ASIGNATURA: MICROCONTROLADORES ING. ELECTROMECÁNICA		
INFORME DE PRÁCTICAS CON ARDUINO Y DISPOSITIVOS ELECTRÓNICOS EN ESPACIO ÁULICO **LISTA DE COTEJO: 30 %**		
INSTITUTO TECNOLÓGICO SUPERIOR DE: SAN ANDRÉS TUXTLA		**GRUPO.** **EQUIPO.**
NOMBRE DEL DOCENTE: BLANCA N. RIOS ATAXCA.	**FECHA:**	**UNIDAD No.**
NOMBRE DE (LOS) ALUMNO (S):	**TEMA:** **-ACTIVACIÓN DE PUERTOS COMO ENTRADAS Y SALIDAS (ARDUINO)** **DISPLAY, LCD, MATRIZ 8X8** **-DISPOSITIVOS DE POTENCIA** **-MOTORES, ETC.**	

INSTRUCCIÓN

Revisar los documentos o actividades que se solicitan y marque en los apartados "SI" cuando la evidencia a evaluar se cumple; en caso contrario marque "NO". En la columna "OBSERVACIONES" ocúpela cuando tenga que hacer comentarios referentes a lo observado.

VALOR DEL REACTIVO %PLANEADO	CARACTERÍSTICA A CUMPLIR (REACTIVO)	CUMPLE			OBSERVACIONES
		SI	NO	%REAL	
2	**Portada: Nombre** de la escuela, logotipo, Nombre del proyecto, Carrera, Asignatura, Profesor, Alumnos Matricula, Grupo, Lugar y fecha de entrega. Título del tema. **Introducción, desarrollo, conclusiones**	✓			
3	**El alumno + equipo reconoce los instrumentos de trabajo en laboratorio y los que requiere para su uso en la elaboración de circuitos electrónicos, y los describe en el documento.**	✓			
5	**Identifica** el tema y cuestionamientos que se le presenta, relacionándolo con la asignatura. **Se observa** el comportamiento del alumno para trabajar de forma individual y en equipo (desempeño). Esto se verifica al entregar un informe completo y ampliamente explicado.	✓			
5	**Conceptos Básicos.** El alumno, en su reporte de prácticas, realiza una descripción breve y concreta de cada dispositivo empleado, características, aplicaciones (leds, LCD, display de 7 segmentos, motor PAP, servomotor y su función en el circuito elaborado.	✓			
5	**Presenta el desarrollo de sus actividades en el laboratorio y montaje de cada circuito con imágenes propias.** **Responde las preguntas que se realizan al final de cada practicas así como las actividades que se indican.**		-2		Faltó reportar (evidencias)la práctica 18
4	**Describe las imágenes propias, las actividades realizadas para lograr la ejecución de la práctica, incluyendo los problemas que tuvo que resolver y los resultados obtenidos.**		-2		

2	**Ortografía**: Aplicación de las normas para redactar textos.				
2	**Presentación.** Limpieza y formalidad, archivo electrónico realizado en Word o LaTex, libreta de apuntes, hoja blanca.				
2	**Puntualidad** en la entrega.				
30%	**Calificación.**				

MAESTRA: Blanca Nicandria Ríos Ataxca

MATERIA: Microcontroladores

UNIDAD: 1 y 2

Falta evidencia de la práctica #18

EQUIPO 1: Angel Abrajan González 221U0135

Erick Candelario Fiscal Ambros 221U0155

José armando cabrera Echavarría 221U0258

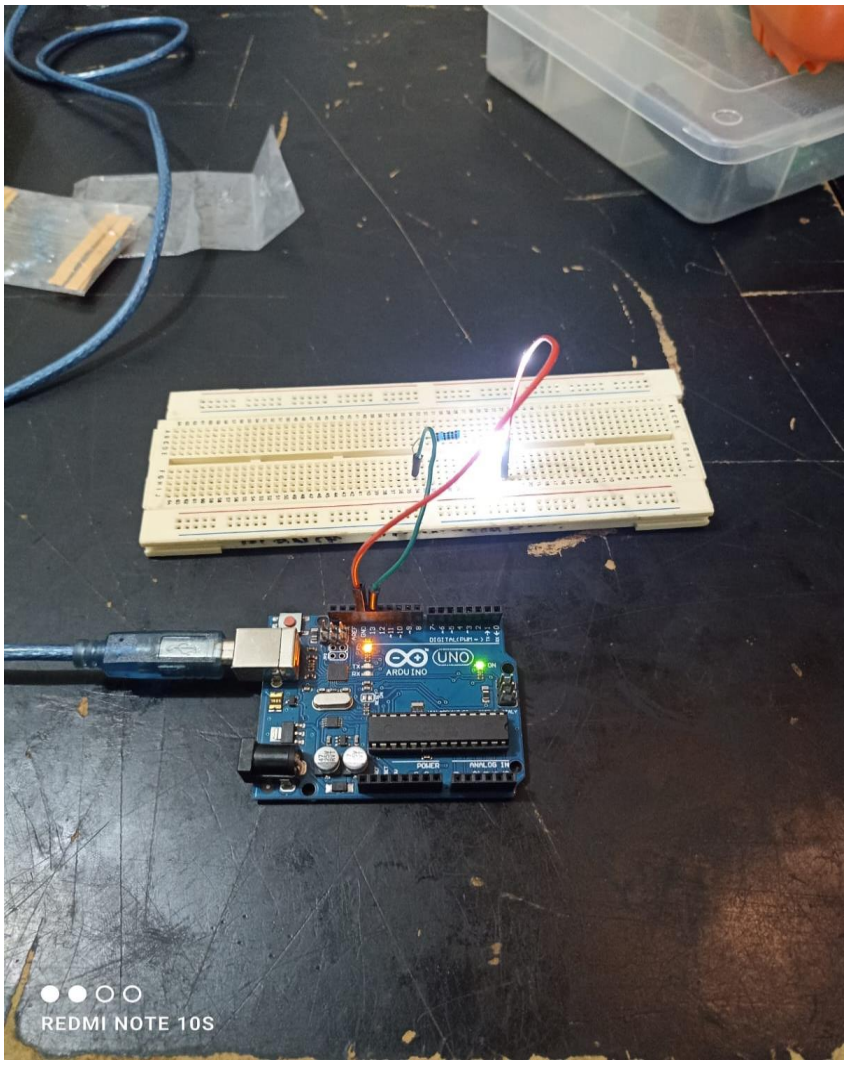
CARRERA: Ingeniería Electromecánica

REPORTE DE PRACTICAS

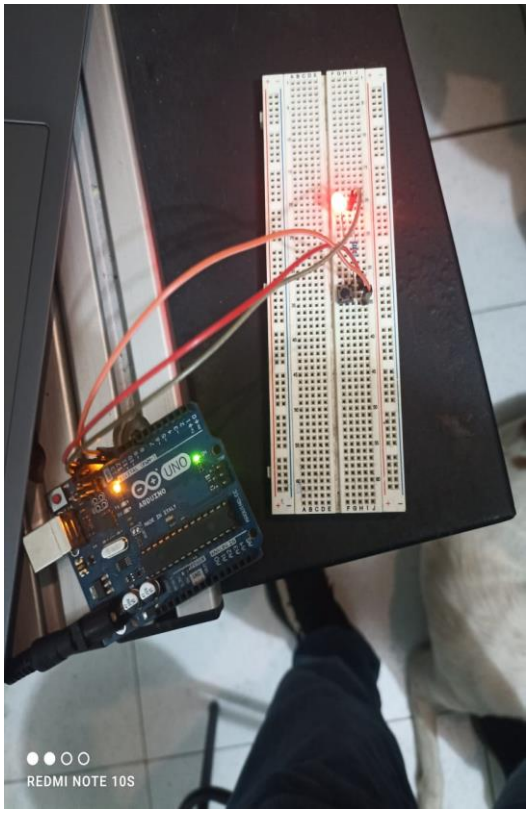
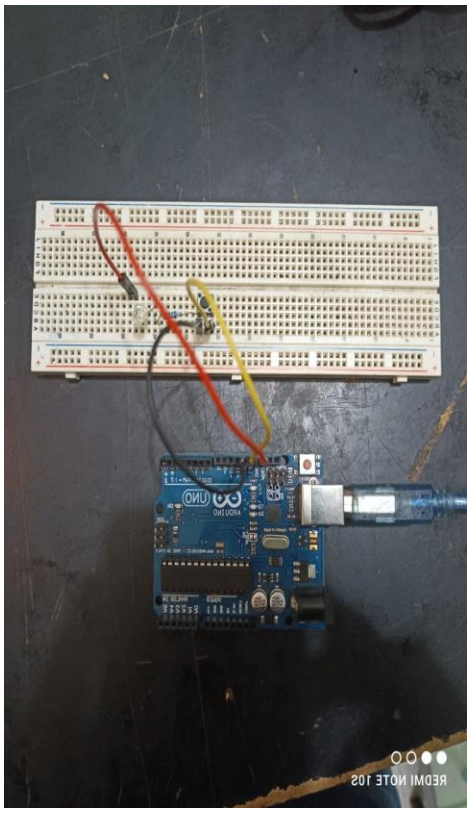
GRUPO: 702-B

