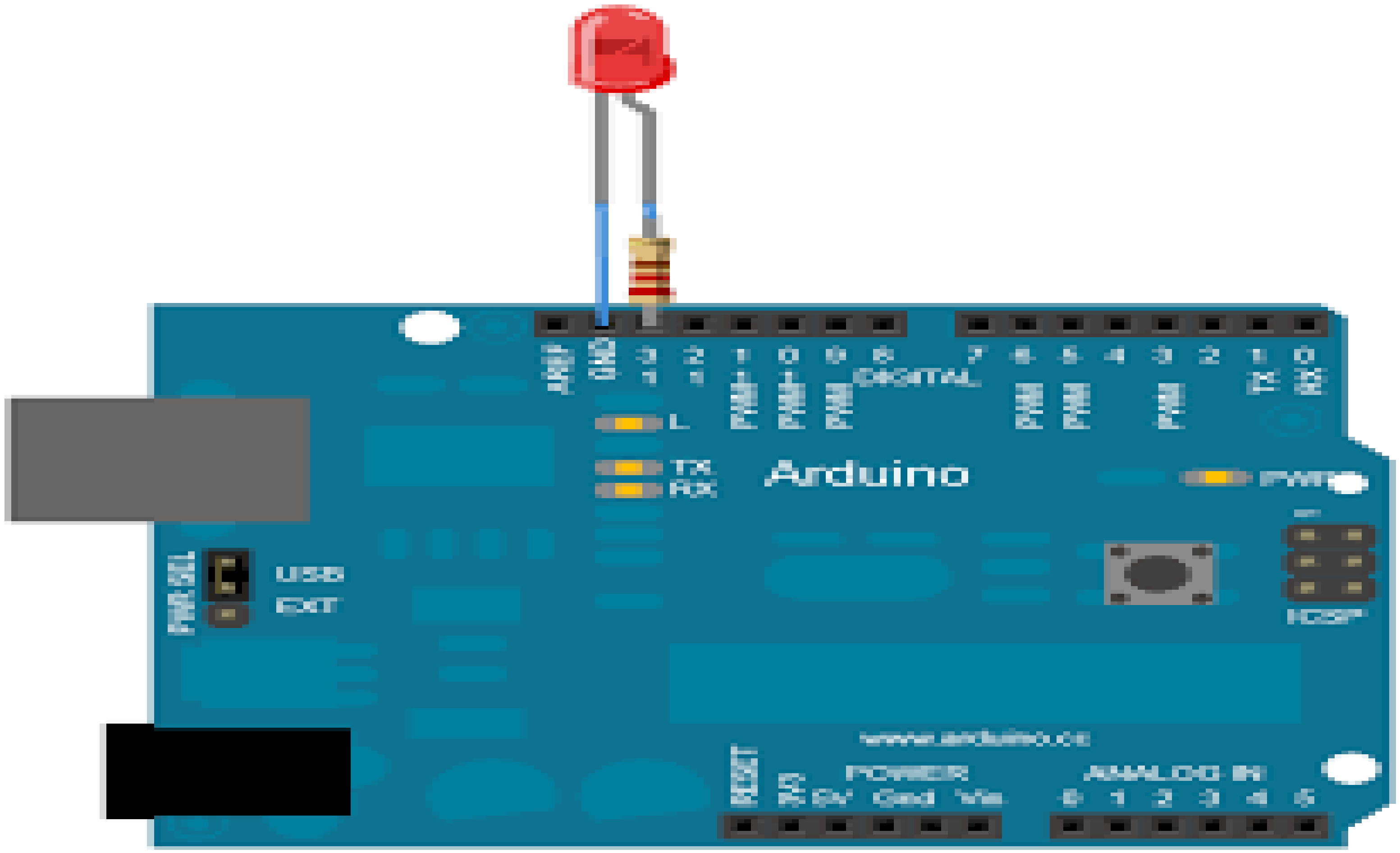

Classic_Blink_LED | Arduino 1.0.5-r2
File Edit Sketch Tools Help
Verify
Classic_Blink_LED §

```
const int LED = 13;

void setup()
{
  pinMode(LED,OUTPUT);
}

void loop()
{
  digitalWrite(LED,HIGH);
  delay(1000);
  digitalWrite(LED,LOW);
  delay(1000);
}
```

