**Desafío 1**

Nicolás Valencia

Karen Mazo

Universidad de Antioquia

2598521: Informática II

Aníbal Guerra

1. de noviembre de 2024

Contenido

[a. Análisis del problema y consideraciones para la alternativa de solución propuesta. 3](#_Toc177587801)

[b. Esquema donde describa las tareas que usted definió en el desarrollo de los algoritmos. 5](#_Toc177587802)

[c. Algoritmos implementados. 6](#_Toc177587803)

[d. Problemas de desarrollo afrontados. 11](#_Toc177587804)

[e. Evolución de la solución y consideraciones para tener en cuenta en la implementación. 12](#_Toc177587805)

# Análisis del problema y consideraciones para la alternativa de solución propuesta.

Como problema principal se tiene la identificación de una señal generada aleatoriamente, esta señal está dada por un generador de función de tinkercad, también se tiene la adquisición de los datos de forma adecuada para lograr identificar dicha señal, adicionalmente se debe pensar en el procesamiento de forma eficiente de los datos adquiridos. El análisis inicial para la solución teniendo en cuenta que la forma de las señales en tinkercad son tres (cuadrada, senoidal, triangular), se debe modelar el comportamiento de cada forma de señal, para hallar las características de cada forma, teniendo en cuenta todos los posibles escenarios sobre el tiempo para adquirir los datos de la señal, se debe analizar como fue el comportamiento de la señal la mayor parte del tiempo y así determinar su forma, además, se debe considerar que, si una señal no coincide con ninguna de las características halladas, se define como señal desconocida. Luego de identificar la forma de la señal se desarrolla un mecanismo de salida mediante una pantalla LCD, que contenga información de la forma de la señal, la frecuencia (Hz) y su amplitud (V).

A continuación, se definen las señales como análisis principal para la solución del problema:

* Señal cuadrada: es una forma de onda periódica que alterna entre dos niveles de voltaje (alto y bajo) con un tiempo de duración constante para cada nivel. La transición entre los niveles es abrupta y se produce en un tiempo muy corto.
* Señal triangular: es una forma de onda periódica que varia linealmente entre dos niveles extremos (alto y bajo). La señal aumenta y disminuye de manera lineal y simétrica, creando picos y valles en intervalos regulares.
* Señal senoidal: es una forma de onda periódica que sigue la función matemática seno, es sueve y continua que varia en forma de una curva sinusoidal representando el valor de la señal en función del tiempo.
* Señal desconocida: es una señal cuya forma de onda no se puede identificar claramente como una de las formas de onda (cuadrada, triangular, senoidal). Puede tener una forma compleja y no periódica, o una forma de onda que resulta de la combinación de varias señales.

# Esquema donde describa las tareas que usted definió en el desarrollo de los algoritmos.

Diagrama

Descripción generada automáticamente

No

No

Sí

Sí

No

Sí

**Figura 1.** Esquema para la solución

# Algoritmos implementados.

Dentro del programa se implementaron algoritmos con el fin de dar solución al problema planteado; teniendo en cuenta que las señales serán dadas por un generador de funciones se debe capturar los datos por el puerto serial, procesar los datos, es decir, analizar los datos capturados usando un algoritmo para analizar la forma de cada señal, además, el correcto manejo de la memoria dinámica, por último la visualización de los datos obtenidos y verificar que correspondan con el tipo de señal esperado. A continuación, se hace una breve descripción de los algoritmos:

1. Captura de datos:

Se usan dos pulsadores donde uno indica el inicio y otro el fin de la adquisición de datos, estos se definen con su tipo y el pin donde están colocados.

Se usa un arreglo bidimensional dinámico para guardar los datos adquiridos, en el cual se tienen dos columnas una almacena el valor y la otra el tiempo de la señal.

1. Procesamiento de datos:

Se analiza las características de la señal como el periodo el cual se calcula detectando los puntos donde la señal pasa por cero; el periodo sirve para calcular la frecuencia (1 / periodo); también se halla los valores máximo y mínimo de la señal para calcular la amplitud.

Para el análisis de las formas de las señales se decidió un orden de identificación iniciando por analizar la señal cuadrada con la cual se tiene como clave los cambios bruscos que tiene entre alto y bajo, luego, la señal triangular donde se usan dos pendientes de la señal teniendo como referencia el punto máximo, donde se espera que sean semejantes y por último para la senoidal se analizó que, si no es una señal triangular, entonces se puede concluir que es una señal senoidal.