FECHA DE ENTREGA: 06/12/2025

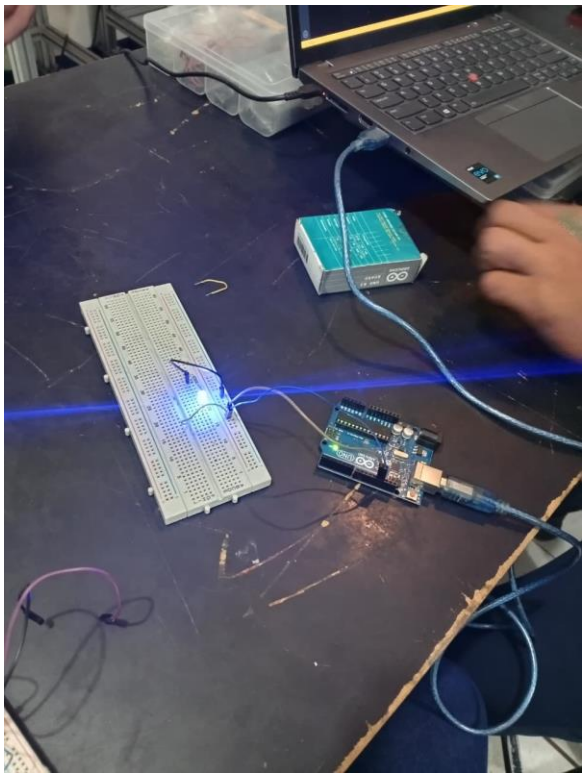
REPORTE DE PRÁCTICA	
CONFIGURACIÓN DE PUERTO DE SALIDA Y ENCENDIDO DE LED CON ARDUINO	
PRACTICA 1	
Objetivo	Configurar un puerto digital de un microcontrolador Arduino como salida y comprobar su funcionamiento mediante el encendido de un LED físico y virtual (en Proteus).
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • 1 LED • 1 Resistencia de 220 • Jumper **Software** • Arduino IDE • Proteus
Actividades Realizadas	**Montaje Físico** El LED se conectó de la siguiente manera: • El ánodo (pata larga) se conectó al pin digital 13 del Arduino. • El cátodo (pata corta) se conectó a la resistencia de 220 Ω. • El otro extremo de la resistencia se conectó a GND (tierra). **Montaje en Proteus** • En la biblioteca de Proteus se seleccionó el componente LED ACTIVE, ya que es ideal para observar el encendido y apagado. • Se colocó un Arduino UNO y se cargó el archivo. hex generado por Arduino IDE. • Se realizaron las conexiones igual que en el circuito físico. • Se ejecutó la simulación para observar el encendido del LED cada segundo
Resultados	• El puerto digital seleccionado quedó correctamente configurado como salida. • El LED encendió y apagó de acuerdo con las instrucciones del programa. • La simulación en Proteus con el componente LED ACTIVE permitió visualizar el comportamiento del sistema sin necesidad de un circuito físico.

	• Se verificó la correcta comunicación entre el software Arduino IDE y la tarjeta mediante la asignación adecuada del puerto COM.
Conclusión	• La configuración del modelo de tarjeta y del puerto de comunicación en Arduino IDE es fundamental para garantizar la correcta carga del programa. • El uso de una resistencia de 220 Ω es obligatorio para proteger el LED y evitar daños en el microcontrolador. • La simulación en Proteus facilita la comprensión del funcionamiento del programa sin necesidad de hardware real.
Ilustración	

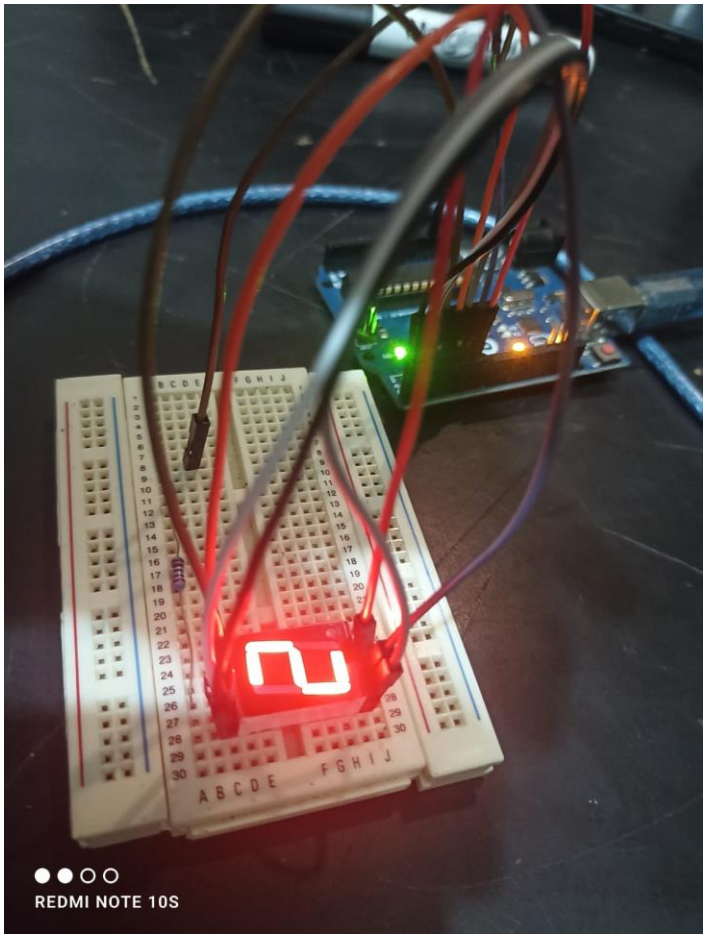
REPORTE DE PRÁCTICA	
PUERTO CONFIGURADO COMO ENTRADA CON ARDUINO	
PRACTICA 2	
Objetivo	Configurar un puerto digital de Arduino como entrada, utilizando un switch o pushbutton para activar o desactivar un LED, verificando el funcionamiento mediante hardware real o simulación.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • 1 Resistencia de 220 Ω • 1 Switch o Pushbutton • 1 LED • Protoboard, • Jumpers
Actividades Realizadas	**Montaje Físico** • Un extremo del botón se conecta al pin digital 2. • El otro extremo se conecta a +5V. • El pin digital 2 se conecta a tierra mediante la resistencia de 220 Ω (funciona como *pull-down*). • Cuando el botón se presiona, el pin recibe 5V → HIGH. • Cuando NO se presiona, la resistencia lo mantiene en 0V → LOW. **Conexión del LED** • Ánodo (pata larga) - Pin 13 del Arduino. • Cátodo (pata corta) - resistencia de 220 Ω - GND. **Montaje en Proteus** • En la biblioteca de Proteus se seleccionó el componente • Se colocó un Arduino UNO y se cargó el archivo. hex generado por Arduino IDE. • Se realizaron las conexiones igual que en el circuito físico. • Se ejecutó la simulación para observar el encendido del LED cada segundo
Resultados	• El pin digital seleccionado fue correctamente configurado como entrada. • El sistema detectó los cambios de estado del pushbutton: • HIGH cuando el botón fue presionado. • LOW cuando el botón no fue presionado. • El LED reaccionó en tiempo real, encendiéndose únicamente cuando el botón estaba activado.

	• Se confirmó la correcta comunicación entre el Arduino IDE y la tarjeta mediante la selección apropiada del puerto COM.
Conclusión	• Configurar pines como entrada permite al microcontrolador recibir información del exterior. • El pushbutton, junto con una resistencia de 220 Ω, permite crear un circuito estable. • Esta práctica demuestra la interacción básica entre usuario y microcontrolador mediante entradas y salidas digitales.
Ilustración	 

REPORTE DE PRÁCTICA	
PUERTO CONFIGURADO COMO SALIDA PWM (MODULACIÓN POR ANCHO DE PULSO)	
PRACTICA 3	
Objetivo	Configurar un puerto digital del Arduino con capacidad PWM para controlar la variación del brillo de un LED mediante el uso de la función analogWrite. Se comprobará el funcionamiento a través del hardware real o mediante el LED ACTIVE en Proteus.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • 1 Resistencia de 220 Ω (protección para el LED) • 1 LED • Jumpers **Software** • Arduino IDE (para programación del microcontrolador) • Proteus (para simulación del circuito con LED ACTIVE)
Actividades Realizadas	**Montaje del Circuito** • Conexión del LED • Ánodo (pata larga) - Pin PWM 9 • Cátodo (pata corta) - Resistencia de 220 Ω • Resistencia – GND **Simulación en Proteus** • Se abrió el software Proteus. • Se insertó un Arduino UNO • Se agregó el componente LED ACTIVE, que permite visualizar variaciones de brillo. • Se cargó el archivo. hex generado por Arduino IDE. • Se realizaron las conexiones del LED al pin PWM. • Al iniciar la simulación, el LED mostró una variación suave e incremental de intensidad, confirmando el funcionamiento del PWM.
Resultados	• El pin digital seleccionado fue correctamente configurado como salida PWM. • La señal PWM moduló la intensidad luminosa del LED, variando desde apagado hasta máximo brillo. • El LED ACTIVE de Proteus mostró la variación de intensidad de manera clara y progresiva.