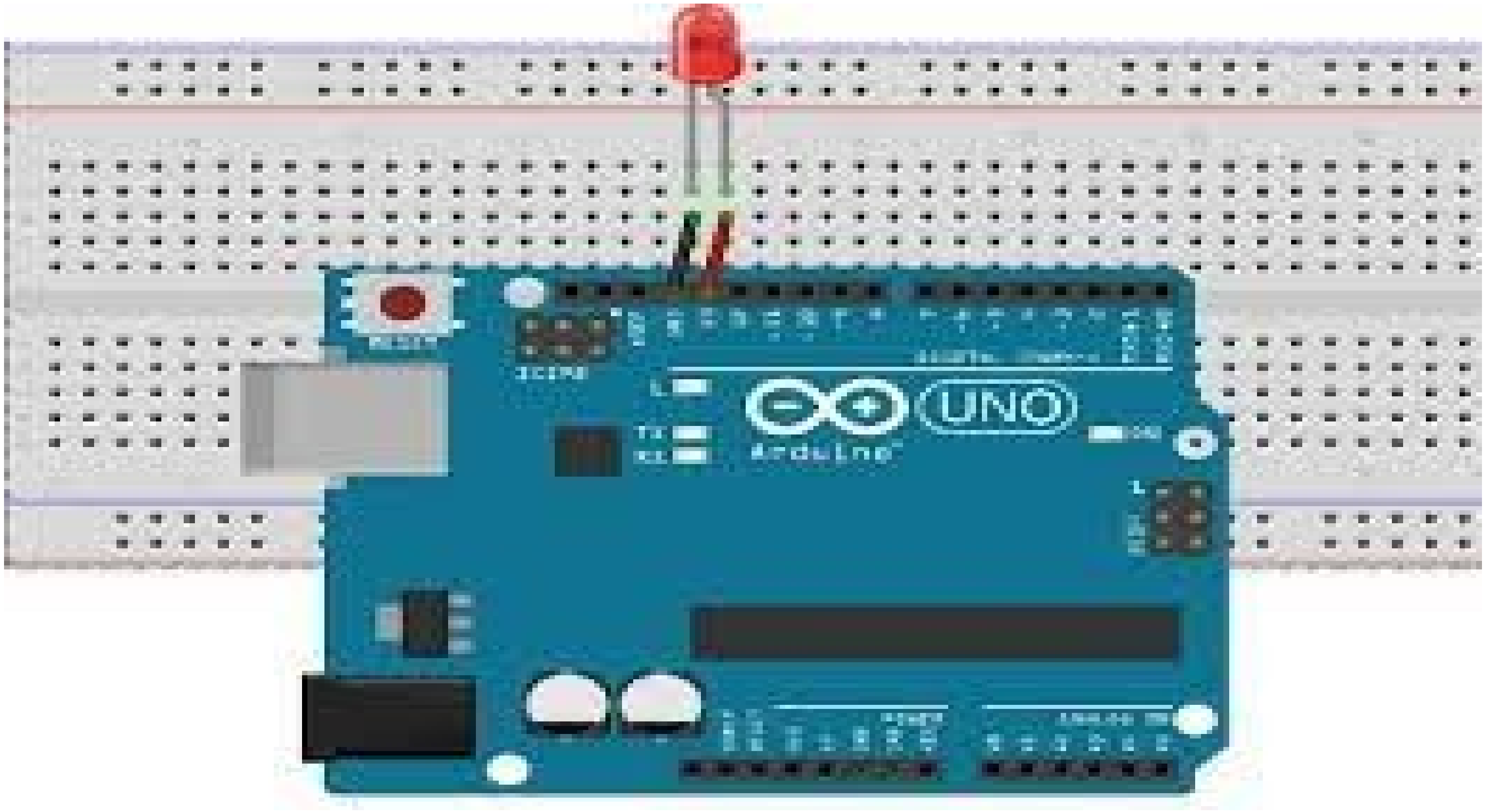
Done compiling.

```
Binary sketch size: 1,076 bytes (of a 32,256 byte maximum)
```

