Universidad de Costa Rica

Laboratorio de Transmisión de Datos

Anteproyecto 1:
Transmisión de datos entre circuitos integrados

Prof. Ing. Teodoro Willink, M.Sc.

Marco Vásquez Ovares - B17032
Franco Castro Chaves - C01886
Grupo: 01

II - 2025

Índice

Índice de figuras

Índice de tablas

1. Presentación

El equipo **Industrial Automaton**, con sede en los astilleros de Naboo, se complace en presentar este anteproyecto para atender los Términos de Referencia del curso IE0528 - *Laboratorio de Transmisión de Datos.*

Inspirados en la tradición de innovación galáctica, hemos decidido emprender el desarrollo de un prototipo experimental bajo el nombre en clave **R2-D2**. Como primer paso en su evolución tecnológica, nuestro prototipo incorpora un **sistema de percepción basado en ultrasonido**, que le permitirá reconocer su entorno inmediato y ubicarse frente a obstáculos.

Para ello, conectaremos un **microcontrolador de propósito general** (Arduino UNO, basado en el ATmega328P) con un **periférico específico de medición** (sensor ultrasónico HC-SR04). Esta integración ejemplifica la comunicación entre dos circuitos integrados y constituye la base del estudio solicitado.

La información de distancia obtenida por el sensor será procesada en tiempo real por el microcontrolador, que activará un conjunto de indicadores luminosos (LEDs de distintos colores) para representar visualmente el rango de proximidad. De esta manera, el prototipo contará con una "percepción inicial" que servirá como fundamento para futuras mejoras de navegación autónoma.

Con este anteproyecto, **Industrial Automaton** reafirma su compromiso con la exploración, la creatividad y la rigurosidad técnica.

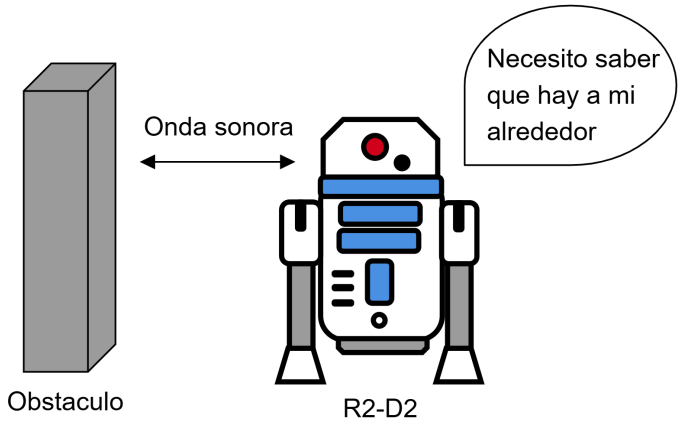

Figura 1: Representación conceptual del prototipo R2-D2 detectando un obstáculo mediante sensor ultrasónico.

2. Equipo de Ingeniería

El desarrollo de este estudio está a cargo de los siguientes profesionales:

- **Ing. Franco Castro Chaves**
  Ingeniero eléctrico con énfasis en electrónica y telecomunicaciones. Actualmente cursa la licenciatura en Sistemas de Comunicación. Apasionado por el arte y la música, fuentes que inspiran su creatividad e innovación. Durante su periodo como estudiante participó en el Laboratorio de Investigación Biomédica como investigador y en proyectos interdisciplinarios como Electrizarte, donde la ciencia y la tecnología se fusionan con expresiones artísticas. En la actualidad se desempeña como ingeniero en el área de Tecnologías de la Información, y ha desarrollado un creciente interés en el mundo del software, complementando así su formación técnica con nuevas competencias digitales.

- **Ing. Marco Vásquez Ovares**
  Ingeniero eléctrico con formación académica en el área de la ingeniería eléctrica y con conocimientos complementarios en programación, sistemas embebidos e ingeniería de computadoras. Mi perfil combina tanto la parte técnica de la electricidad y la electrónica, como el desarrollo de soluciones basadas en microcontroladores y entornos de software. Esta preparación me permite abordar proyectos de forma integral, desde el diseño del hardware hasta la implementación de algoritmos y aplicaciones prácticas en sistemas embebidos.

