INSTITUTO TECNOLÓGICO SUPERIOR DE SAN ANDRÉS TUXTLA

DIVISIÓN DE INGENIERIA MECATRÓNICA

INSTRUMENTACIÓN VIRTUAL

CATEDRATICO

DR. JOSÉ ÁNGEL NIEVES VÁSQUEZ

GRUPO 711-A

FECHA DE ENTREGA: 14/11/2024

PERIODO: AGOSTO 2024 – DICIEMBRE 2024

EVIDENCIAS UNIAD 4

ALUMNO

EDMUNDO LÓPEZ HERNÁNDEZ

D 15 M 10 A 2024

4.1 ACONDICIONAMIENTO DE SEÑAL UTILIZANDO AMPLIFICADORES OPERACIONALES

Este es un proceso clave en la instrumentación, donde se preparan señales eléctricas provenientes de sensores o transductores para que sean compatibles con los sistemas de adquisición de datos o control.
Los amplificadores operacionales (op-amps) son componentes esenciales en este proceso, utilizados en diversas configuraciones para modificar o mejorar estas señales.

Amplificación: Los op-amps pueden amplificar señales débiles provenientes de sensores, lo que mejora la resolución y precisión de la medición.

Filtrado: se emplean en filtros activos (pasa-bajo, pasa-alto, pasa-banda) para eliminar el ruido y frecuencias no deseadas.

Aislamiento: Funciona como seguidor de voltaje (buffer) para separar diferentes etapas del circuito.

14/11/24

TEMA 4.2 FILTROS DE SEÑAL

Son circuitos que permiten modificar una señal o atenuado ciertas frecuencias mientras permiten que otras pasen, según los requisitos de la aplicación. Dentro de estos existen dos tipos de filtros.

Filtros Analógicos

- Se implementan utilizando componentes pasivos como (resistencias, capacitores o inductores).
- Funcionan directamente en señales analógicas.

Filtros Comunes

- Filtro pasa-bajo: Permite el paso de frecuencias por debajo de una frecuencia de corte determinado.
- Filtro Pasa-Alto: Permite el paso de frecuencias de corte determinada.

Filtros Digitales

- Se implementan mediante algoritmos o software, operando sobre señales digitales.
- Tienen mayor flexibilidad y pueden adaptarse a las necesidades específicas del sistema de adquisición de datos.

Aplicaciones:

- Separación De Señales: Se usan para extraer una señal de interes dentro de un rango específico de frecuencias.
- Mejora la Calidad de señal: Al eliminar interferencias o ruido, se mejora la precisión en sistemas de medición.

TEMA 4.3 CARACTERISTICAS DE LA CONVERSION ANALÓGICA-DIGITAL

Este es el proceso de transformar una señal analógica continua en una señal digital discreta.
Esta conversión es crucial para que las computadoras y sistemas digitales pueden procesar señales del mundo real, como temperatura, sonido o presión.

Resolución: Se refiere al número de bits utilizados para representar la señal digital. Una mayor resolución permite representar la señal con más precisión.

Rango De Entrada: Es el intervalo de voltaje que el adc puede aceptar y convertir. Normalmente es especificado como un rango de voltajes.

Linealidad: Indica que tan precisa es la relación entre la entrada analógica y la salida digital. Un ADC lineal convierte la señal de forma proporcional a su entrada en todo el rango de operación.

Ruido: El ruido afecta la capacidad del ADC para realizar conversiónes precisas.
Algunos ADC tienen caracteristicas especiales para minimizar el impacto del ruido de las mediciones.

TEMA 4.2: Adquisicion De Datos

Esto se refiere al proceso de recuperar información de un sistema o fenomeno fisico utilizando sensores, dispositivos de medición y equipos electronicos para analizar y tomar decisiones basadas en esta información

Este proceso Implica:

Sensores: Capturan las señales fisicas (Temperatura y presión)

Transductores: Convierten esas señales fisicas en señales eléctricas.

Sistema de adquisición de datos (DAQ): Un dispositivo que digitaliza las señales electricas para que puedan ser procesadas por una Computadora.

Software: Programas que permiten la Visalización, almacenamiento y analisis de datos adquiridos.

La adquisición de datos es el proceso de recopilar y digitalizar señales fisicas mediante sensores y transductores. Este sistema toma muestra de las señales e intervalos irregulares, las procesa con una resolución especifica y las transmite a una Computadora para su analisis y almacenamiento.

TEMA 4.5: Adquisición de datos Digidates

Este es el proceso de recoger, procesar y almacenar la información que ya que esta en formato digital.
A diferencia de la adquisición de datos analógicos, donde las señales físicas se convierten de digitales.

Características:

Datos discretos: Son valores binarios o secuencias de números digitales, no continuos como señales analógicas.

Velocidad de Captura: Los sistemas pueden operar a altas velocidades de muestreo, lo que permite grabar grandes cantidades de información.

Interfaces: Los datos son capturados y transmitidos mediante interfaces como USB, CAN bus.

Almacenamiento Eficiente: Los datos digitales pueden ser fácilmente almacenados y comprimidos para un análisis posterior.

Aplicaciones en automatización: Se utilizan ampliamente en sistemas automatizados de monitoreo y control, como en redes de sensores y dispositivos IoT.

9

GUIA DE OBSERVACIÓN EXPOSICIÓN

Instrumentación Virtual.

Nombre del estudiante: López Hernández Edmundo.

Tema: Adquisición de Datos.

Explicación	5 %	5 %
Dominio del tema	10 %	10 %
Presentación en tiempo y forma	5 %	5 %
Total	20 %	20 %

LISTA DE COTEJO INVESTIGACIÓN

INSTRUMENTACION VIRTUAL.

Nombre del estudiante: López Hernández Edmundo.

Tema: Adquisición de datos.

Portada	5 %	3 %
Desarrollo y claridad	10 %	10 %
Entrega en tiempo y forma	5 %	5 %
Total	20 %	18 %

LISTA DE COTEJO DE PROYECTO.

Nombre del estudiante: López Hernández Edmundo.

Tema: Detección de nivel utilizando Labview.

Portada	5 %	3 %
Introducción	10 %	10 %
Desarrollo y explicación	30 %	28 %
Conclusiones	5 %	5 %
Referencias	5 %	0 %
Entrega en tiempo y forma	5 %	5 %
Total	60 %	50 %

INSTITUTO TECNOLÓGICO SUPERIOR DE SAN ANDRÉS TUXTLA

DIVISIÓN DE INGENIERIA MECATRÓNICA

INSTRUMENTACIÓN VIRTUAL

CATEDRATICO

DR. JOSÉ ÁNGEL NIEVES VÁSQUEZ

GRUPO 711-A

FECHA DE ENTREGA: 02/12/2024

PERIODO: SEPTIEMBRE 2024 – DICIEMBRE 2024

PROYECTO FINAL LABVIEW

ALUMNOS:

LÓPEZ HERNÁNDEZ EDMUNDO

MORALES AZAMAR ZAIRA ITZEL

HERNANDEZ FLORES MIGUEL ANGEL

PATIÑO BARRIOS JOSE LUIS

GONZALES CAZANOVA JADEN

OBJETIVO, MISION Y VISION

Objetivo del proyecto: El objetivo de este proyecto es desarrollar un sistema automatizado para medir el nivel de un tanque utilizando un sensor conectado a un microcontrolador Arduino. Los datos obtenidos serán procesados y graficados en tiempo real mediante un programa en Python, lo que permitirá un monitoreo preciso y eficiente del nivel del tanque.

Misión del proyecto: La misión de este proyecto es proporcionar una solución accesible y confiable para la medición continua del nivel de tanques en diversos sistemas. Utilizando la plataforma Arduino y Python, se busca optimizar la visualización y análisis de datos, facilitando la toma de decisiones y la implementación de sistemas automatizados de control.