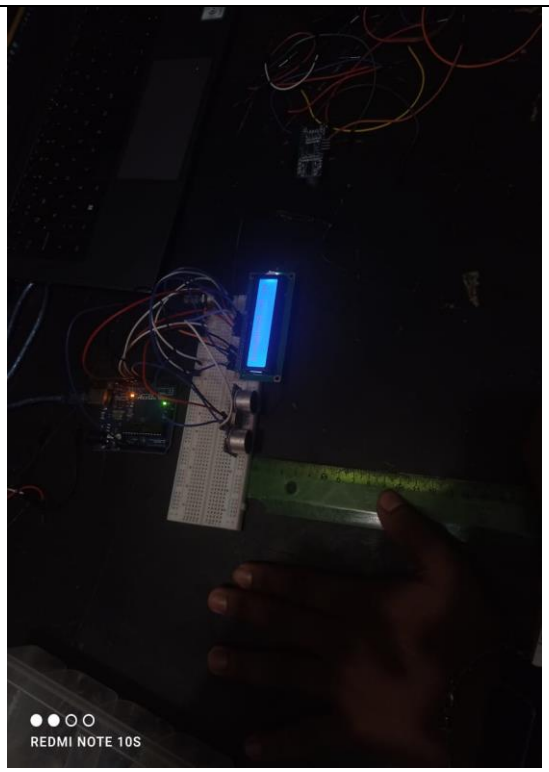
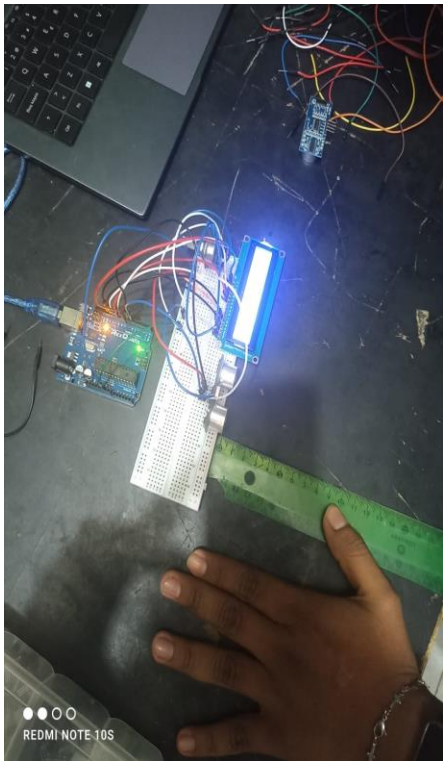
	• Se comprobó el uso adecuado del puerto de comunicación (COM) y la configuración del modelo de tarjeta seleccionada en Arduino IDE. • El ciclo de aumento y disminución del brillo se repitió continuamente sin errores
Conclusión	• La función analogWrite permite generar señales PWM con valores entre 0 y 255, modulando efectivamente la potencia entregada al LED. • El uso de PWM es fundamental cuando se requiere controlar velocidad de motores, brillo de LEDs u otros dispositivos que no requieren señales analógicas reales. • Los pines PWM varían entre modelos de Arduino, por lo que es importante seleccionar un pin correcto según la tarjeta utilizada. • La simulación en Proteus con LED ACTIVE facilita la observación clara de los cambios de brillo y es una herramienta muy útil para validar el programa sin necesidad de un circuito real. • Se verificó la correcta comunicación entre PC y Arduino mediante la selección adecuada del puerto y placa.
	

REPORTE DE PRÁCTICA	
CONEXIÓN DE DISPLAY DE 7 SEGMENTOS	
PRACTICA 4	
Objetivo	Aprender a conectar y controlar un display de 7 segmentos de ánodo común utilizando una tarjeta Arduino. Se mostrará un número en el display activando los segmentos necesarios mediante programación y usando Proteus para observar su funcionamiento.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • 1 Display de 7 segmentos tipo Ánodo Común • Jumpers **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	Un display de ánodo común funciona de la siguiente forma: • El ánodo común (COM) va conectado a +5V**.** • Los segmentos a, b, c, d, e, f, g se activan llevando a LOW el pin correspondiente del Arduino (encendido por nivel bajo). • Cada segmento requiere una resistencia para limitar la corriente. Simulación en Proteus • Se agregó un Arduino UNO. • Se colocó un display de 7 segmentos tipo activo para visualizar los efectos del programa. • Se conectaron los segmentos según la tabla anterior. • Se cargó el archivo .hex generado por Arduino IDE al Arduino virtual. • Al iniciar la simulación, el número 3 se mostró en el display, confirmando la activación correcta de los segmentos.
Resultados	• Aunque inicialmente se utilizó el código proporcionado por la docente, éste no generó el funcionamiento esperado en el display de 7 segmentos, ya que no activaba correctamente los segmentos correspondientes. Debido a ello, fue necesario modificar el programa y elaborar un nuevo código funcional, ajustado al tipo de display utilizado (ánodo común) y a la asignación real de pines del montaje.

Conclusión	• Una vez implementado el nuevo código, el display de 7 segmentos respondió correctamente, mostrando el número deseado y activando únicamente los segmentos correspondientes. • La conexión mediante resistencias de 220–500 Ω permitió proteger los LEDs internos del display, evitando daños por exceso de corriente.
Ilustración	

REPORTE DE PRÁCTICA	
PUERTO CONFIGURADO COMO SALIDA CON LCD 16X2 Y SENSOR ULTRASÓNICO HC-SR04	
PRACTICA 5	
Objetivo	Configurar un puerto digital como salida en Arduino para mostrar información en un display LCD 16x2, además de emplear un sensor ultrasónico HC-SR04 para medir distancias y desplegarlas en pantalla. Se realiza control de contraste mediante un potenciómetro, y se utiliza Proteus para simular el sistema.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • 1 potenciómetro de 10 kΩ (control de contraste del LCD) • 1 Display LCD 16x2 (basado en controlador Hitachi HD44780) • 1 sensor ultrasónico HC-SR04 • Cables de conexión • jumpers **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Conexión del LCD 16x2** • Para conectar el display LCD 16x2 se utilizaron 14 pines principales, además de los dos pines destinados a la retroiluminación. El pin VSS del LCD se conectó a GND del Arduino, mientras que el pin VDD se unió al terminal de 5V. El pin V0, encargado del ajuste de contraste, se enlazó al punto central de un potenciómetro de 10 kΩ. El potenciómetro se configuró con un extremo conectado a GND y el otro extremo a 5V, permitiendo regular la intensidad del contraste en la pantalla. • El pin RS del LCD se conectó al pin digital 12 del Arduino, y el pin E al pin 11. El pin RW se mantuvo a nivel bajo conectándolo directamente a GND para establecer el modo de escritura permanente. Los pines de datos D4, D5, D6 y D7 del LCD se conectaron respectivamente a los pines digitales 5, 4, 3 y 2 del Arduino, utilizando el modo de comunicación de 4 bits. **Conexión del Sensor Ultrasónico HC-SR04** • El sensor ultrasónico HC-SR04 se conectó de la siguiente manera:

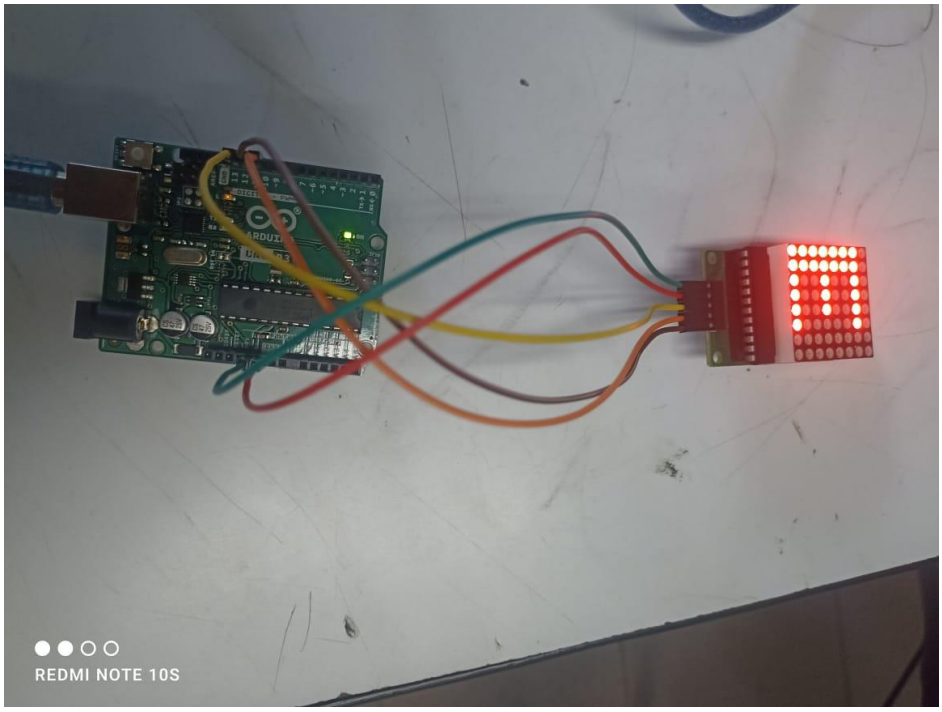
	El pin VCC del módulo se unió a la línea de 5V del Arduino y el pin GND se conectó a tierra común. El pin TRIG del sensor se conectó al pin digital 9 del Arduino, encargado de enviar el pulso ultrasónico. El pin ECHO se conectó al pin digital 10, donde el Arduino recibe el pulso de retorno para calcular la distancia. **Simulación en Proteus** • Se creó el circuito completo con: • Arduino UNO • LCD 16x2 • Potenciómetro • Sensor ultrasónico HC-SR04 (o módulo virtual equivalente) • Se ajustó el contraste mediante el potenciómetro. • Se cargó el archivo. hex generado desde Arduino IDE. • En la simulación, el LCD mostró correctamente: • "Distancia:" en la primera línea • La distancia medida en la segunda línea • Los valores cambiaban según la variación del sensor, confirmando el funcionamiento.
Resultados	• El LCD 16x2 funcionó correctamente mostrando texto y valores de distancia. • El sensor ultrasónico HC-SR04 permitió medir distancias con precisión aceptable. • El potenciómetro de 10 kΩ permitió ajustar correctamente el contraste del display. • La comunicación entre la PC y Arduino se configuró exitosamente mediante el puerto COM. • La simulación en Proteus demostró el funcionamiento real del sistema antes del montaje físico. • Los pines de salida configurados en Arduino enviaron la información correctamente al LCD.
Conclusión	• La práctica permitió comprender el control de un **LCD 16x2**, un dispositivo muy útil para mostrar datos en tiempo real. • Se comprobó el funcionamiento básico del sensor ultrasónico, que es ideal para medición de distancia sin contacto. • La correcta selección de pines y la configuración del software son esenciales para evitar errores en la comunicación del LCD. • El potenciómetro es indispensable para ajustar el contraste del display y asegurar la legibilidad. • Proteus proporciona un entorno seguro para probar el programa antes de implementarlo en hardware real.

