"Introducción a la Robótica para la Enseñanza Técnico Profesional"

Módulo 2 –Fundamentos técnicos y componentes

Lección 6 – Introducción al controlador Arduino Uno en robótica educativa

¿Qué es un controlador y por qué Arduino Uno?
En todo sistema robótico, existe un componente central que se encarga de coordinar las acciones del conjunto: el controlador. Este dispositivo es el encargado de hacer que el robot piense y actúe de forma autónoma o semiautónoma, transformando información del entorno en decisiones lógicas y respuestas físicas.

🔁 **¿Qué hace un controlador?**
Un controlador es, en esencia, un dispositivo electrónico programable que cumple tres funciones fundamentales:

Recibir información (entrada): Lee datos desde sensores que detectan condiciones del entorno, como luz, temperatura, humedad, distancia o presencia.

Procesar esa información (decisión): Aplica instrucciones programadas (el código) que le dicen qué hacer según las condiciones detectadas.

Ejecutar acciones (salida): Controla actuadores, como motores, luces LED, zumbadores, relés o pantallas, para producir una respuesta física.

🗣️ ***Es decir, un controlador permite que el robot o el sistema "sienta, piense y actúe", como un pequeño sistema inteligente.***

¿Qué es Arduino Uno?
Arduino Uno es una placa de desarrollo electrónica basada en el microcontrolador ATmega328P, que se ha convertido en el estándar de enseñanza en electrónica y robótica educativa.
Diseñada originalmente para facilitar la creación de prototipos electrónicos, Arduino combina hardware accesible con un entorno de programación simple, lo que permite programar y controlar dispositivos reales sin necesidad de conocimientos avanzados de ingeniería.

🗣️ ***En robótica educativa, Arduino Uno es la herramienta ideal para introducir la lógica de control, la electrónica y la programación de manera accesible y concreta.***

¿Por qué usar Arduino Uno en educación técnica?
Arduino Uno se destaca por:

✅ **Accesibilidad:** Es de bajo costo, fácil de conseguir, y de código abierto.
✅ **Simplicidad:** Permite comenzar con ejemplos simples (como encender un LED) y avanzar hacia proyectos complejos.
✅ **Versatilidad:** Se puede usar en robótica, domótica, automatización, monitoreo ambiental, salud, y más.
✅ **Compatibilidad:** Funciona con cientos de sensores y módulos disponibles en el mercado.
✅ **Soporte comunitario:** Existen miles de tutoriales, librerías, y foros en español e inglés.
✅ **Simulación virtual:** Se puede usar con plataformas como Tinkercad Circuits para aprender sin hardware.

📌 ***En resumen: Arduino Uno reduce la barrera de entrada a la tecnología, permitiendo a docentes y estudiantes diseñar, programar y controlar sistemas reales de forma concreta, creativa y significativa.***

🔧 2. Características técnicas clave del Arduino Uno

Para comprender cómo funciona Arduino Uno como controlador en un sistema robótico, es fundamental conocer sus componentes técnicos principales. Cada uno de ellos determina qué tipo de proyectos se pueden desarrollar y cómo interactuar con sensores, actuadores y otros dispositivos.

Elemento	Detalle ampliado
Microcontrolador	El ATmega328P es el chip central de la placa. Es un circuito integrado que ejecuta las instrucciones del programa cargado. Tiene 32 KB de memoria flash, 2 KB de RAM y 1 KB de EEPROM.
Entradas/salidas digitales	Posee 14 pines digitales numerados del 0 al 13, que pueden configurarse como entrada (leer señales) o salida (activar componentes). 6 de estos pines (3, 5, 6, 9, 10, 11) pueden generar PWM, lo que permite controlar la velocidad de motores o el brillo de LEDs.
Entradas analógicas	Tiene 6 pines analógicos (A0 a A5) que permiten leer valores variables, como la intensidad de luz, temperatura o humedad, en lugar de solo 0 o 1. Estos valores se traducen en un rango de 0 a 1023.
Voltaje de operación	Arduino trabaja con 5 V, lo que lo hace compatible con la mayoría de sensores y módulos educativos. Los pines de salida también entregan 5 V cuando están en estado "alto" (HIGH).
Alimentación externa	Puede alimentarse mediante el puerto USB o por una fuente externa de 7 a 12 V, usando un conector jack. Esto permite usarlo en proyectos móviles o sin computadora conectada.
Comunicación	Soporta varios protocolos de comunicación: USB (para cargar programas y comunicación con PC), UART (Serial) para monitores o módulos, I2C y SPI para dispositivos como pantallas, sensores inteligentes y memorias.
Programación	Se programa usando el lenguaje Arduino, una versión simplificada de C/C++, a través del Arduino IDE, un entorno gratuito, amigable y multiplataforma. Permite cargar programas fácilmente vía USB.
Compatibilidad virtual	Arduino Uno puede simularse en entornos como Tinkercad Circuits, lo que facilita su enseñanza incluso en entornos sin equipamiento físico. En la simulación se pueden conectar sensores, ejecutar código y observar resultados en tiempo real.

