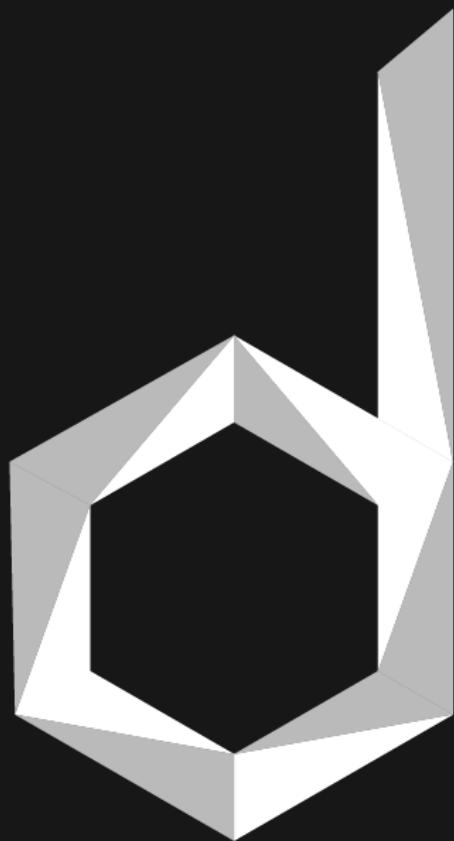


INGENIERÍA MECATRÓNICA



DI_CERO

DIEGO CERVANTES RODRÍGUEZ

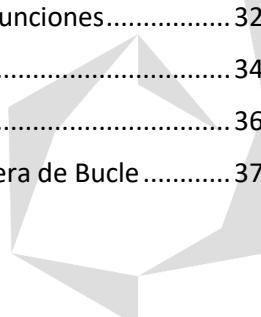
INSTRUMENTACIÓN VIRTUAL

NI LABVIEW 2020 (32-BIT)

WaveChart: Señales Reales y Virtuales
WaveGraph: Arrays y Clusters

Contenido

Introducción Teórica de LabVIEW:.....	3
Introducción al Entorno de LabVIEW:.....	3
Front Panel: Ventana Gris con la Interfaz del Programa	5
Block Diagram: Ventana Blanca con la Lógica del Programa (Bloques)	5
Front Panel o Block Diagram - Show Context Help: Descripción de Bloques	6
Front Panel y Block Diagram: Navegar de una Ventana a Otra	7
Block Diagram - Cambiar Nombre a los Bloques: Nombre de los elementos en el Front Panel	8
Block Diagram - Highlight Execution: Correr Más Lento el Programa.....	9
Coertion dot: Conversión Automática de Datos por Parte de LabVIEW	9
Block Diagram - Clean Up Diagram: Organizar Automáticamente los Bloques del VI	9
Programa: Graficación Waveform Graph y Chart.....	10
Desarrollo del Programa: Creación de una Gráfica con Waveform Graph y Chart	10
Front Panel - Waveform Graph: Ventana que Muestra una Señal (Array)	10
Front Panel - Waveform Chart: Ventana que Muestra una Señal (Dynamic Data)	10
Block Diagram - Wait Until Next ms Multiple: Temporizador en milisegundos.....	11
Block Diagram - Random Number: Creación de Número Aleatorio.....	12
Block Diagram - Build Array: Creación de un Array Vacío de 1 Posición.....	12
Block Diagram - Bucle While: Creación de un Array de 1 Posición con Números Aleatorios	13
Ejecución del Programa: Gráfico de Señal Graph y Chart con un Array	14
Block Diagram - Build Array: Aumento de Tamaño de un Array Vacío de 1 Posición	17
Block Diagram - Bundle: Juntar Varios Tipos de Datos Para Mandarlos a un Cluster.....	18
Block Diagram - DBL Numeric Constant: Creación de Número Tipo Double	18
Block Diagram - Bucle While o For: Si se Usa la Variable i, el Bucle Puede Ser while o for	21
Block Diagram - Bucle While - Túnel (index): Sacar Información del Bucle en un Vector.....	22
Front Panel - Waveform Chart: Clear Chart	24
Front Panel - Waveform Chart o Graph - Graph Palette: Ajuste de Zoom en la Gráfica.....	25
Ejecución del Programa: Gráficos Graph y Chart con Túneles Index.....	30
Block Diagram - Flat Secuence Structure: Ejecución Ordenada Secuencial de Funciones.....	32
Block Diagram - Property Node: Propiedad Específica de Cualquier Bloque.....	34
Ejecución del Programa: Gráficos Graph y Chart Empezando Desde 0	36
Block Diagram - Property Node: Propiedad Específica de Cualquier Bloque Fuera de Bucle	37



Front Panel - Tab Control: Contenedor con Pestañas para Ordenar Elementos.....	40
Front Panel - Decorations: Elementos de Diseño de Interfaz, no afectan al Block Diagram.....	44
Front Panel - SHIFT+Clic Derecho: Pluma para Colorear la Interfaz.....	46



Introducción Teórica de LabVIEW:

LabView sirve para poder usar la computadora como instrumento de medición, monitoreo, control y análisis de procesos y operaciones, esto se hace a través de una frecuencia de muestreo que se relaciona con mediciones de los dispositivos digitales y tiene que ver con la señal de reloj de la tarjeta de desarrollo, indicando cada cuánto tiempo se hará un muestreo de cualquier señal del mundo real.

La diferencia entre los instrumentos virtuales de medición y los reales es más que nada el precio, ya que un osciloscopio cuesta alrededor de \$10,000 y se puede hacer la misma función con LabView y un Arduino, que cuesta alrededor de \$170, además de que es modular, esto implica que se pueden agregar o quitar funcionalidades. La mejor tarjeta de desarrollo para hacer esto es la de NI Instruments, que es la creadora de LabVIEW.

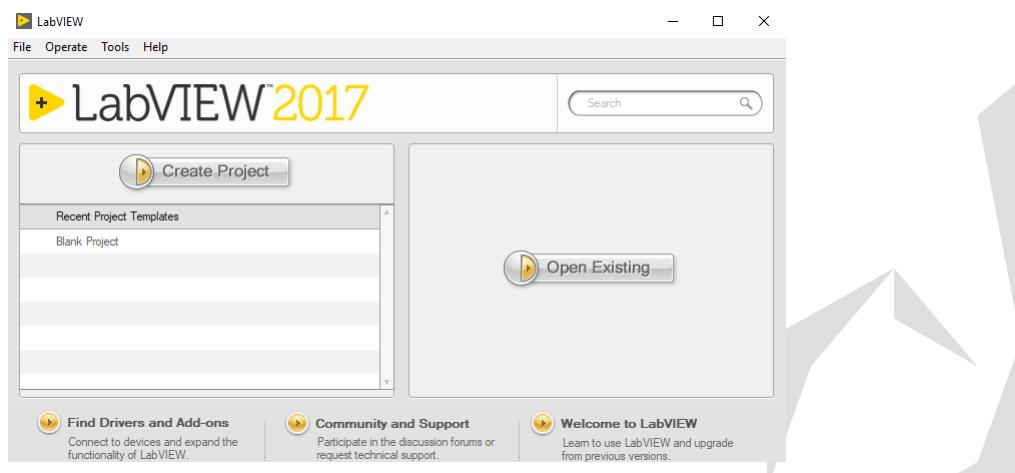
- **Instrumentación Tradicional:** El hardware es más usado, como por ejemplo con los circuitos integrados de un osciloscopio.
- **Instrumentación Virtual:** El software es el más utilizado y sus funciones son modulares, como lo es en una tarjeta de desarrollo de National Instruments.

La instrumentación virtual es empleada para la gestión de sistemas industriales y muy utilizado en compañías como: Ford, SpaceX, Accenture, Bosch, etc.

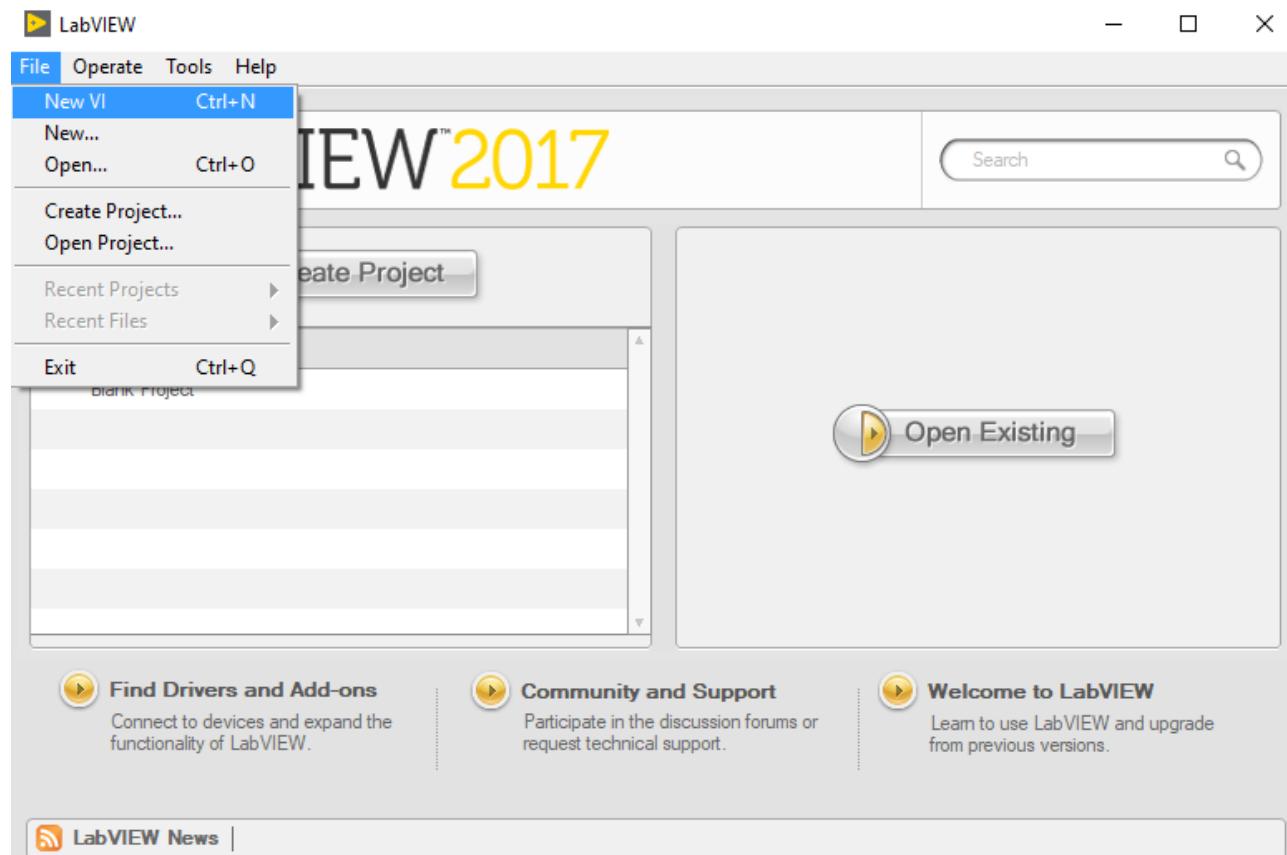


Introducción al Entorno de LabVIEW:

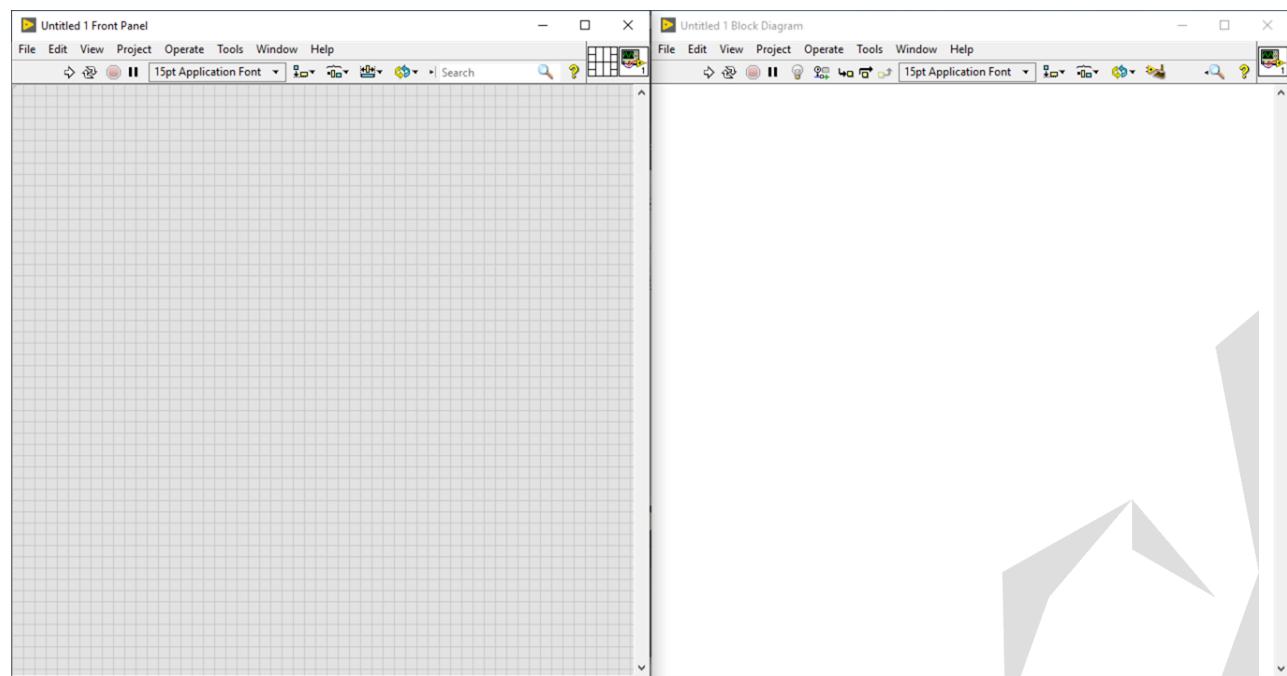
Un nuevo proyecto de LabView se abre por medio del botón de Create project que aparece inmediatamente cuando abra el programa.



VI se refiere a Virtual Instrument.

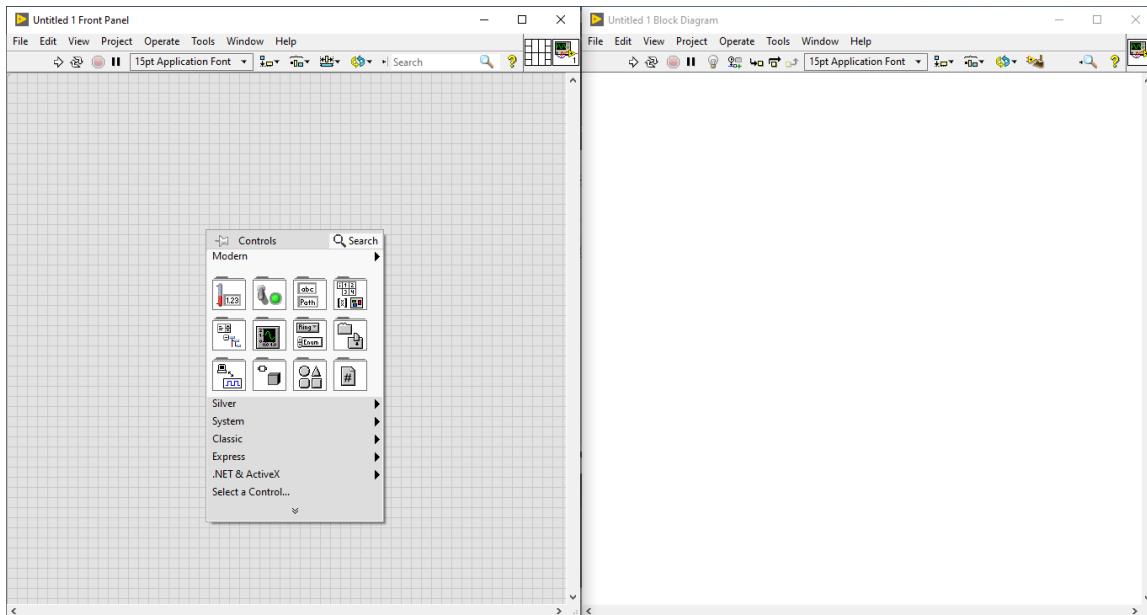


Al hacerlo me abrirá estas dos ventanas, en una de ellas se creará el programa con bloques (Ventana Block Diagram) y en la otra se verá la interfaz (Ventana Front Panel).



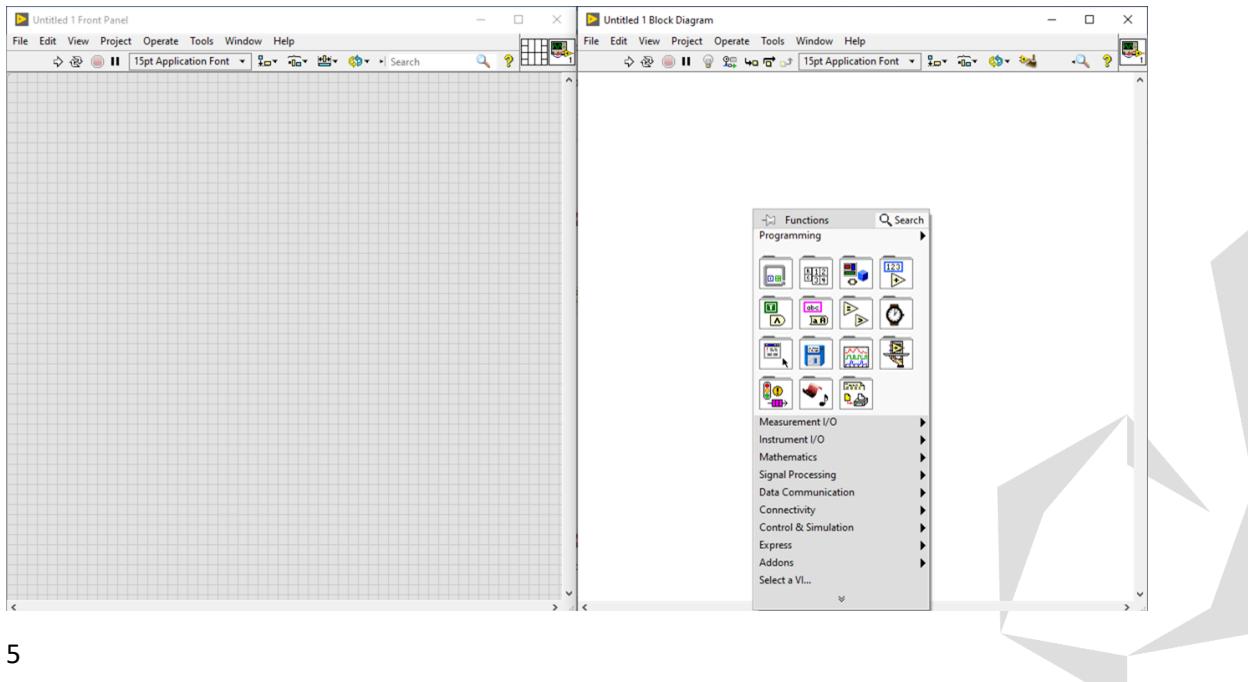
Front Panel: Ventana Gris con la Interfaz del Programa

En la ventana gris llamada **Front Panel**, es donde se observa la interfaz del Programa y se cuenta con el control palette que sirve para poder añadir elementos gráficos a la interfaz y aparece dando clic derecho en la pantalla gris. Si no aparece la otra ventana (blanca) por default, se debe seleccionar la opción Window → Show Block Diagram y con ello aparecerá.



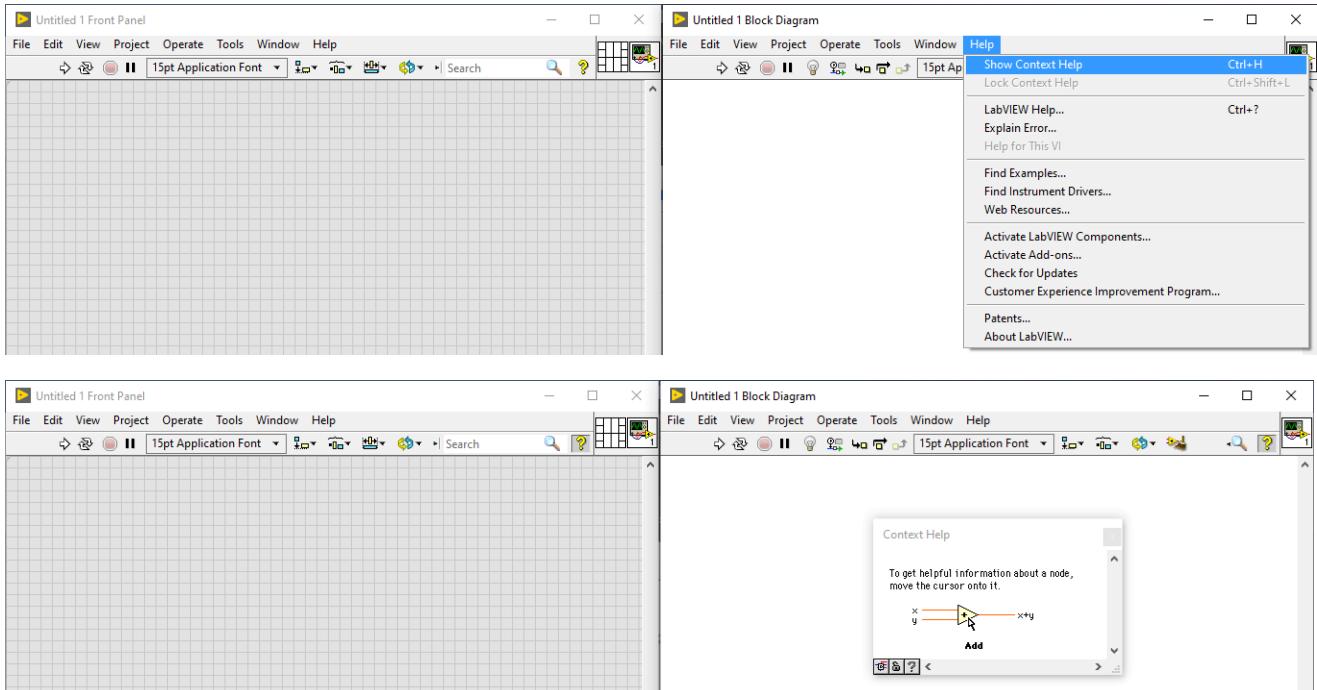
Block Diagram: Ventana Blanca con la Lógica del Programa (Bloques)

En la ventana blanca llamada **Block Diagram** aparece la paleta de funciones que sirve para introducir los elementos de programación en forma de bloques que se conectarán entre ellos y describirán la función del programa, aparece dando clic derecho en la pantalla gris. Si no aparece la ventana gris se debe seleccionar la opción Windows → Show Front Panel y con ello aparecerá.



Front Panel o Block Diagram - Show Context Help: Descripción de Bloques

Seleccionando la opción de Help → Show Context Help, aparecerá una ventana emergente que explicará las propiedades de los bloques que se puede seleccionar, mostrando una descripción de su función, imágenes explicativas y significado de sus pines de entrada y salida.

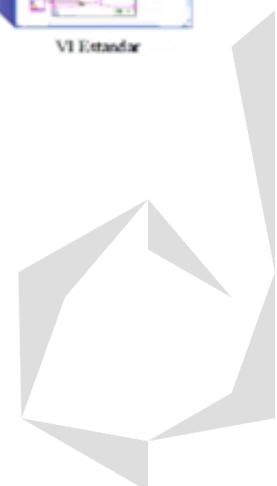


Las funciones o subrutinas son los elementos más básicos que pueden existir en LabView, dentro de ellas existe un código de bloque propio que describe sus funciones, pero además se cuenta con otros elementos:

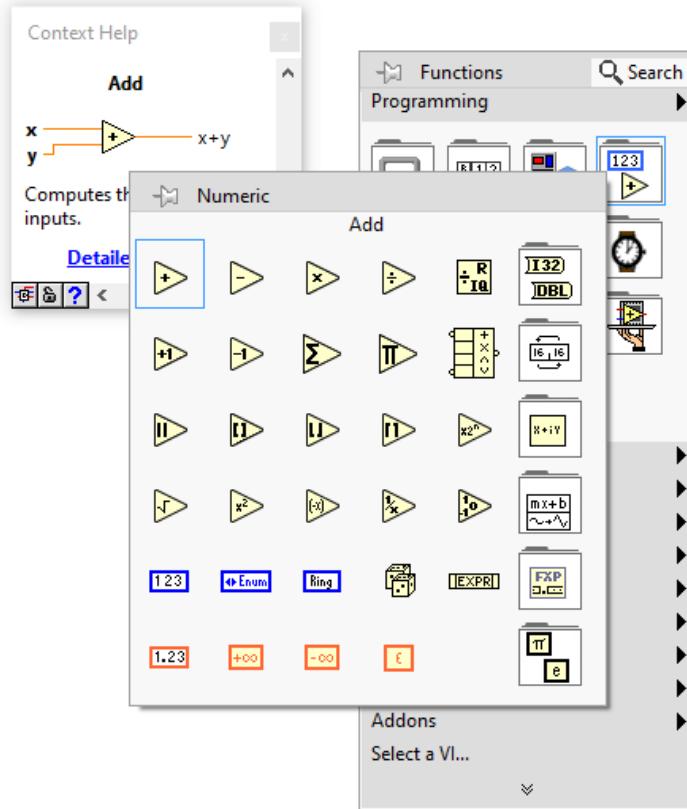
VIs Express, VIs y Funciones



- **VIs Expreso:** VIs interactivos con pagina de dialogo configurable
- **VIs estándar:** VIs modulares y personalizables mediante cableado
- **Funciones:** Elementos fundamentales de operación de LabVIEW; no contiene panel frontal o diagrama de bloque



En un bloque de código, las terminales que aparezcan en negritas son las que a fuerza deben estar conectadas a algo, las que no estén en negritas no deben estar conectadas a nada forzosamente.