Ilustración	 

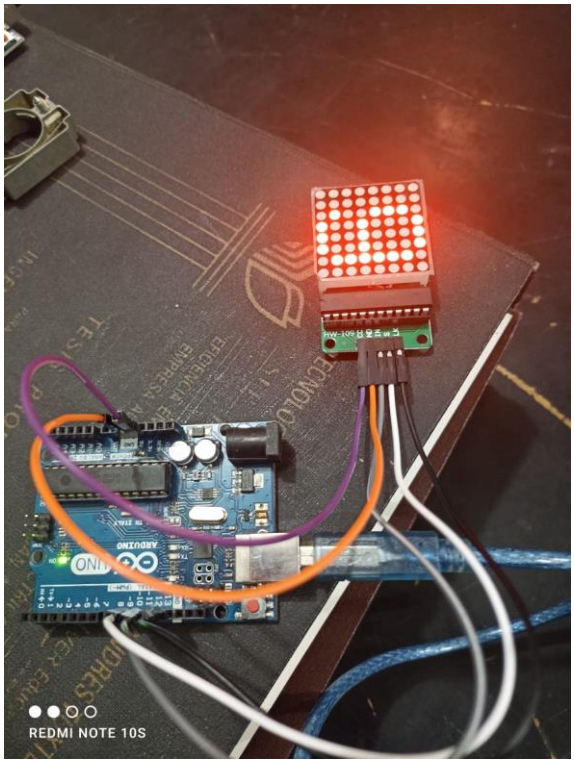
REPORTE DE PRÁCTICA	
TEXTO EN LA PANTALLA LCD	
PRACTICA 6	
Objetivo	Configurar un puerto digital como salida en Arduino para mostrar información en un display LCD 16x2
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • 1 potenciómetro de 10 kΩ (control de contraste del LCD) • 1 Display LCD 16x2 (basado en controlador Hitachi HD44780) • 1 sensor ultrasónico HC-SR04 • Cables de conexión • jumpers **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Conexión del LCD 16x2** • Para conectar el display LCD 16x2 se utilizaron 14 pines principales, además de los dos pines destinados a la retroiluminación. El pin VSS del LCD se conectó a GND del Arduino, mientras que el pin VDD se unió al terminal de 5V. El pin V0, encargado del ajuste de contraste, se enlazó al punto central de un potenciómetro de 10 kΩ. El potenciómetro se configuró con un extremo conectado a GND y el otro extremo a 5V, permitiendo regular la intensidad del contraste en la pantalla. **Simulación en Proteus** • Se creó el circuito completo con: • Arduino UNO • LCD 16x2 • Se cargó el archivo. hex generado desde Arduino IDE. • En la simulación, el LCD mostró correctamente:
Resultados	• El LCD 16x2 funcionó correctamente mostrando texto
Conclusión	• La práctica permitió comprender el control de un **LCD 16x2**, un dispositivo muy útil para mostrar datos en tiempo real.

REPORTE DE PRÁCTICA	
CÓDIGO PARA MOSTRAR CARACTERES ESPECIALES	
PRACTICA 7	
Objetivo	Configurar un puerto digital como salida en Arduino para mostrar información en un display LCD 16x2
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • 1 potenciómetro de 10 kΩ (control de contraste del LCD) • 1 Display LCD 16x2 (basado en controlador Hitachi HD44780) • 1 sensor ultrasónico HC-SR04 • Cables de conexión • jumpers **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Conexión del LCD 16x2** • Es la misma conexión solo se modifica el código • Para conectar el display LCD 16x2 se utilizaron 14 pines principales, además de los dos pines destinados a la retroiluminación. El pin VSS del LCD se conectó a GND del Arduino, mientras que el pin VDD se unió al terminal de 5V. El pin V0, encargado del ajuste de contraste, se enlazó al punto central de un potenciómetro de 10 kΩ. El potenciómetro se configuró con un extremo conectado a GND y el otro extremo a 5V, permitiendo regular la intensidad del contraste en la pantalla. **Simulación en Proteus** • Se creó el circuito completo con: • Arduino UNO • LCD 16x2 • Se cargó el archivo. hex generado desde Arduino IDE. • En la simulación, el LCD mostró correctamente:
Resultados	• El LCD 16x2 funcionó correctamente mostrando texto corrido
Conclusión	• La práctica permitió comprender el control de un **LCD 16x2**, un dispositivo muy útil para mostrar datos en tiempo real.

REPORTE DE PRÁCTICA	
CONTROL DE UNA MATRIZ LED 8X8 CON ARDUINO	
PRACTICA 8	
Objetivo	Implementar el control de una matriz LED 8x8 utilizando una tarjeta Arduino Uno, configurando correctamente el entorno de programación, las conexiones físicas y verificando la activación de los LEDs mediante un programa donde se presentará una palabras o oraciones
Materiales Utilizados	**Hardware** • Tarjeta Arduino Uno • Cable USB tipo A–B para comunicación entre Arduino y la PC • Matriz LED 8x8 • Alambres para conexión (jumpers) • PC con puerto USB disponible **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Conexión** • Para el ensamble del circuito, se consideraron dos opciones según el tipo de matriz LED utilizada. En el caso de una matriz LED con el controlador MAX7219, las conexiones fueron simples, utilizando únicamente tres pines principales del Arduino: DIN, CS y CLK, conectados a los pines 12, 10 y 11 del Arduino respectivamente. Las líneas de alimentación VCC y GND se conectaron a 5V y tierra. Esta configuración permitió enviar datos seriales al controlador, que se encarga de manejar automáticamente filas y columnas de la matriz. **Simulación en Proteus** • Se creó el circuito completo con: • Arduino UNO • Se cargó el archivo. hex generado desde Arduino IDE. • En la simulación,
Resultados	• Se verificó que la comunicación Arduino–PC era correcta. • La matriz LED encendió y respondió de acuerdo a las instrucciones del programa. • Con la matriz MAX7219, el rendimiento fue estable y de fácil control.

Conclusión	• El uso de una matriz LED 8x8 con controlador MAX7219 simplifica significativamente la conexión y programación. • La correcta selección del puerto y tarjeta en el Arduino IDE es esencial para evitar errores de carga. • El uso de resistencias adecuadas es fundamental para proteger los LEDs y asegurar que enciendan correctamente. • La práctica permitió comprender conceptos como multiplexado, control de filas/columnas y comunicación serial con dispositivos externos.
Ilustración	

REPORTE DE PRÁCTICA	
CONTROL DE UNA MATRIZ LED 8X8 CON ARDUINO	
PRACTICA 9	
Objetivo	Implementar el control de una matriz LED 8x8 utilizando una tarjeta Arduino Uno, configurando correctamente el entorno de programación, las conexiones físicas y verificando la activación de los LEDs mediante un programa donde se presentará una secuencia del 1 al 10
Materiales Utilizados	**Hardware** • Tarjeta Arduino Uno • Cable USB tipo A–B para comunicación entre Arduino y la PC • Matriz LED 8x8 • Alambres para conexión (jumpers) • PC con puerto USB disponible **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Conexión** • Para el ensamble del circuito, se consideraron dos opciones según el tipo de matriz LED utilizada. En el caso de una matriz LED con el controlador MAX7219, las conexiones fueron simples, utilizando únicamente tres pines principales del Arduino: DIN, CS y CLK, conectados a los pines 12, 10 y 11 del Arduino respectivamente. Las líneas de alimentación VCC y GND se conectaron a 5V y tierra. Esta configuración permitió enviar datos seriales al controlador, que se encarga de manejar automáticamente filas y columnas de la matriz. **Simulación en Proteus** • Se creó el circuito completo con: • Arduino UNO • Se cargó el archivo. hex generado desde Arduino IDE. • En la simulación,
Resultados	• Se verificó que la comunicación Arduino–PC era correcta. • La matriz LED encendió y respondió de acuerdo a las instrucciones del programa. • Con la matriz MAX7219, el rendimiento fue estable y de fácil control.

Conclusión	• El uso de una matriz LED 8x8 con controlador MAX7219 simplifica significativamente la conexión y programación. • La correcta selección del puerto y tarjeta en el Arduino IDE es esencial para evitar errores de carga. • El uso de resistencias adecuadas es fundamental para proteger los LEDs y asegurar que enciendan correctamente. • La práctica permitió comprender conceptos como multiplexado, control de filas/columnas y comunicación serial con dispositivos externos.
Ilustración	

REPORTE DE PRÁCTICA

CONTROL DE SERVOMOTOR CON PUSHBUTTON

PRACTICA 10

Objetivo

Identificar y utilizar los puertos de entrada y salida del microcontrolador Arduino Uno para controlar un servomotor mediante pulsadores. La práctica permite comprender cómo el microcontrolador recibe señales desde el exterior mediante entradas digitales y cómo envía instrucciones de movimiento a un periférico, en este caso un servomotor, utilizando una señal PWM.

Materiales Utilizados

Hardware

- Tarjeta Arduino UNO
- Servomotor
- 2 Pushbuttons (pulsadores)
- Resistencias de 10 kΩ
- Protoboard

Software

- Arduino IDE (para programación del microcontrolador)
- Proteus

Actividades Realizadas

Entradas y salidas del Arduino

- Entradas digitales: permiten recibir información desde el exterior, por ejemplo, el estado de un botón.
- Salidas digitales: permiten enviar señales que activan o controlan dispositivos externos.
- Señal PWM (Pulse Width Modulation): señal necesaria para el control del servo; son los pines 3, 5, 6, 9, 10 y 11.

Funcionamiento del pulsador (pushbutton)

Los pulsadores permiten introducir señales digitales de:

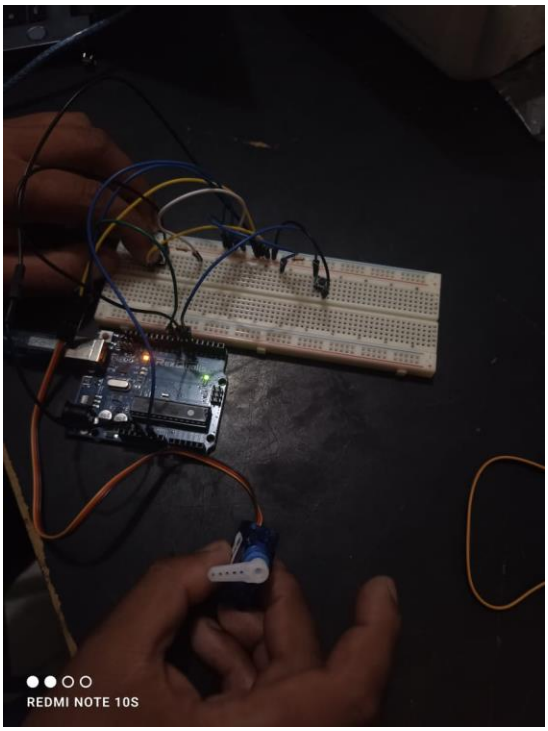
- LOW (0) cuando el botón está suelto.
- HIGH (1) cuando el botón es presionado y conectado a 5V.

