**Desafío I**

**Laura Nataly Oquendo Vera cc 1036684701**

**Brayan David Sabogal Ramírez cc 1030645440**

**Facultad de Ingeniería Universidad de Antioquia**

**Informática II**

**Septiembre 2024**

**PROYECTO INFORMA2**

**Descripción del Problema**

La empresa Informa2 necesita un sistema que permita identificar las características principales de una señal analógica y visualizar los resultados. El sistema debe iniciar la adquisición de datos mediante un pulsador y permitir la solicitud de información de la señal capturada en cualquier momento. Durante el procesamiento de la información, la adquisición de datos puede ser suspendida y debe reanudarse una vez finalizado el procesamiento.

**Requisitos Específicos**

* Iniciar la adquisición de datos con un pulsador.
* Solicitar información de la señal capturada con otro pulsador.
* Visualizar las características de la señal en una pantalla LCD.
* Medir la frecuencia y la amplitud de la señal.
* Identificar la forma de onda de la señal.

**Consideraciones para la Solución Propuesta**

**Alternativa de Solución**

La solución propuesta implica el uso de una plataforma de simulación como Tinkercad para diseñar y probar el sistema. Se utilizará un Arduino para la adquisición y procesamiento de la señal, junto con una pantalla LCD para la visualización de los resultados.

Componentes Utilizados:

* Arduino Uno: Para la adquisición y procesamiento de la señal.
* Generador de Señales: Para generar la señal analógica de entrada.
* Pulsadores: Para iniciar la adquisición de datos y solicitar la información de la señal.
* Pantalla LCD: Para visualizar las características de la señal.
* Librería Adafruit\_LiquidCrystal.h: Para facilitar la interacción con la pantalla LCD.

**Pasos para la Implementación:**

1. **Montaje del Circuito:**
   * Conectar el generador de señales al pin de entrada analógica del Arduino.
   * Conectar los pulsadores a los pines digitales del Arduino.
   * Conectar la pantalla LCD al Arduino utilizando la librería Adafruit\_LiquidCrystal.h.
2. **Desarrollo del Código:**
   * Escribir un código en Arduino para iniciar la adquisición de datos con un pulsador.
   * Implementar funciones para medir la frecuencia y la amplitud de la señal.
   * Desarrollar un algoritmo para identificar la forma de onda de la señal.
   * Mostrar los resultados en la pantalla LCD.
3. **Pruebas y Validación:**
   * Realizar pruebas con diferentes formas de onda generadas por el generador de señales.
   * Validar que el sistema mide correctamente la frecuencia y la amplitud.
   * Verificar que el sistema identifica correctamente la forma de onda y muestra los resultados en la pantalla LCD.

**Esquema de Tareas para el Desarrollo**

**1. Configuración Inicial**

**Tarea**: Configurar los componentes del hardware y las variables iniciales.

* **Subtareas**:
  + Inicializar la pantalla LCD.
  + Configurar pines de botones y pin de entrada analógica.
  + Configurar el buffer de datos.

**Objetivo**: Preparar el sistema para la adquisición y procesamiento de datos.

**2. Iniciar Adquisición**

**Tarea**: Iniciar el proceso de adquisición de datos desde el pin analógico.

* **Subtareas**:
  + Verificar si la adquisición está activa.
  + Limpiar la pantalla LCD y mostrar mensaje de inicio.
  + Leer datos del pin analógico y almacenarlos en el buffer.
  + Controlar el tamaño del buffer y detener la adquisición si es necesario.

**Objetivo**: Capturar datos en tiempo real y almacenarlos para su posterior análisis.

**3. Solicitar Información**

**Tarea**: Procesar y mostrar la información de la señal capturada.

* **Subtareas**:
  + Mostrar mensaje de procesamiento en la pantalla LCD.
  + Pausar temporalmente la adquisición de datos si es necesario.
  + Llamar a la función de análisis de señal.
  + Reanudar la adquisición después del procesamiento.

**Objetivo**: Permitir al usuario solicitar información y realizar el procesamiento necesario para mostrar los resultados.

**4. Analizar Señal**

**Tarea**: Procesar los datos del buffer para extraer características de la señal.

* **Subtareas**:
  + Encontrar los valores máximos y mínimos en el buffer.
  + Calcular la amplitud de la señal.
  + Implementar lógica para determinar la frecuencia y forma de onda.

**Objetivo**: Extraer características importantes de la señal, como amplitud, frecuencia y forma de onda.

**5. Mostrar Resultados**

**Tarea**: Presentar los resultados del análisis en la pantalla LCD.

* **Subtareas**:
  + Identificar la forma de onda basada en los datos analizados.
  + Mostrar la amplitud y frecuencia en la pantalla LCD.
  + Actualizar la pantalla con la forma de onda identificada.

