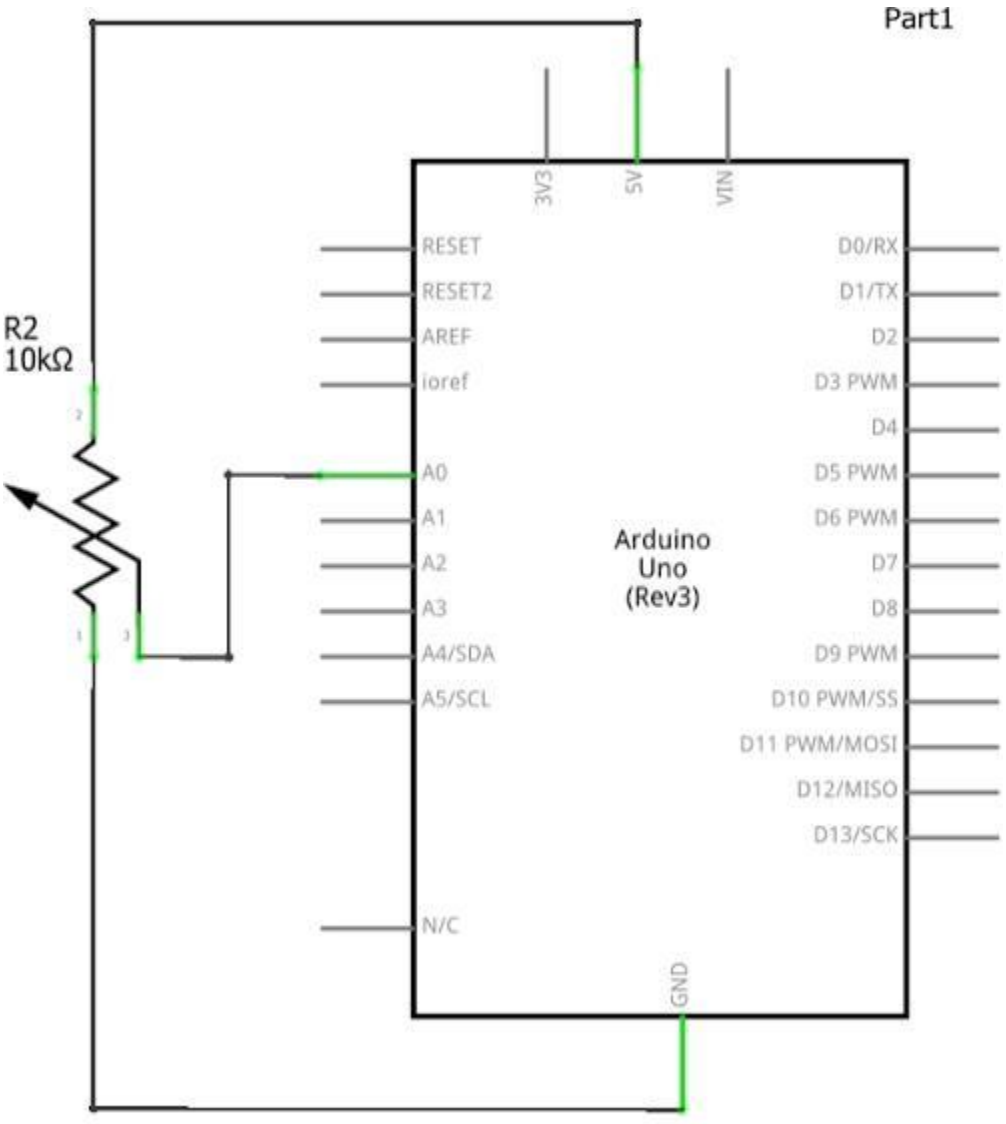
Arduino boards handle analog signals through two main methods: integrated **Analog-to-Digital Converters (ADC)** and, depending on the model, either **PWM-simulated** or **true Digital-to-Analog Converters (DAC)**.

1. ADC (Analog-to-Digital Converter)

The ADC translates a voltage from the physical world into a number the microcontroller can process.



- **Operation:** Most Arduinos use a **Successive Approximation Register (SAR)** ADC, which samples the voltage and compares it against a reference to find its digital value.
- **Resolution & Accuracy:**
    - **Uno R3**