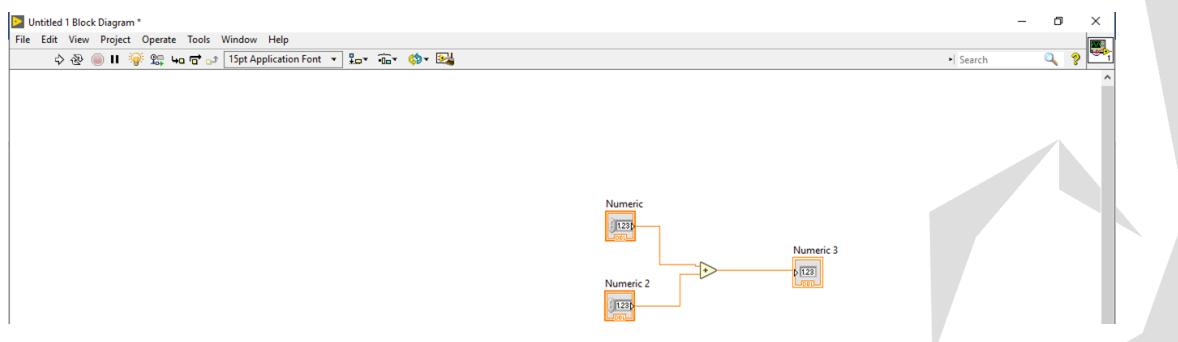


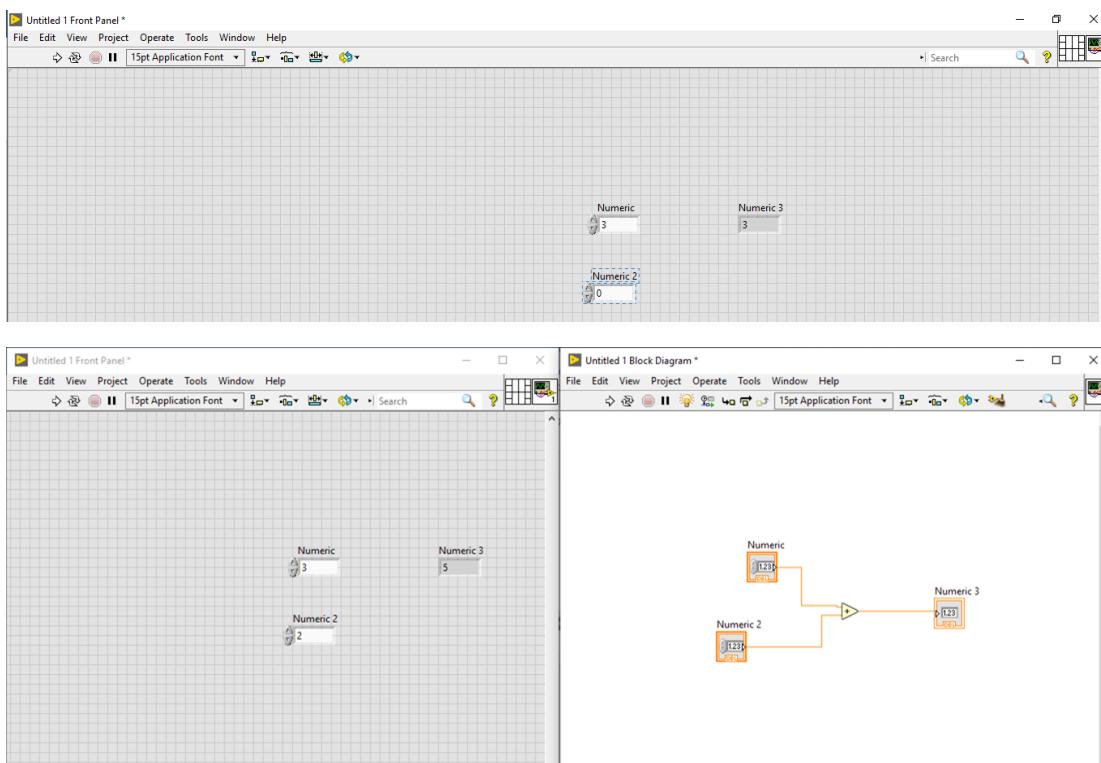
El programa es autocompilable, es decir que se corre por sí solo, por lo que si la flechita aparece rota es porque hay un error en el programa.



Front Panel y Block Diagram: Navegar de una Ventana a Otra

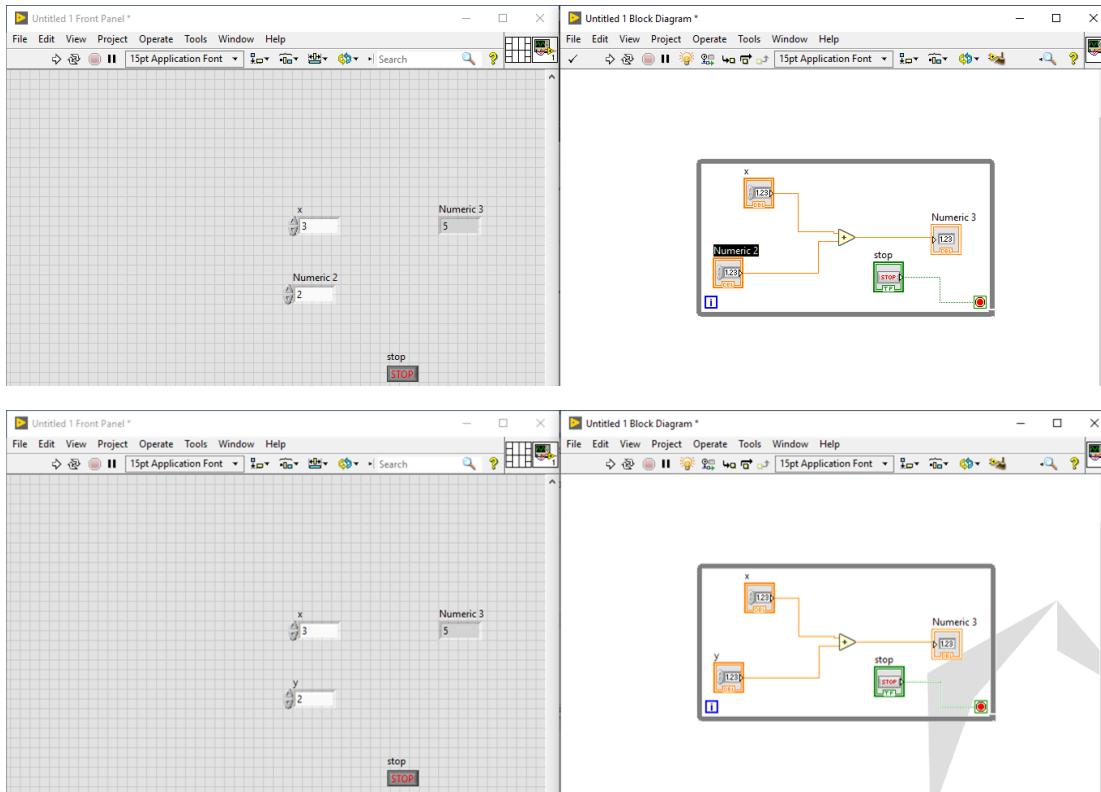
Al dar doble clic en el bloque de la pantalla blanca, me llevará al punto donde se encuentra el mismo bloque, pero en la pantalla gris.

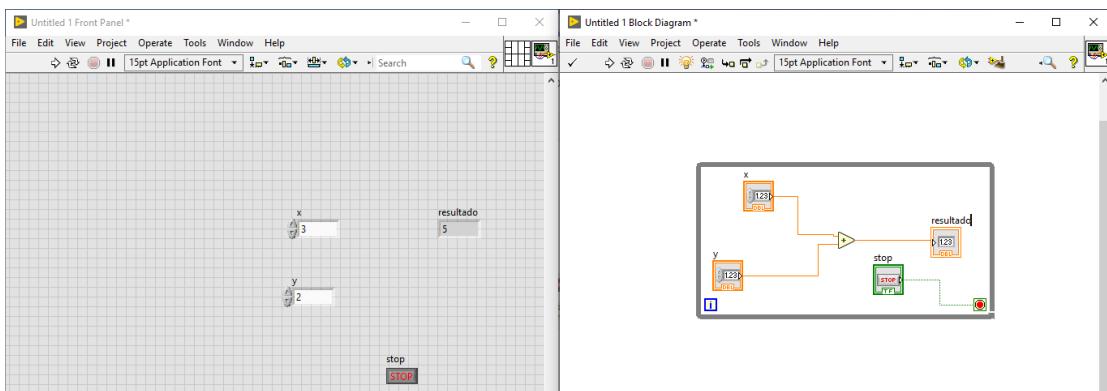




Block Diagram - Cambiar Nombre a los Bloques: Nombre de los elementos en el Front Panel

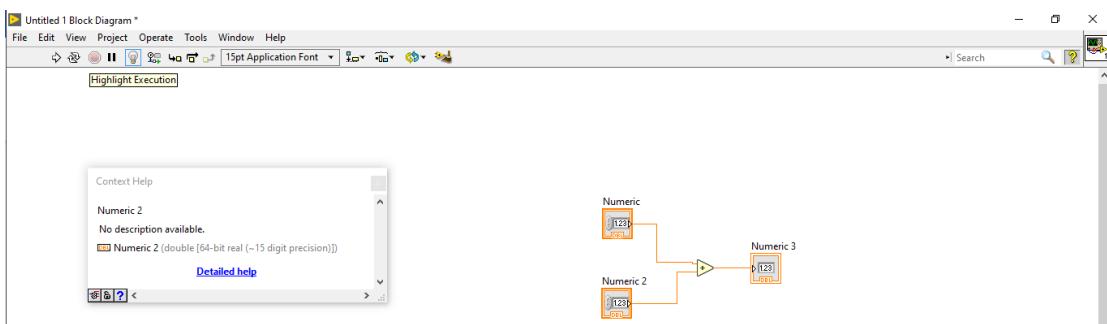
El nombre de los elementos de las interfaces se puede cambiar desde el Block Diagram, cambiándole literal el nombre a los bloques.





Block Diagram - Highlight Execution: Correr Más Lento el Programa

Podemos presionar el foquito del menú superior para ver el funcionamiento de programa de manera más lenta.



Coersion dot: Conversión Automática de Datos por Parte de LabVIEW

Aparece un punto rojo en la terminal del bloque llamado coercion dot, este lo que me dice es que los tipos de datos en la conexión son distintos, por lo que LabVIEW está forzando una conversión de un tipo de dato a otro, el problema es que en este tipo de conversión yo no sé si se están perdiendo datos, por eso debemos evitar el uso de coercion dots porque usa direcciones de memoria o recursos de la computadora sin que yo tenga control de ellos.

Block Diagram - Clean Up Diagram: Organizar Automáticamente los Bloques del VI

Con el botón de Clean Up Diagram que se encuentra en la parte superior derecha del Block Diagram se organizan mejor y de forma automática mis elementos.

Programa: Graficación Waveform Graph y Chart

Generación de gráficas dinámicas y estáticas con las herramientas Waveform Graph y Waveform Chart, además de especificar las diferencias entre ambas.

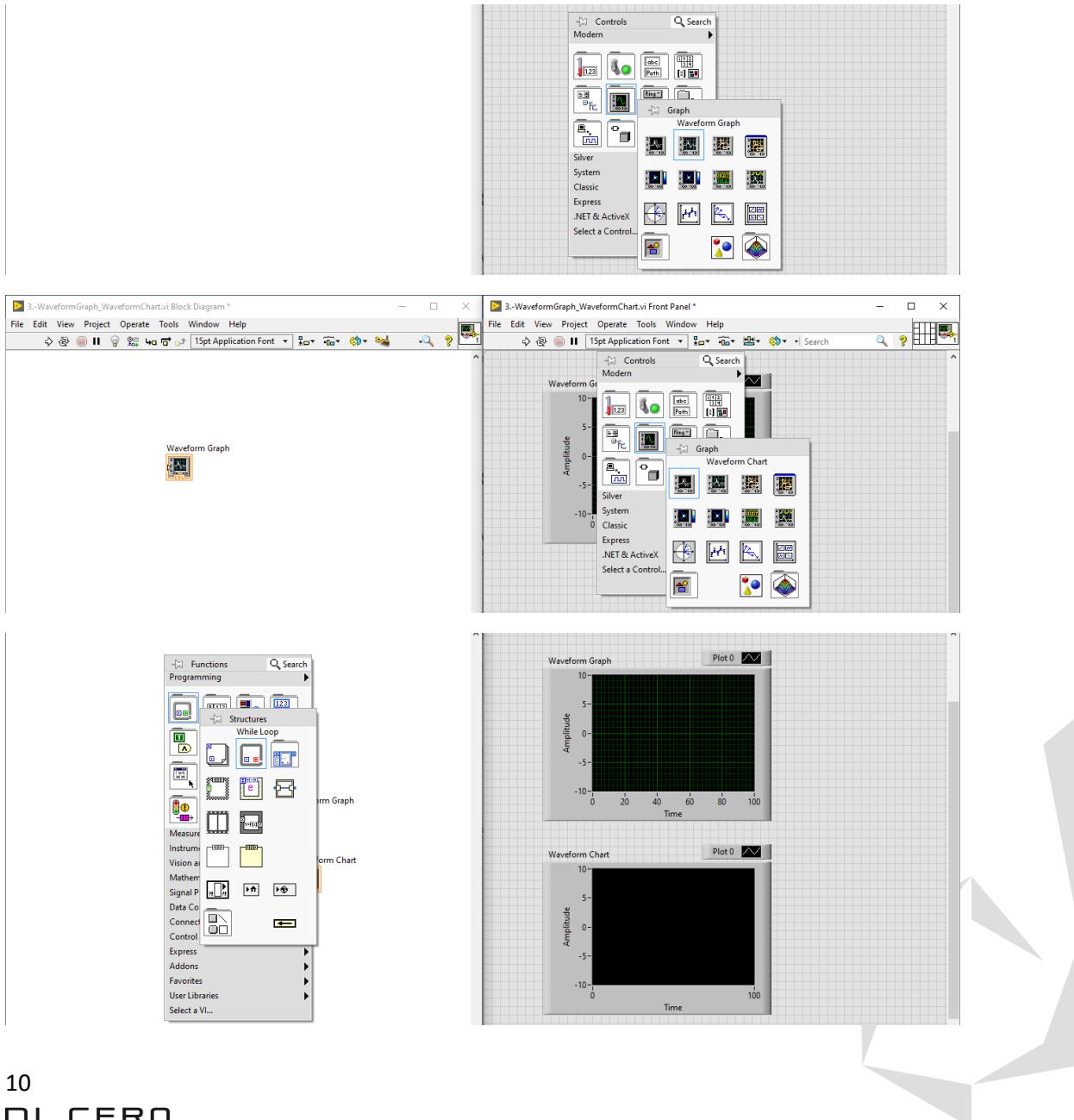
Desarrollo del Programa: Creación de una Gráfica con Waveform Graph y Chart

Front Panel - Waveform Graph: Ventana que Muestra una Señal (Array)

El Waveform Graph muestra gráficas de tipo Array, las cuales son cualquier tipo de grupos de números.

Front Panel - Waveform Chart: Ventana que Muestra una Señal (Dynamic Data)

El Waveform Chart muestra gráficas de tipo Dynamic Data, las cuales son señales virtuales o reales.

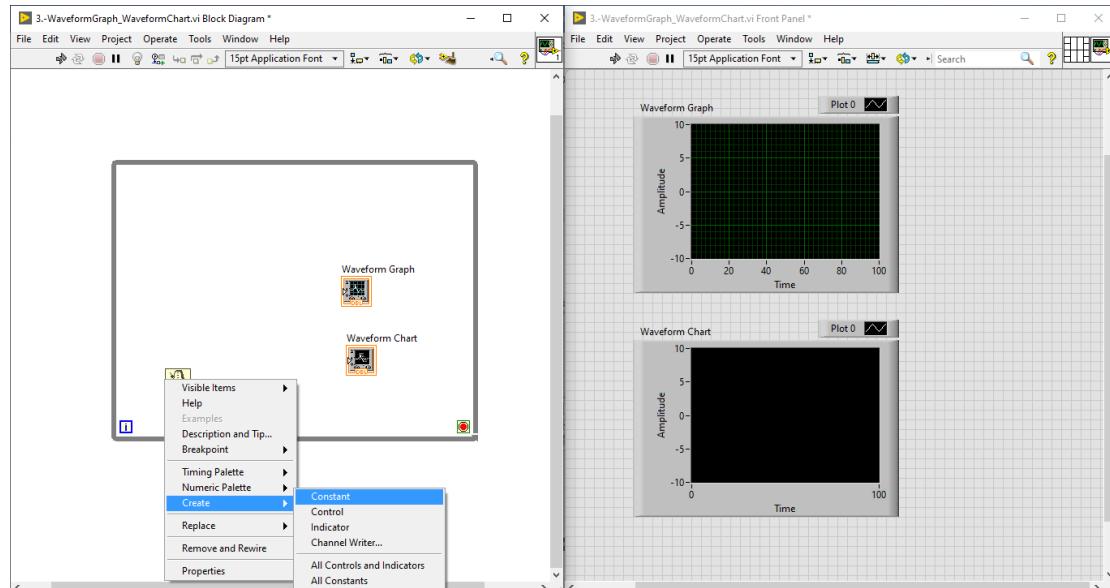
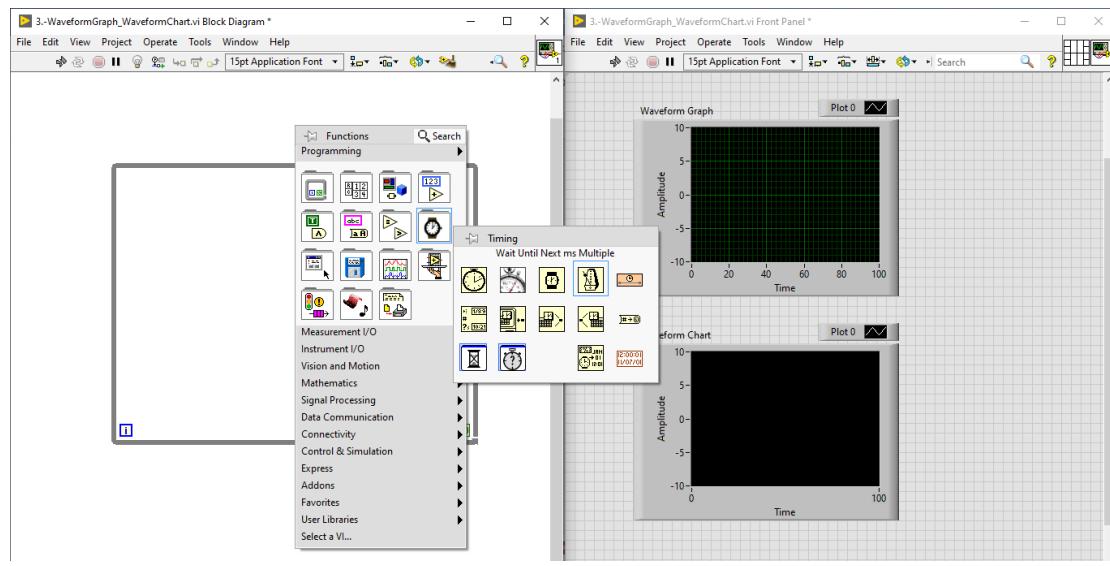


Debemos tomar en cuenta porque se utilizan dos bloques distintos de graficación en LabVIEW:

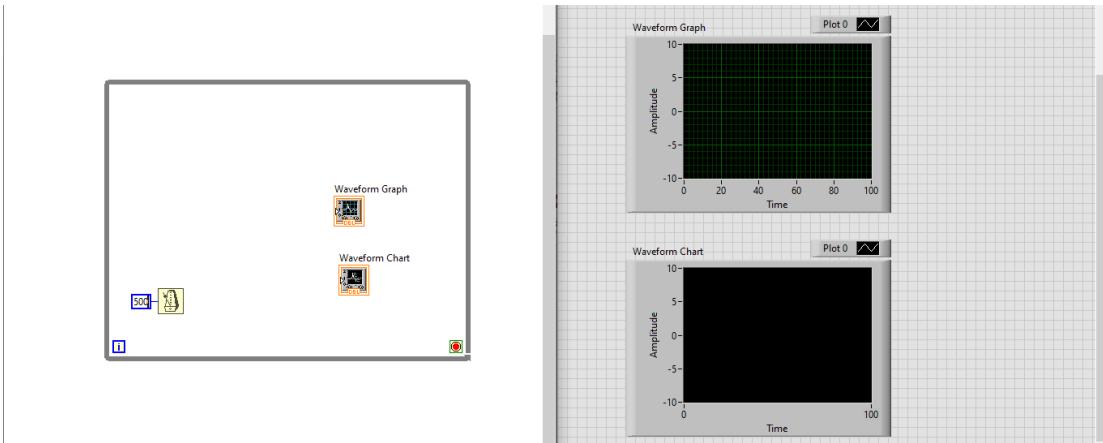
- **Waveform Chart:** Grafica señales de tipo Dynamic Data, las cuales pueden ser señales reales captadas por una tarjeta de desarrollo (Arduino, Nexys (FPGA), tarjeta de desarrollo de National Instruments, etc.) o señales virtuales creadas por cualquier programa.
- **Waveform Graph:** Grafica Grupos de datos numéricos cualquiera, de tipo Array o Cluster (agrupación de varios tipos de datos distintos en uno solo llamado Cluster).

Block Diagram - Wait Until Next ms Multiple: Temporizador en milisegundos

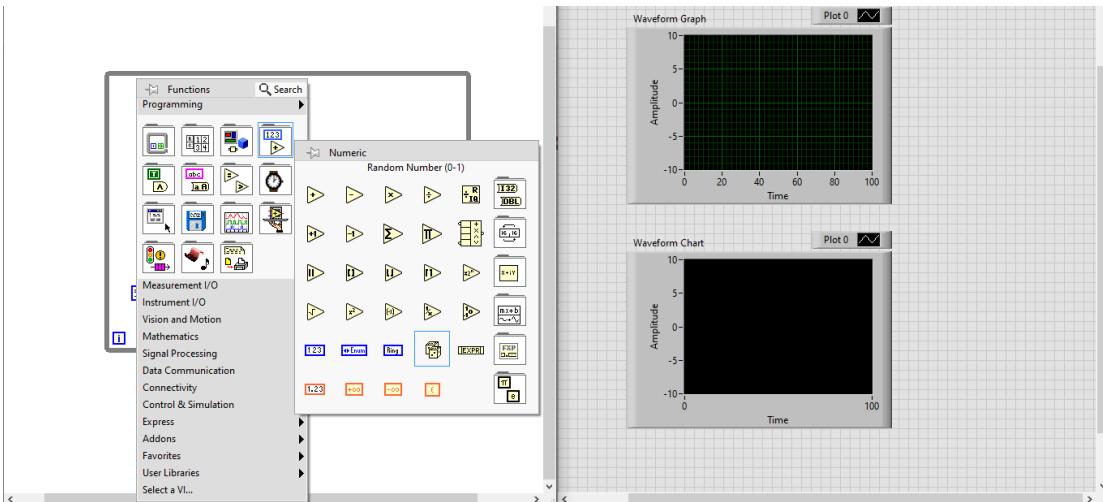
El bloque de wait until se utiliza cuando se debe hacer un retraso de tiempo (delay) por ciertos segundos, para de esta manera parar la ejecución del programa por un cierto tiempo, en específico para que corra este bloque se debe crear una constante dando clic derecho sobre ella y declarando el tiempo en milisegundos.



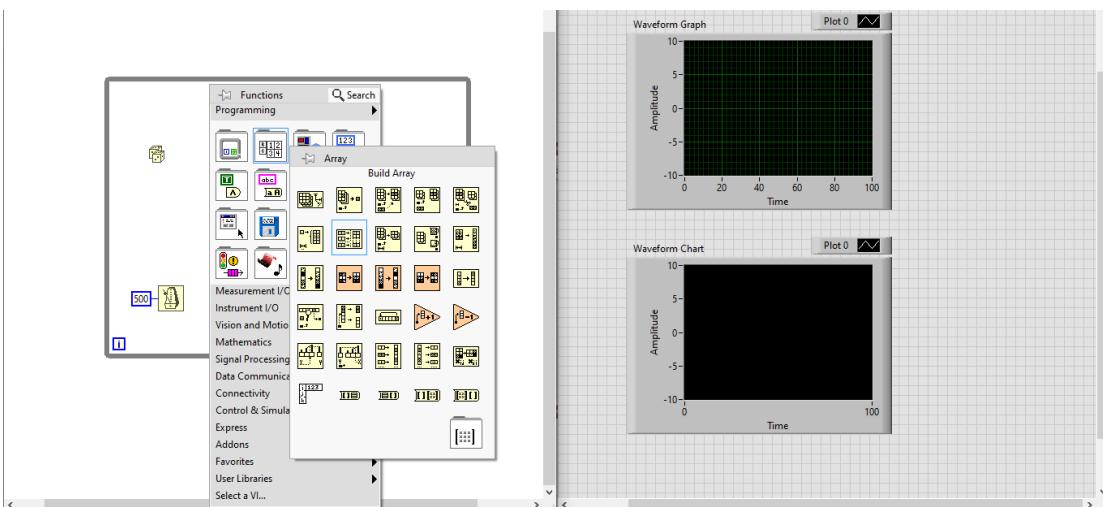
Con esto voy a hacer un muestreo, para que se grafique más lento, esto representa la duración del ciclo en milisegundos, cada muestreo se dará en este caso cada 500 milisegundos.