Arduino Uno combina potencia técnica, simplicidad operativa y versatilidad educativa, lo que lo convierte en una herramienta ideal para iniciar a los estudiantes en el diseño y control de sistemas electrónicos reales. Sus características permiten trabajar desde proyectos simples (como encender un LED) hasta sistemas complejos de automatización y robótica.

🔌 ¿Qué puedo conectar a Arduino Uno?

Uno de los grandes valores de Arduino Uno es su capacidad para interactuar con el mundo físico. Lo logra gracias a sus pines de entrada y salida, que permiten conectar sensores, actuadores y módulos complementarios. Esto transforma al controlador en una verdadera plataforma de desarrollo de sistemas robóticos.

🔍 1. Sensores (entradas)

Los sensores son dispositivos que permiten a Arduino percibir el entorno. Se conectan como entradas, ya que envían información al controlador. Pueden ser digitales (detectan solo encendido/apagado) o analógicos (miden valores variables).

🧳 Ejemplos comunes:

- LDR (resistencia dependiente de luz): Mide iluminación.

- Sensor de temperatura (como LM35): Entrega valores analógicos.
- Sensor ultrasónico: Mide distancia usando ondas de sonido.
- Sensor PIR: Detecta movimiento o presencia.
- Sensor de humedad del suelo: Ideal para proyectos de riego automático.
- Pulsadores o botones: Detectan interacción del usuario.

🔧*Estos sensores se conectan a pines analógicos (A0–A5) o digitales (0–13), según el tipo.*

⚙**Actuadores (salidas)**
Los actuadores son dispositivos que ejecutan una acción física. Se conectan como salidas, ya que Arduino les envía señales para que funcionen. Son esenciales para que un sistema robótico produzca efectos visibles o útiles.

🎒 **Ejemplos comunes:**

- **LEDs:** Emiten luz; usados como indicadores visuales.
- **Zumbadores (buzzer):** Producen sonido.
- **Servomotores:** Giran con precisión en un ángulo controlado (brazo robótico, compuerta).
- **Motores DC:** Generan movimiento continuo (ruedas, ventiladores).
- **Relés:** Permiten encender o apagar dispositivos de mayor voltaje.
- **Pantallas LCD u OLED:** Muestran datos en texto o gráficos.

📡 **3. Módulos complementarios**
Arduino también puede conectarse a módulos que amplían sus capacidades. Estos pueden usarse tanto como entradas, salidas, o interfaces de comunicación.

🎒 **Ejemplos:**

- **Módulo Bluetooth (HC-05/HC-06):** Permite comunicación inalámbrica con el celular.
- **Módulo Wi-Fi (ESP8266):** Conecta a redes para proyectos IoT.
- **Reloj de tiempo real (RTC DS3231):** Proporciona fecha y hora precisa.
- **Lector RFID:** Para identificar tarjetas o personas.
- **Módulo de tarjeta SD:** Para registrar datos (datalogging).

🎓 **Uso pedagógico clave**
Cada pin de Arduino Uno es una puerta de interacción con el entorno: el programador decide si será entrada (sensor), salida (actuador), o canal de comunicación (módulo).