3. Equipo

Tabla 1: Lista de componentes

Componente	Cantidad	Observaciones
Arduino UNO	1	Unidad de procesamiento (ATmega328P)
Sensor ultrasónico HC-SR04	1	Sensor de distancia por ultrasonido
LED color azul	1	Indicador: fuera de rango
LED color rojo	1	Indicador: rango cercano
LED color amarillo	1	Indicador: rango medio
LED color verde	1	Indicador: rango lejano
Resistencias 220Ω	4	Limitadoras de corriente para LEDs
Protoboard	1	Plataforma de prototipado
Cables de conexión	1 set	Interconexión de componentes

4. Objetivos

4.1. Objetivo general

- Diseñar e implementar un sistema de medición de distancia utilizando un sensor ultrasónico y un microcontrolador Arduino, con el fin de visualizar los resultados y analizar su desempeño en la transmisión y procesamiento de datos.

4.2. Objetivos específicos

- Configurar el sensor ultrasónico HC-SR04 y el Arduino para la medición de distancias.
- Implementar la lógica de adquisición y procesamiento de datos en Arduino.
- Visualizar las mediciones en un dispositivo de salida (LEDs y monitor serial).
- Analizar el desempeño del sistema en cuanto a precisión y estabilidad.

5. Metodología

El desarrollo del proyecto se llevará a cabo en las siguientes etapas:

1. **Investigación preliminar:** Como parte de la etapa de investigación preliminar, se consultaron diversas fuentes teóricas y prácticas sobre el funcionamiento del sensor ultrasónico HC-SR04 y la plataforma Arduino UNO. Entre ellas se incluyen:
    1. Tutorial interactivo en línea sobre el sensor HC-SR04 en Tinkercad: https://www.tinkercad.com/things/1E3hXimmjCt-sensor-ultrasonico-hc-sr04
    2. Video explicativo de Arduino en YouTube: https://www.youtube.com/watch?v=SGHFbNRu1BA&t=384s
    3. Guía práctica en *Arduino Project Hub*: https://projecthub.arduino.cc/Isaac100/getting-started-with-the-hc-sr04-ultrasonic-sensor-7cabe1
    4. Hoja técnica del sensor ultrasónico HC-SR04
    5. Hoja técnica de la tarjeta Arduino UNO R3
2. **Diseño de hardware:** conexión física entre Arduino y el sensor ultrasónico, así como los LEDs indicadores.
3. **Diseño de software:** programación en Arduino IDE para el control del sensor, la medición de distancias y la activación de salidas.
4. **Pruebas de funcionamiento:** validación de la lectura de distancias y comportamiento del sistema.
5. **Evaluación de desempeño:** análisis de la precisión de las mediciones en diferentes condiciones.

6. Diseño de hardware

El diseño de hardware del proyecto se fundamenta en la **integración entre el microcontrolador Arduino UNO (ATmega328P)** y el sensor ultrasónico **HC-SR04**, estableciendo un canal de transmisión de datos digitales basado en señales de control. La comunicación se realiza mediante dos pines: `TRIG` (entrada del sensor) y `ECHO` (salida del sensor).

El proceso inicia cuando el Arduino aplica un pulso digital de **10 μs en nivel alto ($\approx$ 5 V)** sobre el pin `TRIG`, el cual ordena al HC-SR04 emitir una ráfaga de **8 ciclos ultrasónicos a 40 kHz**. Posteriormente, el sensor mantiene su pin `ECHO` en nivel alto durante un intervalo de tiempo proporcional a la duración del eco recibido. Dicho intervalo constituye la **codificación digital de la distancia**, ya que el ancho del pulso en `ECHO` representa el doble del tiempo de vuelo de la onda acústica. El Arduino mide este ancho de pulso en microsegundos y, a través de la relación:

$$d[\text{cm}] = \frac{t_{\text{echo}}[\mu s]}{58} \tag{1}$$

convierte el tiempo en una magnitud espacial. De esta forma, la transmisión de datos entre ambos circuitos integrados no se efectúa mediante un protocolo de bytes estructurados (como UART o I^{2}C), sino a través de la **duración de una señal digital**, lo que constituye un esquema sencillo pero efectivo de comunicación maestro–esclavo.