Block Diagram - Random Number: Creación de Número Aleatorio



Block Diagram - Build Array: Creación de un Array Vacío de 1 Posición

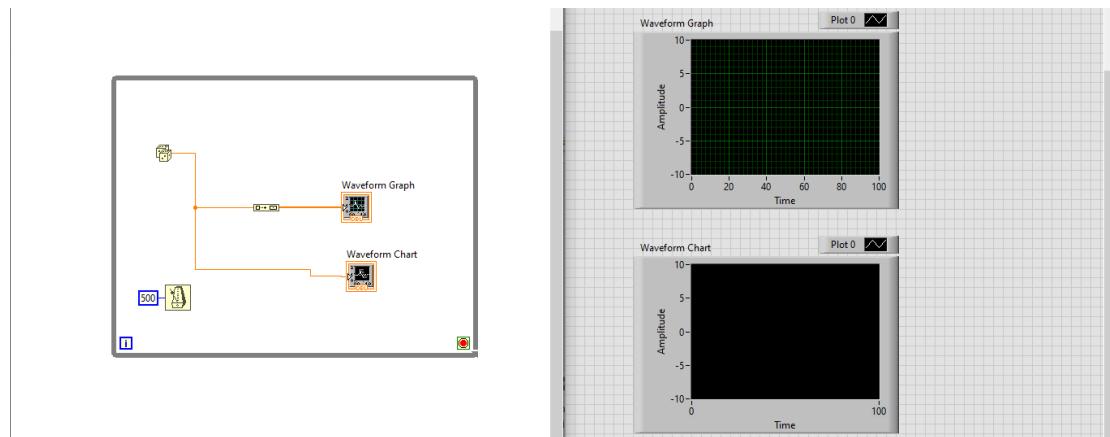


Block Diagram - Bucle While: Creación de un Array de 1 Posición con Números Aleatorios

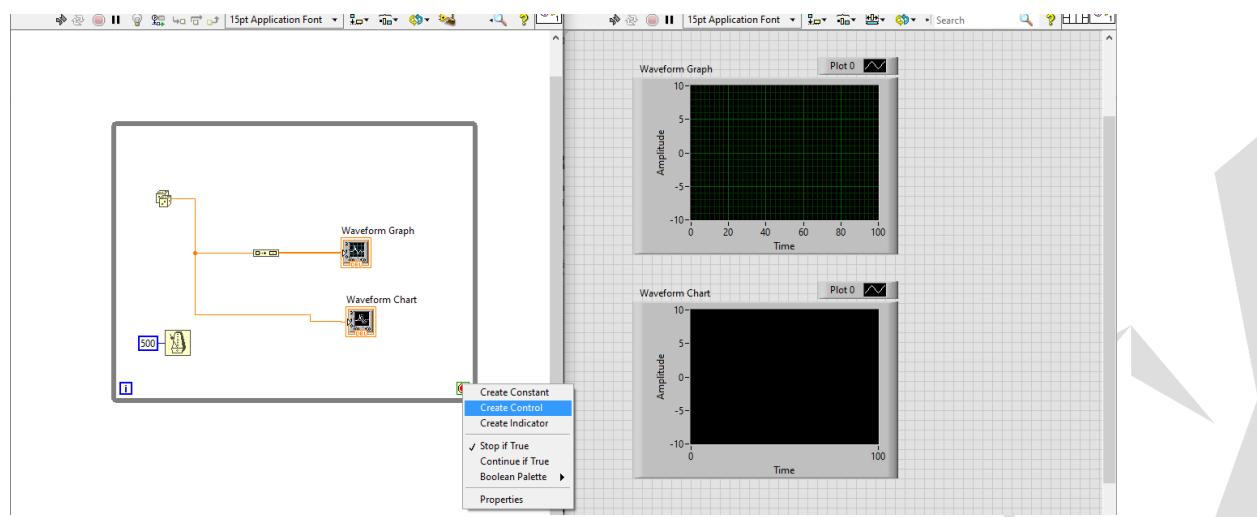
Con el bloque de Random Number se crea un número aleatorio, luego con el bloque de Wait Until se espera cada 500 ms para ejecutar cada vez el bucle While y crear nuevos números, después estos números los meteré dentro del bloque Build Array, que representa un array vacío, para de esta manera crear una serie de valores numéricos agrupados que se podrá graficar en mi bloque de **Waveform Chart** y **Waveform Graph**.

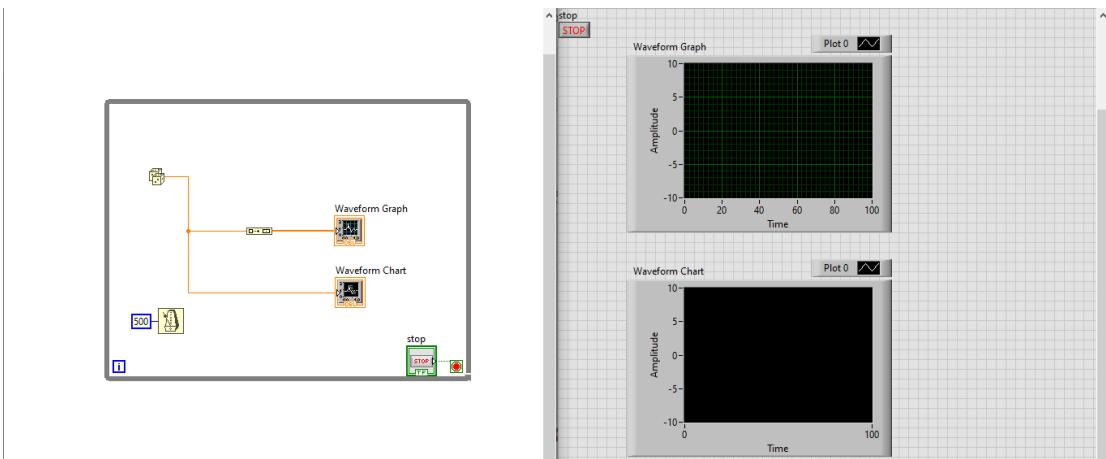
- **Waveform Chart:** Grafica señales de tipo Dynamic Data, las cuales pueden ser señales reales captadas por una tarjeta de desarrollo (Arduino, Nexys (FPGA), tarjeta de desarrollo de National Instruments, etc.) o señales virtuales creadas por cualquier programa.
- **Waveform Graph:** Grafica Grupos de datos numéricos cualquiera, de tipo Array o Cluster (agrupación de varios tipos de datos distintos en uno solo llamado Cluster).

En el Waveform Graph se verá el número individual que se esté graficando en cada momento y en el Waveform Chart se pondrá la gráfica tal cual.

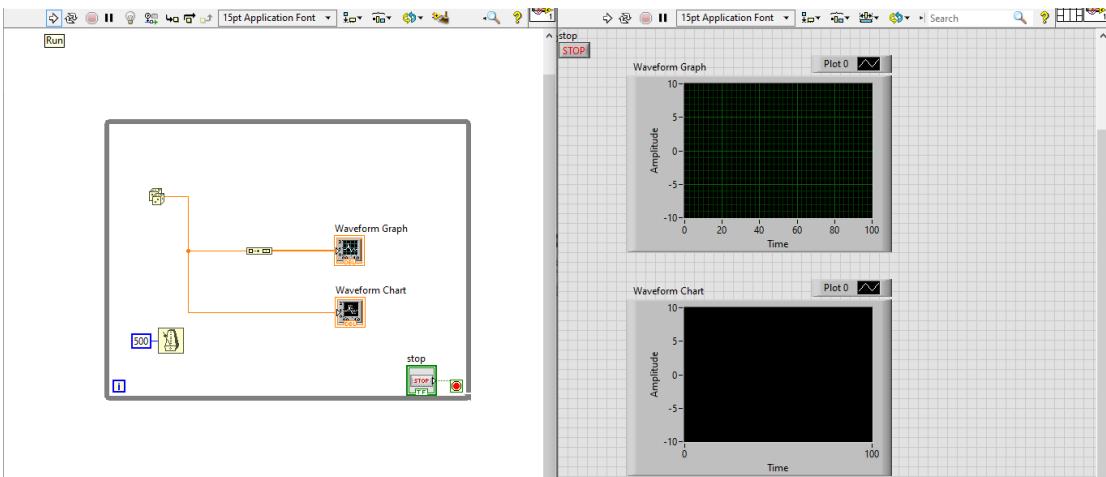


Se usa el ciclo while para que el código se esté repitiendo sin fin, creando así la gráfica, hasta que el usuario le dé clic en el botón de stop que vamos a construir como control del bucle, esto lo haremos dando clic derecho sobre el botón rojo de la esquina superior derecha del bucle while y añadiendo un control, que es en sí el botón de STOP.



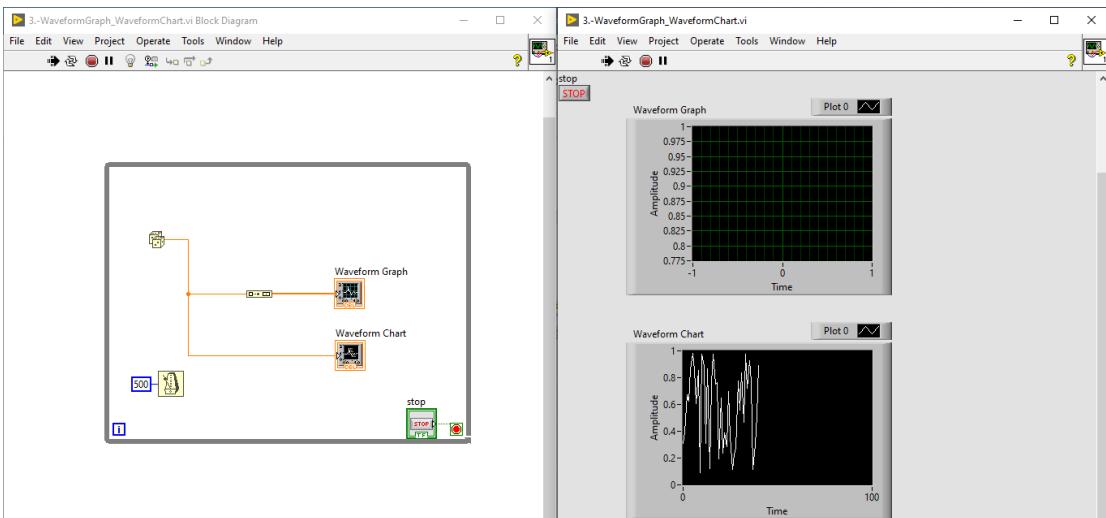


Cuando corra el programa podré ver como se grafica lentamente la señal.

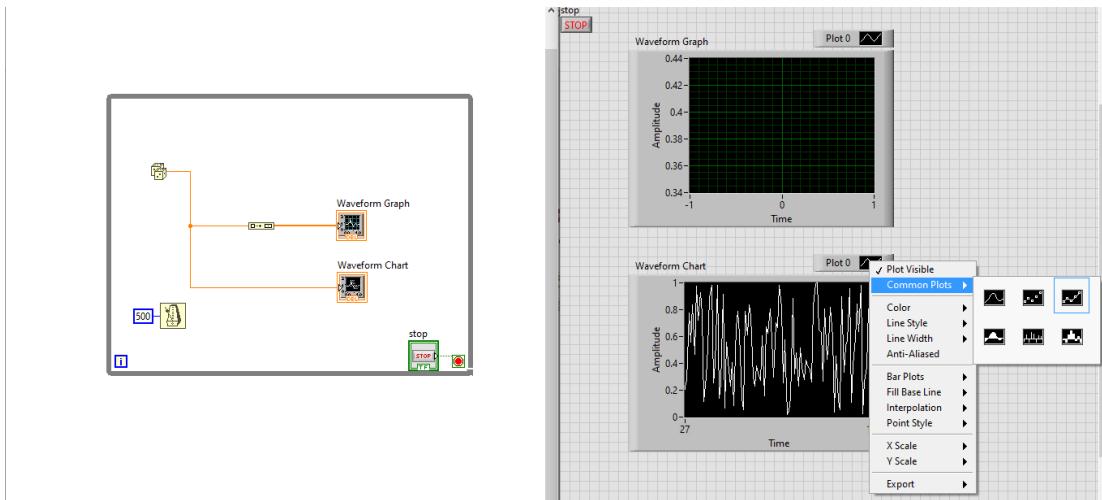


Ejecución del Programa: Gráfico de Señal Graph y Chart con un Array

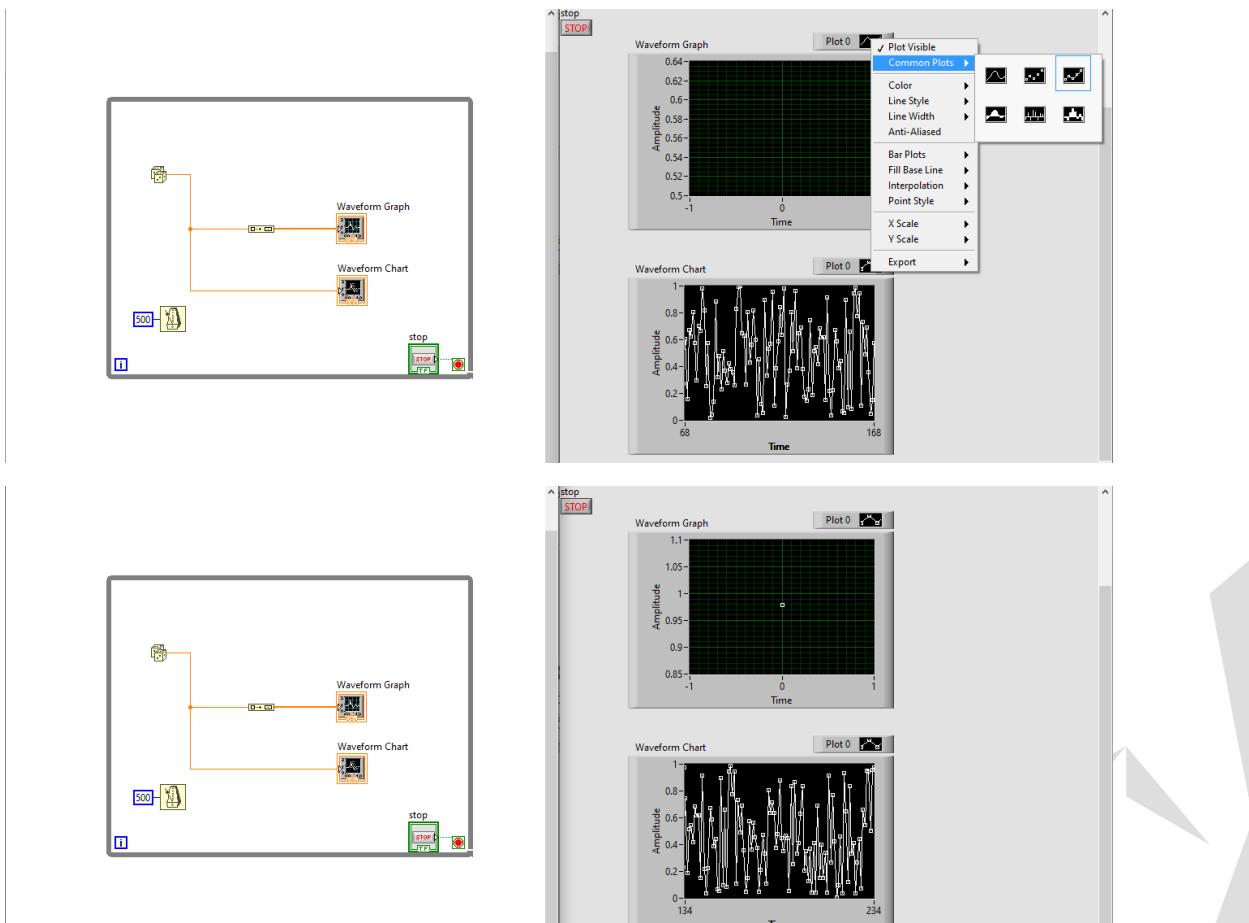
Se grafica lentamente la señal porque le di un periodo de 500 ms al temporizador, si este número lo bajo, la señal se graficará mas rápidamente.

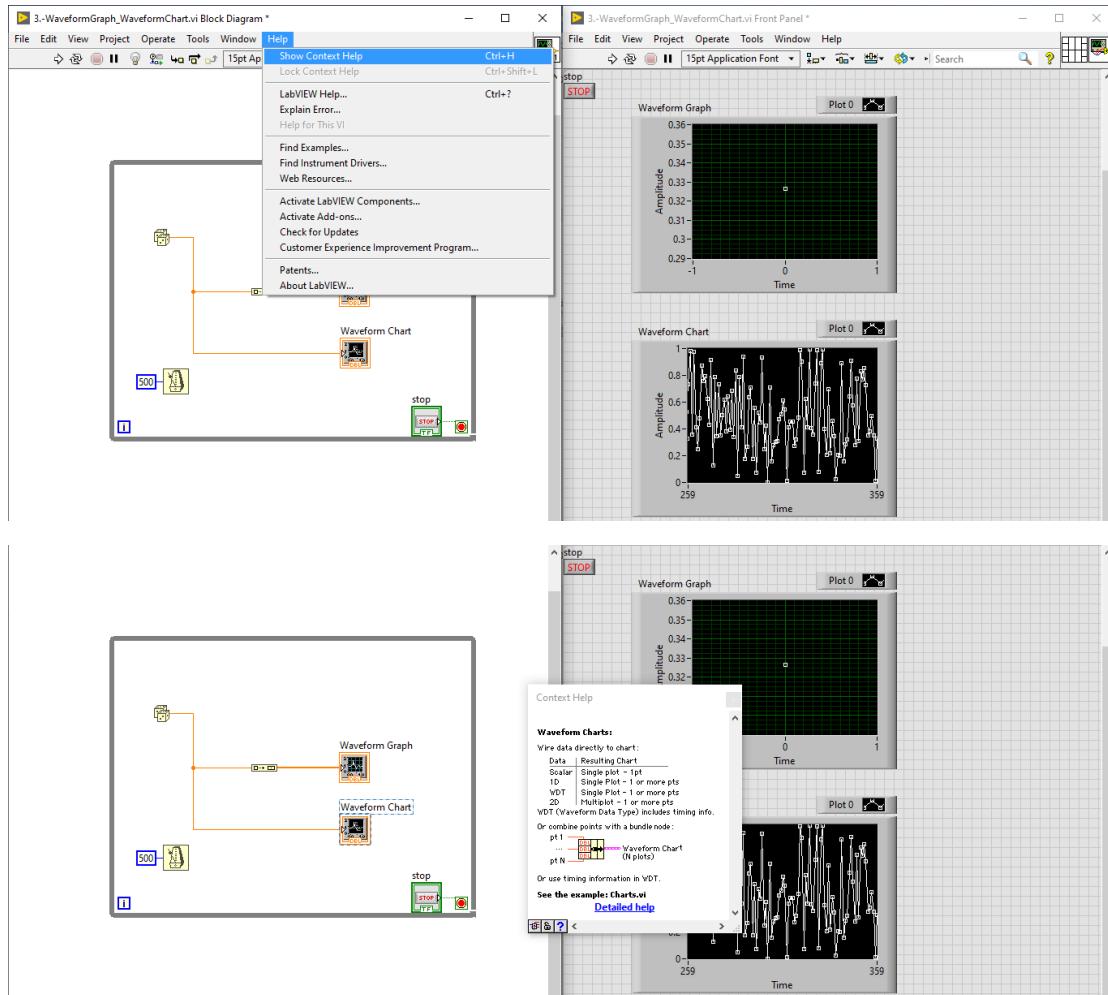


Para que en la gráfica del **Waveform Chart (Grafica Dynamic Data)** se pongan puntos en sus valores, podemos dar clic en su parte superior derecha que dice Plot 0 y seleccionar la opción de Common Plots. Se puede observar la gráfica porque al bloque le llega un simple valor numérico que cambia de valor, eso es un Dynamic Data.

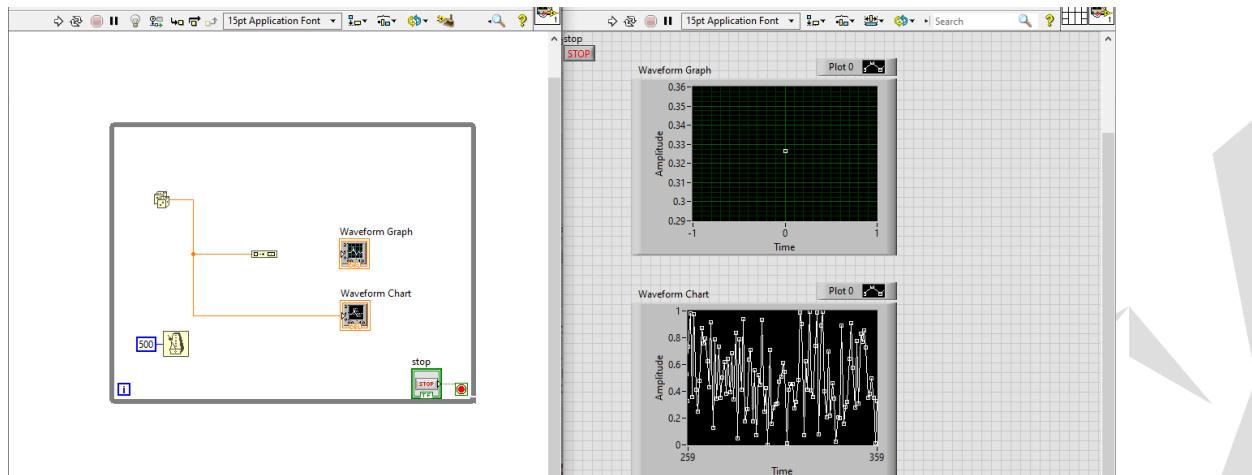


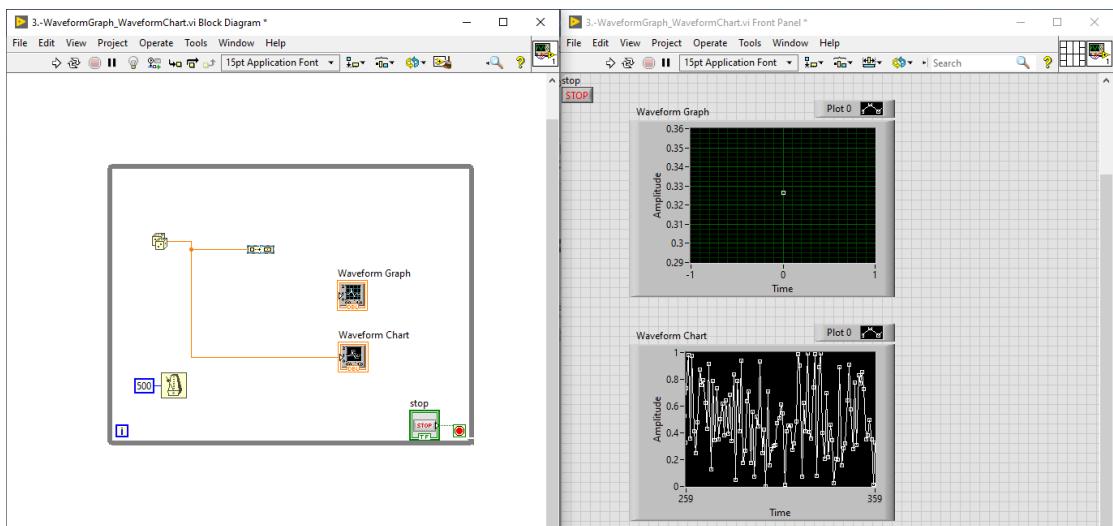
Para que se pueda ver el punto del número graficado en el **Waveform Graph (Grafica Arrays o Clusters)** debo hacer lo siguiente:



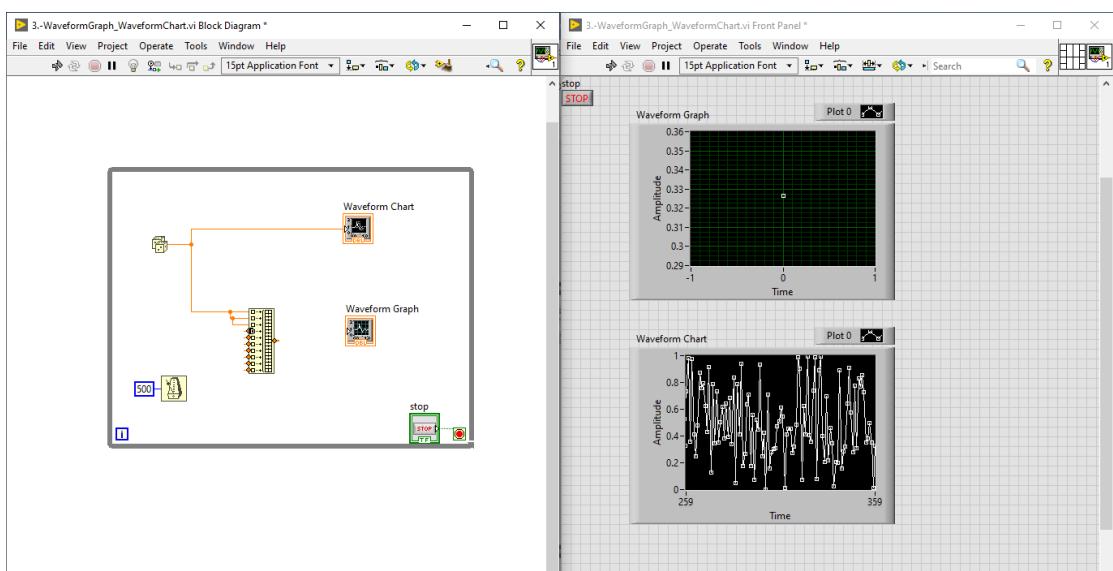
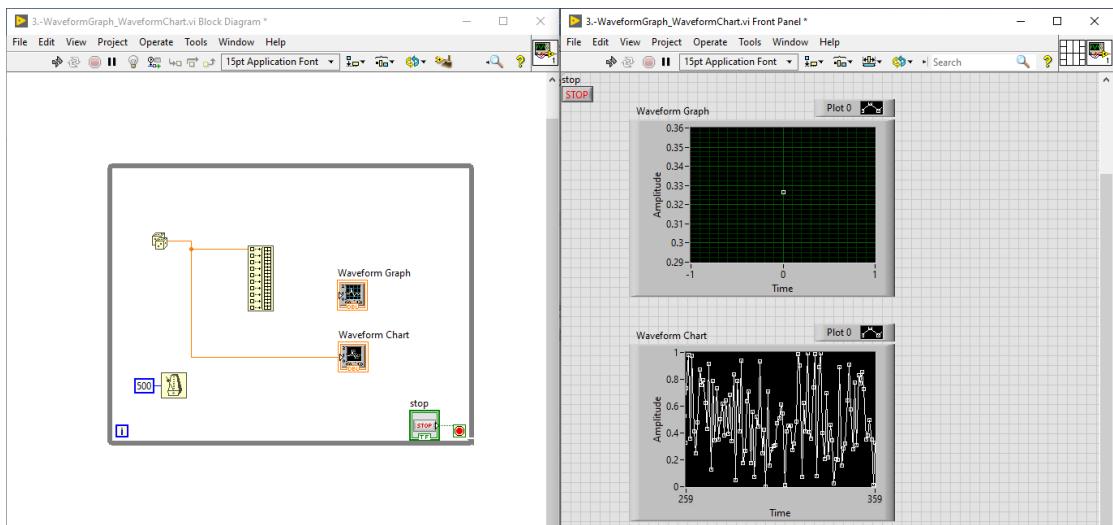


El **Waveform Graph (Grafica Arrays o Clusters)** se usa para graficar una señal después de haber hecho alguna operación matemática, en el ejemplo anterior se usó para analizar lo que salió de una serie de Fourier, si quiero usar este debo usar un bloque llamado **Bundle** para definir bien el vector que quiero graficar, no basta con crear un simple Array, además de que el Array creado es de 1 sola posición, por eso solo se puede almacenar y graficar un punto.