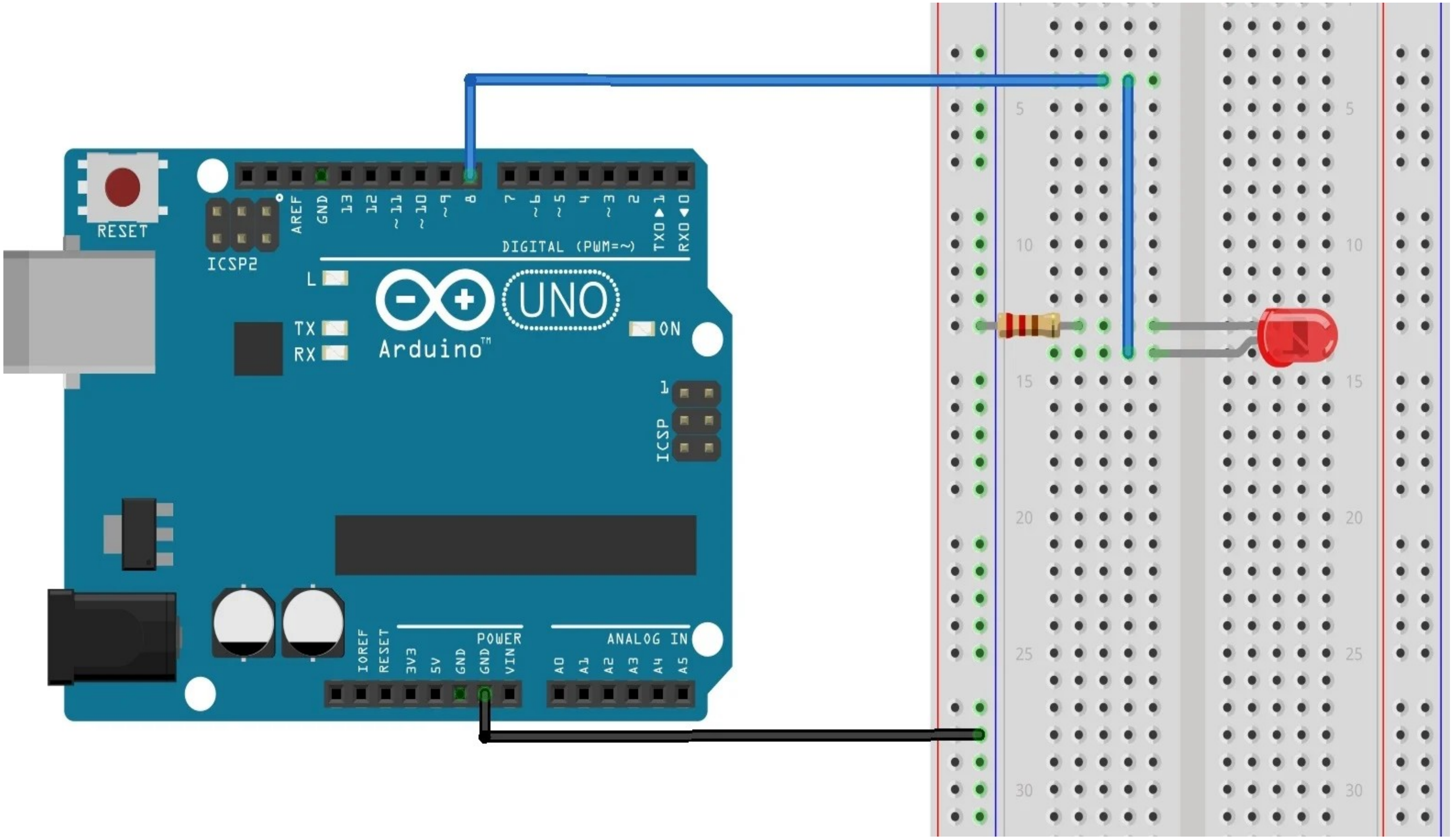
LED Blinking Circuit

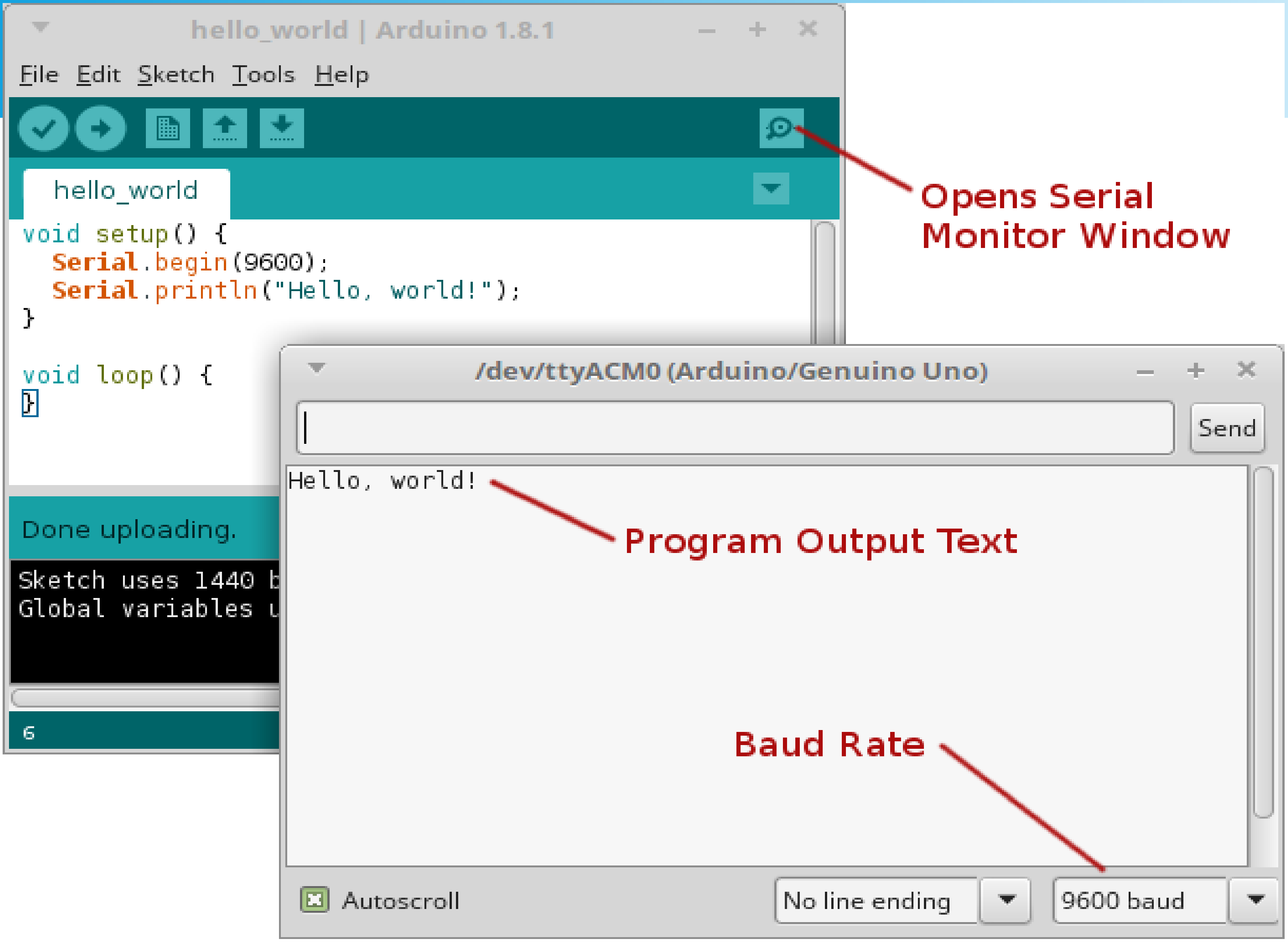


LED Blinking Circuit with Breadboard



LED Blinking Circuit with Breadboard




hello_world | Arduino 1.8.1
File Edit Sketch Tools Help
hello_world
Opens Serial Monitor Window
/dev/ttyACM0 (Arduino/Genuino Uno)
Send
Program Output Text
Baud Rate
Autoscroll
No line ending
9600 baud
Done uploading.
6

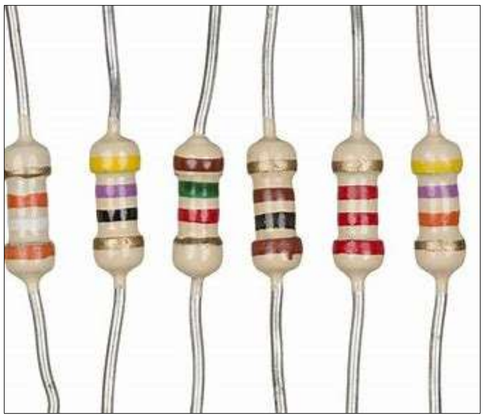
```
void setup() {
  Serial.begin(9600);
  Serial.println("Hello, world!");
}

void loop() {
}
```

```
Sketch uses 1440 b
Global variables u
```

```
Hello, world!
```

Components, Sensors and Modules

Resistor



A resistor is a fundamental electronic component that limits or regulates the flow of electrical current in a circuit. It is one of the most common and widely used components in electronics.

Capacitor

A capacitor is an electronic component that stores and releases electrical energy in a circuit. I

Jumper Wires

Jumper wires are small, flexible electrical wires with connectors at each end, commonly used for making temporary electrical connections.

Types of Connectors:

- **Male to Male (M-M)**: Both ends have male connectors (pins), typically used to connect two female headers or components.
- **Female to Female (F-F)**: Both ends have female connectors (sockets), used to connect male pins on components or headers.
- **Male to Female (M-F)**: One end has a male connector, and the other end has a female connector. This type is versatile for connecting different types of components.

Buzzer

A buzzer is an audio signalling device that emits a sound when it receives an electrical signal. Buzzers are commonly used in a variety of applications, including alarms, timers, and notifications in electronic devices. They can be found in household appliances, vehicles, and industrial equipment.

LED and OLED

LED (Light Emitting Diode)

LEDs (Light Emitting Diodes) are commonly used as indicators, status lights, or for creating visual effects. LEDs are easy to use and versatile, making them a staple component in many beginner and advanced electronics projects. H

OLED (Organic Light Emitting Diode)

An OLED (Organic Light Emitting Diode) display can be a great addition to an Arduino project, offering bright, high-contrast visuals with low power consumption. These displays are popular for showing text, graphics, and even simple animations.

To interface the OLED with the Arduino, you can use libraries like Adafruit_SSD1306 and Adafruit_GFX, which simplify working with these displays.