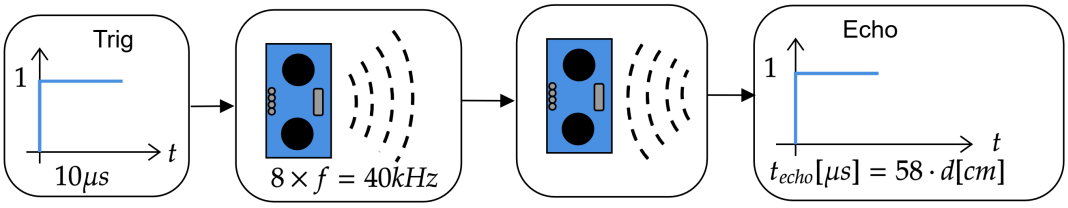

Figura 2: Secuencia de comunicación digital entre el Arduino UNO y el sensor ultrasónico HC-SR04. El pin TRIG recibe un pulso de 10μs en nivel alto, lo que provoca la emisión de 8 ciclos a 40kHz. El pin ECHO permanece en alto durante un tiempo proporcional a la distancia, codificando el resultado como un ancho de pulso digital.

Para la visualización de los resultados, se emplean LEDs indicadores conectados a pines digitales configurados como salidas. Cada LED representa un estado distinto según la distancia medida (por ejemplo: verde para distancia segura, amarillo para precaución y rojo para distancia crítica). Se utilizan resistencias en serie con los LEDs para limitar la corriente y proteger los componentes.

El sistema se alimenta a través del puerto USB del Arduino, lo que simplifica la implementación y reduce el uso de fuentes externas. Este diseño sencillo y modular facilita tanto la comprensión como la posible expansión del proyecto en el futuro, por ejemplo, añadiendo una pantalla LCD o comunicación con otros dispositivos.

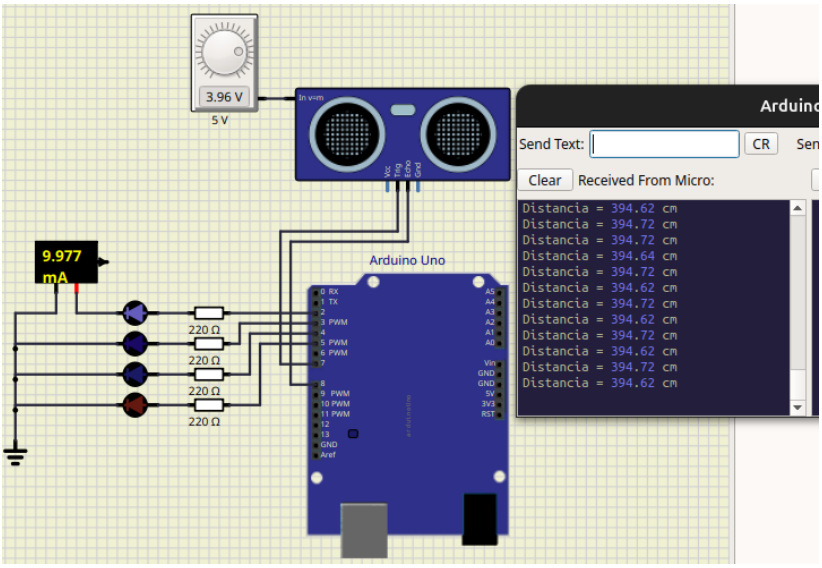

Figura 3: Diseño y funcionamiento del hardware
Elaboración propia

El cálculo de la resistencia se realiza aplicando la ley de Ohm:

$$R = \frac{V_{cc} - V_f}{I_f} \tag{2}$$

donde:

- $V_{cc} = 5\,V$ es el voltaje de salida de los pines digitales del Arduino.
- $V_f \approx 2{,}0\,V$ corresponde al voltaje típico de polarización directa de un LED estándar.
- $I_f \approx 15\,mA = 0{,}015\,A$ es la corriente recomendada para un brillo adecuado sin sobrecargar el microcontrolador.