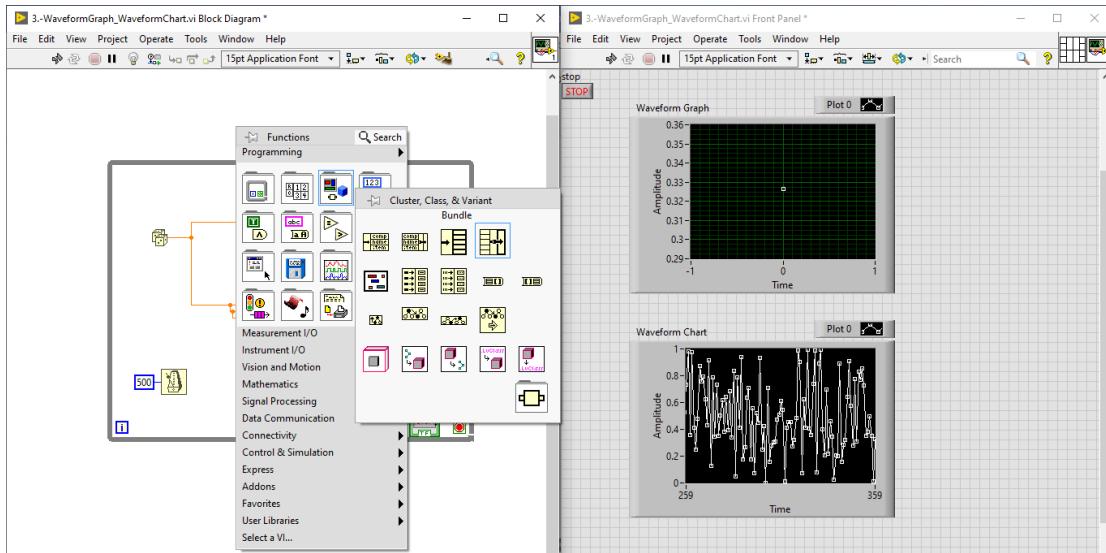


Block Diagram - Build Array: Aumento de Tamaño de un Array Vacío de 1 Posición

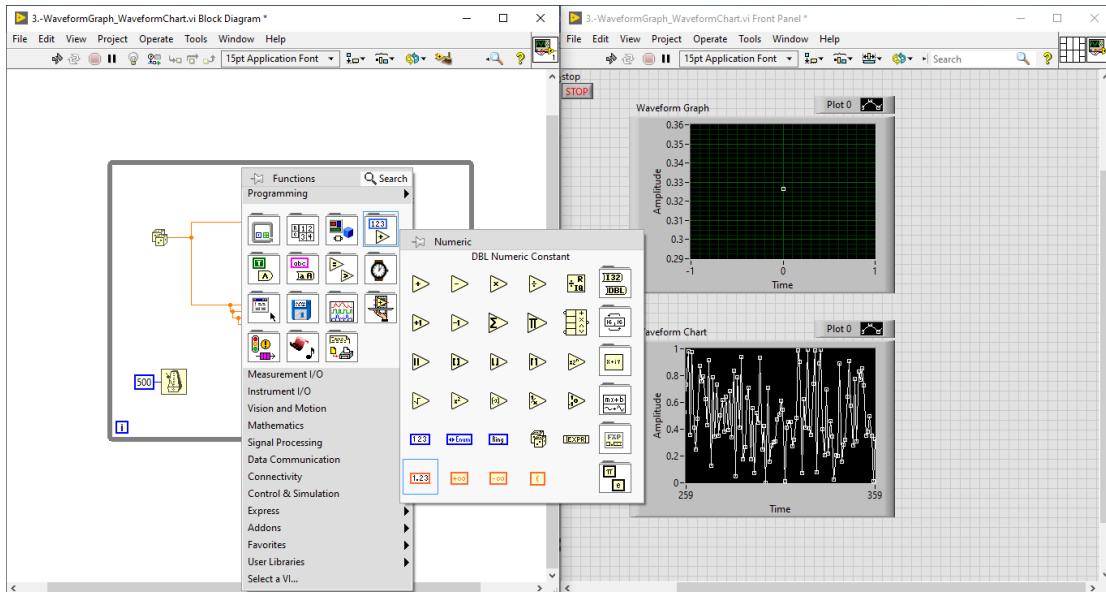


Block Diagram - Bundle; Juntar Varios Tipos de Datos Para Mandarlos a un Cluster

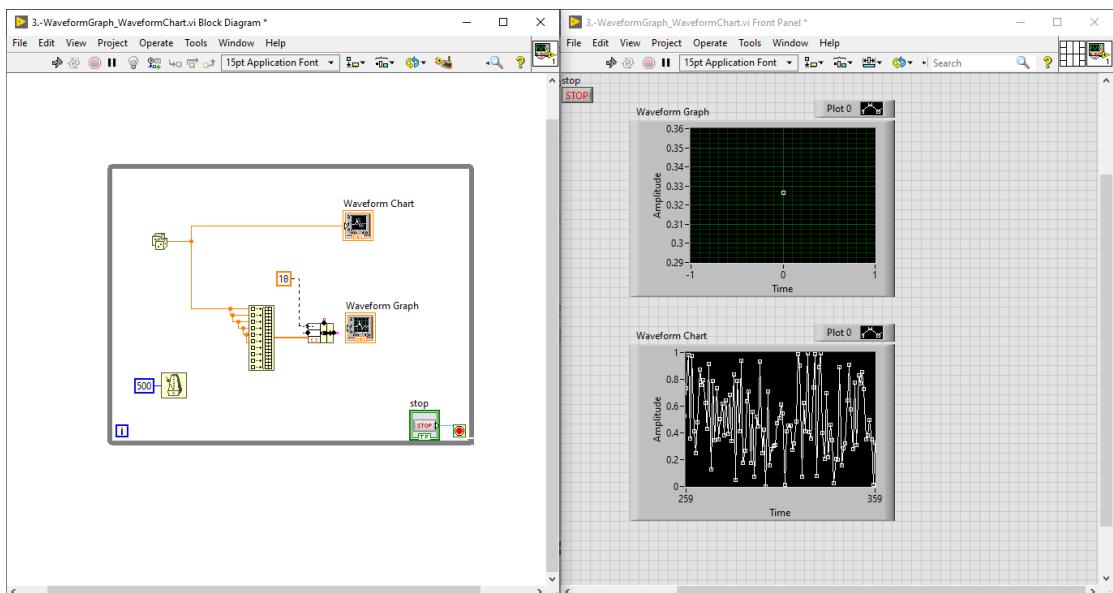
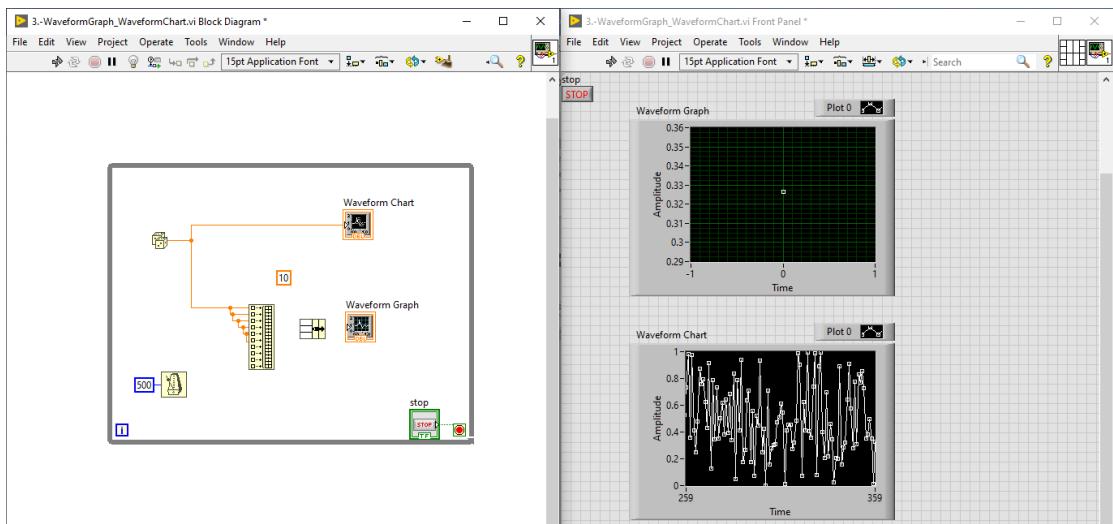
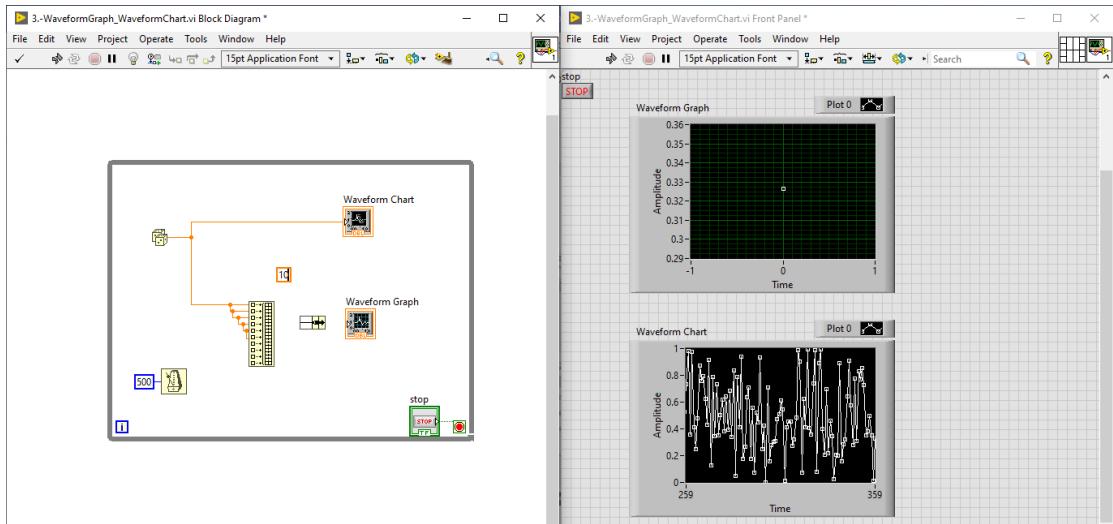
El tipo de dato Cluster es perteneciente únicamente al entorno de desarrollo de LabVIEW y representa un tipo de dato definido por el usuario que recibe y encapsula varios, pero para poder realizar esto se debe incluir un bloque intermedio llamado Bundle, que se encarga de juntar todos los tipos de datos distintos o iguales antes de enviarlos al bloque de Cluster.



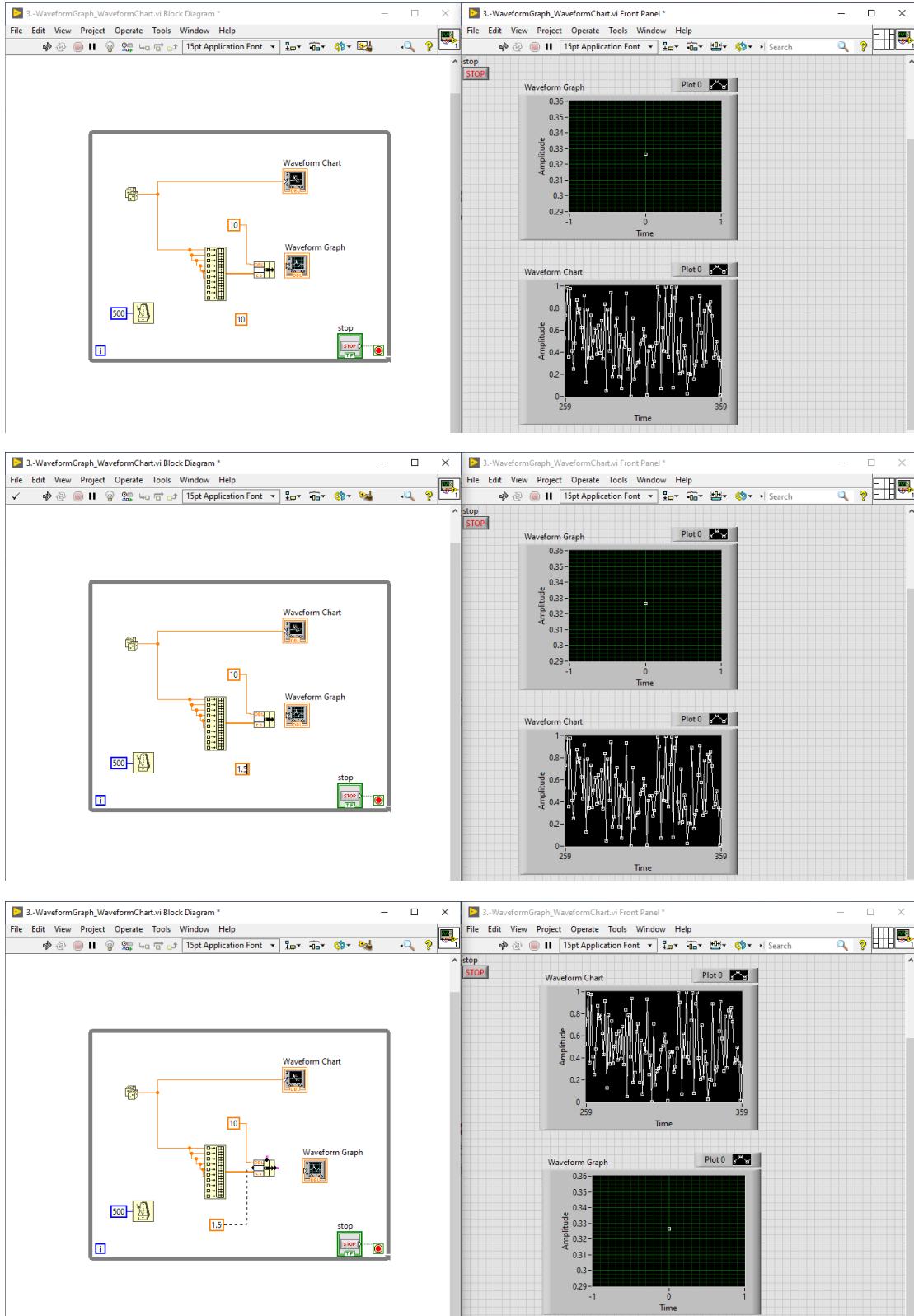
Block Diagram - DBL Numeric Constant: Creación de Número Tipo Double

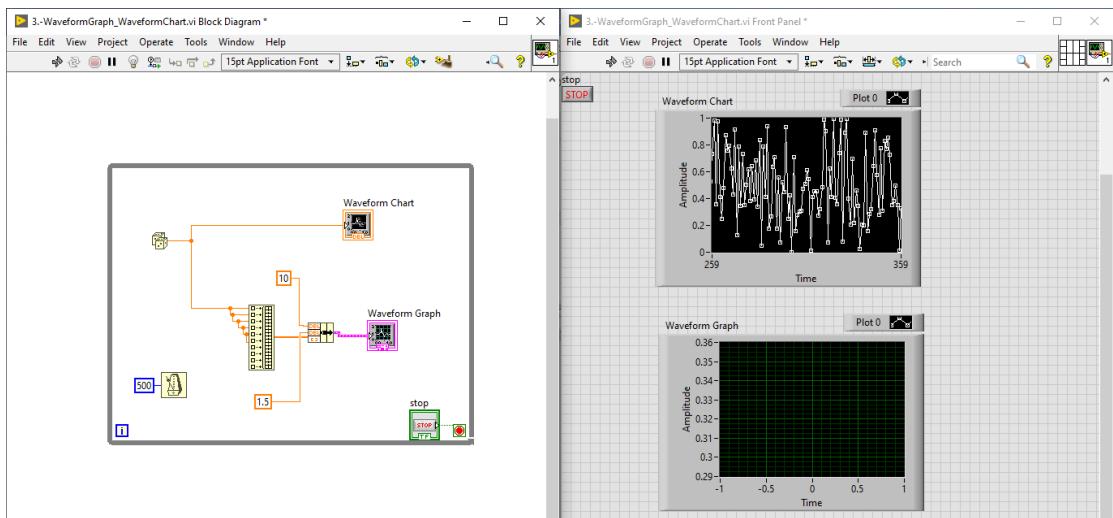


Ahora se van a inicializar los datos del Bundle, incluyendo en 2 números double (decimales y enteros), además de incluir el Array llenado con números aleatorios provenientes del bucle while y el bloque que genera números aleatorios.



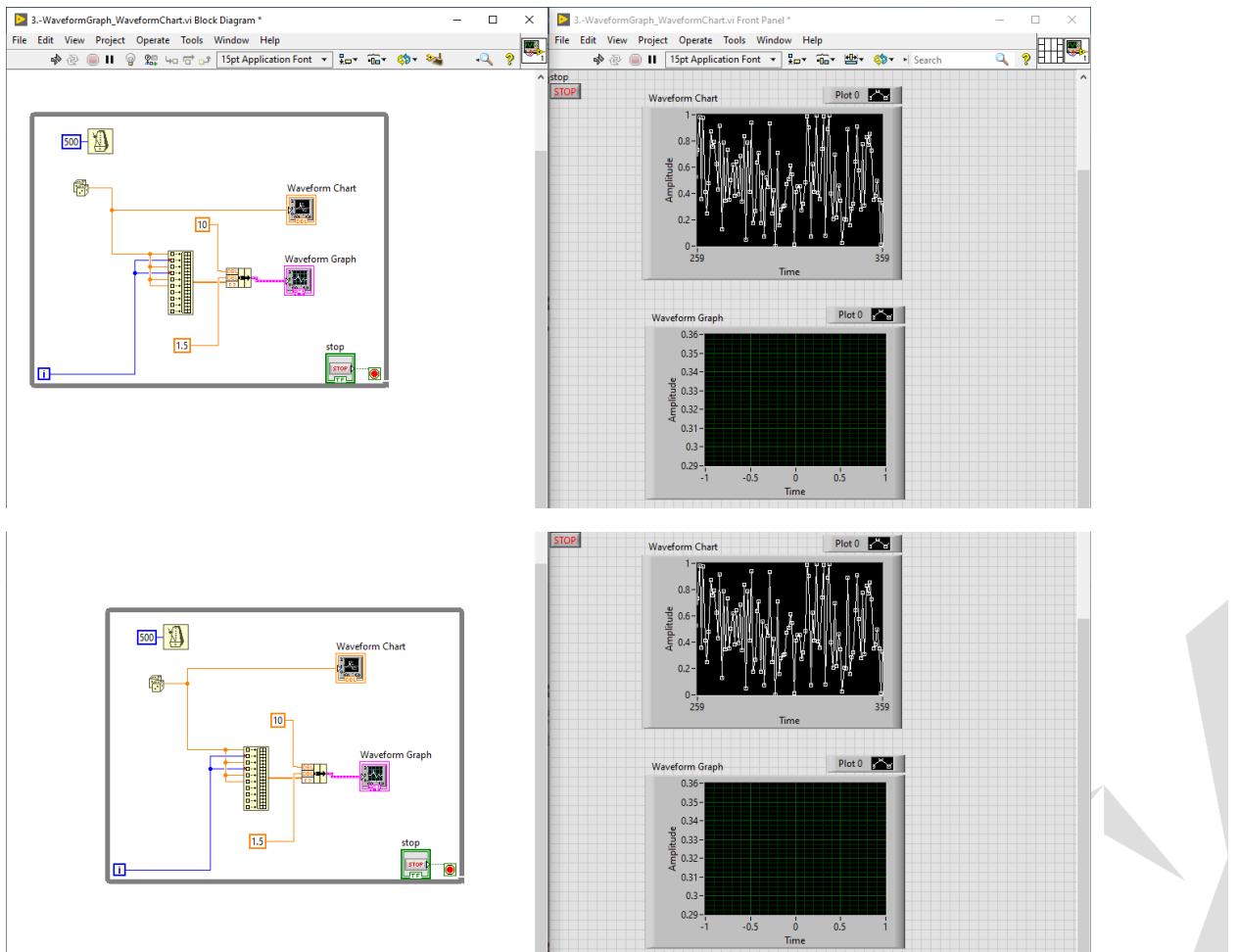
Puedo usar una constante entera y otra decimal para que se tomen como muestreo de la señal, agregadas al bloque del Bundle.





Block Diagram - Bucle While o For: Si se Usa la Variable i, el Bucle Puede Ser while o for

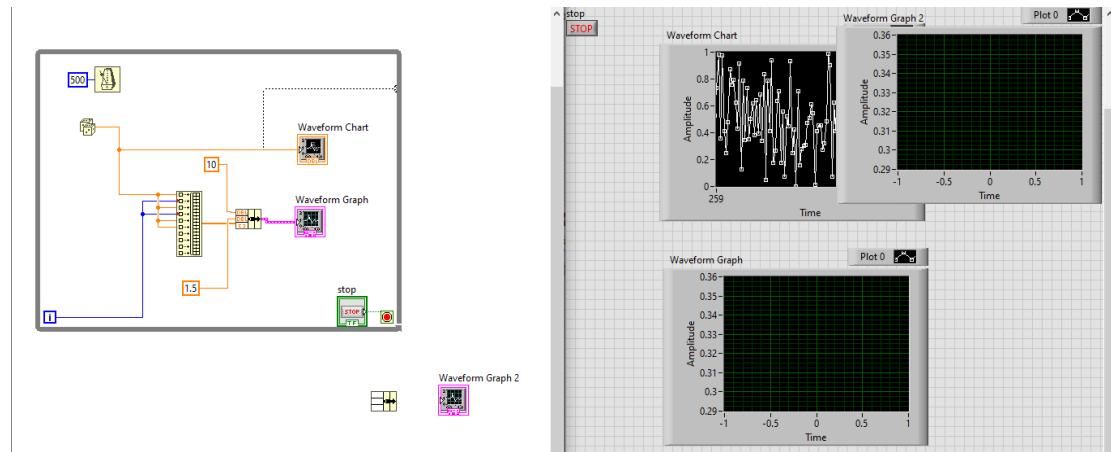
La *i* de la esquina inferior izquierda del bucle while es la iteración, osea que va contando las veces que se ha corrido el bucle, por lo que entonces el mismo bucle while se puede tomar como un bucle for, si es que esta variable *i* es restringida. Aunque en este caso solo se está usando para agarrar otro número.