LCD (Liquid Crystal Display)

LCD (Liquid Crystal Display) is commonly used to provide visual output for data, status messages, or other information. The most frequently used type of LCD in Arduino projects is the character LCD, specifically the 16x2 and 20x4 models, which display 16 characters per line over 2 lines, or 20 characters per line over 4 lines.

Arduino IDE includes a library called LiquidCrystal that simplifies the interface with LCD modules. This library provides functions to initialize the display, send data, and control the cursor and display properties.

Common Commands: Using the LiquidCrystal library, commands can be sent to clear the display, move the cursor, or turn on and off the display. For example:

- lcd.begin(16, 2); initializes a 16x2 LCD.
- lcd.print("Hello, World!"); displays the text "Hello, World!" on the LCD.
- lcd.clear(); clears the display.

Switches

Switches are common input devices used in Arduino projects to control circuits and interact with programs. They can be used to turn devices on or off, trigger events, or select between different modes. In the context of Arduino, switches are typically used to provide a digital input, either HIGH (on) or LOW (off).

Types of Switches

1. **Push-Button Switches**: These are momentary switches that remain in their on or off state only as long as they are pressed. They return to their default state when released. Push-buttons are commonly used for user input, such as starting or stopping a process.

2. **Toggle Switches**: These switches maintain their position until manually changed, providing a stable on or off state. They are used when a constant change in the state is needed, such as powering a device or selecting modes.

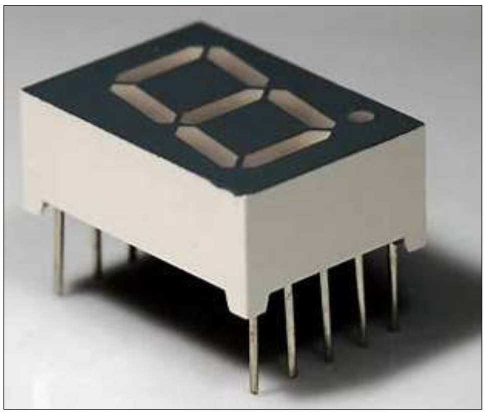
3. **Slide Switches**: Similar to toggle switches, slide switches change state when slid from one position to another and maintain that state until changed again.

4. **Rotary Switches**: These switches allow for multiple positions and are often used for selecting between multiple modes or settings.

Potentiometer

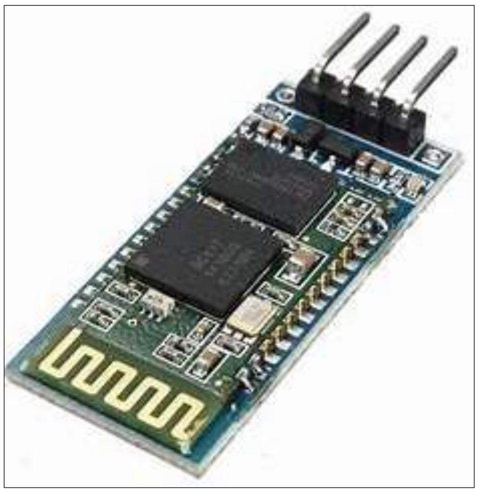
A potentiometer is a variable resistor that can be used to adjust voltage levels or control various parameters in a circuit. It provides an easy way to input analog values to an Arduino, which can be used for adjusting settings, controlling motor speeds, or varying light intensity.

7 Segment Display



Seven-segment displays are popular components for displaying numerical information and some simple characters. They consist of seven LEDs arranged in a figure-eight pattern, plus an optional eighth LED for displaying a decimal point. Each segment can be turned on or off to form numbers from 0 to 9 and some letters.

Bluetooth modules



Bluetooth modules are commonly used with Arduino to enable wireless communication between devices. They are particularly useful for creating wireless interfaces, remote controls, and data transmission applications. The most popular Bluetooth module for Arduino projects is the HC-05, but there are others like the HC-06, and the more advanced HM-10 (Bluetooth 4.0) for BLE (Bluetooth Low Energy) applications.

Applications of Bluetooth Modules in Arduino Projects

- **Wireless Communication**: Send and receive data wirelessly between an Arduino and a smartphone, computer, or another Bluetooth-enabled device.

- **Remote Control**: Control Arduino projects remotely via a Bluetooth-enabled smartphone or tablet.
- **Data Logging**: Send sensor data to a smartphone or PC for real-time monitoring and analysis.
- **User Interfaces**: Create user interfaces for your projects using smartphone apps or custom software.

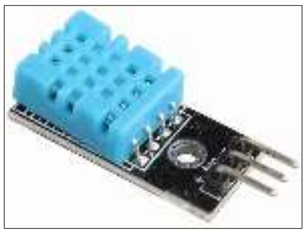
GSM Module

GSM (Global System for Mobile Communications) modules allow Arduino boards to communicate over cellular networks, enabling your projects to send and receive SMS messages, make voice calls, or connect to the internet via mobile data. The most commonly used GSM module with Arduino is the SIM900, but there are other models like the SIM800 and SIM7600 that provide similar functionality with various features.