Sustituyendo valores:

$$R = \frac{5V - 2V}{0{,}015A} \tag{3}$$

$$R \approx 200\,\Omega \tag{4}$$

El valor comercial más cercano es **220** Ω, que proporciona una corriente ligeramente menor ($\approx 13{,}6\,mA$), garantizando brillo suficiente y mayor seguridad tanto para el LED como para la salida del microcontrolador.

7. Diseño de software

7.1. Diagrama de flujo

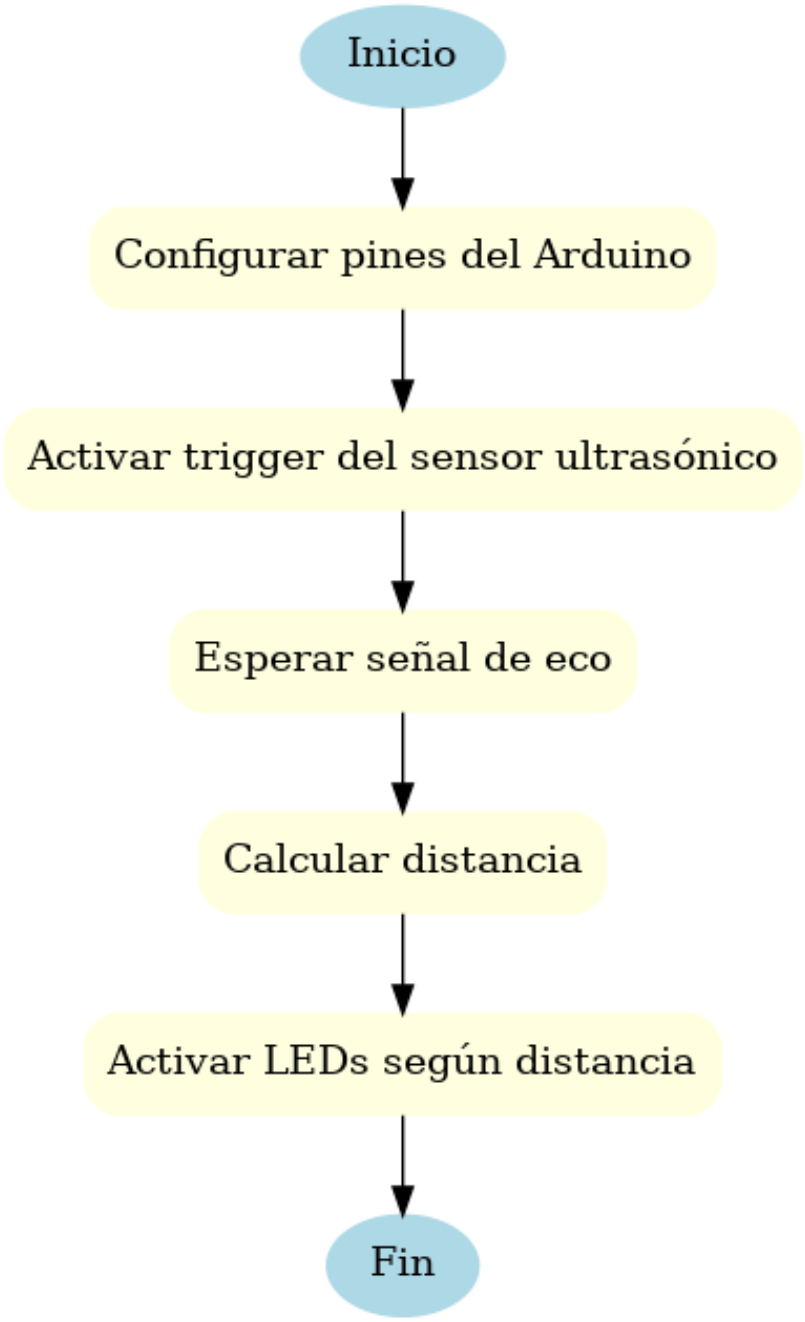

Figura 4: Diagrama de flujo
Elaboración propia

7.2. Definición de pines del sensor ultrasónico

```
const int trigger = 7;   // Pin digital para enviar el pulso (Trigger)
const int echo = 8;      // Pin digital para recibir el eco (Echo)
```

