

Analisis del reto 1.



14 de septiembre de 2024

Cristopher corrales y tomas restrepo

UNIVERSIDAD DE ANTIOQUIA

ANALISIS:  
 El problema me pide que conforme a una onda ingresada por un generador de onda mi programa determine: El tipo de onda, la amplitud de la onda y la frecuencia.

Partiendo de estos tres puntos principales podemos concluir lo siguiente:  
**-Amplitud:** El generador de onda lo que proporciona son amplitudes que las va graficando respecto a un tiempo, entonces una de las posibles soluciones para poder encontrar la amplitud de la onda sabiendo que los datos que me proporciona el programa son amplitudes en intervalos de tiempo es determinar los valores máximos y mínimos que tome la onda (de hecho los valores máximos y mínimos o crestas y valles son muy importantes para hallar casi todo lo que corresponde a la onda).

Sabemos que la amplitud es la variación que tiene una señal respecto a su punto de equilibrio, pero en el programa el punto de equilibrio no siempre va a ser cero, entonces una manera de conocer su amplitud será determinar el valor máximo y mínimo de un ciclo que pueda alcanzar la onda, entonces al valor máximo le voy a restar el valor mínimo y luego eso lo divido entre dos, porque restándole el valor mínimo al máximo lo que obtendría sería una amplitud pico-pico, pero yo necesito la amplitud normal, para esto lo divido entre dos, asi obtendría la amplitud. Para finalizar lo divido en 100 para tener esta amplitud en voltios.

**-Frecuencia:** La frecuencia se define como el número de ciclos que alcanzar a ocurrir en un segundo. Hallar la frecuencia es un poquito mas complejo, debido a que como tal no contamos con la variable tiempo que me diga como cambia la amplitud con respecto al tiempo, pero se puede hacer una estimación, ya que haciendo prueba y error se puede determinar que por cada 10 datos que ingresa el generador de funciones transcurre un segundo, entonces para hallar la frecuencia una relacion me dice que conociendo el numero de picos que muestra la onda y el tiempo que dura la onda puedo hallar su frecuencia, siendo esta la división de la cantidad de picos totales que registre la onda entre el tiempo que tome la onda en hacer el recorrido, esto finalmente lo divido por 100 para encontrar su valor en Hz, sin embargo esta aproximación no es del todo exacta ya que la relacion con el tiempo es una estimación dada acorde a las limitaciones de tinkercard.

-Tipo de onda: Esto es sin duda lo más difícil de relacionar y el problema principal del reto. Se pueden tener estas distinciones según el tipo de onda:

-Función cuadrada: Alterna periódicamente los valores de la onda, pero solo entre máximos y mínimos de manera que no toma valores intermedios haciendo recorridos con cambios extremos, llega a un punto máximo se estabiliza, por lo cual repite este valor una cantidad de veces y luego baja a un mínimo donde su comportamiento es el mismo por lo cual repite este valor una cantidad de veces.

-Función senoidal: la onda senoidal describe una oscilación suave y periódica de tal manera que llega a un punto máximo, pero no de golpe, sino que aproximándose lentamente con una tasa de cambio en sus amplitudes no tan abrupta respecto al tiempo. Lo que implica que tome valores intermedios.

-Función Triangular: la onda triangular tiene un comportamiento similar a la senoidal, sin embargo, en la onda triangular los cambios no son suaves, sino mas bien abruptos, de manera que su llegada a los puntos máximos y mínimos es más escalonada.

Ya definidos los tipos de ondas podemos hacer una diferenciación muy marcada entre la onda de tipo cuadrada respecto a la onda triangular y la onda senoidal, ya que la principal diferencia entre la onda cuadrada con las otras dos es que esta no va a tomar bajo ninguna circunstancia valores intermedios, sino que siempre va a tender a llegar a un máximo, estabilizarse y luego bajar a un mínimo y cumplir con este comportamiento periódicamente, lo que no pasa en la onda senoidal y triangular ya que estas si admiten tomar valores intermedios para conseguir su forma.