Conexión del servomotor

Cable Servo	Función	Conexión
Rojo	Alimentación	5V Arduino
Café/Negro	Tierra	GND Arduino
Amarillo	Señal PWM	Pin 9

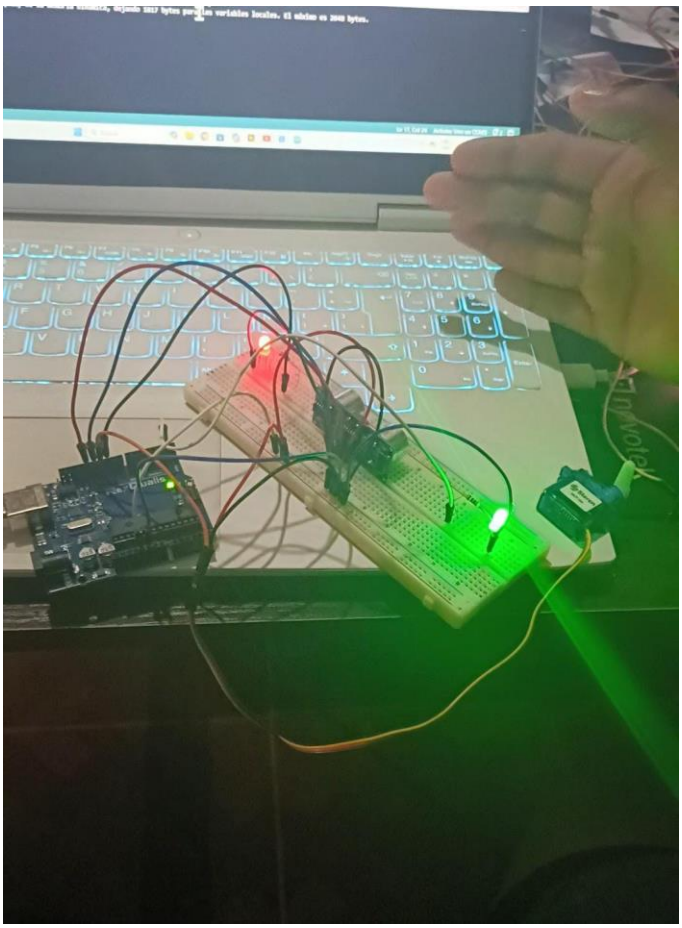
Simulación en Proteus

- Se creó el circuito completo con:
- Arduino UNO
- Se cargó el archivo. hex generado desde Arduino IDE.
- En la simulación

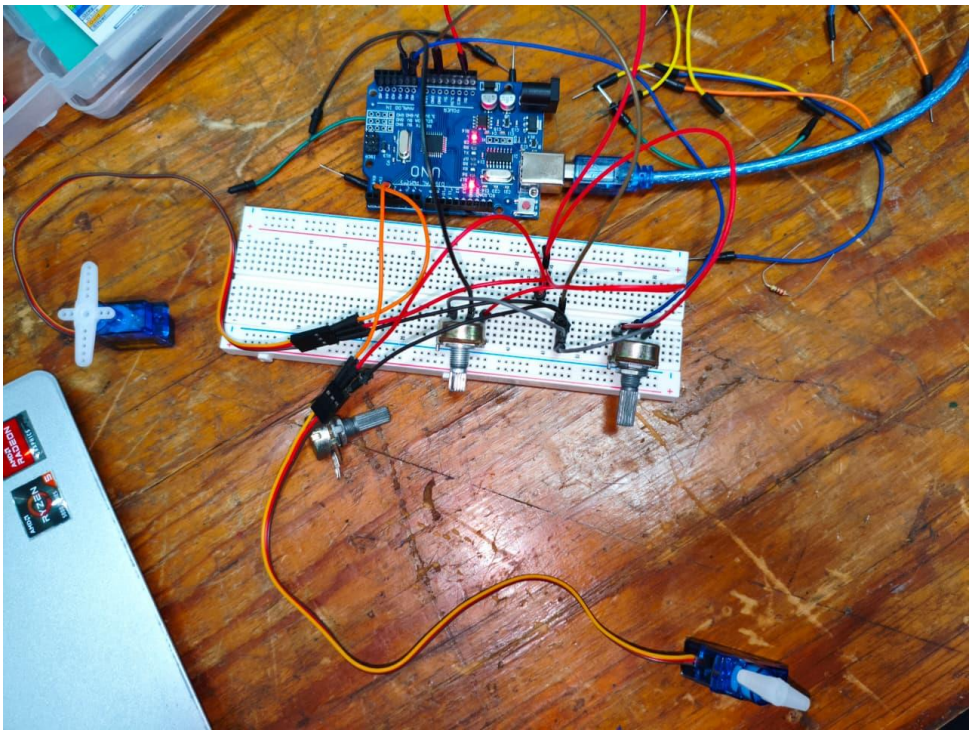
Resultados	• El servomotor se activó únicamente cuando se presionó alguno de los pushbuttons. • Se verificó que el uso de resistencias pull-down evitó activaciones falsas. • El motor alcanzó sin problema las posiciones programadas (0° y 180°). • En Proteus, la simulación funcionó correctamente al emplear el modelo adecuado de servomotor. • La respuesta del servo demostró el correcto manejo de señales PWM por parte del Arduino.
Conclusión	• Los pines de entrada digital del Arduino permiten recibir señales directas desde botones y switches. • Los pines configurados como salida PWM permiten controlar actuadores como servomotores. • La correcta implementación de resistencias pull-down es esencial para evitar lecturas incorrectas. • El uso de la librería Servo simplifica el control del motor y evita manejar manualmente señales PWM. • La práctica confirma el flujo correcto de información: entrada (pushbutton) → procesamiento → salida (movimiento del servo). • Una conexión incorrecta o un pin mal asignado provoca fallas inmediatas en el funcionamiento del servomotor.
Ilustración	

REPORTE DE PRÁCTICA	
SERVOMOTOR CONTROLADO POR SENSOR ULTRASÓNICO HC-SR04	
PRACTICA 11	
Objetivo	Comprender el funcionamiento del sensor ultrasónico HC-SR04 y su integración con un servomotor SG90 o similar, utilizando una tarjeta Arduino UNO. Analizar cómo la distancia medida por el sensor puede activar el movimiento del servo y, adicionalmente, encender o apagar LEDs según los valores detectados. Durante la simulación, se emplearon los botones del modelo del sensor para modificar manualmente la distancia y observar su efecto en el servomotor.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • Servomotor • Sensor Ultrasónico HC-SR04 • 2 Pushbuttons (pulsadores) • Resistencias de 10 kΩ • Protoboard **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Conexión del sensor ultrasónico HC-SR04** **El sensor HC-SR04 se conecta al Arduino de la siguiente manera:** • El pin VCC del sensor se conecta al pin 5V del Arduino para su alimentación. • El pin GND va conectado a tierra (GND) del Arduino. • El pin TRIG, encargado de enviar el pulso de disparo, se conecta al pin digital 7. • El pin ECHO, que recibe el pulso de retorno, se conecta al pin digital 6 del Arduino **Conexión del servomotor** El servomotor requiere tres conexiones principales: • El cable rojo, correspondiente a la alimentación, se conecta al pin 5V del Arduino. • El cable café o negro, que es la tierra, se conecta al pin GND. • El cable amarillo, encargado de la señal PWM, debe conectarse al pin 9 del Arduino, desde donde se controlará el movimiento del servo.

	Conexión de los LEDs Se emplean dos LEDs, cada uno con su resistencia de **220 Ω** para limitar la corriente: • El LED 1 se conecta al pin digital 3 del Arduino y se utiliza para indicar distancias cortas detectadas por el sensor. • El LED 2 se conecta al pin digital 4, y su función es indicar distancias medias. • Arduino UNO • Se cargó el archivo. hex generado desde Arduino IDE. • En la simulación
Resultados	• Los botones del sensor permitieron modificar los valores visualizados entre 25 y 38, rango donde el servo realizó cambios notables. • El servomotor respondió adecuadamente a los valores mostrados por el sensor. • Las transiciones entre 0°, 90° y 180° fueron suaves y precisas. • Los LEDs cambiaron de estado conforme la distancia ingresada. • El sistema permitió observar la interacción entre lectura, procesamiento y actuación del Arduino en tiempo real. •
Conclusión	• El sensor ultrasónico permite medir distancias de forma precisa y su integración con Arduino es sencilla. • El servomotor muestra respuesta inmediata a los cambios en la distancia detectada. • Los rangos entre **25 y 38 cm** fueron los más relevantes para observar el cambio de comportamiento del sistema. • El control simultáneo de servo y LEDs demuestra el manejo de múltiples salidas del microcontrolador. • La simulación mediante los botones del sensor facilita el análisis del comportamiento sin necesidad de hardware físico. • La práctica refuerza la comprensión del uso de entradas (Echo), salidas (PWM del servo y LEDs) y procesamiento mediante código.