**Objetivo**: Visualizar los resultados del análisis de señal de forma clara y comprensible para el usuario.

**6. Identificar Forma de Onda**

**Tarea**: Determinar el tipo de forma de onda de la señal capturada.

* **Subtareas**:
  + Analizar los datos del buffer para detectar patrones de ondas.
  + Comparar los patrones con las formas de onda conocidas (pulso, triangular, cuadrada).
  + Identificar si la forma de onda es desconocida.

**Objetivo**: Clasificar la forma de onda de acuerdo con los patrones predefinidos.

**Descripción de Diagrama de Flujo**

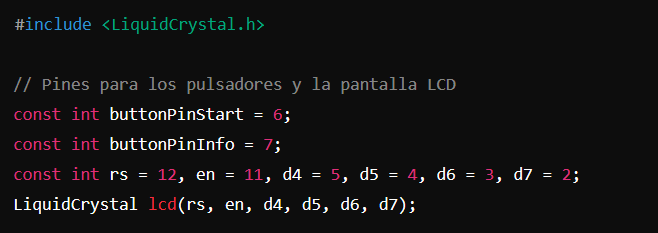
1. **Inicio**: El programa comienza su ejecución.
2. **Configuración Inicial**:
   * Inicializa la pantalla LCD.
   * Configura los pines de los botones y del generador de señales.
3. **Esperar Acción del Usuario**:
   * **Botón de Inicio Presionado**:
     + Comienza la adquisición de datos.
     + Lee datos del pin analógico y almacena en el buffer.
     + Si el buffer está lleno o se detiene la adquisición, pasa al siguiente paso.
   * **Botón de Información Presionado**:
     + Procesa la información (frecuencia, amplitud, forma de onda).
     + Muestra la información en la pantalla LCD.
4. **Identificar Forma de Onda**:
   * Analiza los datos almacenados en el buffer.
   * Identifica la forma de onda (pulso, triangular, cuadrada, desconocida).
5. **Mostrar Resultados**:
   * Muestra la frecuencia, amplitud, y forma de onda en la pantalla LCD.
6. **Reanudar Adquisición**:
   * Continúa la adquisición de datos si fue pausada.
7. **Fin**: El proceso se repite continuamente.

**Diagrama

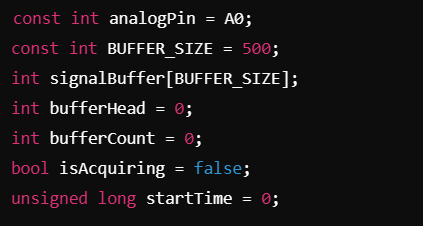
Descripción generada automáticamente**

**Algoritmos implementados**

* + **Inclusión de Librerías y Configuración de Pines**

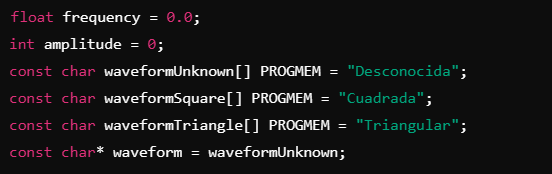


* **Librería LiquidCrystal.h:** Se incluye para controlar la pantalla LCD. Esta librería proporciona funciones para interactuar con la pantalla.
* **buttonPinStart y buttonPinInfo**: Pines digitales conectados a los botones.
* **rs, en, d4, d5, d6, d7**: Pines usados para la comunicación con la pantalla LCD.
  + **Variables para adquisición de datos**

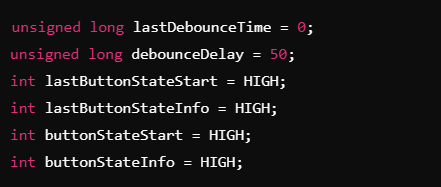


* **analogPin**: Pin analógico donde se leerá la señal.
* **BUFFER\_SIZE**: Tamaño del buffer para almacenar las muestras.
* **signalBuffer**: Array que almacenará las muestras de la señal.
* **bufferHead y bufferCount**: Controlan el índice de lectura y la cantidad de muestras.
* **isAcquiring**: Indica si se está en proceso de adquisición de datos.
* **startTime**: Marca el tiempo de inicio de la adquisición.