La diferencia más difícil de hallar para que el programa pueda tomar decisiones es la de la onda senoidal con la onda triangular, ya que en estas su comportamiento es muy parecido y solo notamos como diferencias que la tasa de cambio de la triangular es mas fuerte o mayor que la tasa de cambio de la senoidal. Su diferencia también podría ir encaminado a la derivada, debido a que la pendiente se relaciona con los cambios abruptos que tenga la función en su trazo, sin embargo, esto tiene bastantes complicaciones puesto que no conocemos la relacion entre cada amplitud con el tiempo exacto en el que esta ocurre.

**Explicación algoritmos:**

El código completo del reto se basa en tres puntos principales que son:  
1. La recolección de datos.

2. El funcionamiento de los botones.

3. El calculo de la amplitud, la frecuencia, pendientes y el tipo de onda.

Cada uno va ligado al otro para su correcto funcionamiento de manera que la recolección de datos funciona a partir del funcionamiento de botones y cada función de cálculo de amplitud, frecuencia y el tipo de ondas depende de la lista que pasa la recolección de datos. Dado lo anterior se explicará el funcionamiento del código parte por parte.

**Inicialización de variables:** Se inicializan todos los datos necesarios para el funcionamiento de cada función, estas se inician conforme se van desarrollando las distintas funciones y se descubren las necesidades de cada una para funcionar.

**VOID SETUP:** se inicializa el modo de los botones para que reciban datos, se inicializa el monitor serial y el LCD.

En el void setup se le dice al LCD que imprima “Presione botón Iz para comenzar” esto debido a que queremos que independientemente de cualquier cosa, siempre que se inicie el funcionamiento del Arduino lo primero que queremos es que se empiecen a recolectar datos, entonces indica al usuario que presione el botón izquierdo para comenzar a recibir datos para determinar todo lo que viene más adelante.

**recodatos:** Esta función se crea con el fin de recolectar los datos y guardarlos en un arreglo dinámico el cual va creciendo según va leyendo datos el serial, como parámetros le pasamos un arreglo dinámico que fue declarado anteriormente y la capacidad de este, para hacer la redimensión utilizamos un IF que ayuda a identificar cuando se llenó la memoria y dentro de este vamos a crear otro arreglo dinámico con un espacio más de memoria que el anterior y haremos la copia de datos del arreglo viejo al nuevo, liberamos la memoria del viejo y después apunta al nuevo (Con estos dos pasos evitamos fuga de memoria) y es así como recolectamos los datos utilizando memoria dinámica.

**funcionamientoBotones:** gestiona las interacciones con el botón de la izquierda que es para iniciar y el de la derecha que sirve para detener la recolección de datos de la señal analógica que le proporciona el generador de funciones al Arduino.

El sistema lee el estado actual de los pines a los que están conectados los botones de iniciar y detener. Esto devuelve HIGH o LOW dependiendo de si el botón esta presionado o no. Utiliza un debounce para evitar falsos cambios de estado debido a fluctuaciones rápidas cuando se presiona el botón se asegura de que un cambio en el estado del botón se mantenga valido por un tiempo mínimo para que se considere valido. Si el estado leído del botón cambia con respecto al último almacenado, se actualiza el tiempo en el que ocurrió este cambio y millis() devuelve el tiempo trascurrido desde que se encendió el Arduino. Si el botón de iniciar esta preionado se muestra el mensaje en la pantalla LCD indicando que la recolección de datos esta lista y que el botón de detener debe ser presionado para terminar, cuando se libera el botón de iniciar automaticante se activa la recolección de datos. Al igual que con el botón de iniciar se verifica si ha habido un cambio en el estado del botón detener y se registra el momento del cambio, luego se actualiza el estado del botón si paso suficiente tiempo desde el ultimo cambio. Si el botón de detener esta en HIGH se desactiva la recolección de datos y finalmente se desactiva la memoria dinámica usada por datos. Si la bandera recibiendo es true, se lee el valor de una señal analógica desde analogPin y se almacena en el arreglo dinámico datos usando la función recodatos. Luego, el valor leído se imprime en el monitor serial y se añade un pequeño retardo para controlar la frecuencia de lectura.