Applications of GSM Modules in Arduino Projects

- **Remote Monitoring**: Send sensor data or status updates via SMS or data connection.
- **Alerts and Notifications**: Send alerts or notifications about system events or errors.
- **Remote Control**: Control devices or systems remotely via SMS commands.
- **Data Logging**: Send logged data to a server or cloud service through mobile data.

DHT-11



What is a DHT11 Sensor?

- The DHT11 is a low-cost digital sensor used to measure temperature and humidity. It is commonly used in various DIY projects, home automation systems, and environmental monitoring applications.

Features of the DHT11 Sensor:

- **Temperature Range:** 0-50°C with ±2°C accuracy.
- **Humidity Range:** 20-90% RH with ±5% accuracy.
- **Operating Voltage:** 3.3V to 5.5V.
- **Sampling Rate:** Once every second (1 Hz).
- **Digital Output:** Provides calibrated digital output signals.
- **Pre-calibrated:** The sensor is pre-calibrated and provides digital outputs, eliminating the need for external components.

Applications of DHT11:

- **Weather Stations:** Monitoring temperature and humidity.
- **HVAC Systems:** Controlling heating, ventilation, and air conditioning.
- **Greenhouses:** Maintaining optimal growing conditions.
- **Home Automation:** Integrating with smart home systems to control the environment.
- **Data Loggers:** Collecting environmental data over time.

Smoke detection sensor

What is a Smoke Sensor?

- A smoke sensor, often referred to as a gas sensor, is a device that detects the presence of smoke, gases, or air quality changes. It converts the concentration of gases or smoke into an electrical signal.

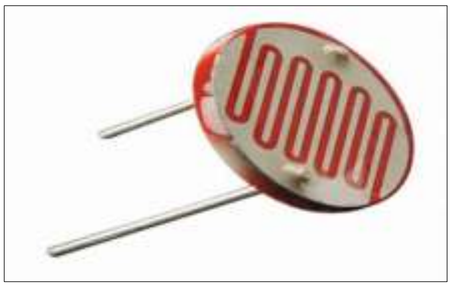
Types of Smoke Sensors:

- **MQ Series Sensors:** The most commonly used smoke sensors in DIY projects and educational purposes are the MQ series sensors, such as MQ-2, MQ-3, MQ-7, etc.
- **MQ-2:** Detects LPG, i-butane, propane, methane, alcohol, hydrogen, and smoke.
- **MQ-3:** Primarily used for alcohol detection.
- **MQ-7:** Used for carbon monoxide detection.

Applications of Smoke Sensors:

- **Fire Alarms:** Detecting smoke from fires and triggering alarms.
- **Industrial Safety:** Monitoring gas leaks in factories.
- **Air Quality Monitoring:** Measuring pollution levels in the environment.
- **Breathalyzers:** Detecting alcohol levels in breath.

Light Sensor (LDR)



An LDR (Light Dependent Resistor), also known as a photoresistor, is a component that changes its resistance based on the amount of light falling on it. The resistance decreases with increasing light intensity.

How Does an LDR Work?

- **High Resistance in Darkness:** When the LDR is in darkness or low light conditions, it has a high resistance (up to several megaohms).
- **Low Resistance in Light:** When the LDR is exposed to light, its resistance drops significantly (to a few hundred ohms).

Applications of LDRs:

- **Light Meter:** Measuring the intensity of light in a given environment.
- **Automatic Lighting:** Turning lights on or off based on ambient light levels (e.g., streetlights).
- **Security Systems:** Detecting changes in light levels for triggering alarms.
- **Photography:** Light-sensitive triggers for cameras.

Ultrasonic Sensor

Definition: An ultrasonic sensor is a device that measures the distance to an object by using ultrasonic sound waves.

Components: Typically consists of a transmitter (which emits the sound waves) and a receiver (which detects the reflected waves).

How Does an Ultrasonic Sensor Work?

- **Sound Wave Emission:** The transmitter sends out a burst of ultrasonic sound waves (typically around 40 kHz).
- **Echo Reception:** These waves travel through the air until they hit an object and reflect back to the sensor.
- **Time Measurement:** The sensor measures the time it takes for the waves to return.
- **Distance Calculation:** The distance to the object is calculated using the formula:

$$\text{Distance} = \frac{\text{Speed of Sound} \times \text{Time}}{2}$$

Characteristics of Ultrasonic Sensors

- **Non-contact Measurement:** Can measure distance to objects without physical contact.
- **Versatility:** Can be used in various environments, including water and air.
- **Range:** Typically effective for distances ranging from a few centimeters to several meters.
- **Accuracy:** Generally provides accurate distance measurements, though accuracy can be affected by factors like temperature and humidity.