Visión del proyecto: Ser un referente en el desarrollo de soluciones de monitoreo y control de niveles de líquidos, ofreciendo sistemas accesibles, precisos y fáciles de usar. A través de la integración de tecnología de vanguardia como Arduino y Python, buscamos mejorar la eficiencia operativa en sectores industriales y domésticos, contribuyendo a la automatización inteligente y la toma de decisiones informadas basadas en datos precisos y en tiempo real.

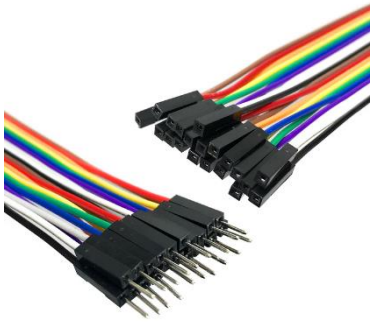
INTRODUCCIÓN

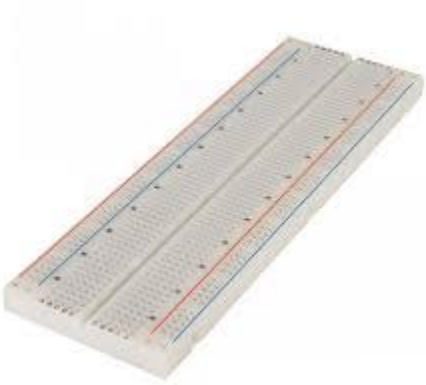
Introducción: En la actualidad, la automatización de procesos industriales y domésticos es un área de creciente importancia. Uno de los aspectos clave en muchos de estos procesos es la medición precisa y continua de los niveles de líquidos en tanques. Este proyecto propone una solución basada en Arduino, donde un sensor de nivel detecta la cantidad de líquido en el tanque, y los datos son enviados a una computadora para su procesamiento. Utilizando Python, se crea una interfaz gráfica que muestra en tiempo real los resultados de la medición, permitiendo al usuario monitorizar el nivel de manera sencilla y eficaz. Este sistema no solo mejora la precisión en la medición, sino que también brinda la posibilidad de realizar ajustes y tomar decisiones informadas basadas en los datos presentados.

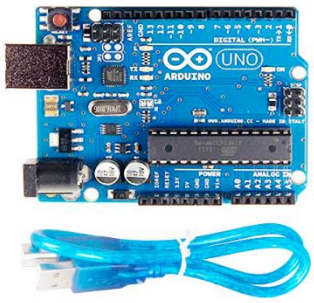
MATERIALES

Para La Construcción De Este Proyecto Fueron Utilizados Los Siguientes Componentes:

- Sensor Ultrasónico HSR-04.
- Cables Para Conexión.
- Placa De Arduino Uno
- Placa De Pruebas
- Software Arduino
- Software De Programación Anaconda
- Leds De Distintos Colores







ANACONDA®

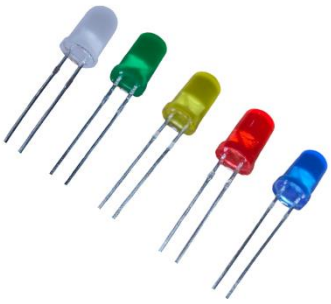
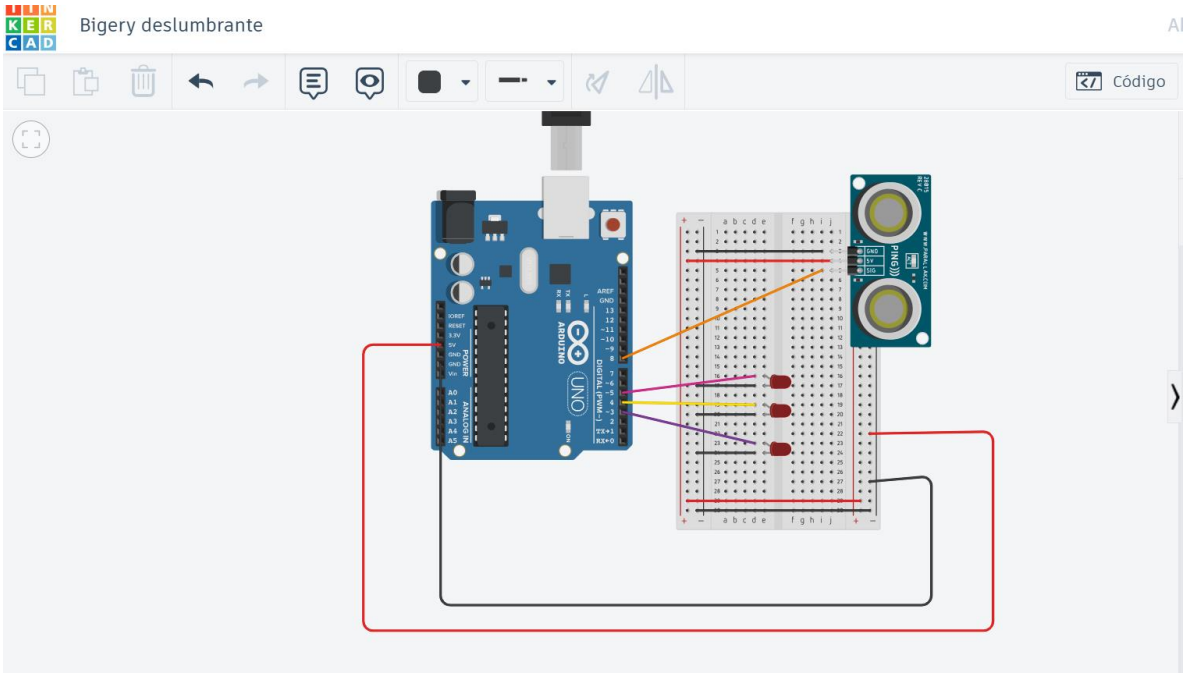


DIAGRAMA DE CONEXIONES



La imagen muestra una conexión de un Arduino Uno con un sensor ultrasónico HC-SR04 y algunos LEDs conectados a una placa de pruebas.

1. **Arduino Uno**:
    - **GND (Tierra)**: Conectado al riel de tierra de la placa de pruebas (breadboard).
    - **Pin 13**: Conectado al cátodo de un LED (con un resistor en serie) para indicar un estado, como un resultado de medición.
    - **Pin 12**: Conectado a otro LED (también con un resistor en serie).
    - **Pin 11**: Conectado a otro LED, para mostrar diferentes indicadores.
    - **Pin 10**: Conectado al pin de "Trigger" del sensor ultrasónico HC-SR04.