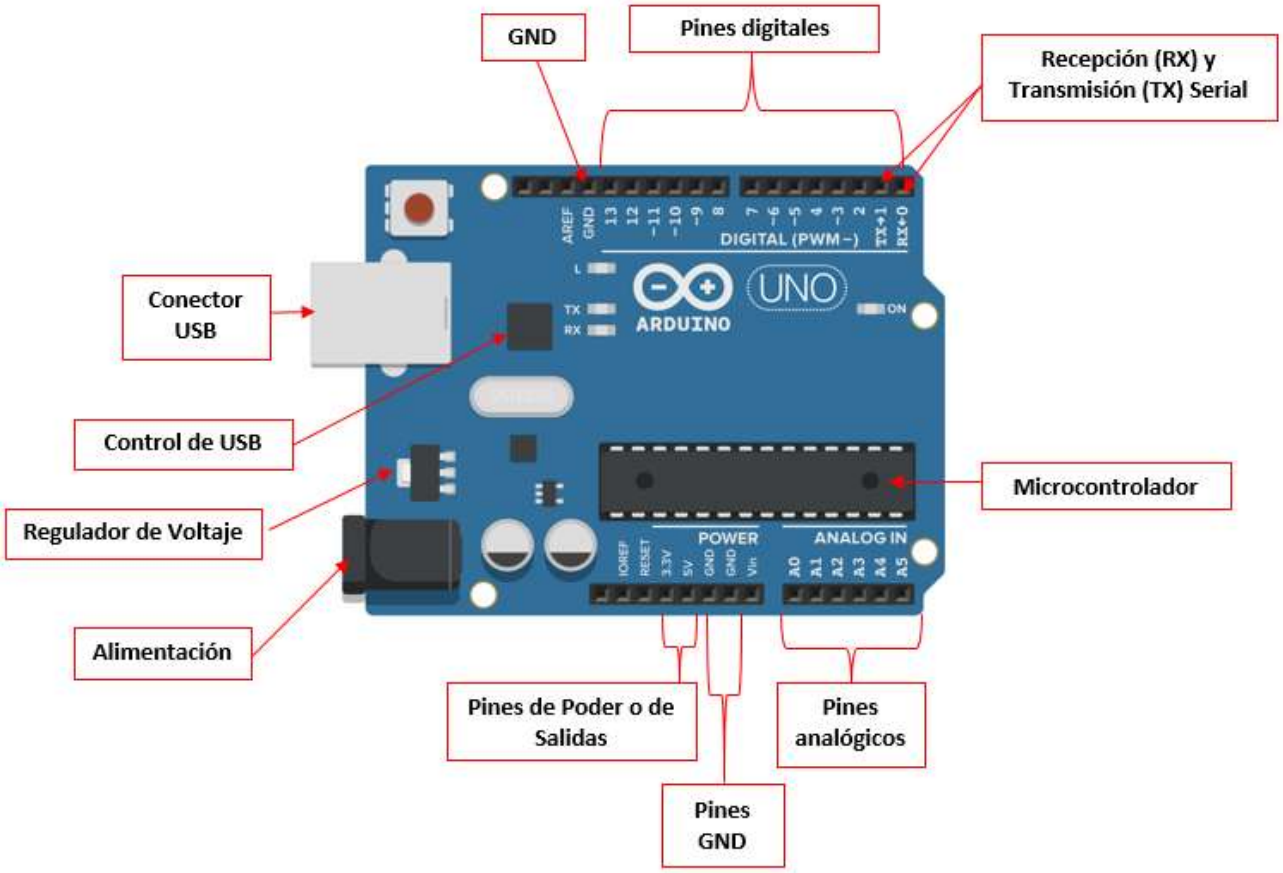
Comprender qué conectar y cómo usarlo permite a los estudiantes:

- Diseñar soluciones técnicas realistas.
- Visualizar el flujo de información (entrada → procesamiento → salida).
- Trabajar en proyectos interdisciplinarios (automatización, domótica, salud, energía, etc.).
- Desarrollar pensamiento sistémico y técnico.

Estructura física de la placa
Es recomendable familiarizarse visualmente con la placa. Entre los componentes principales encontramos:

- Conector USB (programación y alimentación)

- Microcontrolador (chip negro)
- Pines digitales (0 a 13)
- Pines analógicos (A0 a A5)
- Botón de reset
- Pines GND
- Pines de Poder o de Salidas 5 V, 3.3 V
- Control de USB: Moderador entre el Microcontrolador y el software o C
- Alimentación: Permite alimentar la tarjeta con voltaje de Corriente Continua de Siete (7) a Doce (12) voltios.
- Regulador de Voltaje: Permite una salida estable de Cinco (5) voltios independientemente del voltaje de entrada.
- Recepción (RX) y Transmisión (TX) Serial: Esta transmisión se da a través de los Pines Cero (0) y Uno (1).