Estos pines definen la conexión con el sensor ultrasónico HC-SR04:

- `trigger`: se usa para enviar el pulso ultrasónico.
- `echo`: mide el tiempo que tarda en volver la señal reflejada.

7.3. Variables globales

```
float dist;   // Variable que almacena la distancia medida en cm
```

- `dist`: guarda el valor calculado de la distancia entre el sensor y un objeto en centímetros.

7.4. Definición de pines para LEDs indicadores

```
const int LED_AZUL = 2;
const int LED_ROJO = 3;
const int LED_AMARILLO = 4;
const int LED_VERDE = 5;
```

Cada LED se conecta a un pin del Arduino:

- Azul → fuera de rango o sin detección.
- Rojo → distancia corta (<30 cm).
- Amarillo → rango medio (30–150 cm).
- Verde → rango lejano (150–310 cm).

7.5. Configuración inicial (`setup()`)

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

  pinMode(trigger, OUTPUT);
  pinMode(echo, INPUT);

  pinMode(LED_AZUL, OUTPUT);
  pinMode(LED_ROJO, OUTPUT);
  pinMode(LED_AMARILLO, OUTPUT);
  pinMode(LED_VERDE, OUTPUT);
}
```

- Se inicia la comunicación serial a 9600 baudios para mostrar datos en el monitor serie.
- Se configuran los pines del sensor (`trigger` como salida y `echo` como entrada).
- Los pines de los LEDs se configuran como salidas digitales.

7.6. Generación del pulso ultrasónico

```
digitalWrite(trigger, LOW);
delayMicroseconds(5);
digitalWrite(trigger, HIGH);
delayMicroseconds(10);
digitalWrite(trigger, LOW);
```

Se envía un pulso corto al pin `trigger`:

- Bajo → espera inicial.
- Alto (10 µs) → dispara el pulso ultrasónico.
- Bajo → finaliza el pulso.

7.7. Medición del eco y cálculo de la distancia

```
dist = pulseIn(echo, HIGH);
dist = dist / 58;
```

- `pulseIn(echo, HIGH)` mide el tiempo (en microsegundos) que tarda la señal en regresar.
- Se divide entre 58 para convertirlo aproximadamente a centímetros, según la velocidad del sonido.

7.8. Envío de datos al monitor serie

```
Serial.print("Distancia = ");
Serial.print(dist);
Serial.println(" cm");
```

Muestra en pantalla la distancia medida, útil para verificar valores en la PC.

7.9. Apagado de LEDs

```
digitalWrite(LED_AZUL, LOW);
digitalWrite(LED_ROJO, LOW);
digitalWrite(LED_AMARILLO, LOW);
digitalWrite(LED_VERDE, LOW);
```

Antes de decidir qué LED encender, todos se apagan para evitar que queden varios prendidos al mismo tiempo.

7.10. Encendido de LEDs según rango de distancia

```
if (dist == 0 || dist > 310) {
  digitalWrite(LED_AZUL, HIGH);
}
else if (dist < 30) {
  digitalWrite(LED_ROJO, HIGH);
}
else if (dist >= 30 && dist <= 150) {
  digitalWrite(LED_AMARILLO, HIGH);
}
else if (dist > 150 && dist <= 310) {
  digitalWrite(LED_VERDE, HIGH);
}
```

Dependiendo del valor de `dist`, se enciende un solo LED:

- Azul → fuera de rango o sin detección.
- Rojo → muy cerca.
- Amarillo → distancia media.
- Verde → distancia lejana.

7.11. Retardo entre lecturas