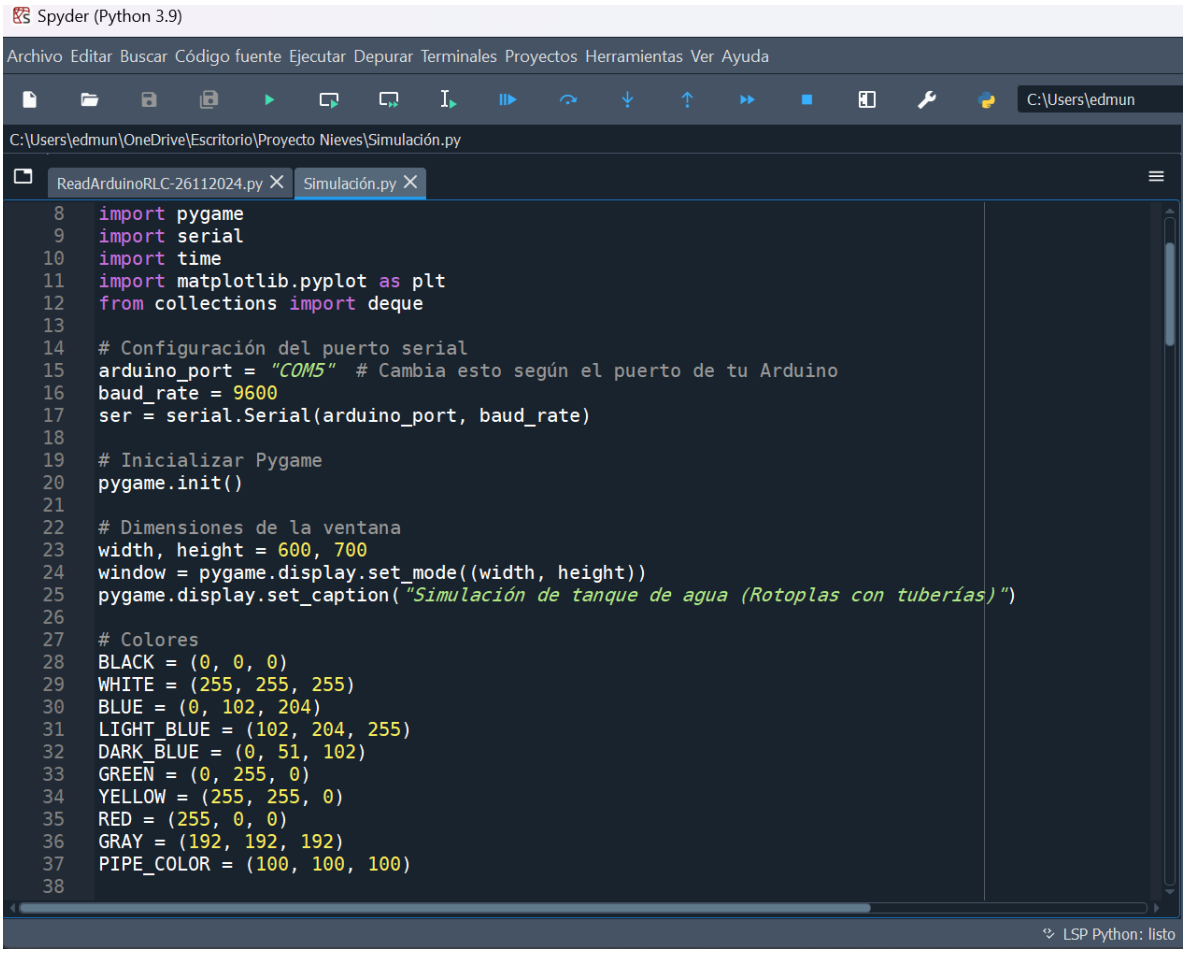
2. **Sensor Ultrasónico HC-SR04**:
    - **VCC**: Conectado al riel de 5V de la placa de pruebas.
    - **GND**: Conectado al riel de tierra de la placa de pruebas.
    - **Pin Trigger (TRIG)**: Conectado al pin 10 del Arduino, utilizado para iniciar la medición del tiempo del pulso del ultrasonido.
    - **Pin Echo (ECHO)**: Conectado al pin 9 del Arduino para recibir el pulso reflejado del ultrasonido y calcular la distancia.
3. **Leds**:
    - Los LEDs están conectados a los pines digitales del Arduino (probablemente 11, 12 y 13), con resistores en serie para limitar la corriente.
    - Cada LED puede representar un estado o indicador del sistema, como la activación del sensor o la distancia medida.