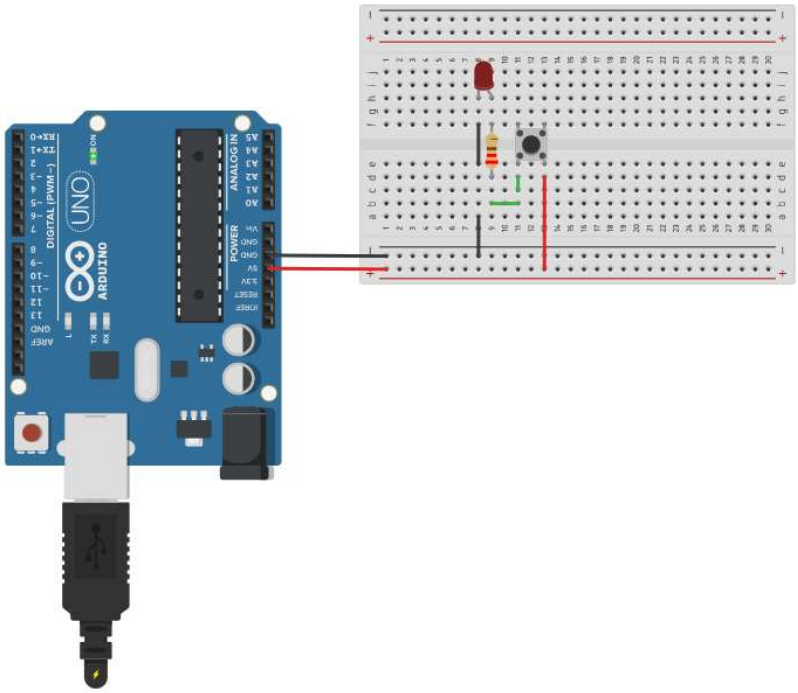


5. Primer ejemplo de uso: LED controlado por botón

Retomemos el ejemplo de la lección anterior. Donde utilizábamos la placa como fuente de alimentación de 5 volt. En este ejemplo no usábamos una salida digital, si no que usábamos un pin de Poder o de Salidas 5 V.

El pin siempre entrega 5 V constantes.
Es una fuente de alimentación estable para sensores, módulos, o componentes que necesitan estar encendidos todo el tiempo.
No se puede controlar por código
Se usa, por ejemplo, para alimentar sensores, botones o un potenciómetro.

📌 ***Ejemplo: conectar una pata del botón a 5 V → el Arduino detecta cuando se cierra el circuito, pero no puede "decidir" cuándo dar o cortar energía.***



A continuación veremos cómo podemos utilizar las entradas y las salidas junto con la programación para que el sistema cumpla la misma función

El pin puede entregar 5 V o 0 V, pero solo cuando se lo indica por código.
Se puede usar para controlar cuándo alimentar un componente (por ejemplo, prender un LED o activar un relé).
Se controla con digitalWrite(pin, HIGH); para ponerlo en 5 V, o digitalWrite(pin, LOW); para apagarlo (0 V).
Puede actuar como fuente "controlada".

📌 **Ejemplo: Si querés encender un LED solo cuando se presione un botón, usás una salida digital para darle 5 V solo cuando el programa lo decide.**

Diferencia clave:

Situación	Pin 5 V	Pin digital (salida)
¿Entrega energía siempre?	Sí, de forma constante	Solo cuando se lo indica por código
¿Se puede encender/apagar?	No	Sí
¿Controlado por código?	No	Sí (digitalWrite)
¿Sirve para entradas (botones)?	Sí	No se recomienda (sería innecesario)
¿Sirve para salidas (LEDs)?	No directamente	Sí, ideal para eso

🔌 **Introducción al montaje físico: LED controlado por botón**
Antes de pensar en programación, es fundamental entender cómo se conectan físicamente los componentes al Arduino Uno y qué función cumple cada uno dentro del sistema. En este caso, vamos a trabajar con dos elementos básicos: Un botón (pulsador) y un LED.

Componentes utilizados:

Componente	Función en el sistema	Conexión física
Arduino Uno	Controlador central que coordina todo	Se alimenta por USB o fuente externa
Botón (pulsador)	Entrada: permite que el usuario active una acción	Uno de sus terminales va al pin digital 2, el otro a GND o +5 V con una resistencia (según configuración)
LED	Salida: se enciende o apaga en respuesta al botón	Ánodo (pata larga) al pin 13 a través de una resistencia; cátodo (pata corta) a GND
Resistencia	Protege el LED de exceso de corriente	Se coloca en serie con el LED, típicamente de 220 Ω a 330 Ω

⚙Descripción del circuito físico

El botón está conectado a un pin de entrada digital (pin 2 en este caso). Al ser presionado, permite que fluya o no una señal hacia el Arduino.

El LED está conectado al pin digital 13, que se utilizará como salida. Arduino enviará corriente a ese pin para encender el LED, o lo pondrá en estado bajo (sin corriente) para apagarlo.

El circuito incluye una resistencia en serie con el LED, para limitar la corriente y evitar que se queme.

Opcionalmente, se puede usar una resistencia pull-down o pull-up con el botón para asegurar lecturas estables, aunque el pin 2 puede configurarse internamente para esto en el código.