```
delay(200);
```

Introduce una pausa de 200 milisegundos para estabilizar las lecturas y evitar parpadeo excesivo de los LEDs.

8. Desempeño esperado

El desempeño del proyecto se mide de acuerdo a los siguientes puntos:

- Medición de distancias en un rango aproximado de 2 cm a 400 cm.
- Activación de LEDs de acuerdo al rango de distancia detectado.
- Respuesta en tiempo real (latencia mínima en la medición).
- Estabilidad de la señal con un margen de error aceptable (± 1 cm en condiciones normales).

9. Presupuesto

9.1. Desglose de horas estimadas

Tabla 2: Horas de trabajo

Actividad	**Horas estimadas**
Investigación y planificación	8 h
Diseño del sistema (hardware/software)	10 h
Desarrollo de firmware / programación	16 h
Pruebas y validaciones	10 h
Documentación y presentación final	6 h
Total	**50 h**

9.2. Tarifa de referencia

Se propone una tarifa académica de ₡4,000 por hora.

9.3. Estimación de costo de mano de obra

$$50 \text{ horas} \times ₡4{,}000/\text{hora} = ₡200{,}000$$

9.4. Materiales y recursos

9.5. Costo total del proyecto

Por lo tanto se propone un costo total de: **₡200,000**

Tabla 3: Materiales

Componente	Cantidad	Proveedor	Costo (₡)
Arduino UNO	1	U.C.R.	0
Protoboard	1	U.C.R.	0
Sensor ultrasónico HC-SR04	1	Industrial Automaton	–
LED color azul	1	Industrial Automaton	–
LED color rojo	1	Industrial Automaton	–
LED color amarillo	1	Industrial Automaton	–
LED color verde	1	Industrial Automaton	–
Resistencias 220Ω	4	Industrial Automaton	–
Cables de conexión	1 set	Industrial Automaton	–

10. Cronograma

Tabla 4: Cronograma

Fecha	**Actividad**
Miércoles (Día 1)	Revisión de los Términos de Referencia (TdR) y definición del alcance.
Jueves (Día 2)	Planteamiento de objetivos y metodología de trabajo.
Viernes (Día 3)	Diseño preliminar del hardware (esquemático y descripción).
Sábado (Día 4)	Selección de componentes y verificación de compatibilidad.
Domingo (Día 5)	Documentación de diseño de hardware.
Lunes (Día 6)	Diseño preliminar del software (flujo lógico y pseudocódigo).
Martes (Día 7)	Desarrollo del código base en Arduino.
Miércoles (Día 8)	Revisión y depuración del código inicial.
Jueves (Día 9)	Integración del hardware y software en el documento del anteproyecto.
Viernes (Día 10)	Redacción del apartado de desempeño esperado y presupuesto estimado.
Sábado (Día 11)	Revisión y ajustes al cronograma dentro del documento.
Domingo (Día 12)	Redacción de conclusiones parciales y revisión general.
Lunes (Día 13)	Corrección de estilo y preparación del documento final.
Martes (Día 14)	Entrega final del proyecto.

Referencias

1. Tinkercad, "Sensor ultrasónico HC-SR04 en simulador," [En línea]. Disponible en: https://www.tinkercad.com/things/1E3hXimmjCt-sensor-ultrasonico-hc-sr04. [Accedido: 27-ago-2025].

2. Arduino, "Uso del sensor ultrasónico HC-SR04," YouTube, [En línea]. Disponible en: https://www.youtube.com/watch?v=SGHFbNRu1BA&t=384s. [Accedido: 27-ago-2025].

3. Isaac100, "Getting started with the HC-SR04 ultrasonic sensor," Arduino Project Hub, [En línea]. Disponible en: https://projecthub.arduino.cc/Isaac100/getting-started-with-the-hc-sr04-ultrasonic-sensor-7cabe1. [Accedido: 27-ago-2025].

4. Handson Technology, *HC-SR04 Ultrasonic Sensor Module User Guide*, Datasheet, 2019.

5. Arduino S.r.l., *Arduino UNO R3*, Datasheet, 2024.