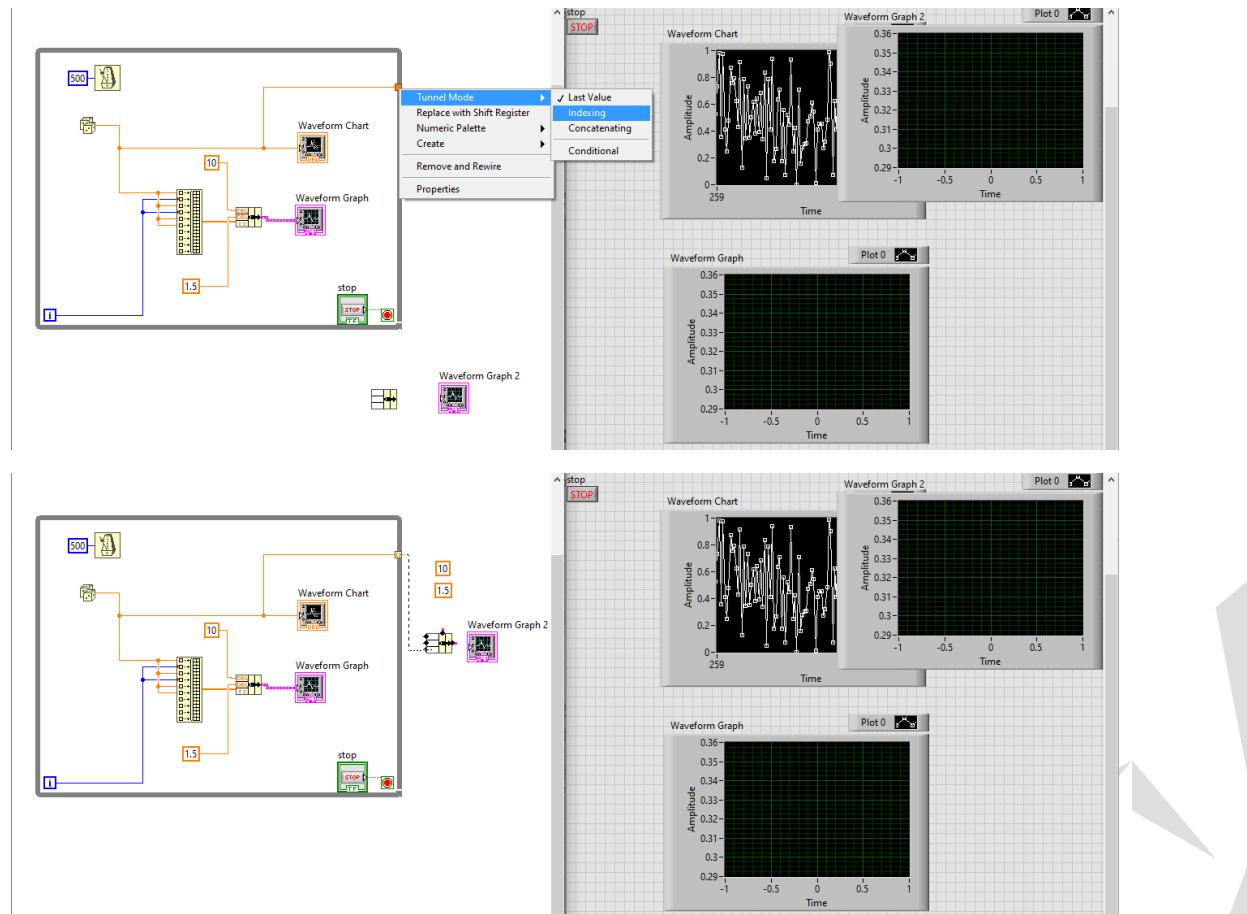
Block Diagram - Bucle While - Túnel (index): Sacar Información del Bucle en un Vector

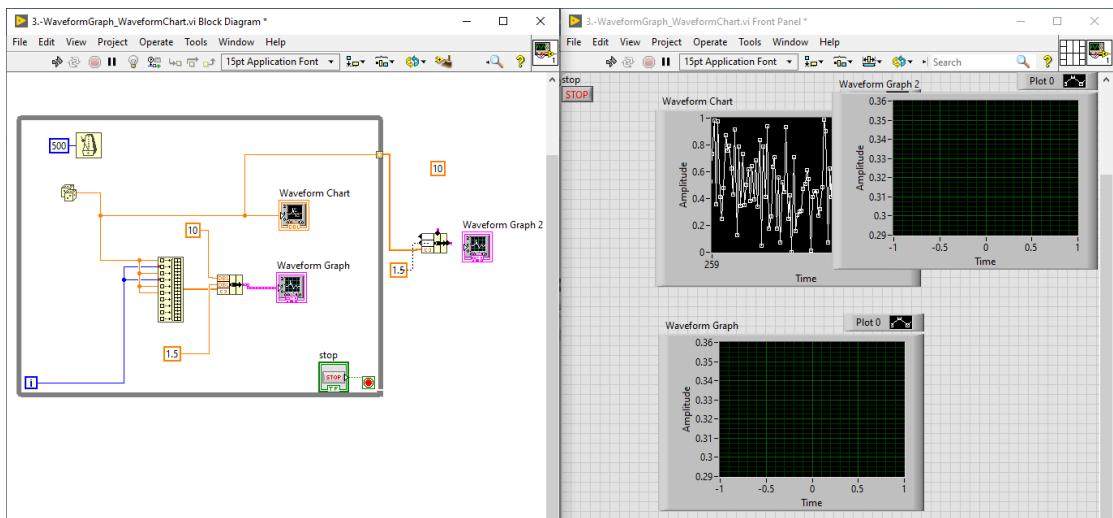
Un túnel es una parte de la pared del bucle que sirve sacar valores de su estructura y que se puedan utilizar afuera, en las demás partes del programa.

Por medio del túnel puedo obtener un escalar que tenga el último valor que se generó del bloque de números aleatorios para sacarlo y mostrarlo.

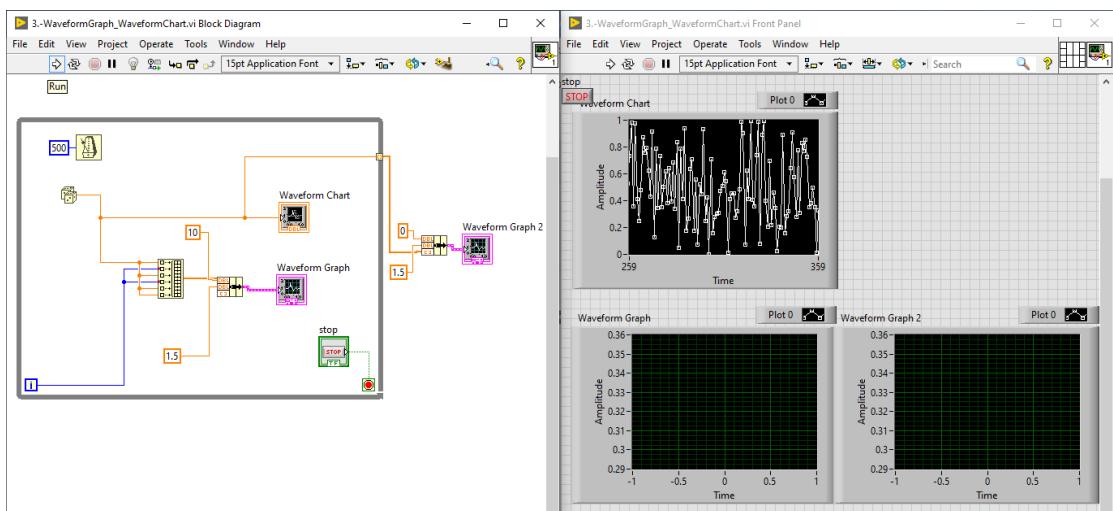
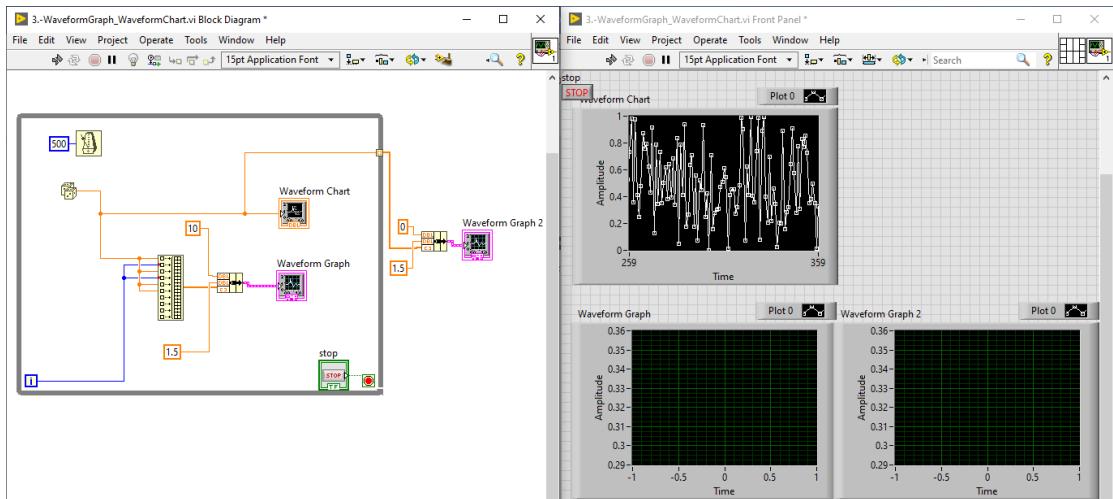


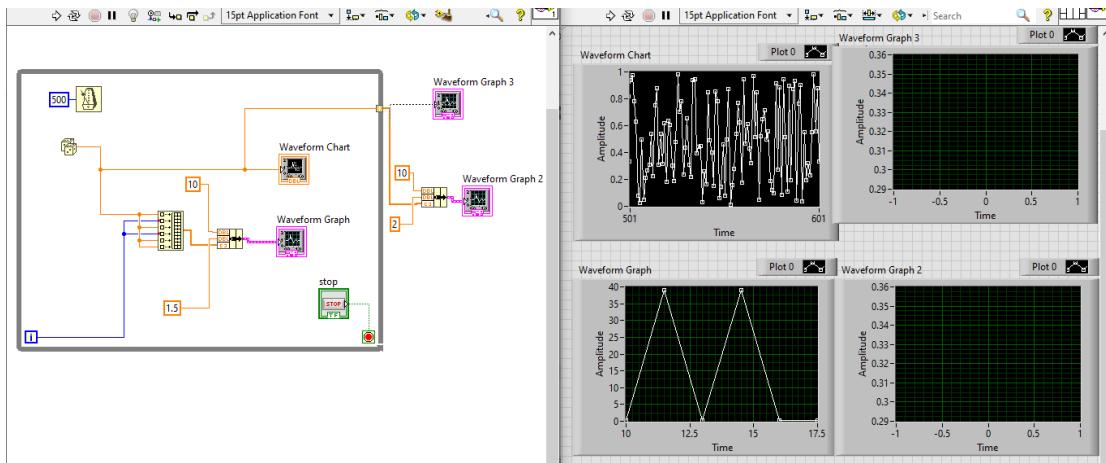
Con la opción de indexing estoy generando un vector que almacene lo obtenido del túnel.





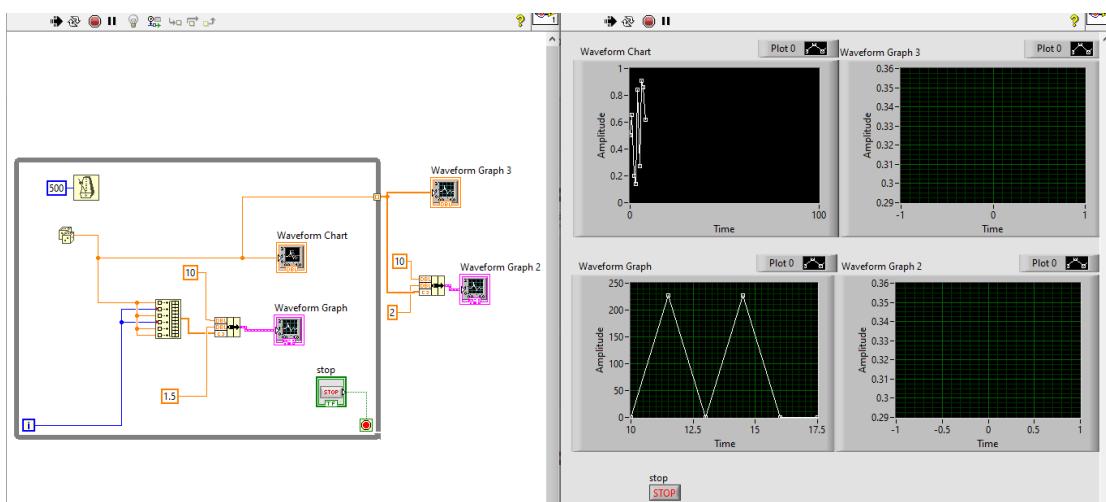
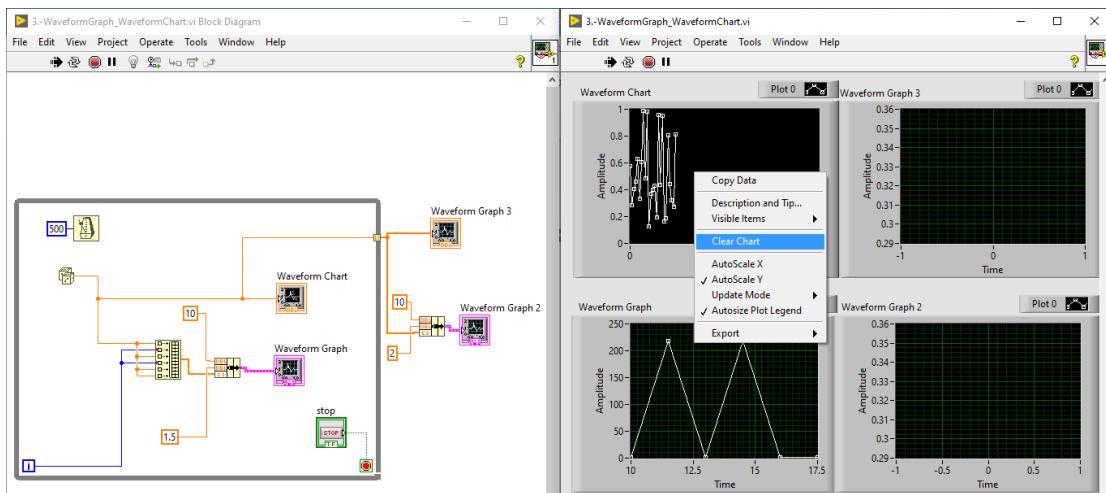
Con lo obtenido del bucle while es que se genera un vector (indexing) y con el túnel lo saqué para poder usarlo fuera, esto empaqueta la señal y le da un punto de partida o condición inicial y que nosotros llamaremos como delta x o separación en el eje horizontal de cada muestreo perteneciente a la señal.

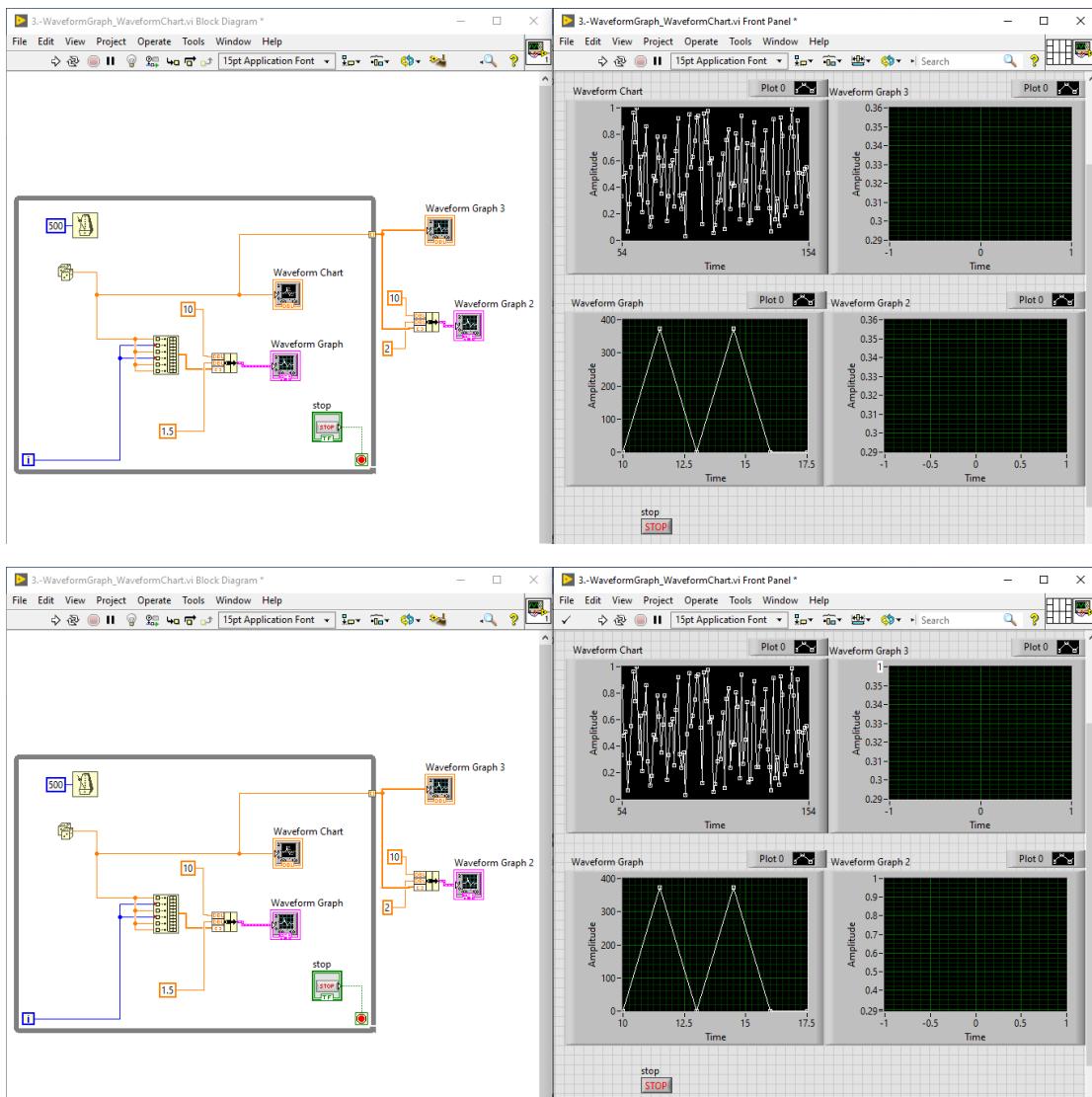




Front Panel - Waveform Chart: Clear Chart

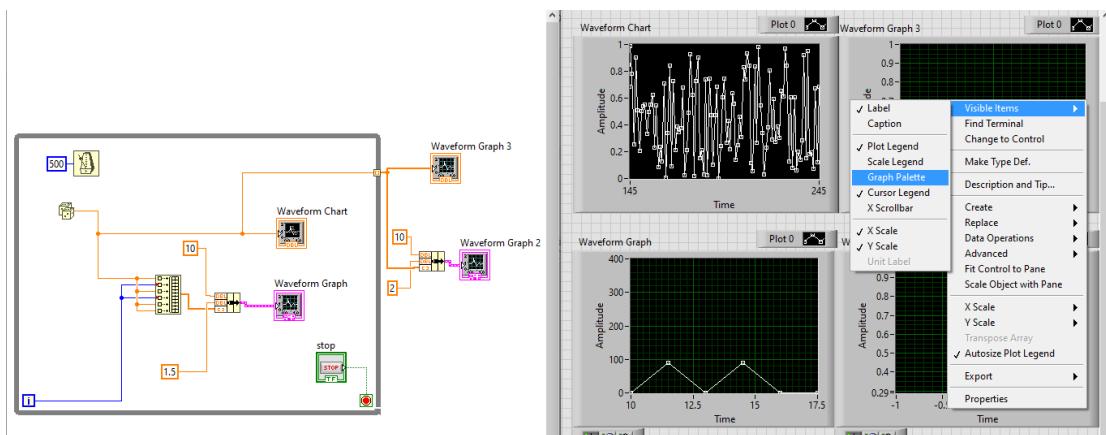
El Waveform Chart muestra gráficas de tipo Dynamic Data, las cuales son señales virtuales o reales, pero al dar clic derecho lo que se hace es borrar lo que haya quedado de la última vez que se corrió el programa.



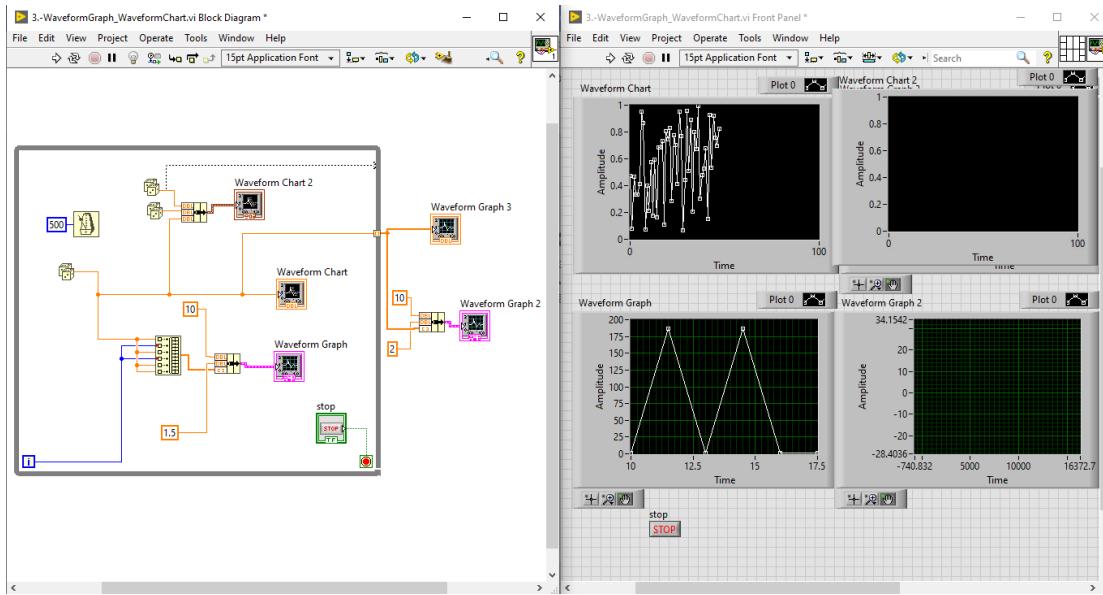


Front Panel - Waveform Chart o Graph - Graph Palette: Ajuste de Zoom en la Gráfica

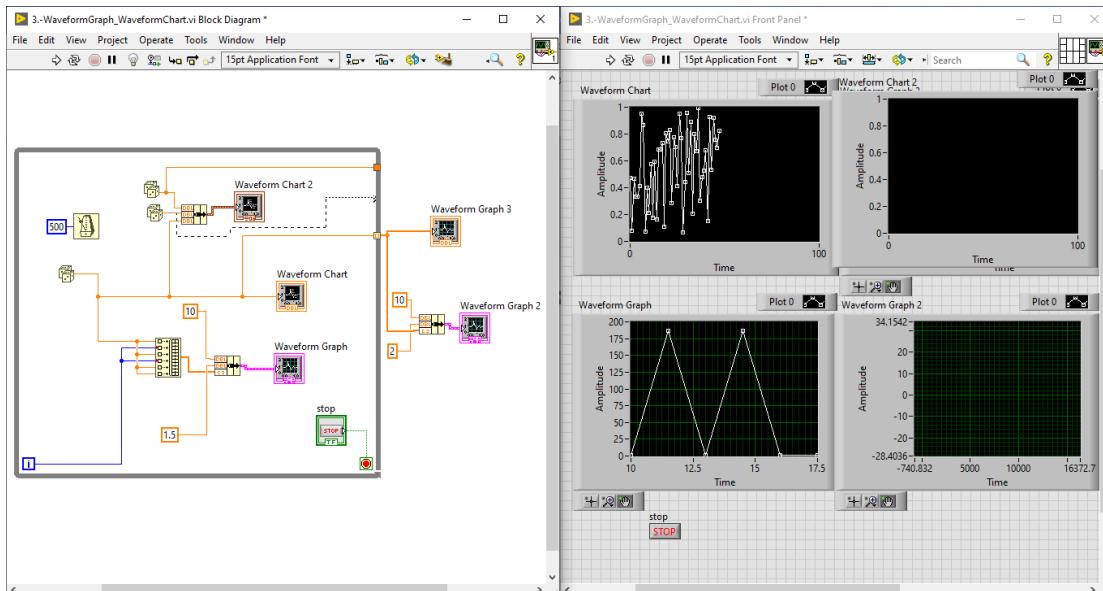
Al dar clic derecho se puede agregar la graph palette a cualquiera de las dos gráficas (Waveform Chart o Graph) y lo que esta hace es permitirme ajustar el zoom de su contenido.

