Arduino Uno Overview

Introduction

The Arduino Uno is a microcontroller board based on the ATmega328P. It is widely used in electronics projects due to its simplicity, ease of use, and large community support. The Uno is ideal for beginners and experienced users alike, offering an open-source platform for prototyping.

Arduino Pin Configuration:

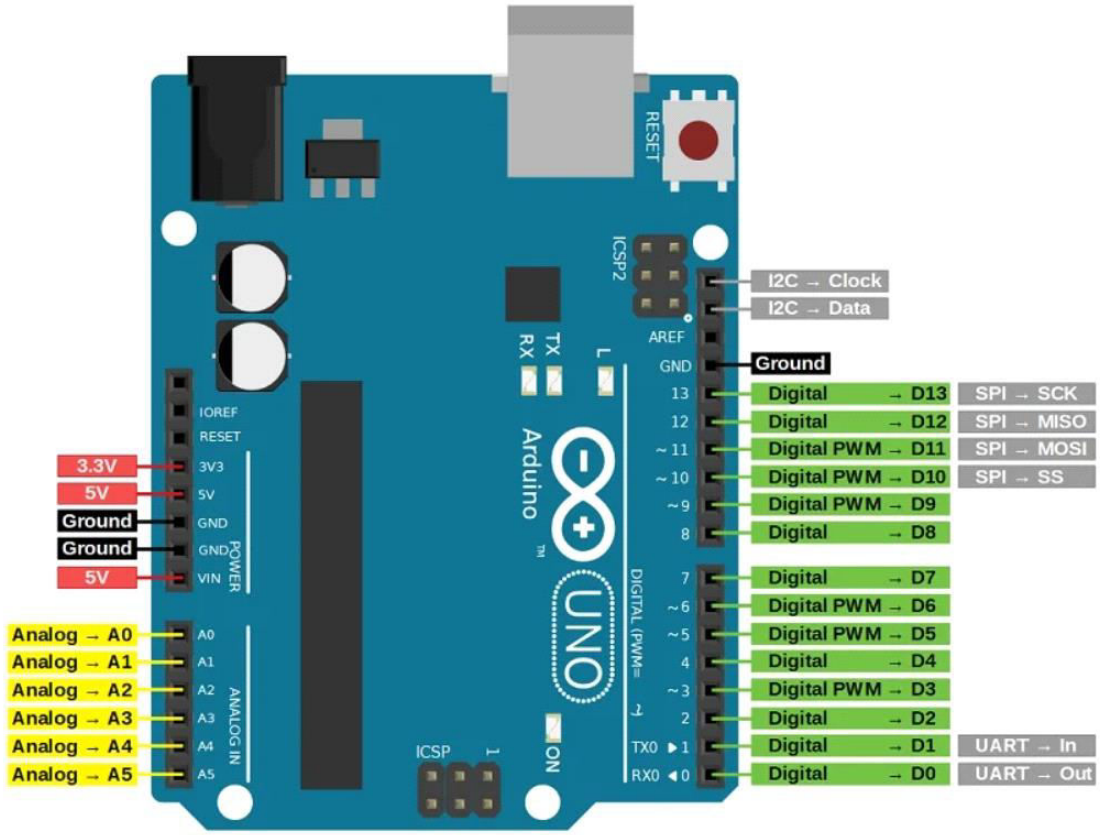

Image Source: diyi0t.com

USB Connection:

Used for uploading code and power supply.

Pinout

Power Pins: VIN, 3.3V, 5V, GND

Digital Pins (0-13): Can be used as general-purpose input/output (GPIO) pins.

PWM Pins: 3, 5, 6, 9, 10, 11 for analog output.

Analog Pins (A0-A5): For reading analog sensors.

TX/RX Pins (0, 1): For serial communication.

Powering Arduino with VIN Pin

For Powering the Arduino with External Voltage.

The VIN pin allows to power the Arduino using an external voltage source (such as a battery or an external power adapter) without using the USB port. This is useful for

standalone applications where you may not have access to a USB connection.

Voltage between 7V to 12V to the VIN pin.The onboard voltage regulator will step this voltage down to the 5V required to power the Arduino's components.

Manual System Reset Using the Reset Pin

It is typically used when we want to reinitialize the system without disconnecting power or pressing the reset button manually.

The reset pin is active low, meaning that it triggers a reset when it is connected to GND (low voltage). When the pin is at a high voltage (5V or floating), the Arduino operates normally.

CODE:

Overview: Here we will see output "Arduino has started or reset" whenever button is pressed.

```
void setup()
{
    Serial.begin(9600);
```

```
    Serial.println("Arduino has started or reset.");
}
void loop()
  {
    // Main program code
}
```

Enhancing Sensor Accuracy with AREF

Some sensors don't use the full 0-5V range. For example, a sensor might give you voltages between 0V and 2V. If you stick with the default 5V reference, you won't get very accurate readings because you're only using part of the full 0-1023 range.

e.g,

LM35 Temperature Sensor

Photocell (LDR - Light Dependent Resistor)

Sharp GP2Y0A21YK0F Infrared Distance Sensor

Measures distance and outputs a variable voltage depending on the proximity.

The output voltage is typically between 0.4V and 2V, making

it suitable for this scenario.

But,

By connecting a lower voltage (like 2V) to the AREF pin, we're telling the Arduino, "Measure everything between 0 and 2V, not 0 to 5V."

So:

0V on the analog pin = 0 in the code.

2V on the analog pin = 1023 in the code.

NOTE:

Without using AREF, the Arduino will treat 2V as if it’s 2/5ths of the total range, and we'll get values up to around 400 out of 1023.

With AREF set to 2V, the Arduino will treat 2V as the maximum, and we'll get values up to 1023 for more accuracy.

Now, our readings will be more precise when our sensor gives values between 0 and 2V.

How to Use AREF

Connect a voltage source (for example, 2V) to the AREF pin.

Library:

```
analogReference(EXTERNAL);
```