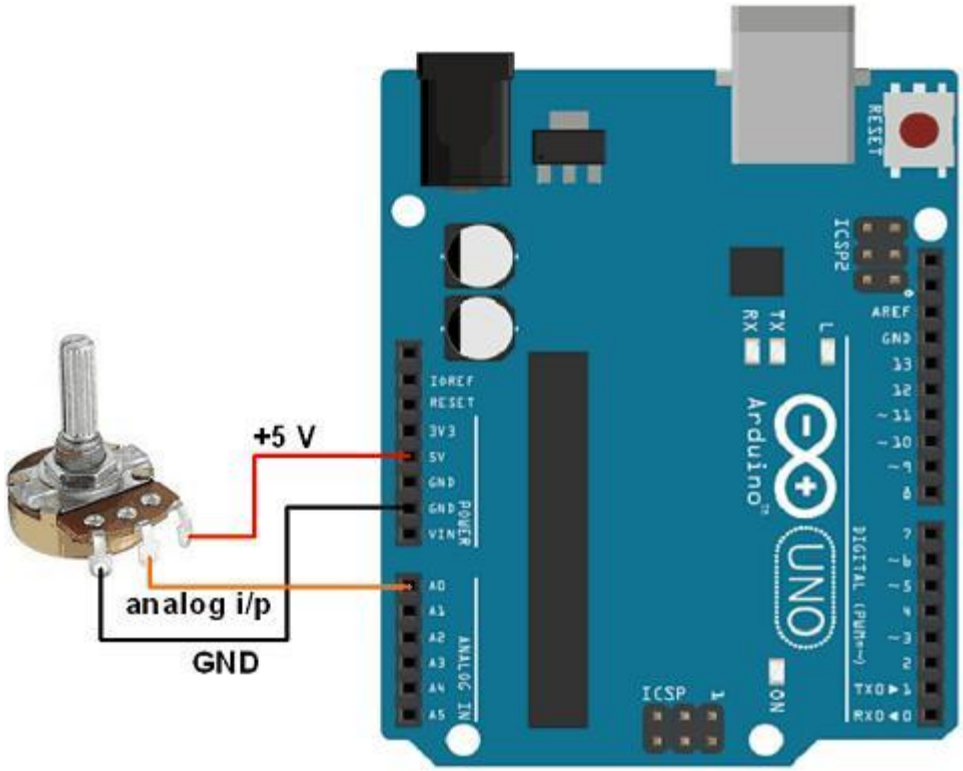
      : 10-bit resolution, providing 1,024 discrete steps (0–1023). At a 5V reference, this equals ~4.9mV per step.
    - **Uno R4/ ESP32**

      :Support up to **14-bit** (0–16,383) or **12-bit** (0–4,095) resolution for higher precision.
- **Sampling Rate:** The standard

  Uno R3 samples at approximately **9,600 Hz** by default, but can be pushed to ~15,400 Hz by adjusting the ADC clock.

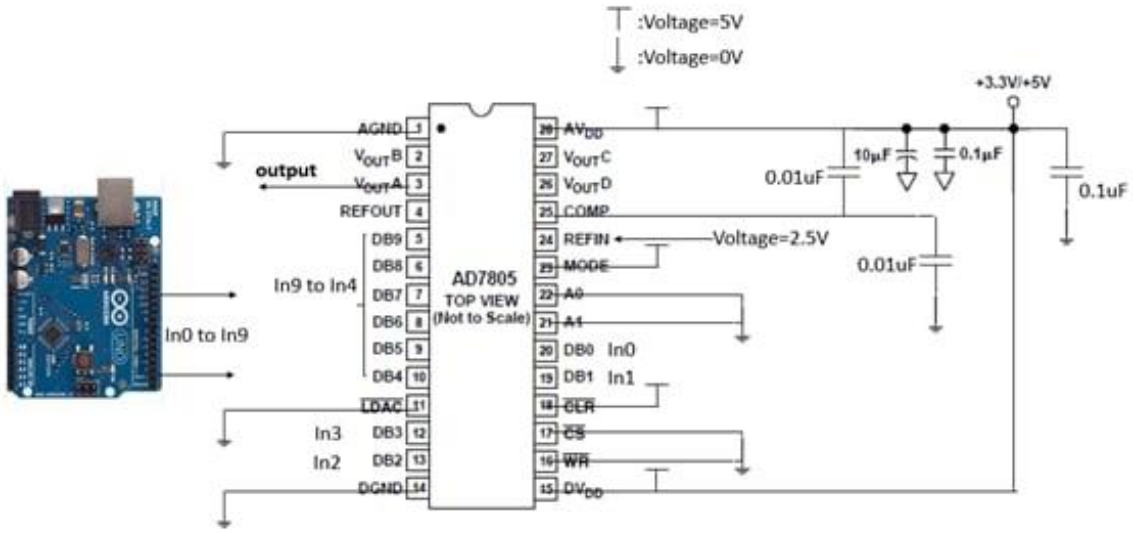
  Keycode

  Int value = analogRead(A0); // Reads voltage from pin A0



2. DAC (Digital-to-Analog Converter)

A DAC performs the reverse of an ADC, outputting a specific voltage based on a digital number.



True DAC vs. PWM:

True DAC: Available on newer boards like the Uno R4 Minima/WiFi (12-bit), GIGA R1 (dual 12-bit), and Due. These output a real, steady analog voltage.

PWM (Simulated): Boards like the Uno R3 lack a true DAC and use Pulse Width Modulation on pins marked with ~. This flickers a signal on and off rapidly to simulate an average voltage, which is fine for LEDs but requires filtering for audio.

Key Features of True DACs:

Waveform Generation: Can generate clean sine, square, or sawtooth waves for audio or laboratory equipment.

DMA Capability: High-end boards like the GIGA R1 can use Direct Memory Access (DMA) to play audio files without slowing down the main CPU.

Key Code (True DAC):

analogWrite(DAC0, 2048); // Outputs mid-range voltage on a 12-bit DAC