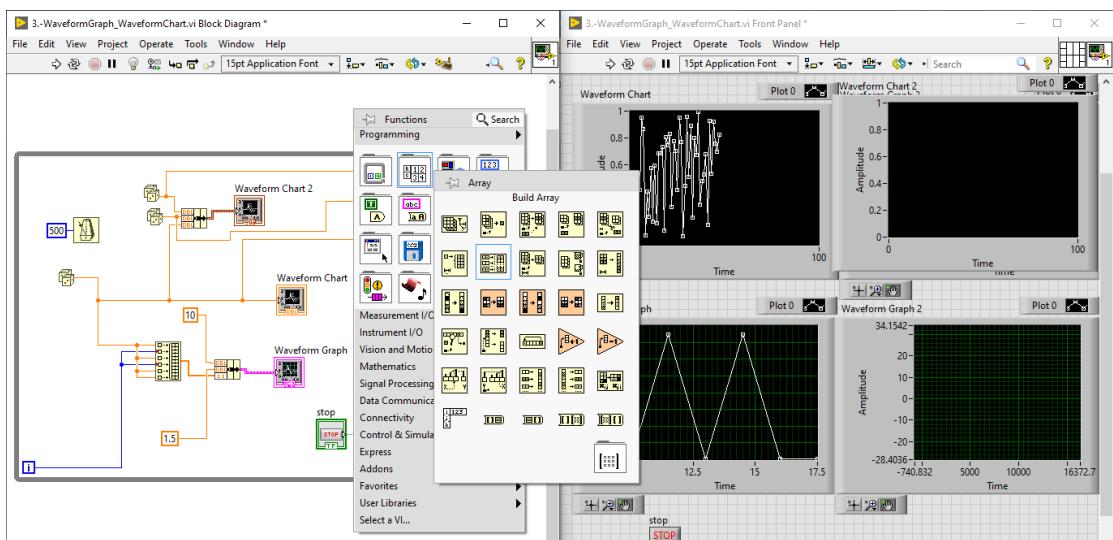
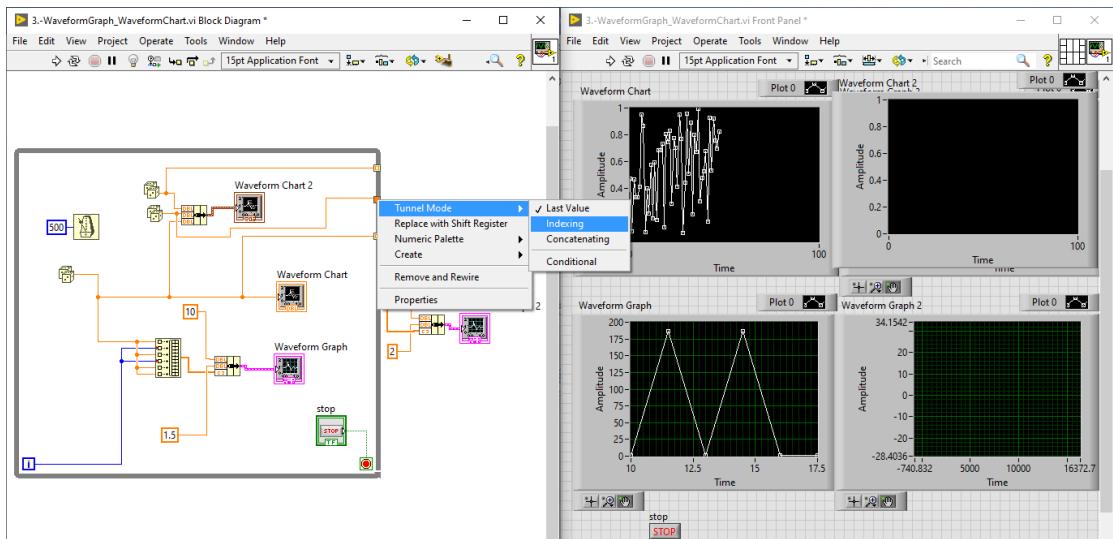
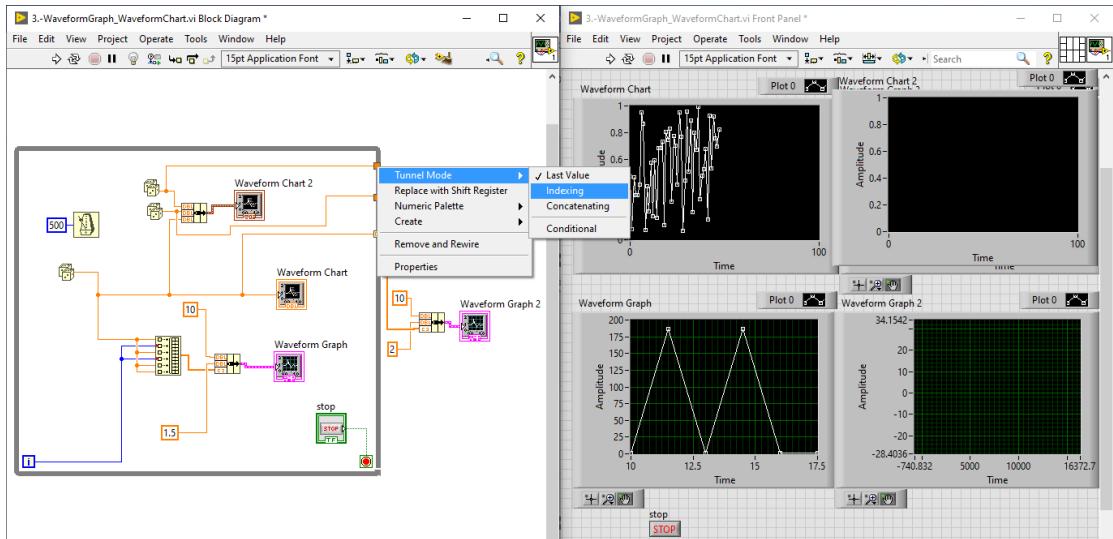


Hasta que yo no dé clic en el botón de stop es cuando en el Graph 3 y 2 (los de la derecha) aparecerá lo que haya almacenado el vector que sale del bucle while en el túnel usando index, mientras no se va a mostrar nada, el vector solamente estará declarado y se irán almacenando más y más valores en el Waveform Graph 3 (**Arrays y Clusters**).

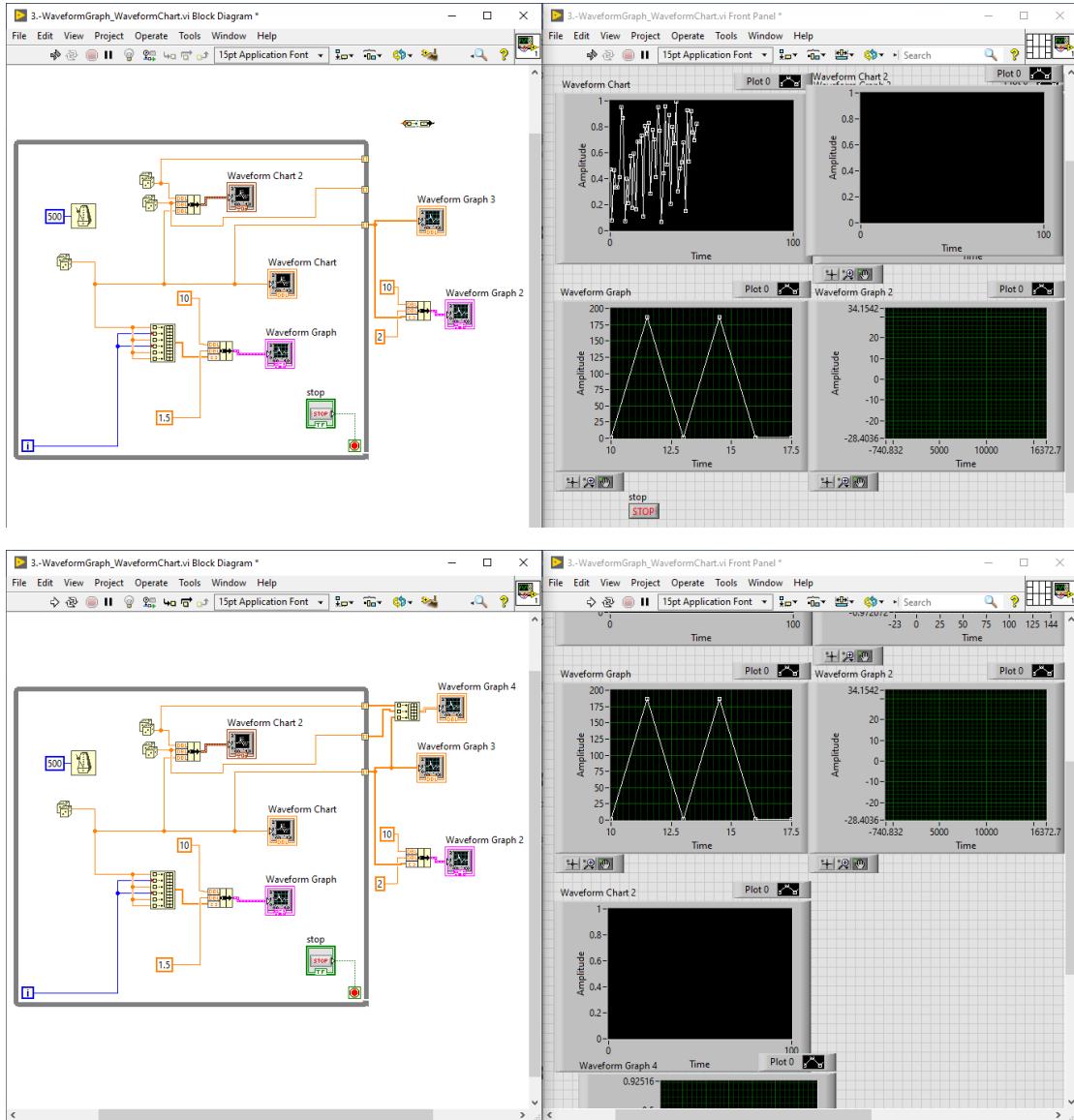


Ahora lo que vamos a hacer es armar un vector de 3 valores aleatorios obtenidos a través del túnel del bucle while, en vez de usar constantes como se ha hecho hasta ahora, para ello se utilizará un nuevo Waveform Graph 4 (**Arrays y Clusters**) que esté afuera y podrá graficar estáticamente un valor que se haya guardado en el vector que sale de los túneles del bucle. Su contenido se mostrará hasta el momento en que demos clic en el botón de STOP, terminando así la ejecución del programa.

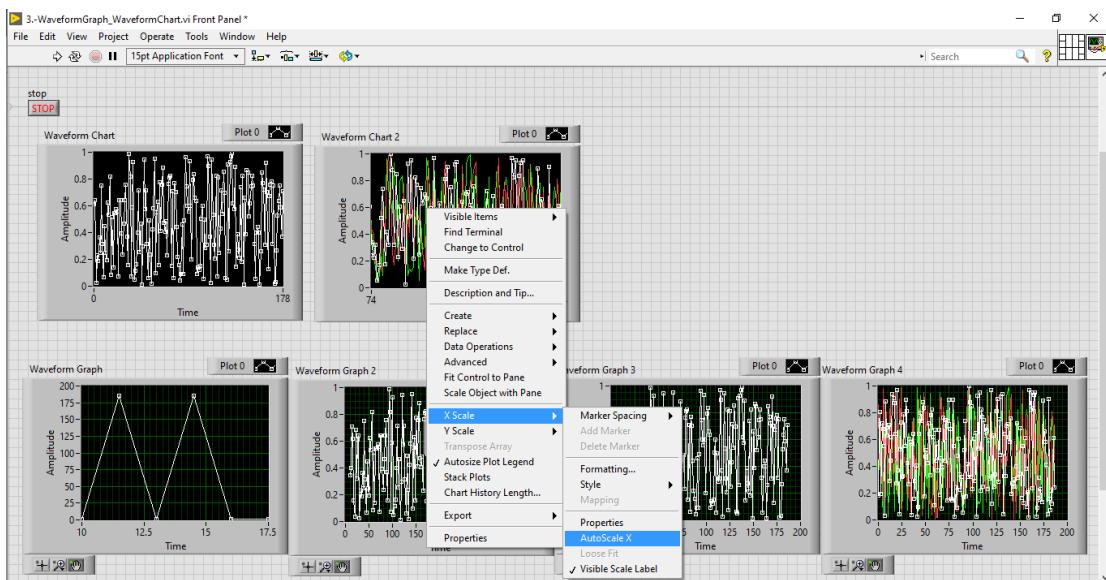
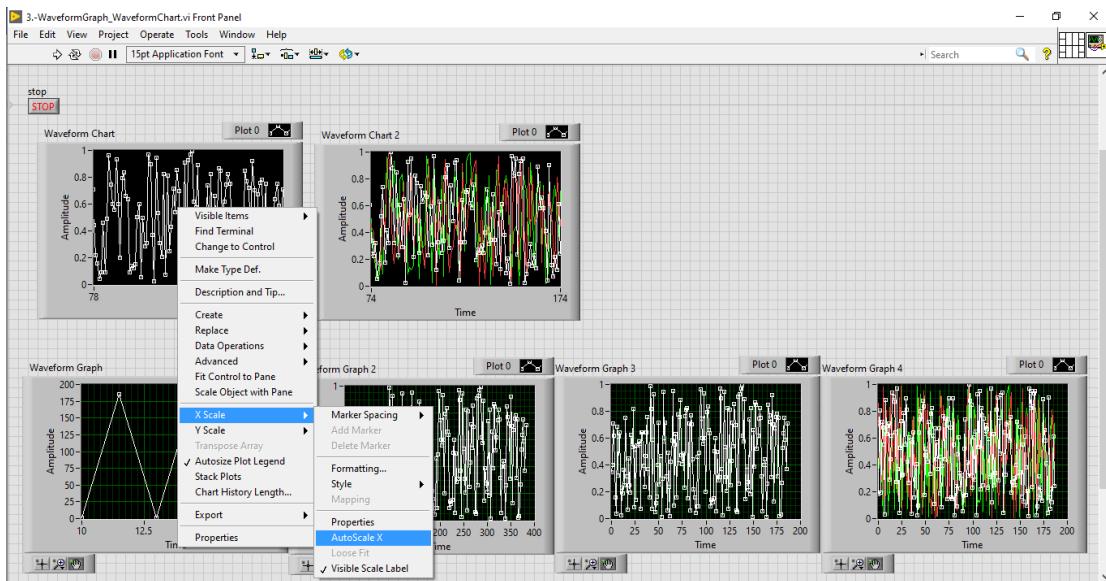
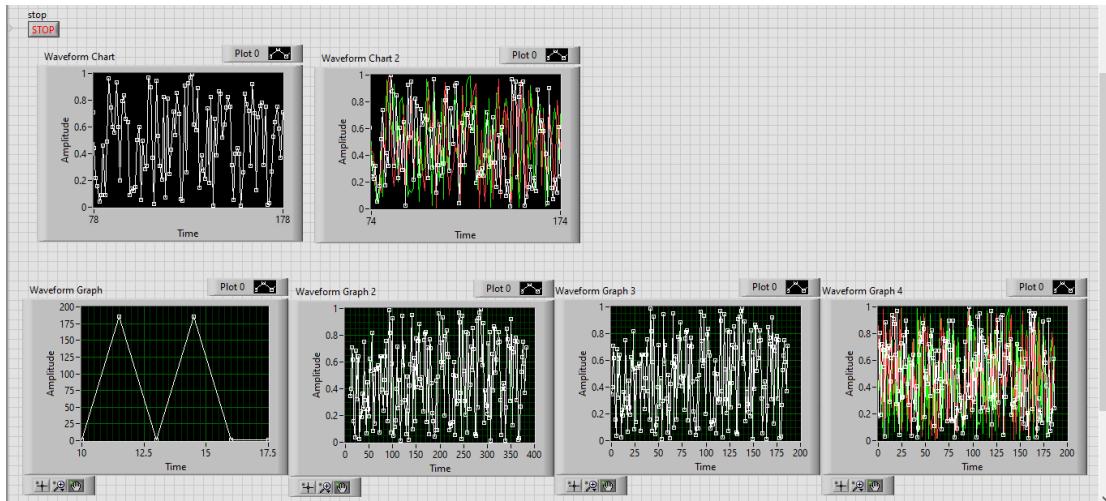


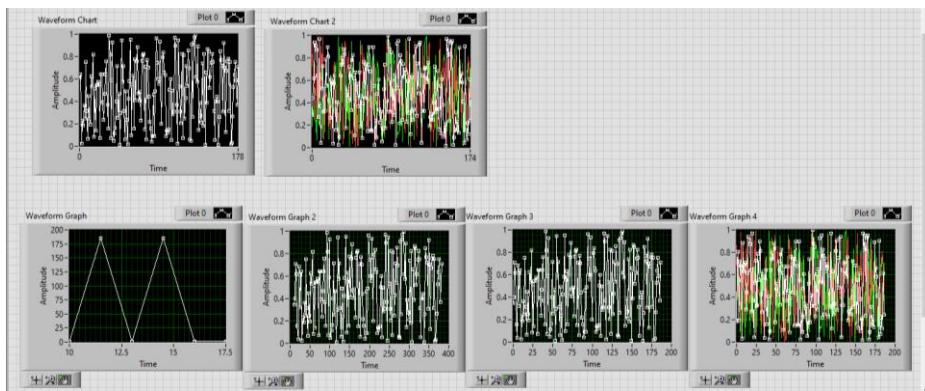


Lo que se hace es un arreglo de arreglos para poder graficar varias funciones a la vez en un mismo Waveform Graph (**Arrays y Clusters**).



- En el 1er **Waveform Chart (Dynamic Data)** se muestran los datos solo conectando directo el bloque que genera números aleatorios sin nada en medio, porque el tipo de dato Dynamic Data representa datos que cambian en el transcurso del tiempo, en el segundo se muestran varias gráficas de distintos colores porque se tiene 3 valores simultáneos que están siendo alimentados por distintos bloques de números aleatorios.
- En el 1er **Waveform Graph (Arrays y Clusters)** se crea por sí sola la señal en la gráfica porque se encuentra dentro del bucle while, en los demás no lo hace hasta que se deja de ejecutar el programa porque están fuera y se tiene que dejar de ejecutar el ciclo while para que se llenen de datos los datos de los vectores que van a mostrarse en los Waveform Graph de afuera del bucle.



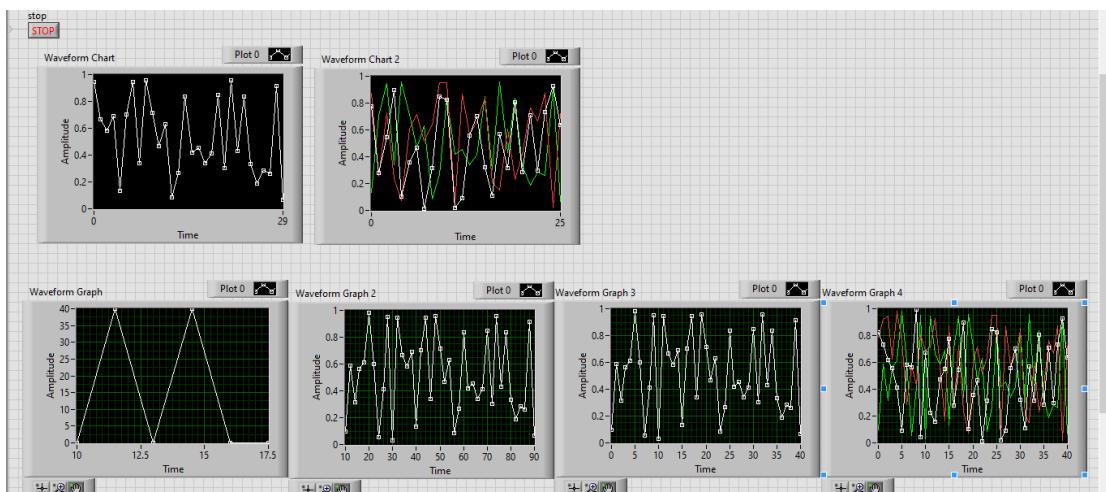


Ejecución del Programa: Gráficos Graph y Chart con Túneles Index

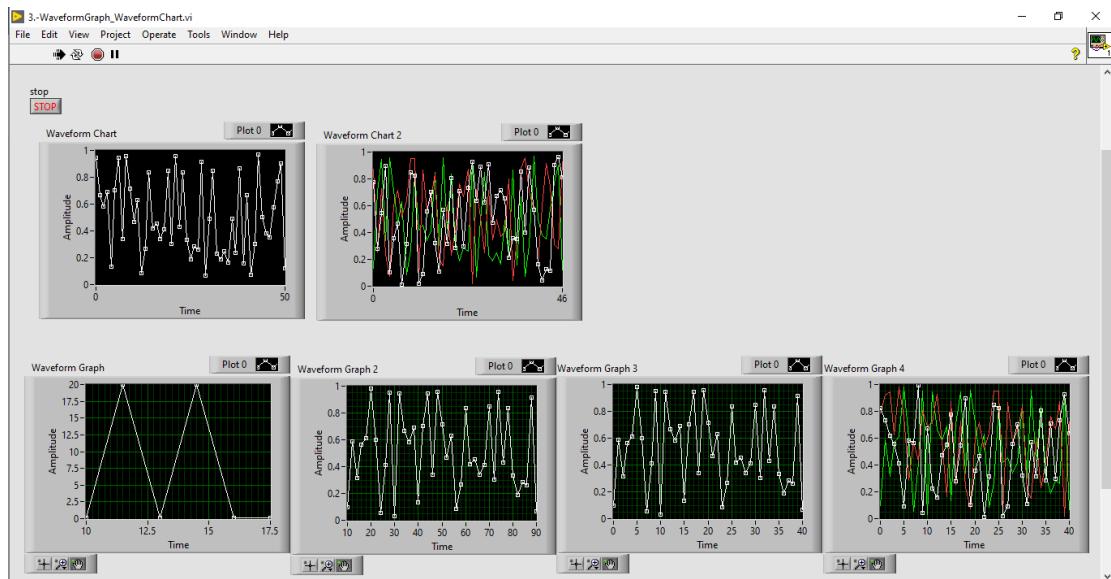
Aquí puedo ver un ejemplo donde se ve cómo después de haber limpiado el waveform chart y graph es cuando se ve el contenido de las gráficas que tienen en túnel index fuera del bucle while, no antes.



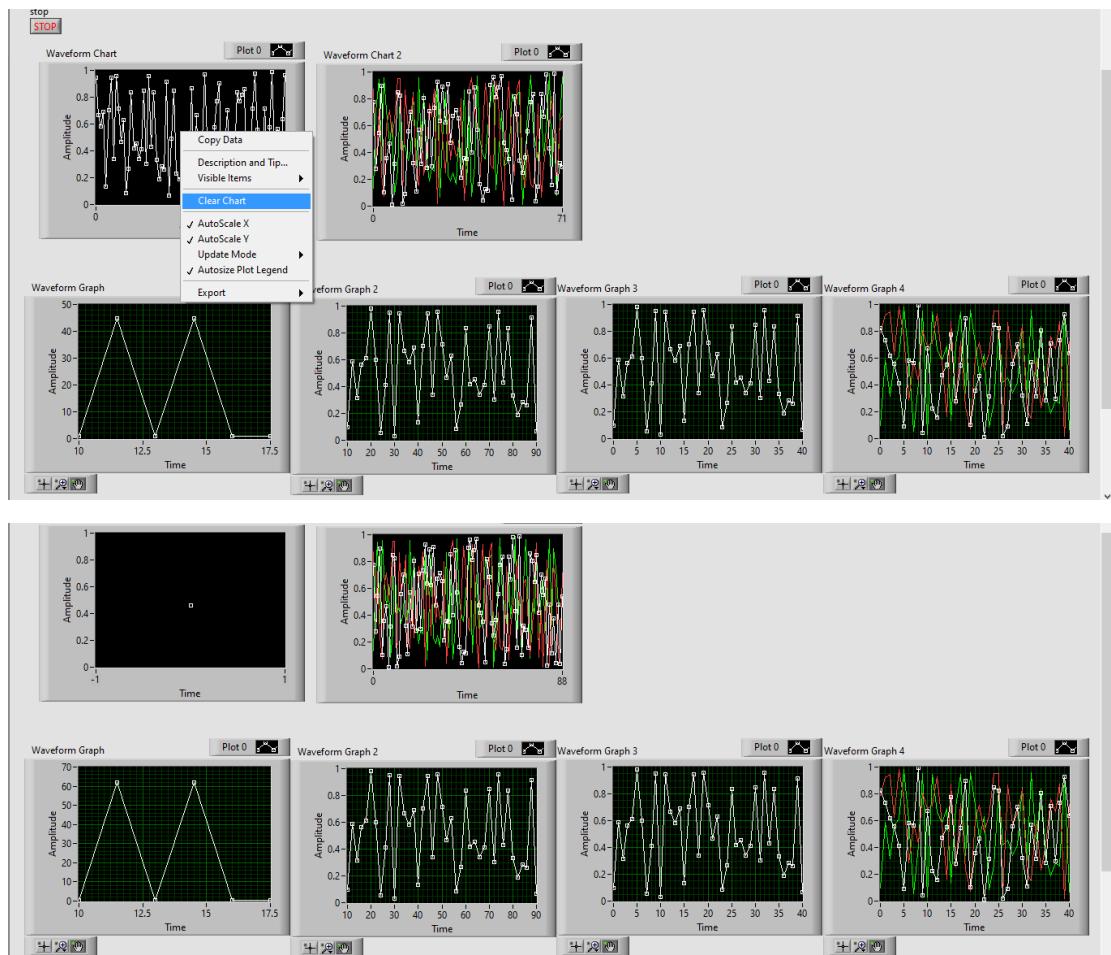
Solo después de dar clic en el botón de STOP aparecerá el contenido de las gráficas.



Doy clic en RUN.



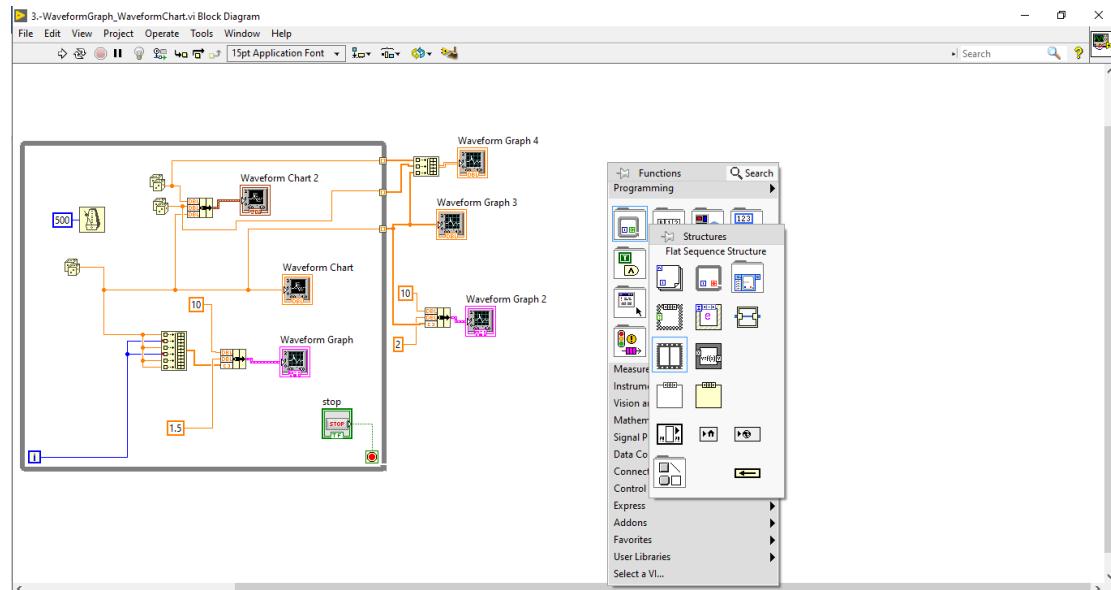
Limpio el chart y graph dando clic derecho en cada elemento y borrando los datos que lleva almacenados y graficados.