Applications of Ultrasonic Sensors

- **Obstacle Detection:** Used in robotics for detecting obstacles and avoiding collisions.
- **Level Measurement:** Used to measure the level of liquids in tanks.
- **Distance Measurement:** Used in parking sensors to help drivers park safely.
- **Object Detection:** Used in various industrial automation processes for detecting objects on conveyor belts.

PIR Sensor

- **Definition:** PIR stands for Passive Infrared. A PIR sensor is an electronic sensor that measures infrared (IR) light radiating from objects in its field of view.
- **Components:** Typically consists of a pyroelectric sensor, a lens (such as a Fresnel lens), and an electronic circuit to process the signals.

How Does a PIR Sensor Work?

- **Infrared Detection:** All objects emit some level of infrared radiation. The PIR sensor detects changes in infrared radiation levels.
- **Motion Sensing:** When an object, such as a human or animal, moves in front of the sensor, the amount of infrared energy detected changes.
- **Signal Processing:** The sensor generates an electrical signal in response to changes in IR levels. This signal is then processed by the electronic circuit to trigger an output (e.g., turning on a light).

Applications of PIR Sensors

- **Security Systems:** Used in burglar alarms and motion-activated security cameras.
- **Lighting Control:** Used in motion-activated lighting systems to turn lights on or off based on occupancy.
- **Home Automation:** Integrated into smart home systems for automated control of lights, heating, and other devices.
- **Energy Conservation:** Helps conserve energy by ensuring lights and appliances are only active when needed.

Sound sensor

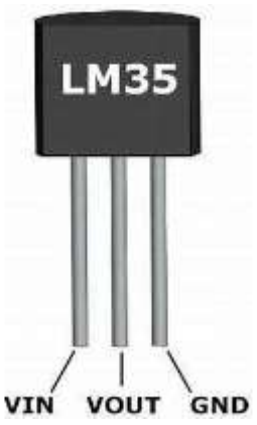
A sound sensor is a device used to detect sound waves and convert them into electrical signals that can be processed by a microcontroller like an Arduino

Applications

- **Sound Level Monitoring:** Measure ambient noise levels.
- **Voice Detection:** Detect presence of sound for voice-controlled applications.
- **Sound-Activated Devices:** Turn on lights, motors, etc., in response to sound.

Temperature sensor

LM35: Analog temperature sensor that outputs a voltage proportional to the temperature.



DHT11/DHT22: Digital temperature and humidity sensors, with simple communication protocol.

LM35

Description

- **Type:** Analog temperature sensor.
- **Output:** Analog voltage proportional to temperature.
- **Requires ADC:** Needs an analog-to-digital converter (ADC) to read values, which is available on most Arduino boards.
- **No Humidity Measurement:** Measures only temperature.

DHT11

Description

- **Type:** Digital temperature and humidity sensor.
- **Output:** Digital signal (temperature and humidity data).
- **Digital Output:** Easier to interface with microcontrollers that lack ADC.
- **Temperature and Humidity:** Measures both temperature and humidity.

Which one should be used?

Use **LM35** if you need higher accuracy and a wider temperature range.

Use **DHT11** if you need to measure both temperature and humidity and can tolerate lower accuracy and a narrower temperature range.

DC Motor

Description

- **Type:** Direct Current Motor.
- **Operation:** Rotates continuously in one direction or the other when powered.
- **Control:** Speed and direction can be controlled using a motor driver or transistor.

Applications

- **Robotics:** Drive wheels or other moving parts.
- **Fans:** Control fan speed.
- **Actuators:** Open or close mechanisms.

Servo Motor

Description

- **Type:** Rotary actuator.
- **Operation:** Provides precise control of angular position.
- **Control:** Controlled by PWM signal from Arduino.

Applications

- **Robotics:** Position control for arms or grippers.
- **Pan-and-Tilt Systems:** Control camera or sensor orientation.
- **Model Railroads:** Operate switches and other features.

Stepper Motor

Description

- **Type:** Motor that moves in discrete steps.
- **Operation:** Provides precise control of rotation angle and position.
- **Control:** Requires a stepper motor driver to control step and direction signals.

Applications

3D Printers: Control the movement of print heads and beds.
CNC Machines: Drive tools and components with precise movements.
Robotics: Position control for legs or arms with high accuracy.

Point to note

- **DC Motors:** Continuous rotation, control speed and direction, simple interface.
- **Servo Motors:** Precise angle control, easy to interface, limited to 180° rotation.
- **Stepper Motors:** Precise step-by-step control, used for applications requiring accurate positioning.

Arduino Flavors

Arduino Uno

Overview

- Most popular and widely used Arduino board.