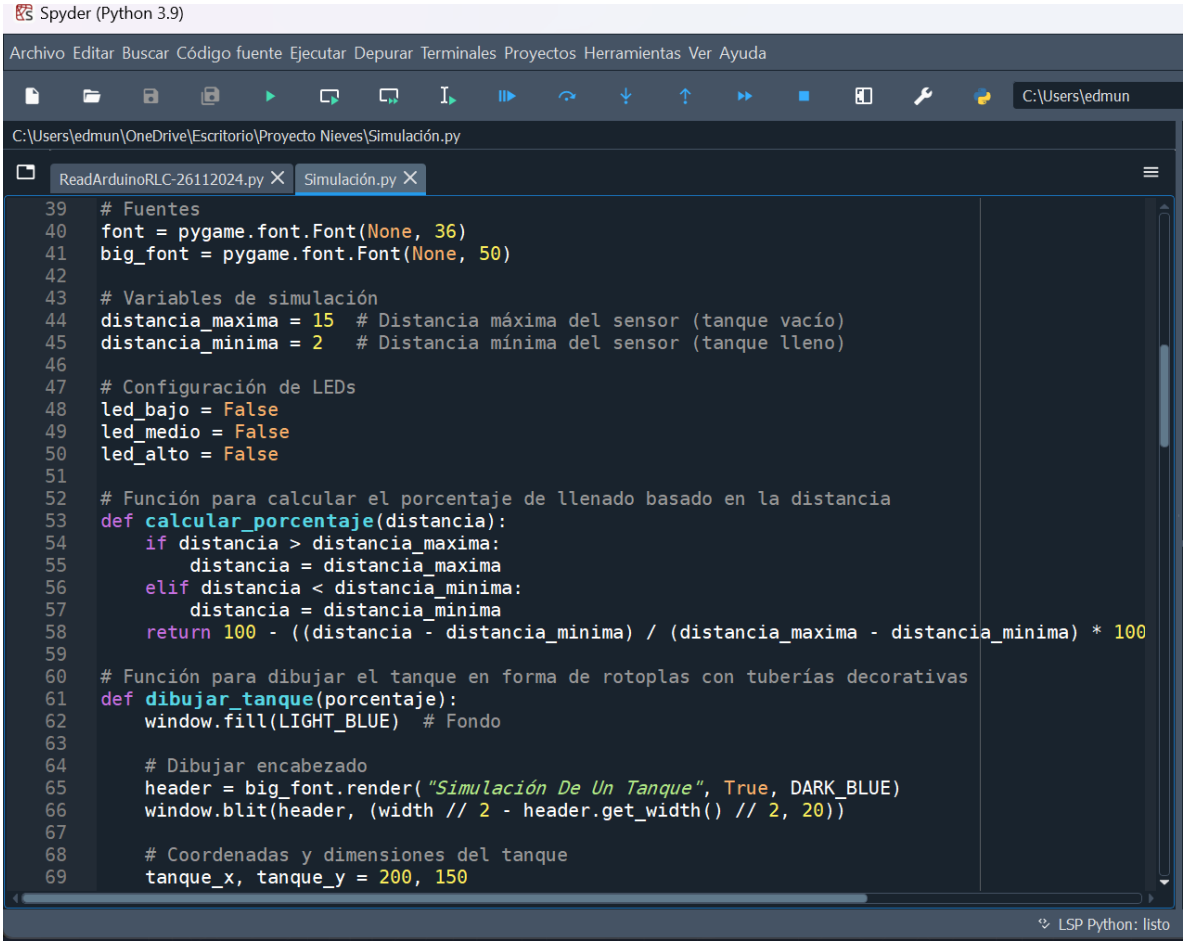
CÓDIGO DE ARDUINO

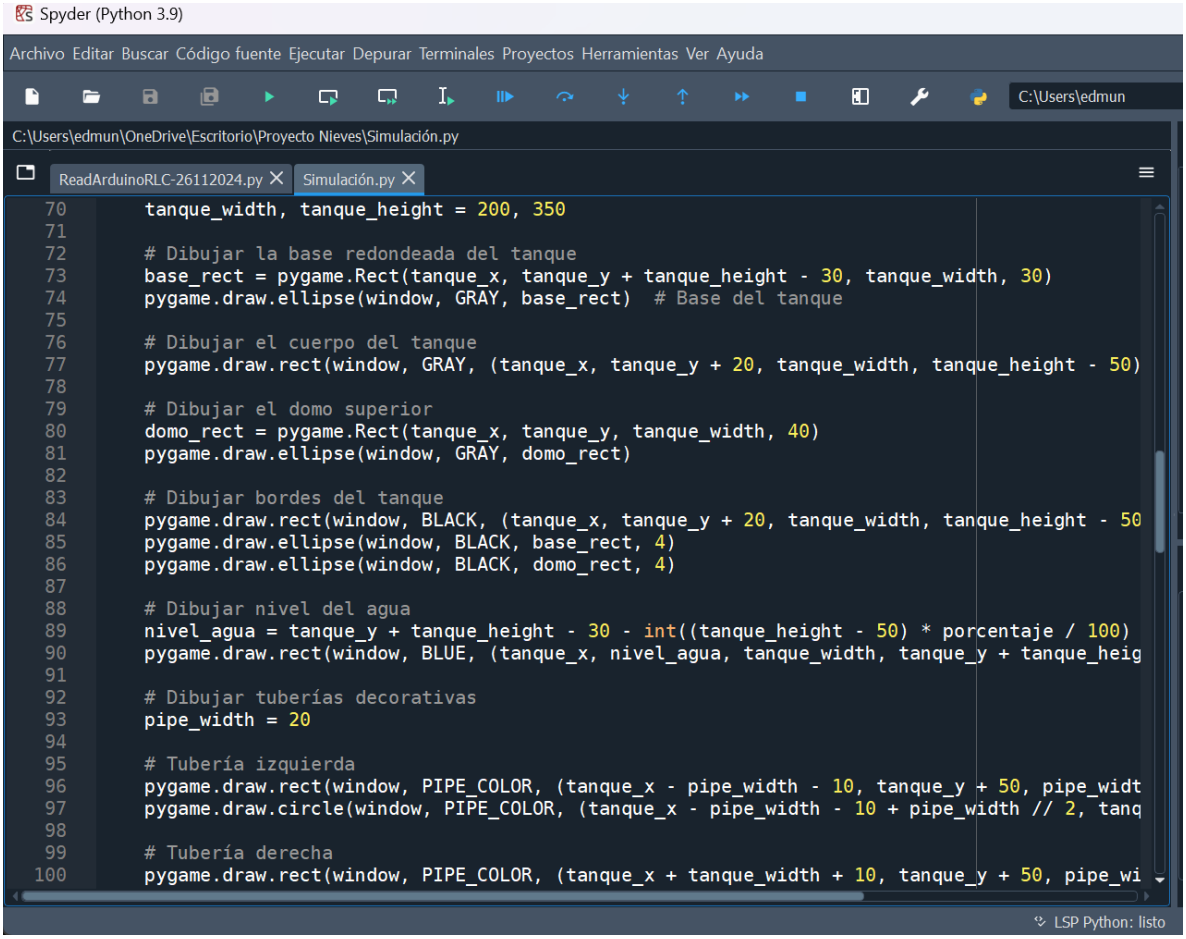
Este código en Arduino está diseñado para trabajar con un sensor ultrasónico HC-SR04 y tres LEDs. El sensor mide distancias y activa diferentes LEDs dependiendo de la distancia detectada. A continuación, una descripción general de las partes del código:

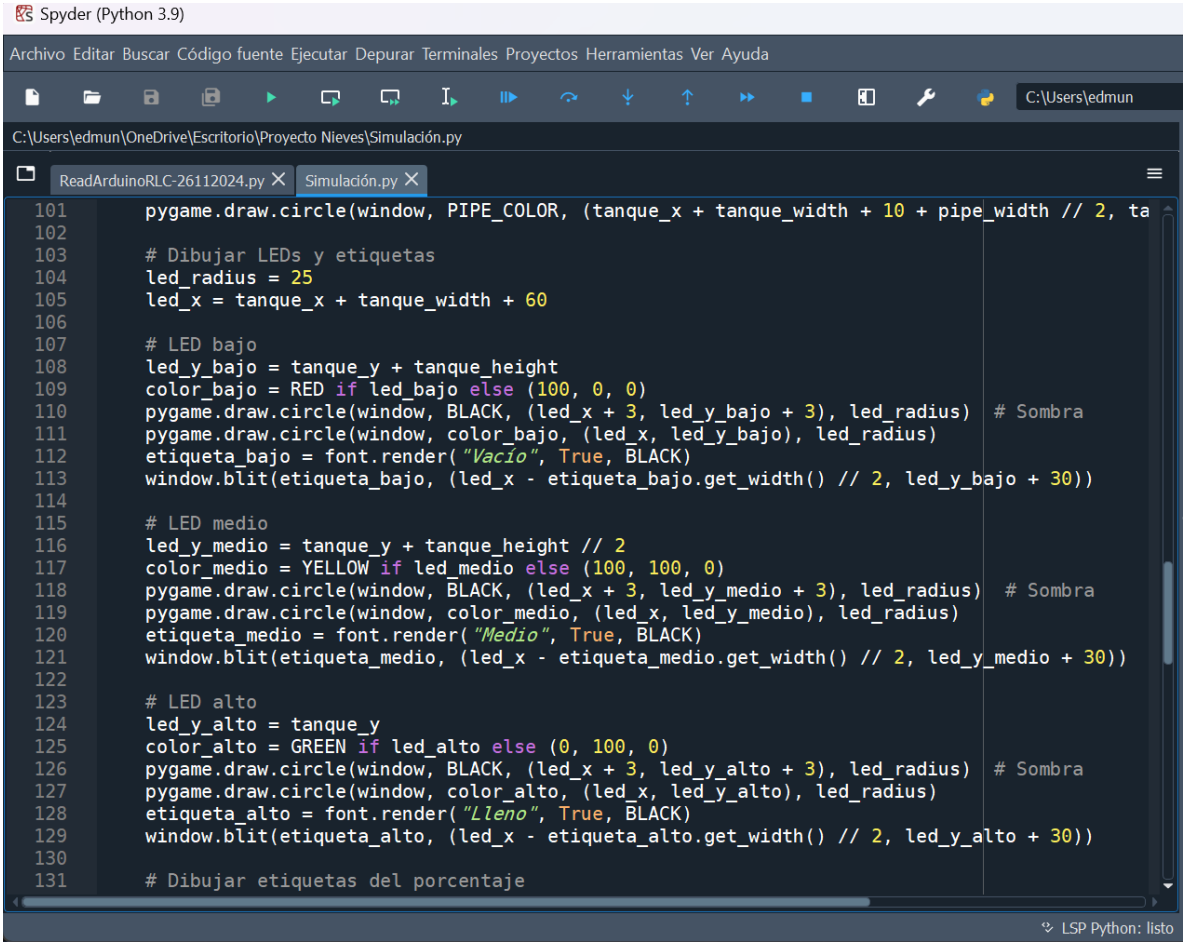
1. **Definición De Pines**:
    - Se definen los pines de conexión del sensor ultrasónico: trigPin (pin 9) y echoPin (pin 10).
    - También se definen los pines de los LEDs: ledCorto (pin 3), ledMedio (pin 4) y ledLargo (pin 5). Estos LEDs indican la distancia detectada: corto, medio y largo.
2. **Configuración En Setup ()**:
    - Los pines del sensor ultrasónico (trigPin y echoPin) se configuran como salida y entrada, respectivamente.
    - Los pines de los LEDs se configuran como salida.
    - Se inicializa la comunicación serial a 9600 baudios, lo cual permite la visualización de los datos en el monitor serial de Arduino.