Ahora vamos a hacer que al momento que inicie el programa, siempre se limpie el graph por sí solo, limpiando el Buffer del programa, el buffer es un canal que puede almacenar datos o que puede almacenar cosas, en LabVIEW el buffer es uno de almacenamiento, donde van a ir datos almacenados, existen buffers de:

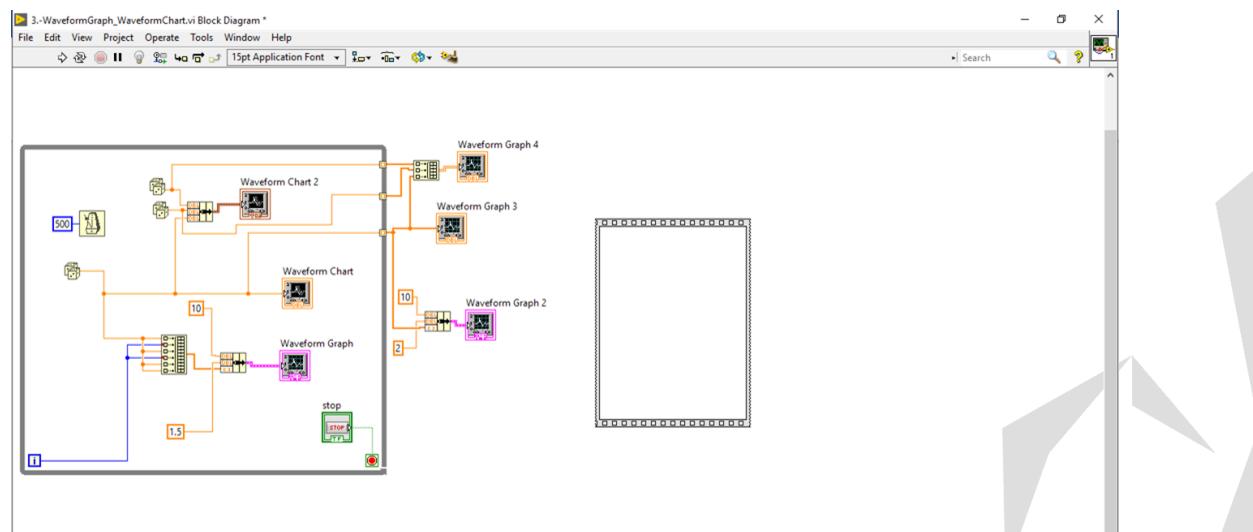
- **Buffer de Almacenamiento:** En el Waveform Chart guarda 1024 datos y luego los libera.
- **Buffers de Paso:** En el Waveform Chart atenuan o amplían la señal.

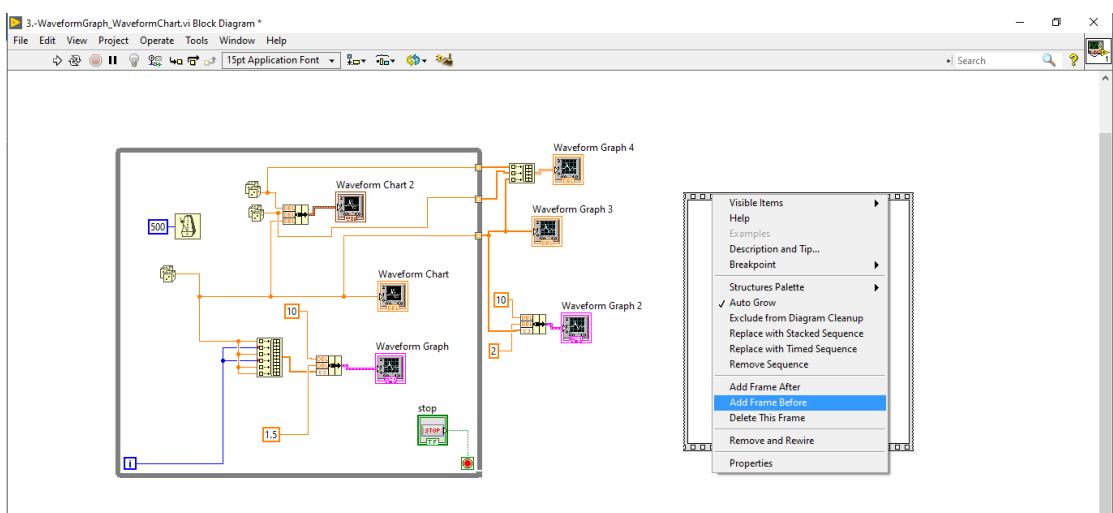
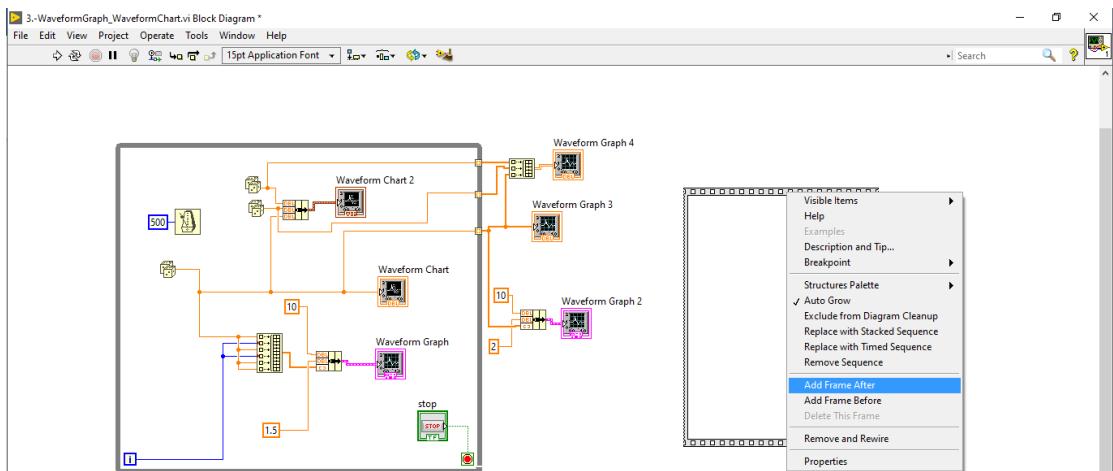
Para ello vamos a añadir un Flat Sequence



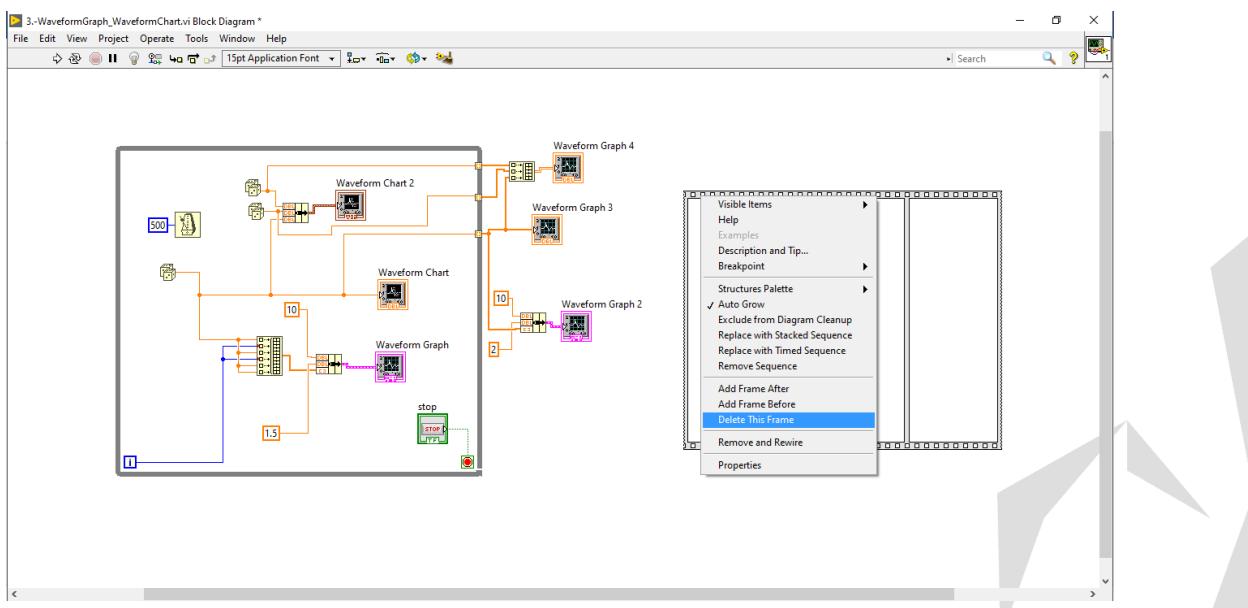
Block Diagram - Flat Secuence Structure: Ejecución Ordenada Secuencial de Funciones

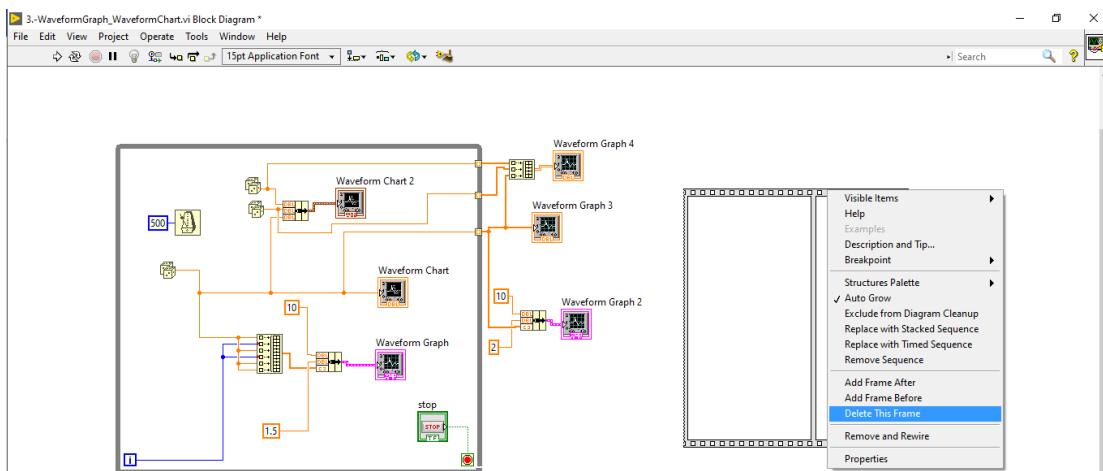
El bloque de Flat Secuence Structure sirve para que cada uno de sus cuadritos ejecute un proceso y hasta que cada uno de sus procesos se ejecuten, se ejecutará el siguiente, siguiendo una programación de escalera, esto significa que se ejecutará de forma secuencial, donde hasta que se termine de ejecutar una instrucción de código no se ejecutará la siguiente.





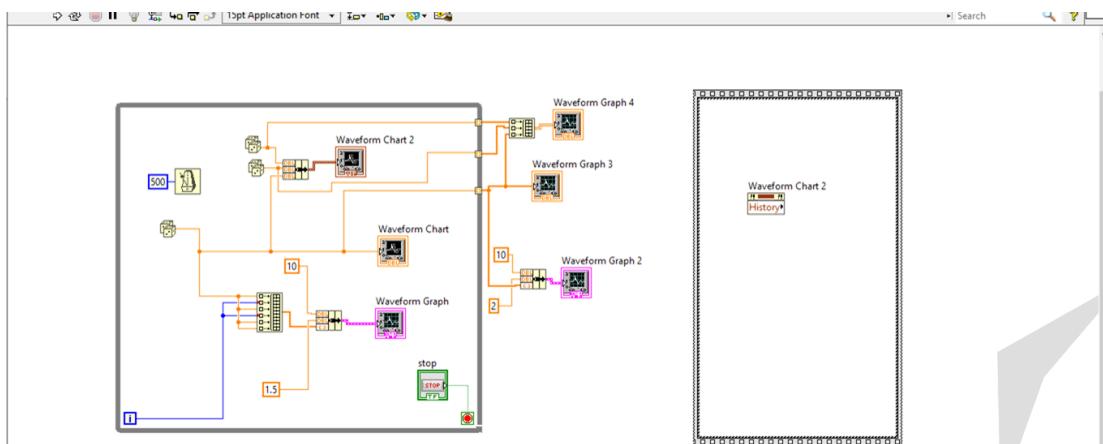
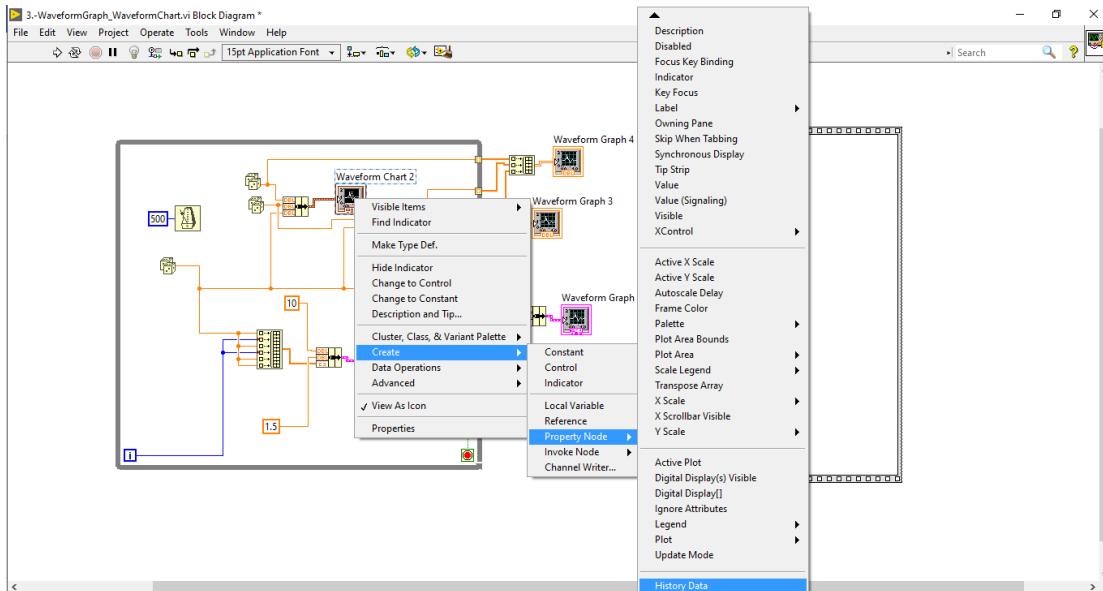
Con estos frames puedo poner un proceso para que se ejecuten de forma ordenada uno después de otro, pero como solo necesitamos uno, vamos a borrar los demás y nos vamos a quedar con solo uno.



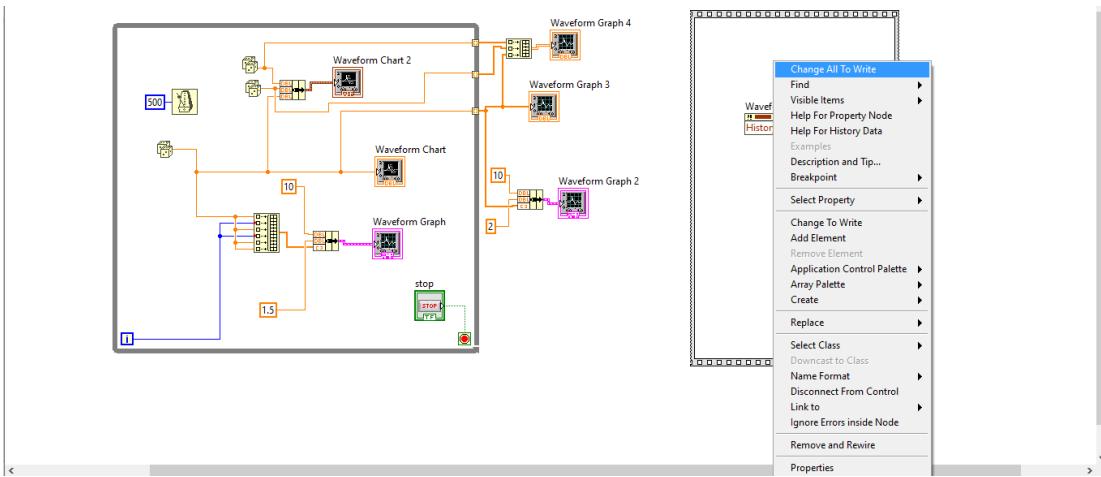


Block Diagram - Property Node: Propiedad Específica de Cualquier Bloque

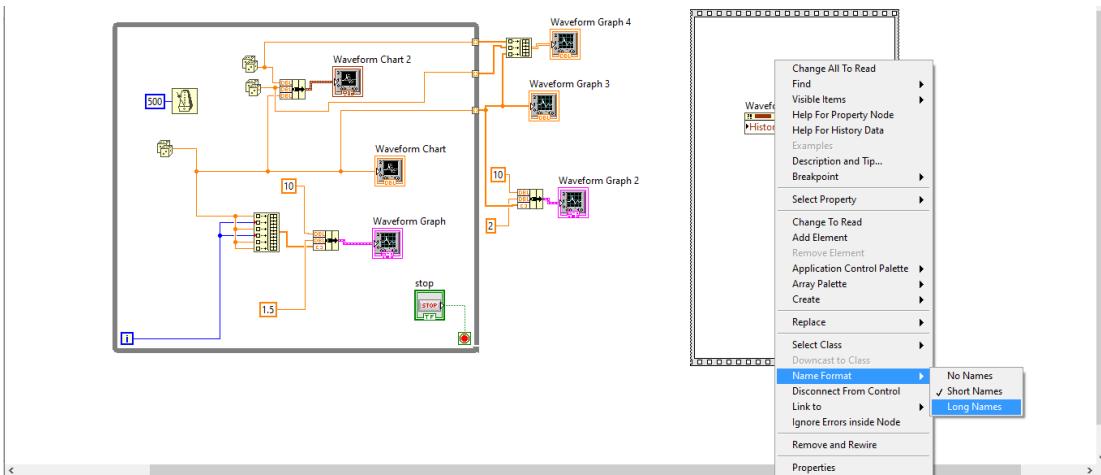
Ahora lo que voy a hacer es acceder a las propiedades de las gráficas dando clic derecho en el bloque de la siguiente manera y seleccionando la opción de Create → Property Node → History Data:



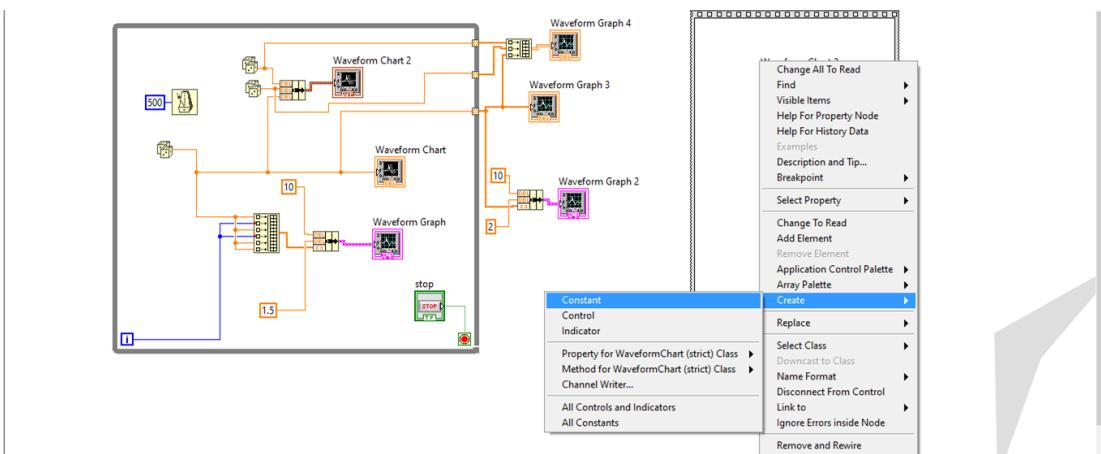
Este nodo (Property Node) de History Data incluye los datos almacenados dentro del Waveform Chart (**Dynamic Data**), pero yo no lo quiero leer, lo quiero escribir o editar, por lo que voy a darle clic derecho y seleccionar la opción de Change All To Write:



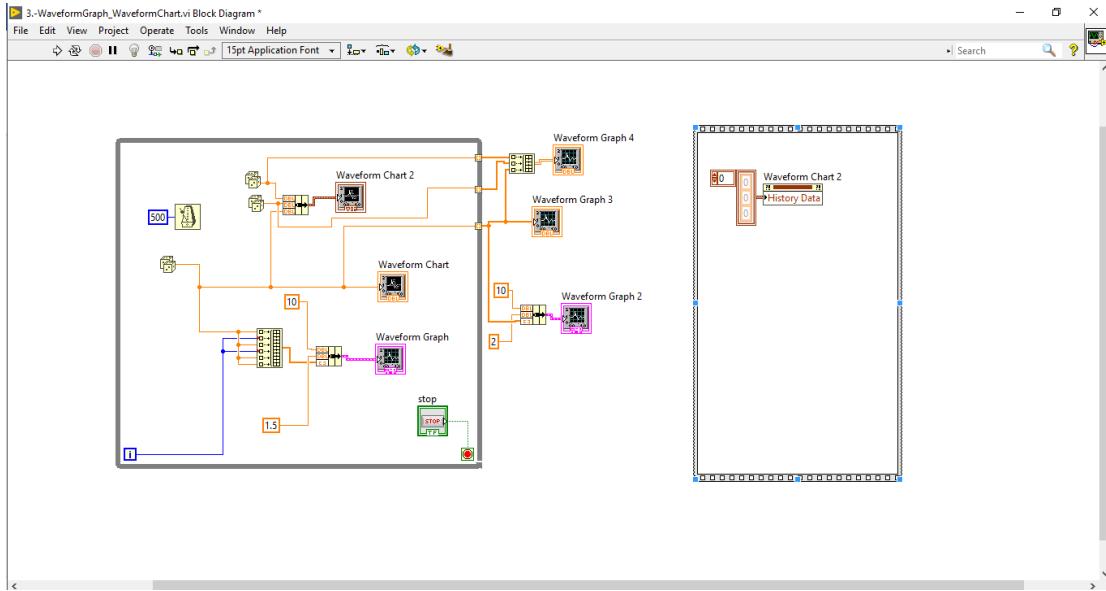
Luego haré que aparezca el nombre del Bloque con letra grande:



Crear una Constante para un Bloque: Clic derecho en la terminal del bloque de interés → Create → Constant. Constante para indicarle al Waveform Graph que borre sus datos almacenados al ejecutarse.



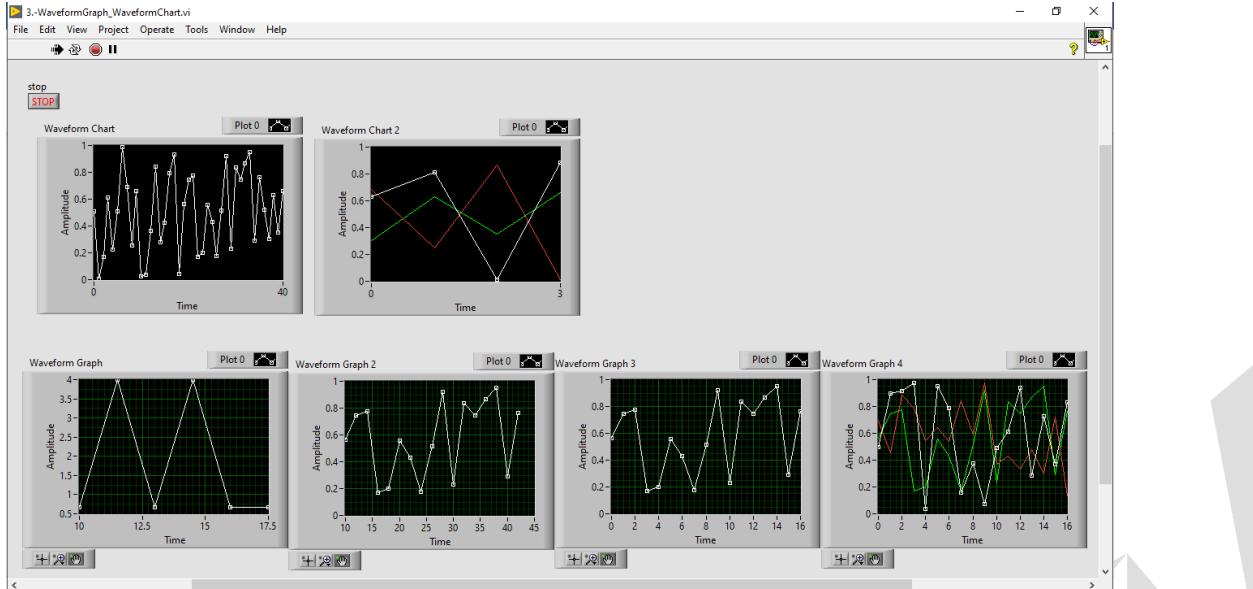
Con la constante 0 se va a regresar las 3 gráficas a cero, borrando su historial, no debo conectarlo a nada porque es un bloque Property Node del mismo Waveform Chart (**Dynamic Data**).