Specifications

- Microcontroller: ATmega328P
- Operating Voltage: 5V
- Digital I/O Pins: 14 (6 PWM output)
- Analog Input Pins: 6
- Flash Memory: 32 KB

Key Features

- Easy to use for beginners.
- Compatible with a wide range of shields and modules.

Common Uses and Projects

- Simple robotics
- Home automation
- Prototyping

Arduino LilyPad

Overview

- Designed for wearable electronics projects.

Specifications

- Microcontroller: ATmega328V
- Operating Voltage: 2.7V to 5.5V
- Digital I/O Pins: 14
- Analog Input Pins: 6
- Flash Memory: 16 KB

Key Features

- Flexible and sewable.
- Low power consumption.

Common Uses and Projects

- Wearable technology
- Interactive clothing
- E-textiles

Arduino Bluetooth (Arduino BT)

Overview

- Arduino board with integrated Bluetooth module.

Specifications

- Microcontroller: ATmega328
- Operating Voltage: 5V
- Digital I/O Pins: 14 (6 PWM output)
- Analog Input Pins: 6
- Flash Memory: 32 KB
- Integrated Bluetooth Module

Key Features

- Wireless communication.
- Easy integration with other Bluetooth devices.

Common Uses and Projects

- Remote control systems
- Wireless data logging
- IoT projects

Arduino Mega

Overview

- Designed for projects requiring more I/O pins.

Specifications

- Microcontroller: ATmega2560
- Operating Voltage: 5V
- Digital I/O Pins: 54 (15 PWM output)
- Analog Input Pins: 16
- Flash Memory: 256 KB

Key Features

- Large number of I/O pins.
- Suitable for complex projects.

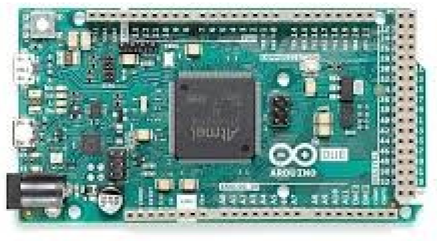
Comparison with Arduino Uno

- More I/O pins and memory.
- Suitable for more complex projects.

Common Uses and Projects

- Large-scale robotics
- 3D printers
- Data acquisition systems

Arduino Due



Overview

- First Arduino board based on a 32-bit ARM core microcontroller.

Specifications

- Microcontroller: AT91SAM3X8E
- Operating Voltage: 3.3V
- Digital I/O Pins: 54 (12 PWM output)
- Analog Input Pins: 12
- Flash Memory: 512 KB

Key Features

- High processing power.
- More memory and faster performance.

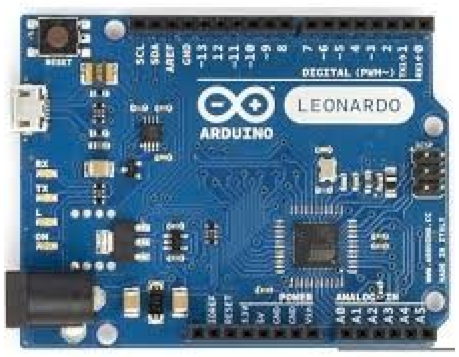
Differences from Other Arduino Boards

- 32-bit ARM core.
- Higher clock speed and memory.

Common Uses and Projects

- Advanced robotics
- Real-time data processing
- High-performance applications

Arduino Leonardo



Overview

- Arduino board with built-in USB communication.

Specifications

- Microcontroller: ATmega32u4
- Operating Voltage: 5V
- Digital I/O Pins: 20 (7 PWM output)
- Analog Input Pins: 12
- Flash Memory: 32 KB

Key Features

- Built-in USB communication.
- Can act as a mouse or keyboard.

Common Uses and Projects

- Human Interface Devices (HID)
- Interactive projects
- USB device emulation

Side-by-Side Comparison

Feature	Uno	LilyPad	Bluetooth	Mega	Due	Leonardo
Microcontroller	ATmega328P	ATmega328V	ATmega328	ATmega2560	AT91SAM3X8E	ATmega32u4
Operating Voltage	5V	2.7V to 5.5V	5V	5V	3.3V	5V
Digital I/O Pins	14 (6 PWM)	14	14 (6 PWM)	54 (15 PWM)	54 (12 PWM)	20 (7 PWM)
Analog Input Pins	6	6	6	16	12	12
Flash Memory	32 KB	16 KB	32 KB	256 KB	512 KB	32 KB
Key Features	Popular, easy to use	Flexible, sewable	Integrated Bluetooth	Large number of I/O pins	32-bit ARM core	Built-in USB
Common Uses	Prototyping, robotics	Wearable tech	Wireless comm.	Large-scale robotics	Advanced robotics	HID, interactive

Choosing the Right Arduino

Content:

- Factors to consider: Project requirements, I/O pins needed, processing power, memory, and special features.
- Examples of projects and the most suitable Arduino board for each.