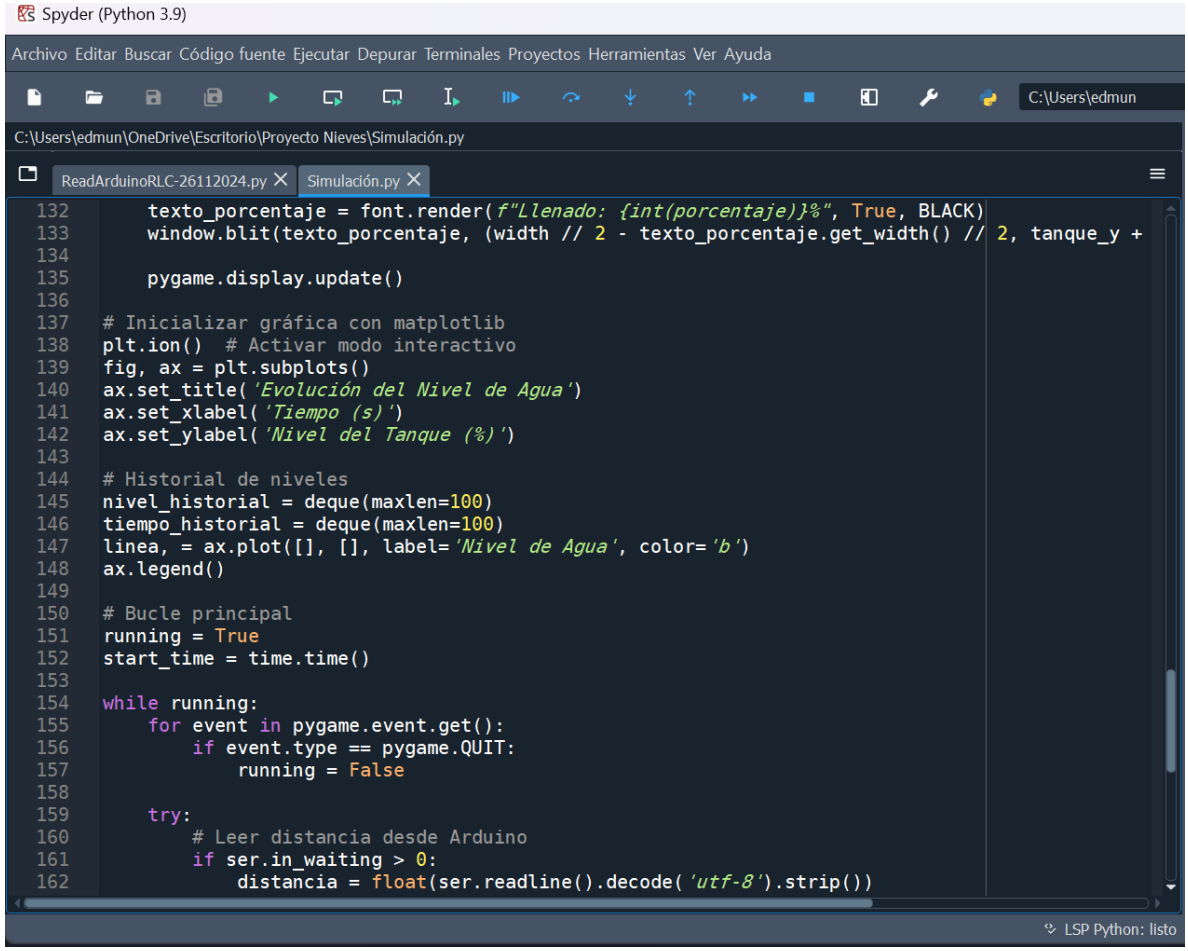
CÓDIGO DE PYTHON PARA LA SIMULACIÓN

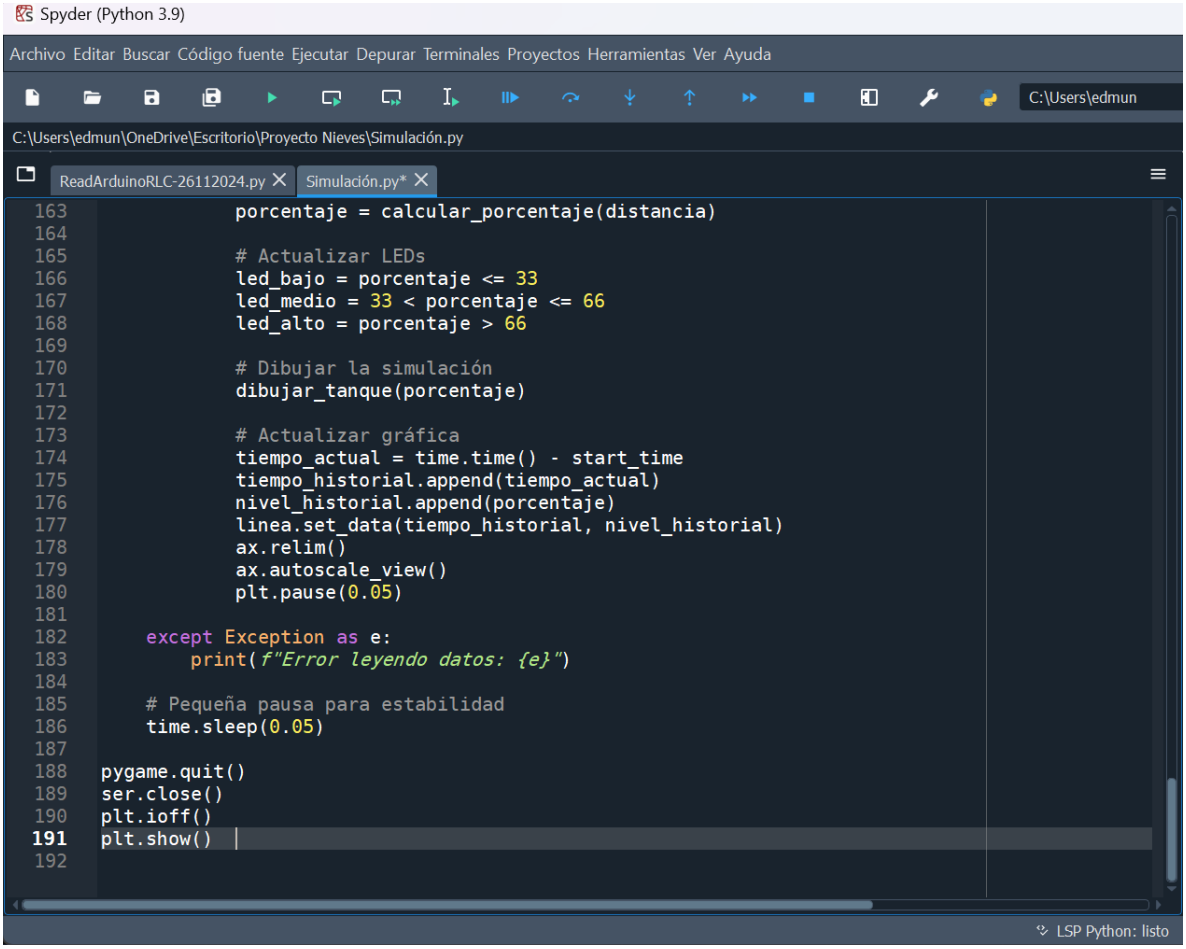



Spyder (Python 3.9)
Archivo Editar Buscar Código fuente Ejecutar Depurar Terminales Proyectos Herramientas Ver Ayuda
C:\Users\edmun
C:\Users\edmun\OneDrive\Escritorio\Proyecto Nieves\Simulación.py
ReadArduinoRLC-26112024.py
Simulación.py
39 # Fuentes
40 font = pygame.font.Font(None, 36)
41 big_font = pygame.font.Font(None, 50)
42
43 # Variables de simulación
44 distancia_maxima = 15 # Distancia máxima del sensor (tanque vacío)
45 distancia_minima = 2   # Distancia mínima del sensor (tanque lleno)
46
47 # Configuración de LEDs
48 led_bajo = False
49 led_medio = False
50 led_alto = False
51
52 # Función para calcular el porcentaje de llenado basado en la distancia
53 def calcular_porcentaje(distancia):
54   if distancia > distancia_maxima:
55   distancia = distancia_maxima
56   elif distancia < distancia_minima:
57   distancia = distancia_minima
58   return 100 - ((distancia - distancia_minima) / (distancia_maxima - distancia_minima) * 100
59
60 # Función para dibujar el tanque en forma de rotoplas con tuberías decorativas
61 def dibujar_tanque(porcentaje):
62   window.fill(LIGHT_BLUE) # Fondo
63
64   # Dibujar encabezado
65   header = big_font.render("Simulación De Un Tanque", True, DARK_BLUE)
66   window.blit(header, (width // 2 - header.get_width() // 2, 20))
67
68   # Coordenadas y dimensiones del tanque
69   tanque_x, tanque_y = 200, 150
LSP Python: listo