Ilustración	

REPORTE DE PRÁCTICA	
SERVOMOTOR CONTROLADO POR SENSOR ULTRASÓNICO HC-SR04	
PRACTICA 12	
Objetivo	Analizar el funcionamiento de dos servomotores controlados por señales PWM generadas por Arduino UNO, utilizando dos potenciómetros de 10 kΩ como entradas analógicas. Comprender cómo la variación del voltaje en los pines A0 y A1 permite modificar el ángulo de giro de cada servomotor y observar su comportamiento en la simulación.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • Servomotor • 2 Pushbuttons (pulsadores) • 2 potenciómetros de 10 kΩ • 2 servomotores SG90 o similares **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Conexión de los potenciómetros** Cada potenciómetro: • Terminal 1 - GND • Terminal 3 - +5V • Terminal central - A0 (Potenciómetro 1) y A1 (Potenciómetro 2) **Conexión del servomotor** El servomotor requiere tres conexiones principales: • El cable rojo, correspondiente a la alimentación, se conecta al pin 5V del Arduino. • El cable café o negro, que es la tierra, se conecta al pin GND. • El cable amarillo, encargado de la señal PWM, debe conectarse al pin 9 del Arduino, desde donde se controlará el movimiento del servo.
Resultados	• Se observó que cada potenciómetro controló de manera independiente el movimiento de su servomotor. • Al girar el potenciómetro lentamente, el servo respondió con un movimiento suave y controlado. • Las variaciones rápidas generaron cambios bruscos en el giro del servo. • Al acercar el potenciómetro al valor máximo, el servo alcanzó aproximadamente los 180°, mientras que con valores mínimos regresó alrededor de los 0°. • No se presentaron fallas en la simulación, aunque se comprobó que si ambos servos fueran alimentados por el 5V

	del Arduino, la placa podría reiniciarse, lo cual valida la recomendación del uso de una fuente externa
Conclusión	• Los potenciómetros permiten modificar el voltaje de entrada y, por lo tanto, controlar con precisión el ángulo del servomotor. • La lectura analógica de Arduino es fundamental para convertir señales físicas en acciones programables. • El uso de la función map() facilita la adaptación de los valores leídos a los rangos requeridos por los actuadores. • Es indispensable considerar el consumo de corriente de los servomotores; dos o más motores deben alimentarse externamente para evitar daños o fallas en la tarjeta Arduino. • La práctica permitió comprender de forma clara la interacción entre sensores analógicos, salidas PWM y elementos mecánicos.
Ilustración	

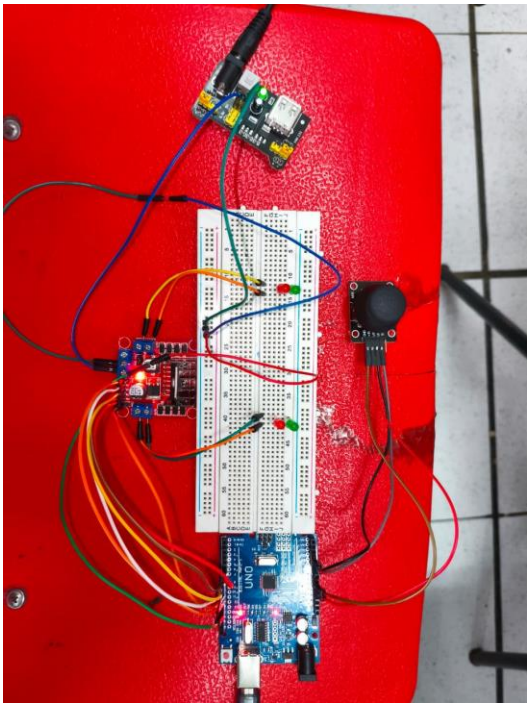
REPORTE DE PRÁCTICA	
ACTIVACIÓN DE UN MOTORES DC CONTROLANDO AVANCE, PARO Y REVERSA MEDIANTE DRIVER L298N	
PRACTICA 13	
Objetivo	Identificar y configurar las entradas y salidas del microcontrolador Arduino UNO para controlar el movimiento de uno o dos motores de corriente continua (DC), utilizando un driver L298N como puente H. Se busca lograr el control de avance, paro y reversa mediante pulsadores, sensores o joystick, comprendiendo el funcionamiento del driver y la lógica de control de los motores.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • Driver L298N (puente H doble) • 1 o 2 motores DC • Pulsadores (push buttons) • Joystick analógico o sensor (opcional según actividad) • Fuente externa de 9–12 V para los motores • Cables de conexión • Protoboard **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Con una sola fuente** • Conectar la fuente al pin 12V (mismo voltaje que soporta el motor). • Dejar puesto el jumper que habilita el regulador a 5V. • No conectar nada al pin de 5V (ya tiene 5V internos. • Recomendado solo si la fuente es menor a 12V para evitar sobrecalentamiento. **Con dos fuentes** • Quitar el jumper del regulador. • Conectar 5V externos al pin de 5V (por ejemplo desde Arduino). • Conectar la fuente del motor al pin de 12V. **Control del módulo** • **MOTOR A**: ENA, IN1, IN2 → salidas OUT1 y OUT2 • **MOTOR B**: ENB, IN3, IN4 → salidas OUT3 y OUT4 • **ENA y ENB**: habilitan motores y permiten controlar la **velocidad (PWM)**. • Si no usarás PWM, coloca los **jumpers** para mantener los motores habilitados.

Resultados	• Al presionar el pulsador de avance, ambos motores giraron hacia adelante. • Al presionar el pulsador de reversa, los motores invirtieron su giro inmediatamente. • El pulsador de paro detuvo ambos motores deteniendo la alimentación de IN1–IN4. • La velocidad pudo ajustarse modificando el ancho del pulso PWM en los pines ENA y ENB. • Se comprobó que el driver L298N soporta cómodamente motores pequeños de 6–12 V.
Conclusión	• El L298N permite controlar uno o dos motores DC de manera eficiente gracias a su estructura de puente H. • La identificación correcta de entradas (pulsadores/joystick) y salidas (pines IN1–IN4 y ENA/ENB) es esencial para un funcionamiento adecuado. • El uso de INPUT_PULLUP simplifica la conexión de pulsadores, evitando el uso de resistencias externas. • El control de avance, paro y reversa se logra únicamente manipulando los estados lógicos de los pines del driver. • Se verificó que el Arduino por sí solo no puede manejar la corriente necesaria para los motores, por lo que el uso del driver es indispensable. • La práctica permitió comprender la importancia de los puentes H en aplicaciones de robótica y control de movimiento.
Ilustración	

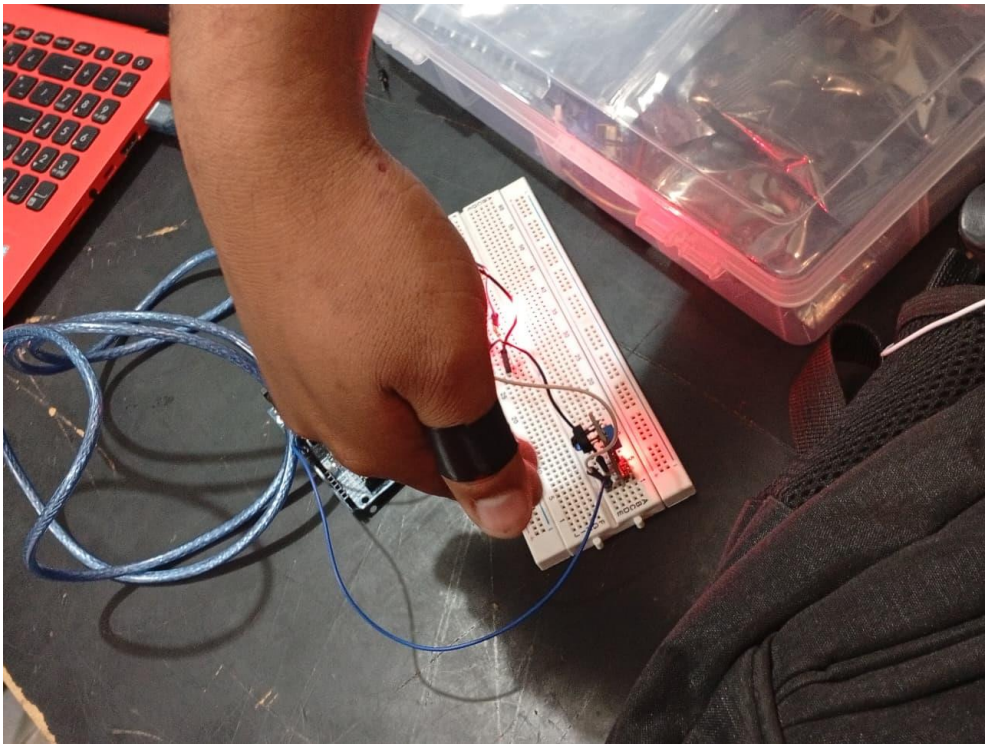
REPORTE DE PRÁCTICA	
ACTIVACIÓN DE DOS MOTORES DC CONTROLANDO AVANCE, PARO Y REVERSA MEDIANTE DRIVER L298N	
PRACTICA 14	
Objetivo	Identificar y configurar las entradas y salidas del microcontrolador Arduino UNO para controlar el movimiento de uno o dos motores de corriente continua (DC), utilizando un driver L298N como puente H. Se busca lograr el control de avance, paro y reversa mediante pulsadores, sensores o joystick, comprendiendo el funcionamiento del driver y la lógica de control de los motores.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • Driver L298N (puente H doble) • 1 o 2 motores DC • Pulsadores (push buttons) • Joystick analógico o sensor (opcional según actividad) • Fuente externa de 9–12 V para los motores • Cables de conexión • Protoboard **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Con una sola fuente** • Conectar la fuente al pin 12V (mismo voltaje que soporta el motor). • Dejar puesto el jumper que habilita el regulador a 5V. • No conectar nada al pin de 5V (ya tiene 5V internos), • Recomendado solo si la fuente es menor a 12V para evitar sobrecalentamiento. **Con dos fuentes** • Quitar el jumper del regulador. • Conectar 5V externos al pin de 5V (por ejemplo desde Arduino). • Conectar la fuente del motor al pin de 12V. **Control del módulo** • **MOTOR A**: ENA, IN1, IN2 → salidas OUT1 y OUT2 • **MOTOR B**: ENB, IN3, IN4 → salidas OUT3 y OUT4 • **ENA y ENB**: habilitan motores y permiten controlar la **velocidad (PWM)**. • Si no usarás PWM, coloca los **jumpers** para mantener los motores habilitados.