Esta función tiene como limitante que después de aproximadamente 30 segundos el programa por el mismo funcionamiento de Arduino y sus limitaciones pierde funcionalidad, sin embargo, en el tiempo preestablecido es totalmente funcional y cumple a la perfección.

**funAmplitud:** esta variable va a ser la encargada de calcular la amplitud de la función acorde a los datos que le ingrese la lista que retorna el funcionamiento de botones.

Esta función pide como datos de entrada el arreglo dinámico de datos y el tamaño de dicho arreglo de datos, que va a ser la cantidad de datos que ingresen en el arreglo. La función calcula el dato mayor que hay en la lista recorriendo la lista y buscando el numero mas grande que haya en el arreglo, luego va a encontrar el numero menor del arreglo de la misma manera. Finalmente, para hacer el calculo de la amplitud lo que hace la función es que agarra el numero mayor y le resta el numero menor, el resultado de esta resta lo divide entre dos, de manera que conozco la distancia que hay entre el dato mayor y el menor y lo divido en dos para tener cuanto es el valor hasta el que abre. Ya luego lo que se hace es pasarlo a voltios dividiendo por 100 para tenerlo en las medidas requeridas.

El me retorna el los voltios que es la variable amplitud pasada a voltios y es de tipo flota para aumentar su precisión debido a que por las limitaciones del muestreo y a su aleatoriedad no siempre agarra como valores máximos los valores máximos que debería agarrar, sin embargo, coge unos cercanos, entonces para que se aprecie esto se deja el resultado final con decimales.

Este algoritmo tiene como limitaciones el cambio en el desface debido a que no puede superar ciertos desfaces ya que los máximos valores que puede agarrar el monitor serial están definidos por 1023 por ende si el valor correcto que debería tener la amplitud en ese momento es mayor a 1023 simplemente se va a repetir este ultimo numero ya que es su límite, además cuando se hace el desface no agarra el valor mínimo como debería por lo que a la hora de mostrar el resultado en vez de mostrar la amplitud correcta mostrara el desface que hay. Sin embargo, el hecho de que no agarre el valor mínimo cuando presenta un desface no es por descuido del código ya que se probo el mismo algoritmo fuera de tinkercad y funciona con normalidad.

**funFrecuencia**: Esta función se crea con el fin de hallar la frecuencia, como parámetros le pasamos un arreglo dinámico que fue declarado anteriormente y la capacidad de este, analizamos este arreglo y encontraremos la posición del segundo pico (Se busca el segundo pico ya que por lo general los datos del primer pico son muy inestables) y buscamos la posición del valle seguido a este(Se encuentra el valle después del pico por lo mismo que encontramos el segundo pico), de estos dos podemos hallar el tiempo total restando la posición del valle con la del segundo pico (Para saber cuántos datos hay de por medio) seguido a esto lo dividimos por diez (Se divide por 10 porque en la ejecución nos dimos cuenta de que es un promedio de 10 datos por segundo que se ingresa al arreglo dinámico) para después multiplicarlo por 2 (Se multiplica por 2 para completar un ciclo), después de hallar este tiempoT podemos hallar la frecuencia dividiendo a 1 por el tiempoT y el resultado de este se multiplica por 100 y es así como hallamos la frecuencia

**funPendiente:** Esta función se crea con el fin de hallar la pendiente entre punto, pero para este caso la vamos a limitar para hallar solo las 10 pendiente antes de llegar al punto del valle (Incluyendo el valle), como parámetros le pasamos dos arreglos (Uno donde están los datos y otro donde vamos a guardar las pendientes) y pasaremos la capacidad del arreglo de los datos.