Spyder (Python 3.9)
Archivo Editar Buscar Código fuente Ejecutar Depurar Terminales Proyectos Herramientas Ver Ayuda
C:\Users\edmun
C:\Users\edmun\OneDrive\Escritorio\Proyecto Nieves\Simulación.py
ReadArduinoRLC-26112024.py
Simulación.py
70 tanque_width, tanque_height = 200, 350
71
72 # Dibujar la base redondeada del tanque
73 base_rect = pygame.Rect(tanque_x, tanque_y + tanque_height - 30, tanque_width, 30)
74 pygame.draw.ellipse(window, GRAY, base_rect) # Base del tanque
75
76 # Dibujar el cuerpo del tanque
77 pygame.draw.rect(window, GRAY, (tanque_x, tanque_y + 20, tanque_width, tanque_height - 50)
78
79 # Dibujar el domo superior
80 domo_rect = pygame.Rect(tanque_x, tanque_y, tanque_width, 40)
81 pygame.draw.ellipse(window, GRAY, domo_rect)
82
83 # Dibujar bordes del tanque
84 pygame.draw.rect(window, BLACK, (tanque_x, tanque_y + 20, tanque_width, tanque_height - 50
85 pygame.draw.ellipse(window, BLACK, base_rect, 4)
86 pygame.draw.ellipse(window, BLACK, domo_rect, 4)
87
88 # Dibujar nivel del agua
89 nivel_agua = tanque_y + tanque_height - 30 - int((tanque_height - 50) * porcentaje / 100)
90 pygame.draw.rect(window, BLUE, (tanque_x, nivel_agua, tanque_width, tanque_y + tanque_heig
91
92 # Dibujar tuberías decorativas
93 pipe_width = 20
94
95 # Tubería izquierda
96 pygame.draw.rect(window, PIPE_COLOR, (tanque_x - pipe_width - 10, tanque_y + 50, pipe_widt
97 pygame.draw.circle(window, PIPE_COLOR, (tanque_x - pipe_width - 10 + pipe_width // 2, tanq
98
99 # Tubería derecha
100 pygame.draw.rect(window, PIPE_COLOR, (tanque_x + tanque_width + 10, tanque_y + 50, pipe_wi
LSP Python: listo


Spyder (Python 3.9)
Archivo Editar Buscar Código fuente Ejecutar Depurar Terminales Proyectos Herramientas Ver Ayuda
C:\Users\edmun
C:\Users\edmun\OneDrive\Escritorio\Proyecto Nieves\Simulación.py
ReadArduinoRLC-26112024.py
Simulación.py
101 pygame.draw.circle(window, PIPE_COLOR, (tanque_x + tanque_width + 10 + pipe_width // 2, ta
102
103 # Dibujar LEDs y etiquetas
104 led_radius = 25
105 led_x = tanque_x + tanque_width + 60
106
107 # LED bajo
108 led_y_bajo = tanque_y + tanque_height
109 color_bajo = RED if led_bajo else (100, 0, 0)
110 pygame.draw.circle(window, BLACK, (led_x + 3, led_y_bajo + 3), led_radius) # Sombra
111 pygame.draw.circle(window, color_bajo, (led_x, led_y_bajo), led_radius)
112 etiqueta_bajo = font.render("Vacío", True, BLACK)
113 window.blit(etiqueta_bajo, (led_x - etiqueta_bajo.get_width() // 2, led_y_bajo + 30))
114
115 # LED medio
116 led_y_medio = tanque_y + tanque_height // 2
117 color_medio = YELLOW if led_medio else (100, 100, 0)
118 pygame.draw.circle(window, BLACK, (led_x + 3, led_y_medio + 3), led_radius) # Sombra
119 pygame.draw.circle(window, color_medio, (led_x, led_y_medio), led_radius)
120 etiqueta_medio = font.render("Medio", True, BLACK)
121 window.blit(etiqueta_medio, (led_x - etiqueta_medio.get_width() // 2, led_y_medio + 30))
122
123 # LED alto
124 led_y_alto = tanque_y
125 color_alto = GREEN if led_alto else (0, 100, 0)
126 pygame.draw.circle(window, BLACK, (led_x + 3, led_y_alto + 3), led_radius) # Sombra
127 pygame.draw.circle(window, color_alto, (led_x, led_y_alto), led_radius)
128 etiqueta_alto = font.render("Lleno", True, BLACK)
129 window.blit(etiqueta_alto, (led_x - etiqueta_alto.get_width() // 2, led_y_alto + 30))
130
131 # Dibujar etiquetas del porcentaje
LSP Python: listo


Spyder (Python 3.9)
Archivo Editar Buscar Código fuente Ejecutar Depurar Terminales Proyectos Herramientas Ver Ayuda
C:\Users\edmun
C:\Users\edmun\OneDrive\Escritorio\Proyecto Nieves\Simulación.py
ReadArduinoRLC-26112024.py
Simulación.py
texto_porcentaje = font.render(f"Llenado: {int(porcentaje)}%", True, BLACK)
window.blit(texto_porcentaje, (width // 2 - texto_porcentaje.get_width() // 2, tanque_y +
pygame.display.update()
# Inicializar gráfica con matplotlib
plt.ion() # Activar modo interactivo
fig, ax = plt.subplots()
ax.set_title('Evolución del Nivel de Agua')
ax.set_xlabel('Tiempo (s)')
ax.set_ylabel('Nivel del Tanque (%)')
# Historial de niveles
nivel_historial = deque(maxlen=100)
tiempo_historial = deque(maxlen=100)
linea, = ax.plot([], [], label='Nivel de Agua', color='b')
ax.legend()
# Bucle principal
running = True
start_time = time.time()
while running:
for event in pygame.event.get():
if event.type == pygame.QUIT:
running = False
try:
# Leer distancia desde Arduino
if ser.in_waiting > 0:
distancia = float(ser.readline().decode('utf-8').strip())
LSP Python: listo




```
            porcentaje = calcular_porcentaje(distancia)

            # Actualizar LEDs
            led_bajo = porcentaje <= 33
            led_medio = 33 < porcentaje <= 66
            led_alto = porcentaje > 66

            # Dibujar la simulación
            dibujar_tanque(porcentaje)

            # Actualizar gráfica
            tiempo_actual = time.time() - start_time
            tiempo_historial.append(tiempo_actual)
            nivel_historial.append(porcentaje)
            linea.set_data(tiempo_historial, nivel_historial)
            ax.relim()
            ax.autoscale_view()
            plt.pause(0.05)

    except Exception as e:
        print(f"Error leyendo datos: {e}")

    # Pequeña pausa para estabilidad
    time.sleep(0.05)

pygame.quit()
ser.close()
plt.ioff()
plt.show()
```

Este código en Python simula un sistema de monitoreo de un tanque de agua utilizando un sensor ultrasónico conectado a un Arduino. El sistema visualiza el nivel de agua del tanque en una interfaz gráfica usando **Pygame** y también genera un gráfico en tiempo real del nivel de agua mediante **matplotlib**.

1. Configuración Del Puerto Serial:

- Se establece la comunicación con el Arduino a través del puerto serie (configurado como "COM5", que debe ajustarse según el puerto del sistema) y se usa una tasa de baudios de 9600.
- serial. Serial se utiliza para leer los datos enviados por Arduino (distancia medida por el sensor ultrasónico).

2. Inicialización De Pygame:

- Pygame se usa para crear una ventana de simulación en la que se muestra el nivel de agua de un tanque y las condiciones del sistema (LEDs indicadores).
- La ventana tiene dimensiones de 600x700 píxeles.

3. Variables Y Funciones:

- **Variables de simulación**: Se define el rango de distancias que el sensor puede medir (de 2 a 15 metros), y un porcentaje de llenado basado en la distancia detectada.
- **Función calcular_porcentaje**: Calcula el porcentaje de llenado del tanque basándose en la distancia medida por el sensor.
- **Función dibujar_tanque**: Dibuja una representación gráfica del tanque (con forma de rotoplas) y visualiza el nivel del agua, además de tres LEDs (rojo, amarillo y verde) que indican si el nivel de agua está bajo, medio o alto.

4. Gráfica En Tiempo Real Con Matplotlib:

- Se inicializa una gráfica interactiva usando matplotlib para mostrar la evolución del nivel de agua en el tiempo.
- Se actualiza cada vez que se recibe una nueva medida desde el sensor.