1. Manejo de memoria:

Se usa arreglos bidimensionales dinámicos, donde se busca tener un manejo eficiente de la memoria y un uso correcto para evitar fugas de memoria.

1. Visualización de datos:

Se envían los datos a una LCD, donde se espera ver la frecuencia, la amplitud y la forma de la señal.

|  |  |  |
| --- | --- | --- |
| **FUNCIONES** | **PARAMETROS** | **HACE** |
| calcularAmplitud | * Float\*\* arreglo (Puntero a un puntero que almacena un arreglo dinámico bidimensional de tipo float.) * Unsigned short int size (Variable de tipo entero corto sin signo, que contiene el tamaño del arreglo) | La función calcula la amplitud de la señal dada, usando los valores máximo y mínimo. |
| calcularPeriodo | * Float\*\* arreglo (Puntero a un puntero que almacena un arreglo dinámico bidimensional de tipo float.) * Unsigned short int size (Variable de tipo entero corto sin signo, que contiene el tamaño del arreglo) | La función calcula el periodo de la señal, teniendo en cuenta el tiempo en que la señal hizo cruces por cero. |
| calcularFrecuencia | * float periodo (variable de tipo float, contiene el valor del periodo de la señal) | La función calcula la frecuencia de la señal, usando el periodo de esta. |
| predominante | * int cuadra (Contador entero que almacena el número de veces que la señal se comportó como cuadrada) * int triangu (Contador entero que almacena el número de veces que la señal se comportó como triangular) * int senoi (Contador entero que almacena el número de veces que la señal se comportó como senoidal) | La función calcula las veces que la señal se comportó como cuadrada, triangular o senoidal. |
| senialTriangular | * float\*\* arreglo (Puntero a un puntero que almacena un arreglo dinámico bidimensional de tipo float) * unsigned short int fila (Entero corto sin signo, contiene el número de filas) | La función verifica si la señal es triangular, usando dos pendientes de la señal teniendo como referencia el punto máximo de la señal, donde se espera que sean semejantes |
| verificarcuadrada | * float\*\* arreglo (Puntero a un puntero que almacena un arreglo dinámico bidimensional de tipo float) * unsigned short int filas (Entero corto sin signo, contiene el número de filas) | La función verifica si la señal es cuadrada usando los cambios bruscos entre alto y bajo. |

El tipo de variable se eligió pensando en el mayor ahorro de memoria posible, dentro de las cuales están las siguientes:

Unsigned short int: Es un tipo de dato entero sin signo que generalmente ocupa 2 bytes. Puede almacenar valores desde 0 hasta 65,535.

Const unsigned short int: Similar al unsigned short int, pero la palabra clave const indica que el valor de la variable no puede ser modificado una vez establecido. Al igual que unsigned short int, puede almacenar valores desde 0 hasta 65,535.

Short int: Es un tipo de dato entero con signo que generalmente ocupa 2 bytes. Puede almacenar valores desde -32,768 hasta 32,767.

Signed short: Se utiliza cuando se sabe que los valores requeridos estarán dentro de ese rango, y se quiere ahorrar memoria. Usualmente ocupa 2 bytes (16 bits). Puede almacenar números desde -32,768 hasta 32,767.

Float: Es un tipo de dato de punto flotante que generalmente ocupa 4 bytes, utilizado para representar números decimales. Puede almacenar valores que van desde aproximadamente -3.4e38 hasta 3.4e38, con una precisión de alrededor de 7 dígitos decimales.

Bool: Es un tipo de dato que representa un valor booleano, es decir, verdadero (true) o falso (false). Solo puede almacenar 2 valores: true (generalmente representado como 1) y false (representado como 0).