Para esta función hallamos la posición del valle recorriendo el arreglo de los datos, este valle lo buscaremos después del segundo pico, según sea la posición del valle lo que hacemos es recorrer desde esa posición hacia atrás en el arreglo de los datos y vamos hallando las pendientes entre el dato en el que estamos parados y el anterior(Esto lo haremos hasta calcular 10 pendientes), está pendiente la encontramos restando la posición actual con la anterior y después dividiéndola por 0.1 (Tenemos un promedio de 10 datos por segundo entonces entre dato y dato paso 0.1 segundos).

-Para identificar el tipo de función que es analíticamente de las cosas más complicadas del reto, lo que hicimos fue crear una función que me defina cuando cumple los requisitos para ser cada una de los tres tipos de funciones es decir por cada tipo de onda se creó una función que defina su comportamiento y añadido a eso se creó otra función que simplemente certifique cuando es una función u otra para imprimirlo en el LCD.

**esCuadrada:** esta función tiene como finalidad únicamente decirme si el tipo de función es cuadrada, esto basándonos en el análisis previo que se hizo desde un principio que dice que la función cuadrada solo toma valores extremos y que nunca tiene valores intermedios.

Para determinar si es cuadrada esta función me pide el arreglo de datos y el tamaño de este arreglo, ahora simplemente se encuentras los valores extremos llamados nivel bajo y nivel alto, el algoritmo lo que hará es verificar que todos los valores del arreglo cumplan con la condición de ser un valor de nivel alto o de nivel bajo, de manera que si cumple esta condición se puede considerar una función de tipo cuadrada, sin embargo si por el contrario encuentra un valor intermedio ya no podrá considerarse como función cuadrada.

**Seno:** Esta función se crea con el fin de saber si es una función senoidal, como parámetros le pasamos un arreglo el será donde están guardadas las pendientes y la capacidad de este, para hallar si es una función senoidal encontramos una peculiaridad y es que en esos 10 datos pendientes que fueron guardad en el arreglo tienen un orden en el 70%, es decir, que el 70% de esos datos guardan un orden que va de mayor a menor.

Para la clasificación de si es senoidal o no utilizamos toda la información que tenemos anteriormente, lo que hacemos es inicializar un contador que va y vamos a recorrer las pendientes comparando las posiciones actuales con las siguientes y si es mayor o igual a la actual sube el contador y si al final este es mayor al 70% se clasifica como senoidal retornando un true y en caso contrario se devuelve un false

**Triangular:** Esta función se crea con el fin de saber si es una función triangular, como parámetros le pasamos un arreglo el será donde están guardadas las pendientes y la capacidad de este, para hallar si es una función triangular encontramos una peculiaridad y es que en esos 10 datos pendientes que fueron guardad en el arreglo no tienen un orden en el 70%, es decir, que el 70% de esos datos no guardan un orden que va de mayor a menor.

Para la clasificación de si es triangular o no utilizamos toda la información que tenemos anteriormente, lo que hacemos es inicializar un contador que va y vamos a recorrer las pendientes comparando las posiciones actuales con las siguientes y si es mayor o igual a la actual sube el contador y si al final este es menor al 70% se clasifica como triangular retornando un true y en caso contrario se devuelve un false.

**determinarFuncion:** el funcionamiento de esta función va a ser muy sencillo debido a que su única tarea es decirme el tipo de función que es basándose en lo obtenido anteriormente por las funciones que definían el comportamiento de cada tipo de onda. De manera que tiene cuatro condiciones, una para cada tipo de función, si es cuadrada me retornara 1, si es senoidal me retornara 2 y si es triangular me retornara 4, si no es ninguna de las anteriores retornara -1.

Void loop: procuramos no llenar de lógica el void loop y ser lo mas organizados posible con las funciones, de manera que en el void loop solo se invocan las funciones y se crea un condicional para que una vez me imprima el primer valor de cada cosa no se quede iterando infinitamente y se invocan dentro de este condicional las funciones necesarias y luego dentro de este condicional se imprimen las condiciones en el LCD dependiendo de lo que retorne la función ”determinarFuncion”.