Resultados	• Al presionar el pulsador de avance, ambos motores giraron hacia adelante. • Al presionar el pulsador de reversa, los motores invirtieron su giro inmediatamente. • El pulsador de paro detuvo ambos motores deteniendo la alimentación de IN1–IN4. • La velocidad pudo ajustarse modificando el ancho del pulso PWM en los pines ENA y ENB. • Se comprobó que el driver L298N soporta cómodamente motores pequeños de 6–12 V.
Conclusión	• El L298N permite controlar uno o dos motores DC de manera eficiente gracias a su estructura de puente H. • La identificación correcta de entradas (pulsadores/joystick) y salidas (pines IN1–IN4 y ENA/ENB) es esencial para un funcionamiento adecuado. • El uso de INPUT_PULLUP simplifica la conexión de pulsadores, evitando el uso de resistencias externas. • El control de avance, paro y reversa se logra únicamente manipulando los estados lógicos de los pines del driver. • Se verificó que el Arduino por sí solo no puede manejar la corriente necesaria para los motores, por lo que el uso del driver es indispensable. • La práctica permitió comprender la importancia de los puentes H en aplicaciones de robótica y control de movimiento.
	•

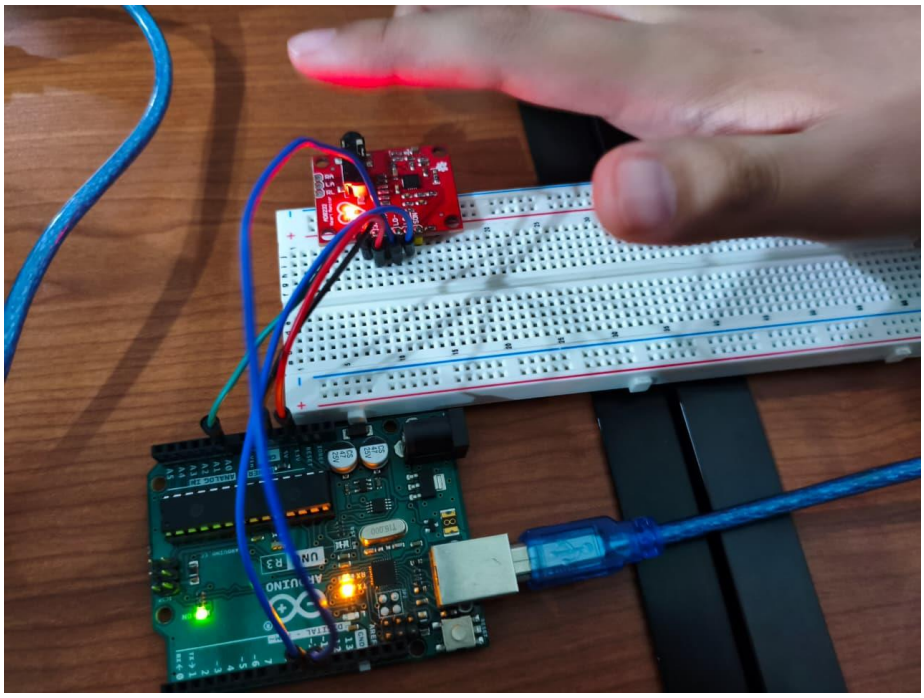
REPORTE DE PRÁCTICA	
ACTIVACIÓN DE DOS MOTORES DC CONTROLANDO AVANCE, PARO Y REVERSA MEDIANTE DRIVER L298N	
PRACTICA 15	
Objetivo	Controlar un motor a pasos (Stepper Motor) empleando un joystick analógico como dispositivo de entrada, identificando las señales de los ejes X y Y para generar movimiento en un sentido, otro, o detener el motor. Además, comprender las aplicaciones del joystick en control de movimiento y su integración con el Arduino.
Materiales Utilizados	**Hardware** • Tarjeta Arduino UNO • Driver L298N (puente H doble) • 1 o 2 motores DC • Pulsadores (push buttons) • Joystick analógico o sensor (opcional según actividad) • Fuente externa de 9–12 V para los motores • Cables de conexión • Protoboard **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Con una sola fuente** • Conectar la fuente al pin 12V (mismo voltaje que soporta el motor). • Dejar puesto el jumper que habilita el regulador a 5V. • No conectar nada al pin de 5V (ya tiene 5V internos), • Recomendado solo si la fuente es menor a 12V para evitar sobrecalentamiento. **Con dos fuentes** • Quitar el jumper del regulador. • Conectar 5V externos al pin de 5V (por ejemplo desde Arduino). • Conectar la fuente del motor al pin de 12V. **Control del módulo** • **MOTOR A**: ENA, IN1, IN2 → salidas OUT1 y OUT2 • **MOTOR B**: ENB, IN3, IN4 → salidas OUT3 y OUT4 • **ENA y ENB**: habilitan motores y permiten controlar la **velocidad (PWM)**. •
Resultados	• El motor paso a paso logró moverse suavemente en ambos sentidos según la lectura del joystick. • Al mover el eje X hacia la derecha, el motor giró en sentido horario.

	• Al mover el eje X hacia la izquierda, el motor giró en sentido antihorario. • Cuando el joystick quedó en posición central, el motor se mantuvo detenido. • El botón del joystick permitió ejecutar una acción adicional (regresar a posición inicial). • Debido a que Proteus no contenía un joystick físico, se simularon movimientos mediante pulsadores conectados a los pines analógicos, logrando reproducir los mismos efectos.
Conclusión	• El joystick analógico es un dispositivo muy versátil para controlar motores debido a que entrega señales analógicas y un pulsador digital. • La lectura de los ejes VRX y VRY permite determinar velocidad y dirección de forma precisa. • El motor paso a paso funciona con señales discretas, lo cual lo hace ideal para movimientos controlados y repetibles. • El uso del driver ULN2003 es indispensable, ya que el motor no puede ser alimentado directamente desde el Arduino. • La práctica permitió identificar claramente entradas (joystick) y salidas (secuencias del motor). • La simulación en Proteus puede adaptarse fácilmente usando pulsadores para emular el comportamiento del joystick cuando el modelo no está disponible.
	

REPORTE DE PRÁCTICA	
SENSOR INFRARROJO Y CONTROL DE SERVOMOTOR CON ARDUINO UNO	
PRACTICA 16	
Objetivo	• Identificar y configurar las entradas y salidas digitales del microcontrolador Arduino Uno. • Comprender el funcionamiento del sensor infrarrojo (IR) como dispositivo de detección. • Controlar un servomotor mediante la señal que genera el sensor IR al detectar la presencia de un objeto. • Analizar el comportamiento del sistema mediante la construcción de un circuito básico y la ejecución de un programa en Arduino.
Materiales Utilizados	**Hardware** • Tarjeta Arduino Uno • Sensor infrarrojo (módulo IR) • Servomotor • Resistencia de 220 Ω • Protoboard **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	**Sensor Infrarrojo (IR)** El sensor infrarrojo detecta la presencia de un objeto mediante **dos principios**: **Reflexión de luz IR:** • El emisor envía un haz infrarrojo invisible. • Si un objeto está cerca, la luz rebota y es captada por el receptor. • Esta variación provoca un cambio de nivel lógico (HIGH o LOW).. --- El sensor infrarrojo detecta la presencia de un objeto delante del emisor/ receptor. **Al detectar presencia:** • El sensor envía una señal al pin de entrada del Arduino. • El Arduino procesa la señal y activa el servomotor, moviéndolo a un ángulo determinado (por ejemplo 90°). • Se activa un LED indicador para mostrar que el sistema ha detectado presencia. **Cuando no se detecta objeto:** • El servomotor regresa a su posición inicial (0°). • El LED se apaga

Resultados	• El sensor infrarrojo logró detectar correctamente la presencia de objetos. • El servomotor respondió de manera inmediata al cambiar de ángulo cuando se detectó presencia. • El LED funcionó como indicador claro del estado del sensor. • El sistema completo operó sin problemas al suministrar 5V desde Ar
Conclusión	• El uso del sensor infrarrojo facilita la detección de objetos o personas mediante señal digital. • El servomotor se puede controlar eficientemente usando salidas PWM del Arduino. • La práctica permitió identificar claramente entradas y salidas del microcontrolador. • Se logró integrar un sistema de detección y respuesta que puede escalarse para aplicaciones de automatización o robótica. • El circuito es seguro, de bajo consumo y fácil de implementar.
Ilustración	

REPORTE DE PRÁCTICA	
CAPTURA DE SEÑALES BIOPOTENCIALES CON SENSOR ECG AS8232	
PRACTICA 17	
Objetivo	Aprender el funcionamiento, conexión, ajuste y lectura del sensor ECG AS8232 para la detección de señales biopotenciales, así como comprender los principios de amplificación, filtrado y digitalización de señales fisiológicas para fines didácticos.
Materiales Utilizados	**Hardware** • Módulo sensor ECG **AS8232** • 3 electrodos adhesivos para biopotenciales • Cables Dupont hembra–hembra • Arduino UNO/Nano • Cable USB • Protoboard (opcional) **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	La alimentación del sensor AS8232 se conecta al pin de 3.3V del Arduino, mientras que la tierra (GND) se conecta al GND del Arduino. La salida de señal ECG (pin OUT) debe conectarse a la entrada analógica A0 del Arduino. El pin SDN, que controla el modo de apagado, debe conectarse también a 3.3V para mantener el sensor activado. Finalmente, los pines LO+ y LO−, que sirven para la detección de electrodos desconectados (Lead-Off), pueden conectarse opcionalmente a pines digitales del Arduino si se desea monitorear esa función.
Resultados	Al ejecutar el código y visualizar la señal en el monitor serial o en el Serial Plotter se obtiene una gráfica que representa el ritmo cardíaco del usuario. • Ondas repetitivas con el patrón P–QRS–T • Oscilación dentro de valores ~ 350–650 (nivel ADC) • Señal con picos bien definidos después del ajuste de ganancia
Conclusión	• El sensor ECG AS8232 es una herramienta útil para comprender el funcionamiento básico de un electrocardiograma. • Permite observar señales biopotenciales reales en tiempo real mediante Arduino. • Requiere alimentación de 3.3 V, correcto ajuste de ganancia y colocación adecuada de electrodos. • Su uso es exclusivamente didáctico, no médico.