5. Bucle Principal:

- El programa entra en un bucle donde constantemente:
    - Lee los datos del sensor ultrasónico desde Arduino.
    - Calcula el porcentaje de llenado basado en la distancia recibida.
    - Actualiza los LEDs (dependiendo del porcentaje de llenado).
    - Dibuja el tanque y su nivel de agua en la ventana de Pygame.
    - Actualiza la gráfica con el tiempo real del nivel del agua.
- Si se presiona el botón de cierre de la ventana, el programa termina.

6. Manejo De Errores:

- Se incluye un bloque try-except para manejar posibles errores al leer los datos del sensor, asegurando que el programa no se detenga inesperadamente.

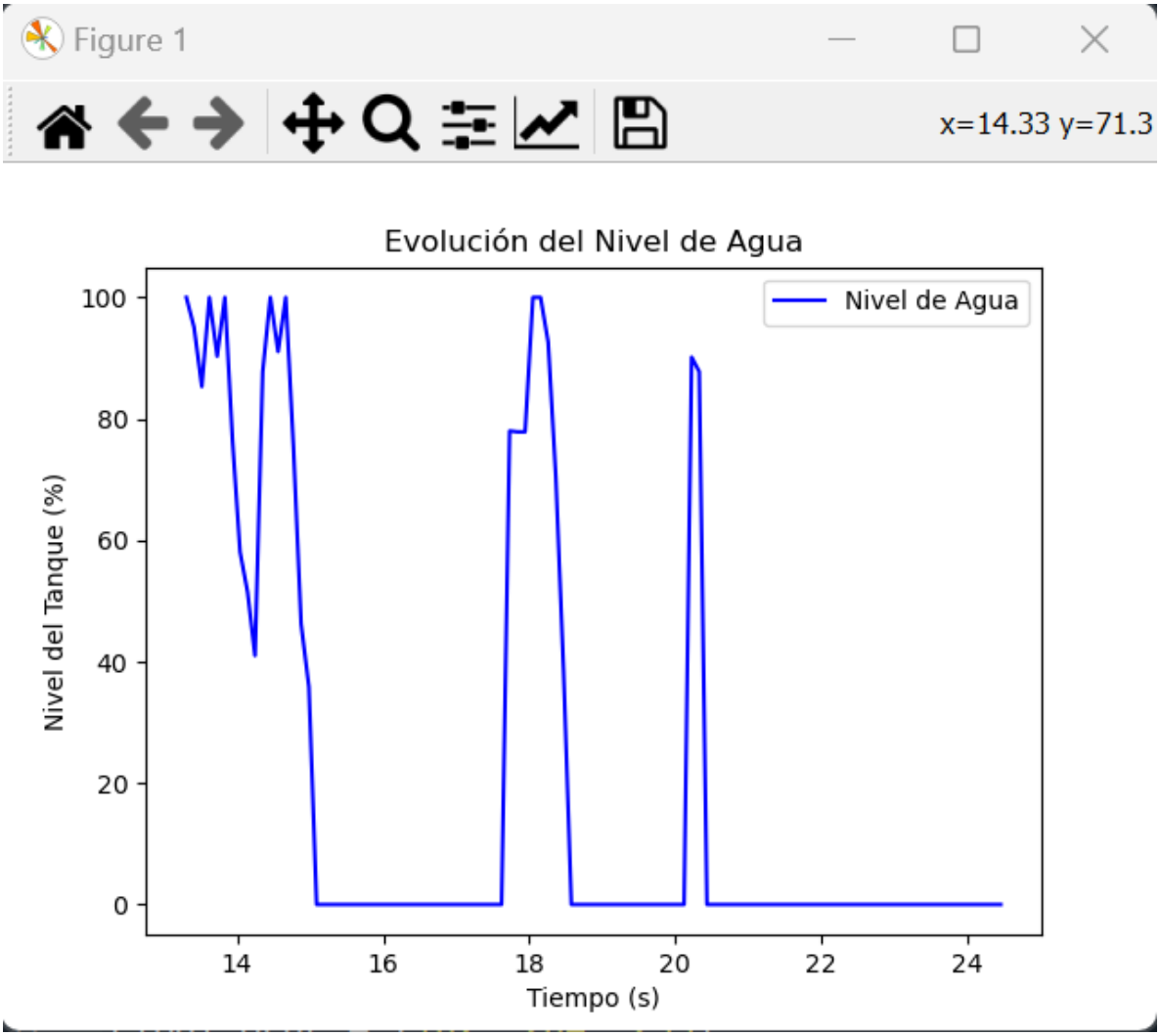
7. Cierre de recursos:

- Al terminar la simulación, Pygame y la conexión serial con Arduino se cierran adecuadamente.

Este código proporciona una simulación visual del nivel de agua en un tanque, con representación gráfica en tiempo real y seguimiento de los datos del sensor, lo cual es útil para monitorear y analizar el comportamiento de un sistema automatizado de llenado de tanque.

RESULTADOS DE LA SIMULACIÓN

Simulación Mostrada Por El Código



Grafica Que Indica Cómo Se Comporta El Nivel De Nuestro Tanque De Almacenamiento

CONCLUSIÓN

El experimento realizado demuestra con éxito cómo utilizar un sensor ultrasónico HC-SR04 para medir distancias y visualizar los resultados a través de indicadores LED conectados a un Arduino. Al integrar el sensor ultrasónico con el microcontrolador, se logra una medición precisa del tiempo de vuelo de las ondas sonoras, lo que permite calcular la distancia al objeto que refleja las ondas. El uso de los LEDs como indicadores facilita la visualización del estado de la medición, proporcionando una forma intuitiva de observar los cambios en la distancia detectada.

El código desarrollado para este experimento se encarga de activar el sensor, medir la distancia y actualizar el estado de los LEDs en función de la distancia medida, lo que simula un sistema de monitoreo visual. Esta simulación no solo valida la funcionalidad del hardware, sino también el comportamiento lógico del código, permitiendo a los usuarios observar los resultados en tiempo real.

El experimento demuestra la viabilidad de crear sistemas de medición de distancia simples pero efectivos utilizando Arduino, proporcionando una base para futuros proyectos de automatización o control que requieran detección de distancia en aplicaciones prácticas. La implementación del código y la simulación confirma que la combinación de hardware y software puede realizar mediciones precisas y ser controlada de manera eficiente a través de señales visuales.

TEMA 4.4 ADQUISICIÓN DE DATOS ANALÓGICOS

INSTITUTO TECNOLÓGICO SUPERIOR DE SAN ANDRÉS TUXTLA

DIVISIÓN DE INGENIERIA MECATRÓNICA

INSTRUMENTACIÓN VIRTUAL

CATEDRATICO

DR. JOSÉ ÁNGEL NIEVES VÁSQUEZ

GRUPO 711-A

PERIODO: AGOSTO 2024 – DICIEMBRE 2024

PROYECTO LABVIEW

INTEGRANTES:

LÓPEZ HERNÁNDEZ EDMUNDO

MORALES AZAMAR ZAIRA ITZEL

HERNANDEZ FLORES MIGUEL ANGEL

PATIÑO BARRIOS JOSE LUIS

GONZALES CAZANOVA JADEN

INTRODUCCIÓN

La adquisición de datos analógicos convierte señales continuas, como temperatura o presión, en información digital mediante sensores y un ADC. Es clave para sistemas de monitoreo, control y automatización.

¿CÓMO FUNCIONA LA ADQUISICIÓN DE DATOS?

La adquisición de datos analógicos funciona capturando una señal continua con un sensor, que la convierte en una señal eléctrica proporcional. Esta señal pasa por un acondicionador que la amplifica o filtra y luego es digitalizada por un conversor analógico-digital (ADC), transformándola en datos binarios que un sistema puede procesar.