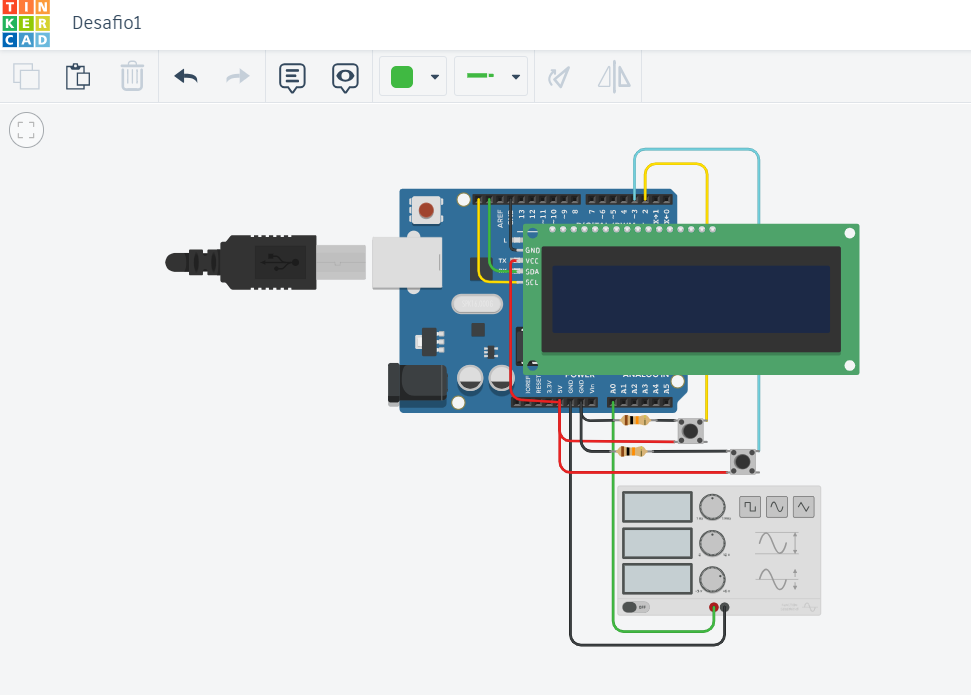
# d. Problemas de desarrollo afrontados.

Dentro del desarrollo del desafío se enfrentaron muchos retos, como el análisis del problema y los pasos a seguir para la correcta solución de este. El uso de arreglos dinámicos ha sido clave para la gestión de la memoria, ya que en la memoria estática no se puede aumentar el almacenamiento en tiempo real de ejecución del programa se opto por el uso de memoria dinámica, implementando técnicas para expandir el tamaño del arreglo cuando se alcanza su capacidad máxima, lo cual se ha hecho mediante el reemplazo de memoria, es decir, no solo se requiere asignar memoria al inicio del programa, sino también debe tener la capacidad de reemplazar el arreglo conforme los datos se van llenando. El proceso del reemplazo de memoria implica crear un nuevo arreglo con más espacio, copiar los datos del arreglo anterior en el nuevo y liberar la memoria que se había ocupado con el arreglo anterior. Esto ayuda a hacer un uso eficiente de la memoria, pero contiene complejidades a la hora de introducirlo al código, por ejemplo, la posibilidad de generar fugas de memoria. También se genera dificultades a la hora de calcular la frecuencia, en la cual se calcula los picos de la señal; pero a la hora de hacer las comparaciones en ocasiones no alcanza a calcular el segundo punto máximo de la señal. Esto es punto para mejorar dentro del desafío, otro punto para mejorar son los cálculos para hallar las señales también se han encontrado algunos problemas. A la hora de llevar a cabo la implementación del desafío, es indispensable más tiempo de análisis, para llegar a mejores resultados.

El uso de la memoria dinámica en este desafío ha sido un reto que ha requerido la atención suficiente para la asignación y liberación de los recursos, además, también se enfrentaron limitaciones de hardware al trabajar con Arduino.

# Evolución de la solución y consideraciones para tener en cuenta en la implementación.

La evolución de la solución del desafío ha abordado el desarrollo, ajuste y refinamiento de los algoritmos implementados, además del uso de técnicas para analizar las señales para determinar su forma. En la fase inicial del proceso, se hace un análisis donde se define los objetivos y los requisitos del desafío, como capturar señales, almacenar y procesar, y por último visualizar los resultados. Para esto se debe tener en cuenta la implementación de un circuito en tinkercad que ayudará en el proceso de captura y visualización de los datos.



**Figura 2. Circuito para la simulación.**

Para armar el circuito de la figura 2, se usó un Arduino uno, al cual se conectan una LCD I2C, dos pulsadores y un generador de funciones, el cual se encargará de generar las señales que se necesitan para llevar a cabo el desafío.

Ahora bien, para la implantación se considera la gestión de la memoria dinámica, donde se debe implementar técnicas de asignación y liberación de memoria, además, se debe tener en cuenta la frecuencia de muestreo y así hacer una captura de datos con más precisión, al mismo tiempo se debe llevar a cabo técnicas para manejar errores, por ejemplo, datos fuera del rango. Igualmente se debe verificar la funcionalidad de los algoritmos de manera unitaria, es decir, a medida que se van creando, y a la vez de manera integrada con el resto del código. Los algoritmos implementados deben permitir el análisis de la señal registrada, cubriendo las necesidades fundamentales para la solución del problema planteado.