Ilustración	

REPORTE DE PRÁCTICA	
MÓDULO DETECTOR DE RITMO CARDÍACO Y ACTIVACIÓN DE SERVOMOTOR CON ARDUINO	
PRACTICA 18	
Objetivo	Implementar un sistema básico de medición de ritmo cardíaco utilizando un sensor fotoeléctrico de pulsaciones y controlar un servomotor a partir de la detección del pulso, aplicando conceptos de electrónica, sensórica y programación con Arduino
Materiales Utilizados	**Hardware** • Arduino Uno • Módulo detector de ritmo cardíaco (sensor PPG) • Cables Dupont • Servomotor SG90 o similar • Protoboard • Cable USB **Software** • Arduino IDE (para programación del microcontrolador) • Proteus
Actividades Realizadas	El sensor de pulso cuenta con tres pines principales: el pin VCC, que se conecta a la alimentación de 5V o 3.3V del Arduino según lo indique la hoja de datos; el pin GND, que debe ir a GND del Arduino; y el pin OUT o Signal, que se conecta a la entrada analógica A0 para leer la señal del pulso. Es importante revisar el datasheet para asegurarse de que el voltaje de operación sea compatible con el Arduino. Para la conexión del servomotor, el cable rojo se conecta al pin de 5V del Arduino, el cable café o negro se conecta a GND, y el cable amarillo o blanco, que corresponde a la señal de control, debe conectarse al pin digital 9, que permite enviar la señal PWM necesaria para mover el servo.
Resultados	**Al correr el programa y colocar el dedo en el sensor:** • La señal analógica muestra picos regulares cada latido. • La gráfica en Serial Plotter permite visualizar claramente las pulsaciones. • El servomotor responde moviéndose en cada pico detectado. • La frecuencia cardiaca aproximada puede calcularse midiendo el tiempo entre picos. **Observaciones típicas:** • Si hay ruido, puede ser por movimiento del dedo. • Señales muy bajas pueden indicar mala colocación.

	• Servomotor puede necesitar fuente externa si consume demasiado.
Conclusión	• El sensor de ritmo cardíaco permite captar pulsaciones utilizando la técnica fotoeléctrica PPG. • Arduino procesa adecuadamente los datos, permitiendo visualizar la señal en tiempo real. • La integración con un servomotor permite generar respuestas físicas a un evento biológico (latido). • El sistema es útil para entender conceptos de sensórica biomédica, aunque no debe usarse para fines clínicos.

Google Classroom

USO DE SOFTWARE Y SIMULACIONES

Presentación de software PROTEUS

Entrega los archivos generados en el software proteus

Excelente	Medio	Suficiente	No cumple
15 puntos	*10 puntos*	*5 puntos*	*0 puntos*
El alumno entrega el archivo correspondiente a cada circuito solicitado en las guías de prácticas. En total son 20 archivos funcionales. Los archivos tienen	El alumno entrega el archivo correspondiente a cada circuito solicitado en las guías de prácticas. En total son un máximo de 15 archivos funcionales. Los archivos tienen	El alumno entrega el archivo correspondiente a cada circuito solicitado en las guías de prácticas. En total son un máximo de 10 archivos medianamente	El alumno entrega 5 o menos archivos medianamente funcionales.

Simulación de código en IDE ARDUINO

El alumno entrega los archivos generados en el IDE ARDUINO (sketch))

Excelente	Medio	Suficiente	No cumple
15 puntos	*10 puntos*	*5 puntos*	*0 puntos*
El alumno entrega los archivos generados en el IDE ARDUINO. Presenta el código que	El alumno entrega los archivos generados en el IDE ARDUINO. Presenta el código que	El alumno entrega los archivos generados en el IDE ARDUINO. Presenta el código que	El alumno no entrega el material solicitado o es menos de 5 archivos.

U2. PROGRAMACIÓN DE ENTRADAS Y SALIDAS DEL MICROCONTROLADOR.
MICROCONTROLADORES 2025 702 B

Cuentan las asistencias al laboratorio realizando las prácticas que se mencionan a continuación:

Realizaron las prácticas pero la núm. 3.1.6 no la completaron, por lo tanto en la lista de cotejo del reporte de prácticas se descuentan 2 puntos.

1.1 Puerto configurado como salida (ENCENDER - APAGAR LED)

1.2 Puerto configurado como entrada (ENCENDER –APAGAR LED SWITCH ANALÓGICO)

1.3 Puerto configurado como PWM

1.4 CONEXIÓN DISPLAY DE 7 SEGMENTOS+MONITORSERIE

1.5 PUERTO CONFIGURADO COMO SALIDA. Uso de display LCD

1.5.1 LCD muestra caracteres

1.5.2 Despliega un texto como saludo

1.5.3 Despliega texto con desplazamiento

1.5.4 Uso de sensor ultrasónico

2.1 MATRIZ LED 8X8

2.1.1 RECONOCIMIENTO DE CADA LED DE LA MATRIZ

2.1.2 MUESTRA TEXTO EN DESPLAZAMIENTO

2.1.3 NUMERACIÓN DEL 0 AL 9

2.1.4 MUESTRA UNA FLECHA

3.1 SERVOMOTRES

3.1.1 SERVOMOTOR AVANCE, REVERSA EMPLEANDO PUSH BUTTOM

3.1.2 SERVOMOTOR ACTIVADO MEDIANTE SENSOR ULTRASÓNICO

3.1.3 DOS SERVOMOTORES ACTIVADOS MEDIANTE POTENCIÓMETRO MOTOR CD A 12 V

3.1.4 ACTIVACIÓN EMPLEANDO CONTROLADOR O DRIVER L298N

3.1.5 ACTIVACIÓN DE DOS MOTORES DE 12 V EN CD EN AVANCE, PARO Y REVERSA **led**

3.1.6 ACTIVACIÓN DE MOTOR EMPLEANDO SENSOR DE TEMPERATURA LM35 MOTOR DE PASOS O STEPPER **solo sensor

3.1.7 CONTROL DE MOTOR A PASOS EMPLEANDO JOYSTICK **led, servomotor

- **CONTROL DE MOTOR A PASO EMPLEANDO PUENTE H (L298N)**

Falta evidencia de la practica 18.

40

José Armando Cabrera Echavarría 23/09/2025
Angel Abraham Gonzales
Erick candelario fiscal Ambros

Realizada en protoboard.

Practica 1 configuración de entradas y salidas (puertos)
Falta la simulación → Realizada con éxito 24/09/25

Realizó activación de LED con switch 24/09/25
en Protoboard con éxito

Evaluación
24 SEP 2025
ITSSAT ITSSAT
REVISADO

- Display 7 segmentos realizado en Proteus
  - Contador 0 a 9
  - Comunicación serial → Escribe en el monitor numérico y despliega display
- Armaron el circuito en protoboard, funcionando
  Erick Candelario
  Armando Echavarría 06/10/2025
  Angel enfermo → justificar

09/10/25 Simulación en Proteus: Mensaje de Texto, Display LCD 16x2 - Sensor Ultrasónico.
~~Falta código de caracteres~~ completo
~~Falta en protoboard~~ completo

Evaluación
09 OCT 2025
ITSSAT ITSSAT
REVISADO

Presentaron Matriz Led 8x8
en Proteus funcionando
" protoboard funcionando
→ Texto, reconocimiento de leds
Falta contador 0 a 9

06/11/2025 Práctica Matriz 8x8 contador de 0 a 9
- Diagrama en Proteus realizado
- Montaje en protoboard funcionando
→ Se equivocaron al conectar los pines las primeras pruebas
Armando, Ángel y Erick Conde

Servomotor controlado por pushbottom
- Proteus → Simulación realizada funcionando
- Montaje en protoboard funcionando
Armando, Ángel, Erick

12/NOV/2025 Práctica - control de Servomotor empleando Sensor Ultrasónico.
- Práctica en Proteus funcionando Simulación
- Montaje en protoboard funcionando
Ángel, Erick, Armando

12/Nov/2025 Práctica - control servomotor empleando potenciómetros (2 pzas)
- Práctica realizada en Proteus
- Montaje en Protoboard - funcionando
Ángel, Erick, Armando

13/NOV/2025 2 Motores CD
- Simulación en Proteus (pendiente)
- Montaje en Protoboard - funcionando
Erick, Armando, Ángel

U2

Sensor de temperatura Módulo DHT11 26/NOV/2025
- Simulación en Proteus → funcionando
- Elaboración en Protoboard, funcionando
- Comunicación serie
Armando, Candelario, Ángel

Sensor infrarrojo (fotodiodo, detector de obstáculos) → Protoboard 26/11/2025
Funcionando
Ángel, Erick, Armando

* Uso de sensor ECG - ritmo cardiaco 26/11/2025
- Elaboración/Montaje en Protoboard
- Trasmisión serie
- Monitor serie
- Gráfica en herramienta plotter (NO)
Ángel, Erick, Armando

Sensor ritmo cardiaco (fotodiodo)
- Control de servomotor
- Montaje en Protoboard (funcionando) 26/NOV/25
Armando, Ángel, Erick

Uso de Joystick - Montaje en Protoboard
- Uso de leds / NO usaron motores Y2
Armando, Ángel, Erick