UNCUYO
UNIVERSIDAD NACIONAL DE CUYO

FACULTAD DE INGENIERÍA

Técnicas y Herramientas Modernas II

Informe Módulo II

“Internet de las cosas”

2023

GRUPO N°1
ALVAREZ , Martina.
BÁSCOLO , Felicitas.
DE MEZZO, Facundo.
ROYO, Lourdes.
VIDELA, Ana.

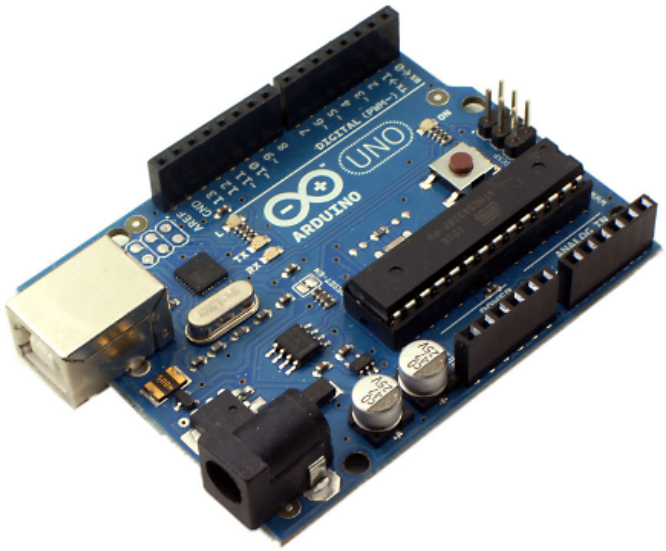
UNCUYO
UNIVERSIDAD
NACIONAL DE CUYO

FACULTAD
DE INGENIERÍA

Módulo 2 - Programación de Hardware IIOT

Microcontroladores - uso básico de Arduino

Arduino es utilizado como un microcontrolador, cuando tiene un programa descargado desde un ordenador y funciona de forma independiente de éste. Controla y alimenta determinados dispositivos y toma decisiones de acuerdo al programa descargado. Interactúa con el mundo físico gracias a sensores y actuadores.



Durante este módulo controlamos diferentes elementos a través de un Arduino UNO que fue programado desde la aplicación Arduino IDE.
Inicialmente, se estudió el comportamiento de un potenciómetro, y se revisó su funcionamiento mediante un multímetro. Luego se conectó el potenciómetro a un pin analógico y una luz LED a un pin digital del Arduino y se programó con un código ejemplo de la aplicación llamado AnalogInOutSerial.

```
Analog input, analog output, serial output

  Reads an analog input pin, maps the result to a range from 0 to 255 and uses
  the result to set the pulse width modulation (PWM) of an output pin.
  Also prints the results to the Serial Monitor.

  The circuit:
  - potentiometer connected to analog pin 0.
    Center pin of the potentiometer goes to the analog pin.
    side pins of the potentiometer go to +5V and ground
  - LED connected from digital pin 9 to ground through 220 ohm resistor
```

```
  created 29 Dec. 2008
  modified 9 Apr 2012
  by Tom Igoe

  This example code is in the public domain.

  https://www.arduino.cc/en/Tutorial/BuiltInExamples/AnalogInOutSerial
*/

// These constants won't change. They're used to give names to the pins used:
const int analogInPin = A0;   // Analog input pin that the potentiometer is
attached to
const int analogOutPin = 9;  // Analog output pin that the LED is attached to

int sensorValue = 0;  // value read from the pot
int outputValue = 0;  // value output to the PWM (analog out)

void setup() {
  // initialize serial communications at 9600 bps:
  Serial.begin(9600);
}

void loop() {
  // read the analog in value:
  sensorValue = analogRead(analogInPin);
  // map it to the range of the analog out:
  outputValue = map(sensorValue, 0, 1023, 0, 255);
  // change the analog out value:
  analogWrite(analogOutPin, outputValue);

  // print the results to the Serial Monitor:
  Serial.print("sensor = ");
  Serial.print(sensorValue);
  Serial.print("\t output = ");
  Serial.println(outputValue);

  // wait 2 milliseconds before the next loop for the analog-to-digital
  // converter to settle after the last reading:
  delay(2);
```

}

Con este código se logró regular la intensidad de luz de LED según la posición del potenciómetro.

Luego, se utilizó un servomotor, también conocido como servo, que es un dispositivo que permite controlar con máxima precisión la posición y movimientos de su eje. Esto significa que este se puede mover en un ángulo, posición y a una velocidad determinada en cada momento, algo que no puede hacer un motor eléctrico normal.

Los pasos para conectar el Servo fueron los siguientes:
1. Conectar la Alimentación y Tierra:
- Conectar el cable de alimentación del servo (generalmente rojo o naranja) a una fuente de alimentación adecuada.
- Conectar el cable de tierra del servo (generalmente marrón o negro) a la tierra (GND) del Arduino.
2. Conectar el Control:
- Conectar el cable de control del servo (generalmente amarillo o blanco) a un pin de salida PWM del Arduino. Se utilizó el pin PWM 9.
3. Programar el Arduino: el código utilizado fue:

```
#include <Servo.h>

Servo miServo;  // Crea un objeto servo para controlar un servo motor

int anguloInicial = 90;  // Define el ángulo inicial del servo

void setup() {
  miServo.attach(9);  // Asigna el pin 9 para controlar el servo
  miServo.write(anguloInicial);  // Mueve el servo a la posición inicial
}

void loop() {
  // Mueve el servo gradualmente
  for (int angulo = anguloInicial; angulo <= 180; angulo += 1) {
```

```
    miServo.write(angulo);
    delay(15);
  }

  delay(1000);  // Espera 1 segundo

  // Vuelve a la posición inicial gradualmente
  for (int angulo = 180; angulo >= anguloInicial; angulo -= 1) {
    miServo.write(angulo);
    delay(15);
  }

  delay(1000);  // Espera 1 segundo
}
```

4. Energizar el Circuito: Conectar el Arduino a la computadora a través del cable USB y cargar el código en la placa.
5. Verificar el Funcionamiento: El servo debe comenzar a moverse según lo indicado en tu código.