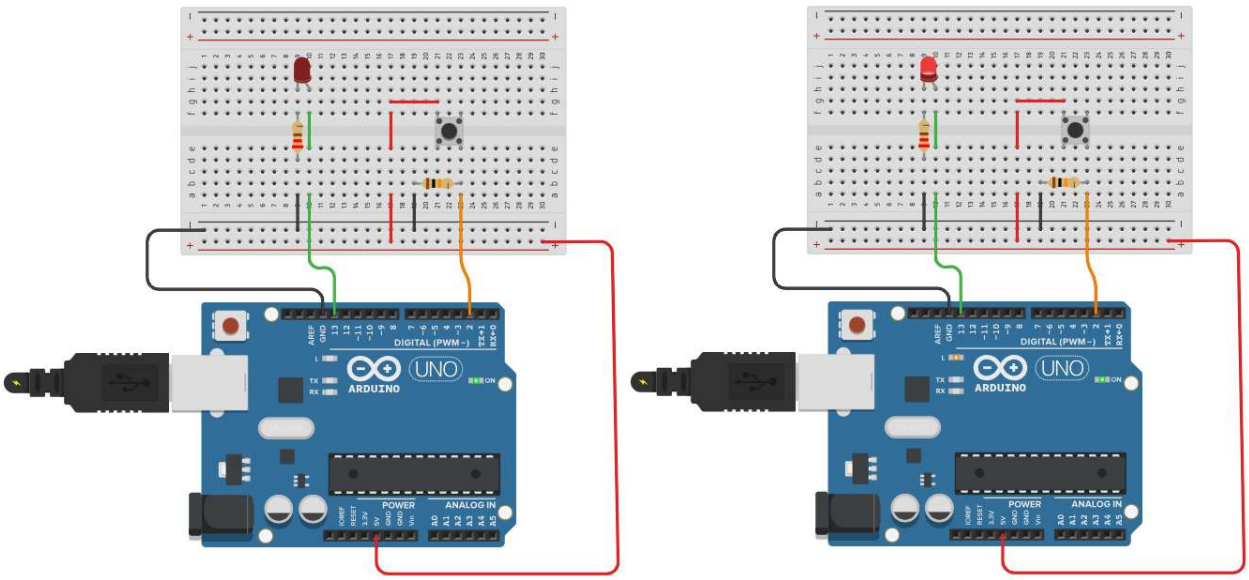
🔍 Flujo físico del sistema

El usuario interactúa con el botón.

El Arduino detecta un cambio en el estado eléctrico del pin 2.

Según ese estado, el pin 13 activa o desactiva el LED.

📌Lo importante en esta etapa es comprender qué elementos están involucrados, cómo están físicamente conectados, y qué rol desempeñan dentro del sistema.



```ino
const int boton = 2;
const int led = 13;

void setup() {
  pinMode(boton, INPUT);
  pinMode(led, OUTPUT);
}

void loop() {
  if (digitalRead(boton) == HIGH) {
    digitalWrite(led, HIGH);
  } else {
    digitalWrite(led, LOW);
  }
}
```

https://www.tinkercad.com/things/5hXAPs7RKGB-leccion-6-de-uso-entradas-y-salidas

¿Qué hace este código?

Lee el estado del botón conectado al pin 2.
Si el botón está presionado, enciende el LED en el pin 13.
Si no, lo apaga.

Este es el primer paso para entender cómo una entrada puede controlar una salida, algo que será la base de toda la lógica de control.

¿Por qué enseñar Arduino en la escuela técnica?
Arduino no solo es accesible y gratuito, también tiene un enorme valor pedagógico:

Aporta a...	Porque permite...
Electrónica	Trabajar con circuitos reales (resistencias, LEDs, sensores)
Programación	Desarrollar lógica secuencial, condicional y de repetición
Pensamiento computacional	Modelar procesos, abstraer situaciones, depurar errores
Interdisciplinariedad	Integrar matemática, física, diseño técnico y comunicación
Autonomía y creatividad	Diseñar soluciones propias a problemas técnicos

📌 ***Aprender Arduino es aprender a pensar técnicamente y a diseñar sistemas inteligentes reales.***

Bibliografía Opcional Recomendada.

Manual de Programación Arduino. Guía completa para principiantes que explica cómo comenzar con Arduino, sus características, y cómo programarlo para controlar dispositivos como LEDs, motores, y más.
Guía práctica sobre el mundo de Arduino. El propósito de esta guía es abordar el concepto de computación física que es la capacidad de interacción y comunicación de una máquina con los humanos
Aprende Arduino en un fin de semana. Manual de aprendizaje de Arduino