VENTAJAS DE LA ADQUISICIÓN DE DATOS

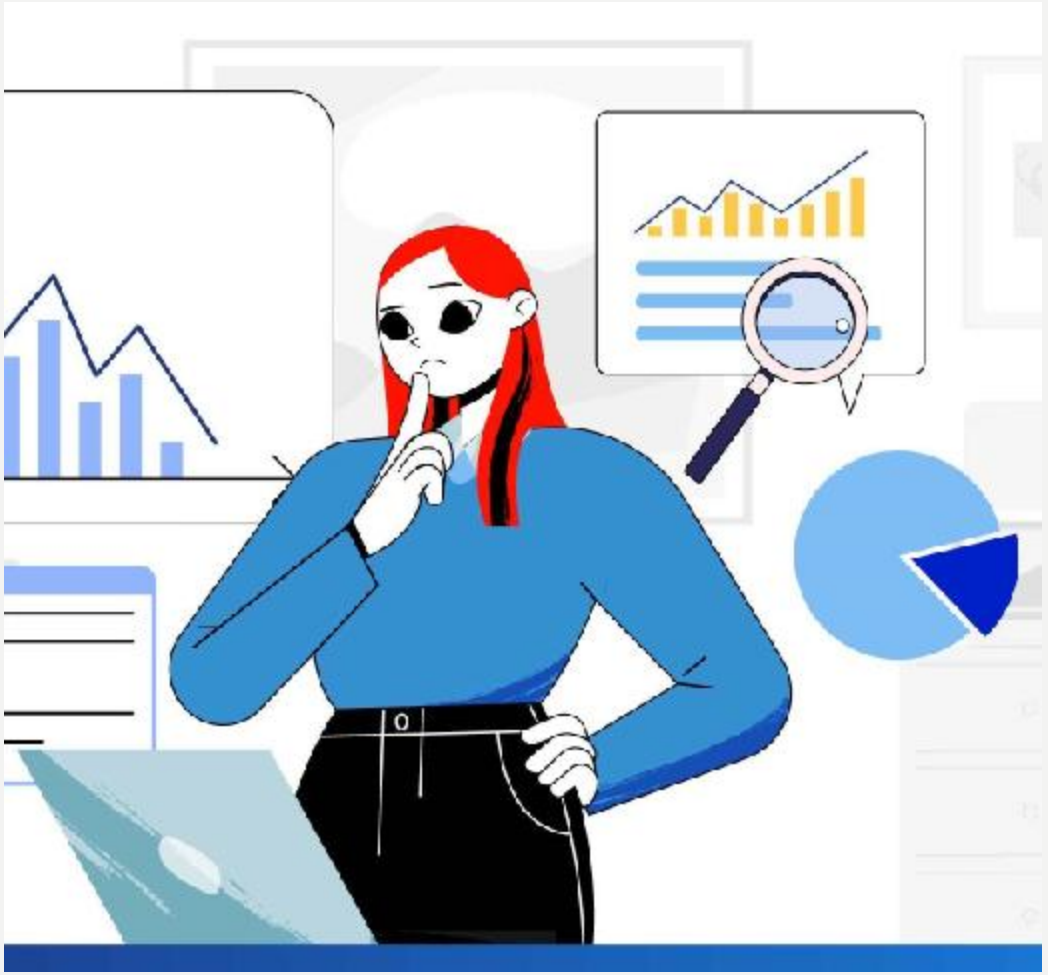
- **Monitoreo en tiempo real:** La adquisición de datos analógicos permite captar y procesar señales continuamente, lo que es crucial para sistemas que requieren respuestas inmediatas, como el control de procesos industriales o sistemas de seguridad.
- **Alta precisión:** Los conversores analógico-digital (ADC) modernos ofrecen una resolución alta, lo que permite obtener mediciones detalladas y minimizar los errores al capturar señales del mundo real.

Versatilidad: Este sistema puede trabajar con una amplia variedad de sensores para medir variables físicas como temperatura, presión, velocidad o flujo, adaptándose a diferentes aplicaciones.

Automatización: Al digitalizar las señales, los datos pueden ser utilizados por sistemas de control automático, lo que optimiza procesos y reduce la intervención humana.

Facilidad de análisis: Al convertir las señales en datos digitales, se facilita el almacenamiento, procesamiento y análisis de la información mediante software especializado para mejorar la toma de decisiones.

Integración tecnológica: Permite conectar dispositivos analógicos con sistemas digitales, como computadoras o controladores programables, creando soluciones más avanzadas e inteligentes.



DESVENTAJAS DE LA ADQUISICIÓN DE DATOS

Costo inicial elevado: La adquisición de datos analógicos requiere equipos como sensores, ADC y acondicionadores de señal, lo que puede implicar una inversión significativa en proyectos pequeños o iniciales.

Complejidad de configuración: La integración de sensores y sistemas de acondicionamiento puede ser compleja, especialmente si las señales requieren filtrado, amplificación o calibración específica.

Limitación Por Ruido: Las señales analógicas son sensibles al ruido eléctrico, lo que puede afectar la precisión de las mediciones si no se implementan las técnicas de reducción de interferencia.

Dependencia De La Resolución Del ADC: Aunque los ADC avanzados son precisos, la resolución limitada en algunos casos puede comprometer la exactitud de los datos obtenidos.

Requiere mantenimiento constante: Los sensores y dispositivos analógicos pueden desgastarse o descalibrarse con el tiempo, afectando la fiabilidad del sistema.

Latencia en sistemas complejos: En configuraciones donde se requiere mucho procesamiento, puede haber retrasos que afecten la respuesta en tiempo real de los sistemas.

APLICACIONES

Automatización industrial: La adquisición de datos analógicos es esencial para monitorear y controlar variables críticas en procesos industriales, como temperatura, presión, caudal y nivel de líquidos. Esto permite optimizar la eficiencia de las operaciones, reducir desperdicios y garantizar la calidad del producto final. Por ejemplo, en una planta química, los sensores miden y ajustan las condiciones del proceso en tiempo real.

Sistemas médicos: Se emplea en dispositivos de monitoreo, como electrocardiógrafos, sensores de presión arterial o pulsioxímetros. Estos convierten señales fisiológicas en datos digitales para su interpretación y almacenamiento. Esto es vital en hospitales y entornos clínicos, donde se requiere supervisión continua y precisa para la salud de los pacientes.

Ingeniería automotriz: En el diseño y fabricación de vehículos, la adquisición de datos es clave para medir y analizar parámetros como velocidad, aceleración, posición de los pedales y condiciones del motor. Estos datos alimentan sistemas avanzados como los frenos ABS o el control de estabilidad, mejorando el rendimiento y la seguridad del automóvil.

Investigación científica: Los sistemas de adquisición de datos se utilizan para medir fenómenos naturales y experimentales. Por ejemplo, en pruebas de laboratorio se recogen datos de vibraciones, flujos de aire o transferencia de calor. Estos datos permiten validar modelos teóricos, analizar comportamientos complejos y desarrollar nuevas tecnologías.

Energía renovable: Los sistemas solares, eólicos e hidroeléctricos dependen de sensores para monitorear variables como intensidad solar, velocidad del viento o caudal de agua. La adquisición de estos datos permite optimizar la generación y distribución de energía, además de predecir condiciones que puedan afectar el rendimiento.

IoT (Internet de las cosas): En el contexto del Internet de las cosas, la adquisición de datos analógicos es fundamental para integrar sensores en dispositivos inteligentes. Estos sistemas permiten medir y transmitir variables como temperatura, humedad o movimiento en tiempo real, mejorando aplicaciones como hogares inteligentes, ciudades sostenibles y monitoreo remoto industrial.

CONCLUSIÓN

La adquisición de datos analógicos es una tecnología esencial que conecta el mundo físico con los sistemas digitales, permitiéndonos medir, analizar y controlar nuestro entorno con precisión. Su impacto se extiende a innumerables áreas de nuestra vida diaria, desde el monitoreo de la salud y la optimización de procesos industriales hasta la eficiencia energética y las aplicaciones del Internet de las cosas. Gracias a esta tecnología, es posible mejorar la calidad de vida, incrementar la seguridad y promover la sostenibilidad, demostrando su importancia como una herramienta clave en el avance de la sociedad moderna.