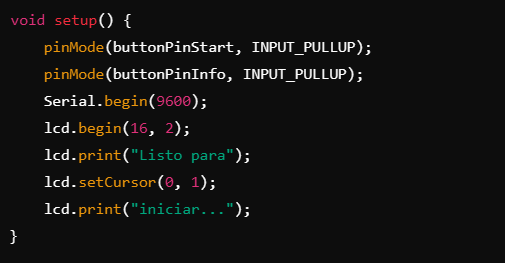
* **Variables para características de la señal**



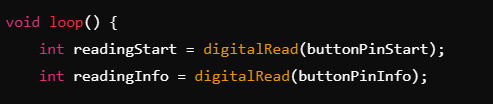
* **frequency y amplitude:** Almacenan la frecuencia y amplitud de la señal**.**
* **waveformUnknown, waveformSquare, waveformTriangle:** Cadenas de texto para identificar diferentes formas de onda.
* **waveform:** Puntero a la forma de onda actual.
* **Prototipos:** Declaraciones anticipadas de funciones que se implementarán más adelante. Esto permite que las funciones sean llamadas en cualquier parte del código antes de que se definan.
* **Variables de antirrebote para botones**



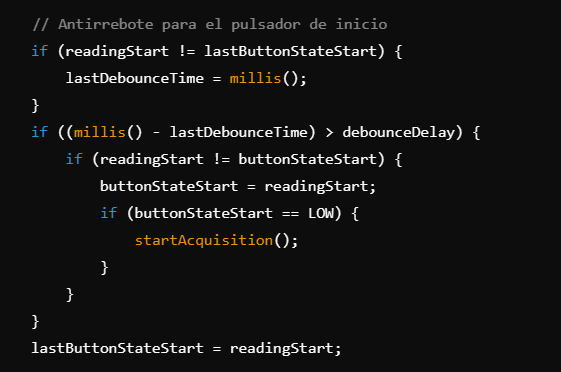
* **lastDebounceTime:** Tiempo del último cambio de estado en el botón.
* **debounceDelay:** Tiempo para considerar un cambio de estado válido.
* **lastButtonStateStart, lastButtonStateInfo:** Guardan el último estado de los botones.
* **buttonStateStart, buttonStateInfo:** Guardan el estado actual de los botones.
* **Función setup()**



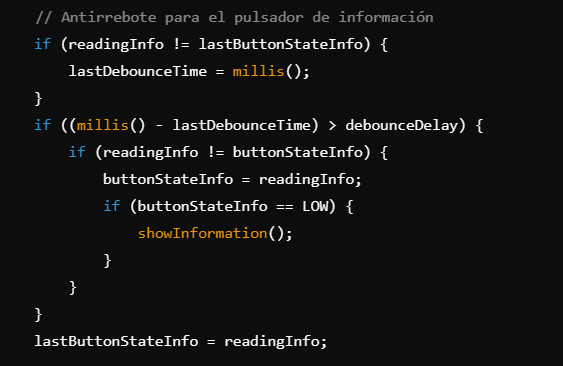
* Configura los pines de los botones como entradas con resistencia pull-up.
* Inicializa la comunicación serial y la pantalla LCD, mostrando un mensaje de preparación.
* **Bucle principal loop()**



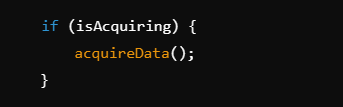
* Lee el estado de los botones.
* **Antirrebote para el botón de inicio**



* Implementa lógica de antirrebote. Si se detecta un cambio en el estado, se actualiza el tiempo y el estado del botón. Si el botón de inicio es presionado, se llama a startAcquisition().
* **Antirrebote para el botón de información**



* Similar al anterior, pero para el botón de información. Si es presionado, llama a showInformation().
* **Adquisición de datos**

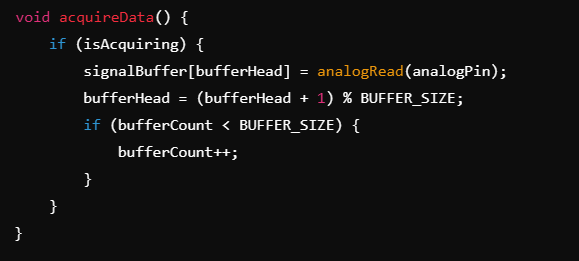


* Si se está adquiriendo, se llama a acquireData() para leer la señal.
* **Función startAcquisition()**

Texto

Descripción generada automáticamente

* Inicia la adquisición de datos, reseteando el buffer y mostrando un mensaje en la LCD.
* **Función acquireData()**



* Lee un valor del pin analógico y lo almacena en el buffer. Maneja el índice circular y cuenta las muestras.
* **Función showInformation()**



* Detiene la adquisición y analiza la señal. Muestra los resultados en la LCD y en el monitor serial.
* **Función analyzeSignal()**

**Esta función realiza varios cálculos sobre los datos adquiridos:**

* **Frecuencia:** Usando autocorrelación para detectar el período.
* **Amplitud:** Calcula la diferencia entre el máximo y mínimo del buffer.
* **Identificación de forma de onda:** Basada en el ciclo de trabajo, simetría y un análisis espectral simple.