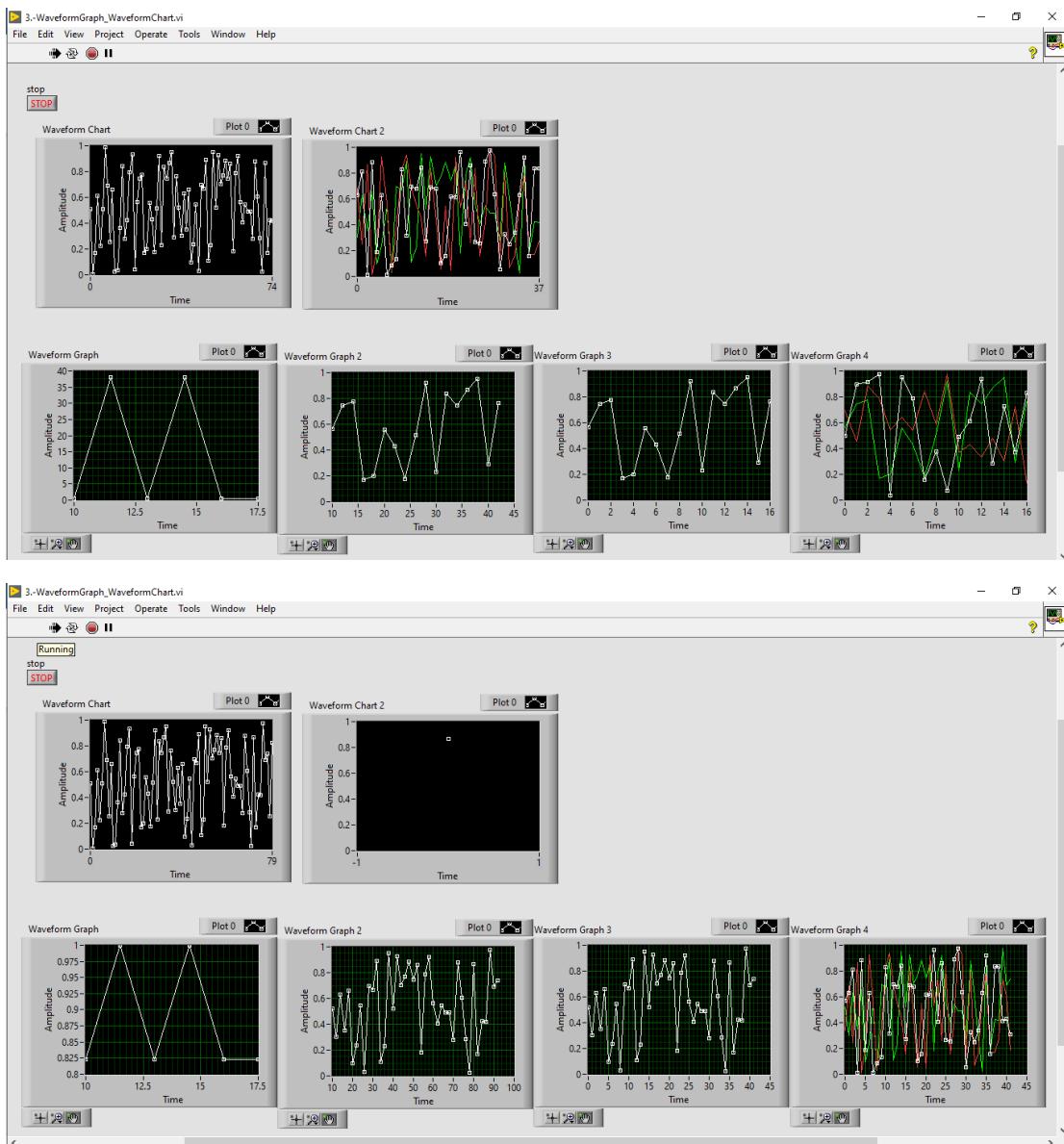


Si yo quiero modificar el hecho de que se borre el contenido de las gráficas de forma automática cada que se ejecute de nuevo, debo repetir este proceso para cada una de las gráficas.

Ejecución del Programa: Gráficos Graph y Chart Empezando Desde 0

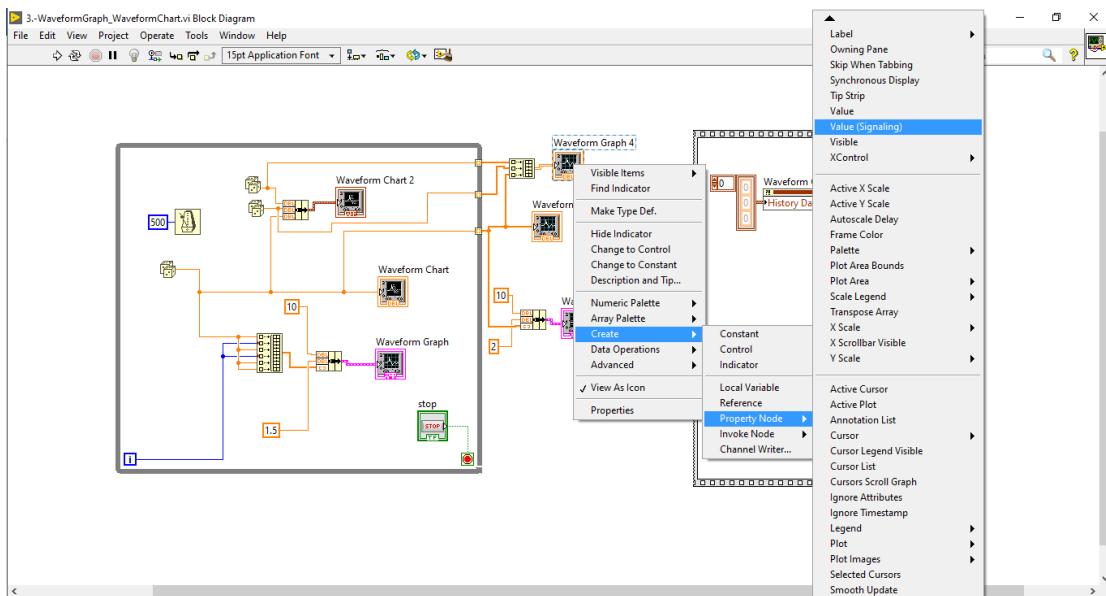
Ahora las 3 gráficas ya se corren desde cero, cuando antes se tenía que borrar su historial de forma manual.



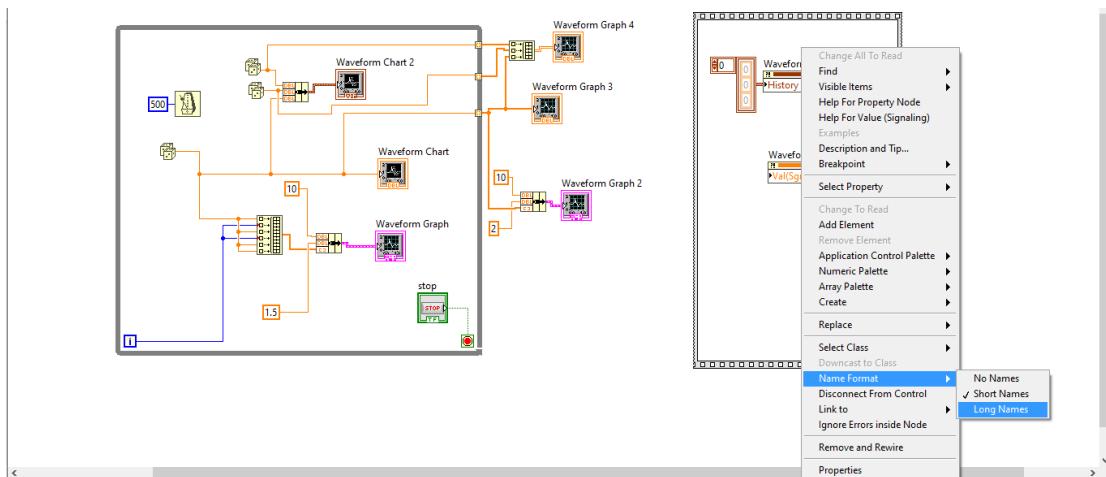


Block Diagram - Property Node: Propiedad Específica de Cualquier Bloque Fuera de Bucle

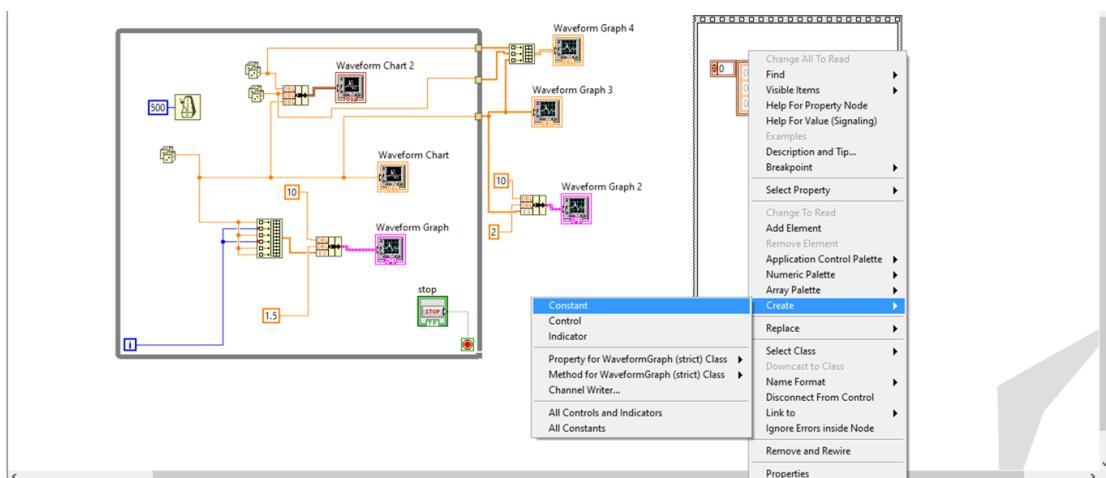
- Este proceso se aplicó a un Waveform Chart (**Dynamic Data**), que se grafica respecto al tiempo y se encuentra dentro del bucle while.
 - Se utilizó el Property Node de Create → Property Node → History Data.
 - Este nodo accede al historial del Waveform Chart (**Dynamic Data**) y así lo podemos borrar.
- Ahora vamos a hacer lo mismo con algún Waveform Graph (**Arrays y Clusters**) que esté fuera del bucle while.
 - Se utilizará el Property Node de Create → Property Node → Value (Signaling).
 - Este nodo accederá al valor del Waveform Graph (**Arrays y Clusters**) y así lo podremos borrar, se hace de esta manera porque es un dato estático.



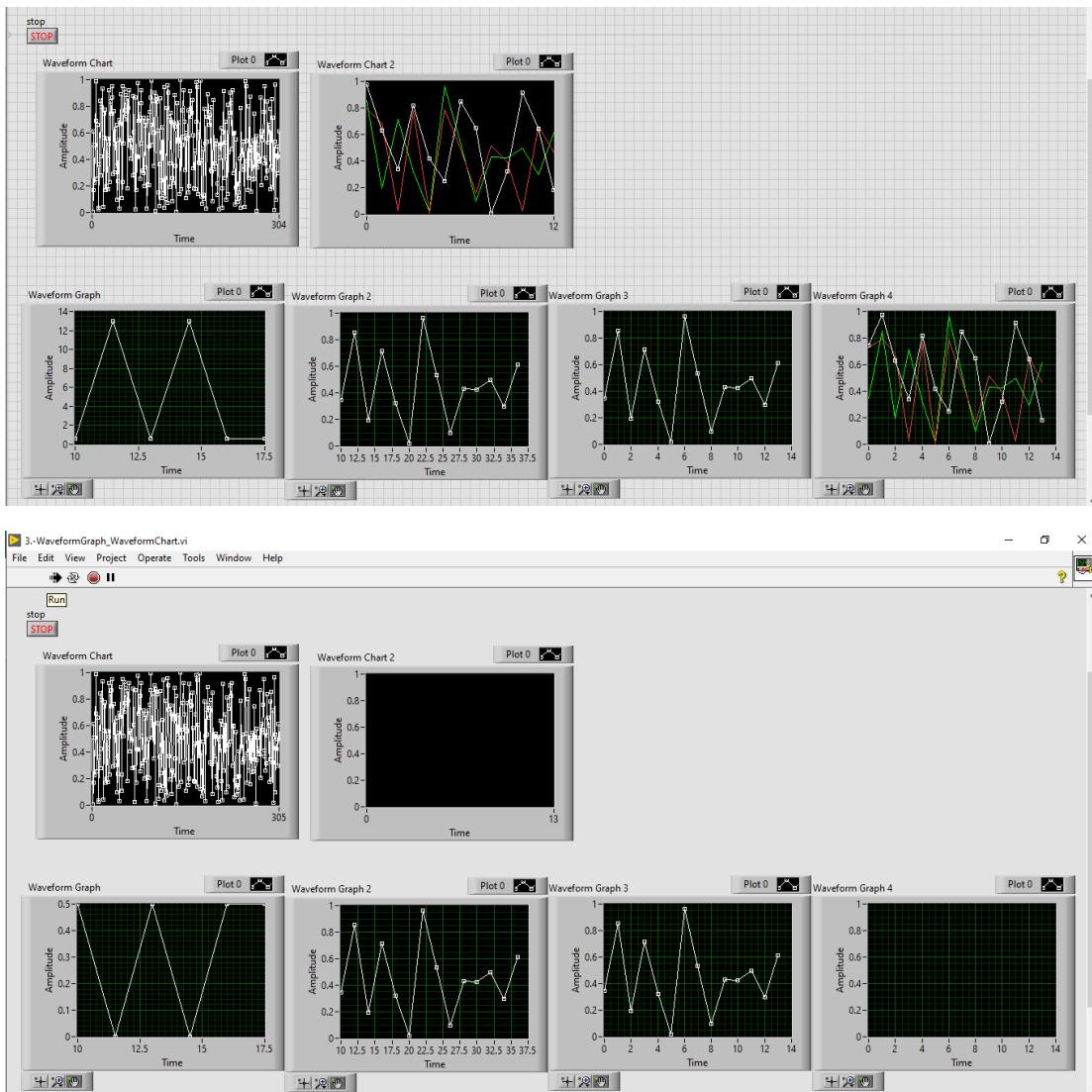
Luego haré que aparezca el nombre del Bloque con letra grande:



Y finalmente crearé una constante para indicarle que borre sus datos almacenados al ejecutarse.



Esto va a limpiar lo que haya dentro del Waveform Graph (**Arrays y Clusters**), que son datos estáticos.



Podemos ver que, al ejecutar el programa, el contenido de los Waveform Graph (**Arrays y Clusters**) se limpió.

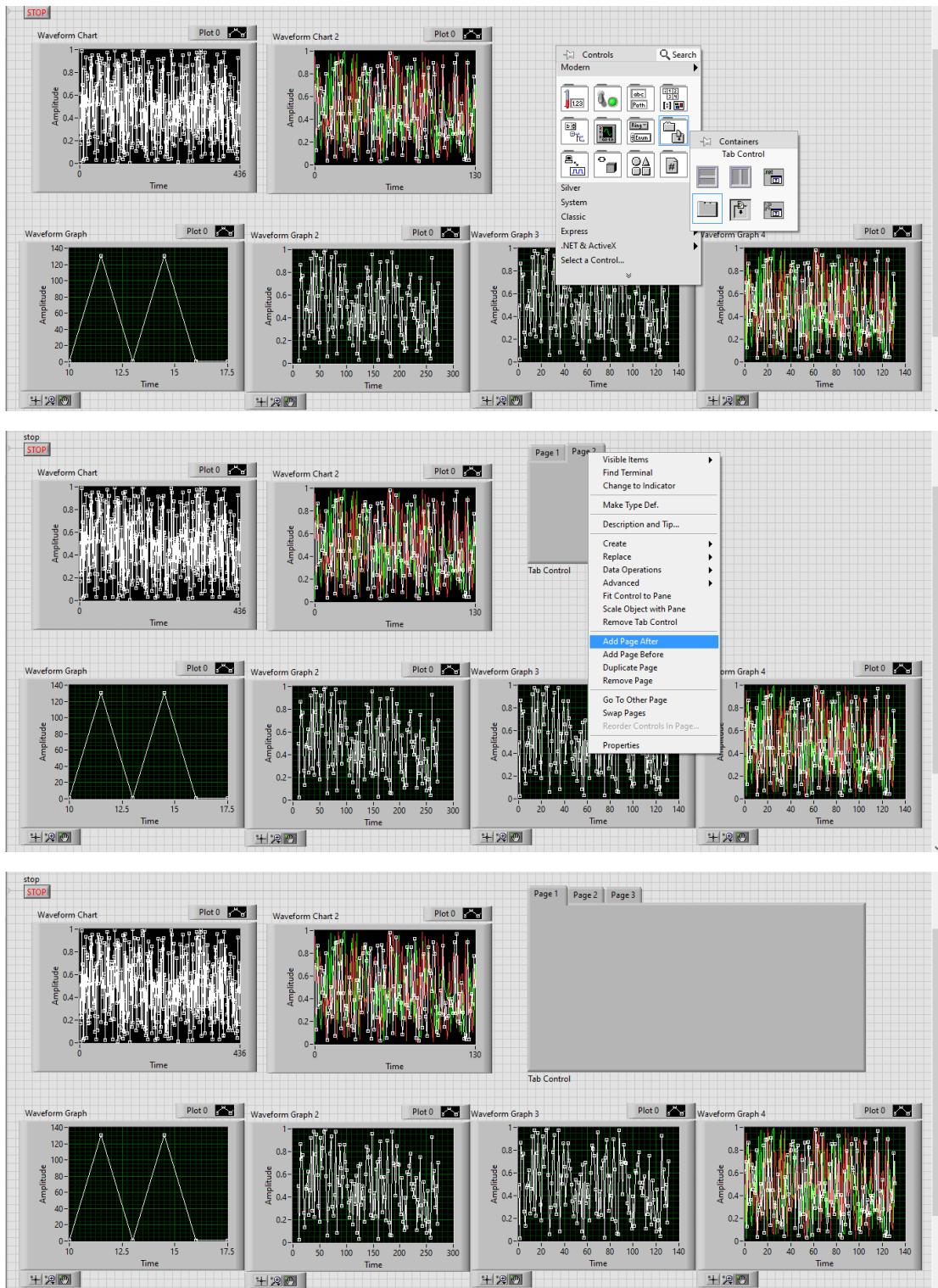
En LabVIEW la gran utilidad de esto es que no solo podemos leer y analizar señales reales o virtuales, sino que además se podrían guardar en un Archivo o en un SHIFT Register (Memoria del programa de LabVIEW), pero el problema es que al momento de para el programa, esto va a desaparecer.

Una forma de resolver el hecho de que al parar el programa de LabVIEW todo el contenido de las gráficas analizadas se pierda es crear un archivo ya sea txt, archivo de Excel, Word, etc. para que después se pueda hacer un cálculo, se guarden estos datos del resultado de análisis de la señal realizado con la Instrumentación Virtual.

Pero además de eso se puede mostrar una interfaz estética para organizar mejor mis gráficas o elementos de mis programas y para ello puedo utilizar toda la gama de elementos estéticos de diseño que proporciona el programa de LabVIEW.

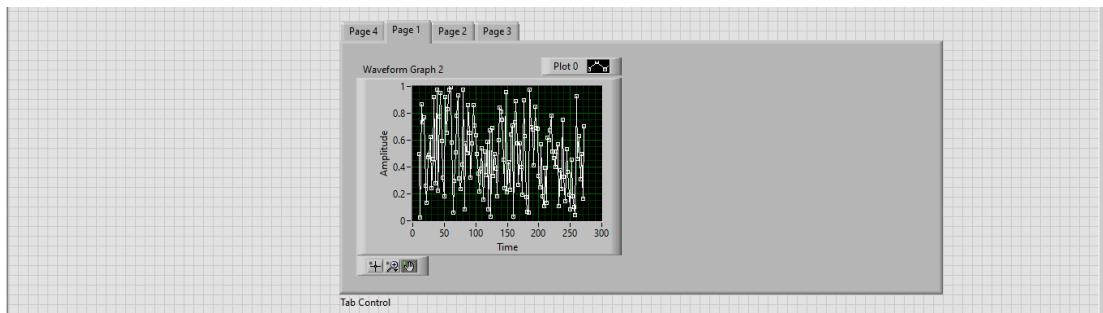
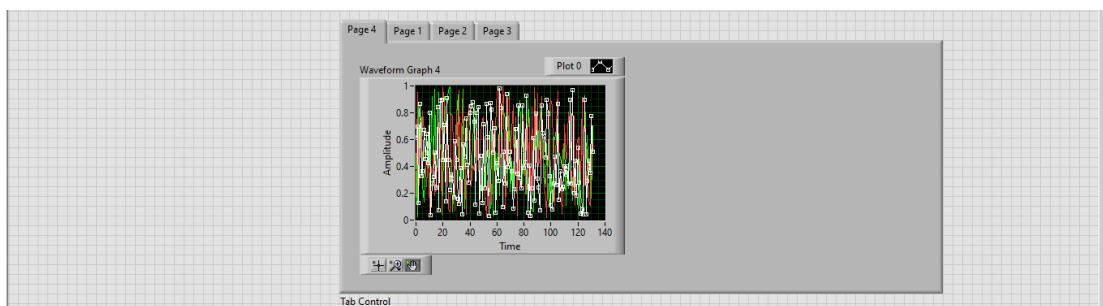
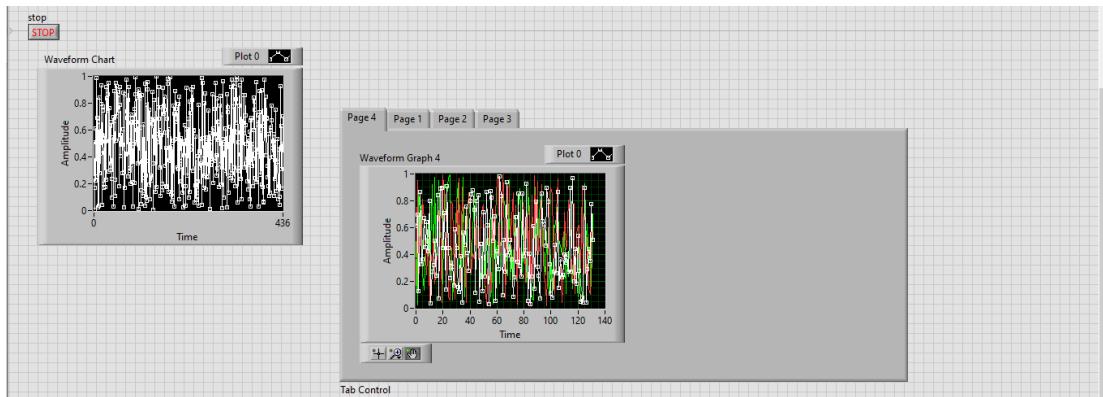
Front Panel - Tab Control: Contenedor con Pestañas para Ordenar Elementos

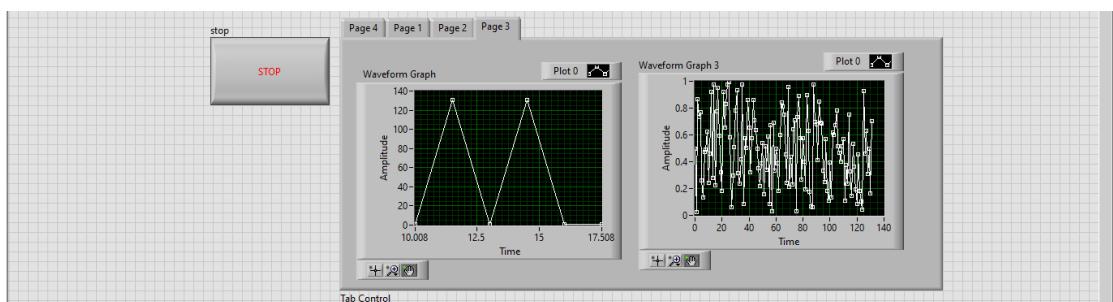
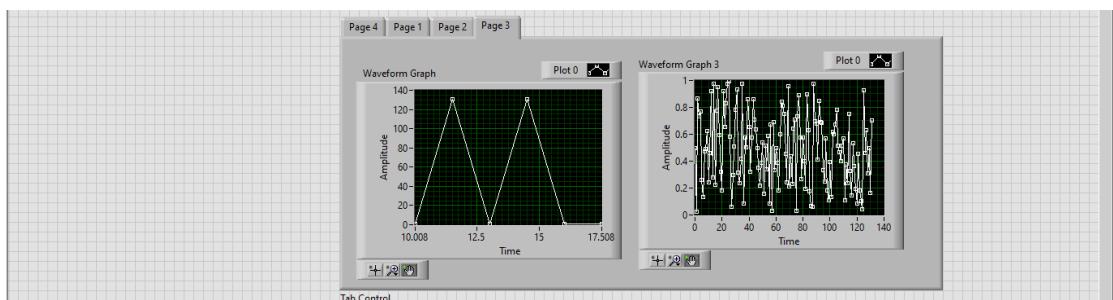
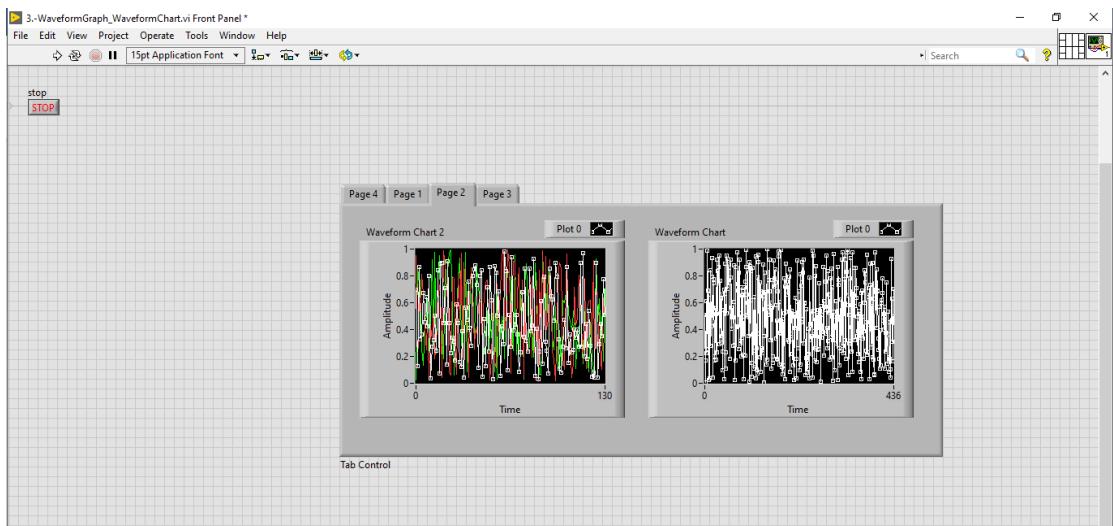
El Tab Control me permite crear un contenedor que ordene todos los elementos incluidos en mi programa.



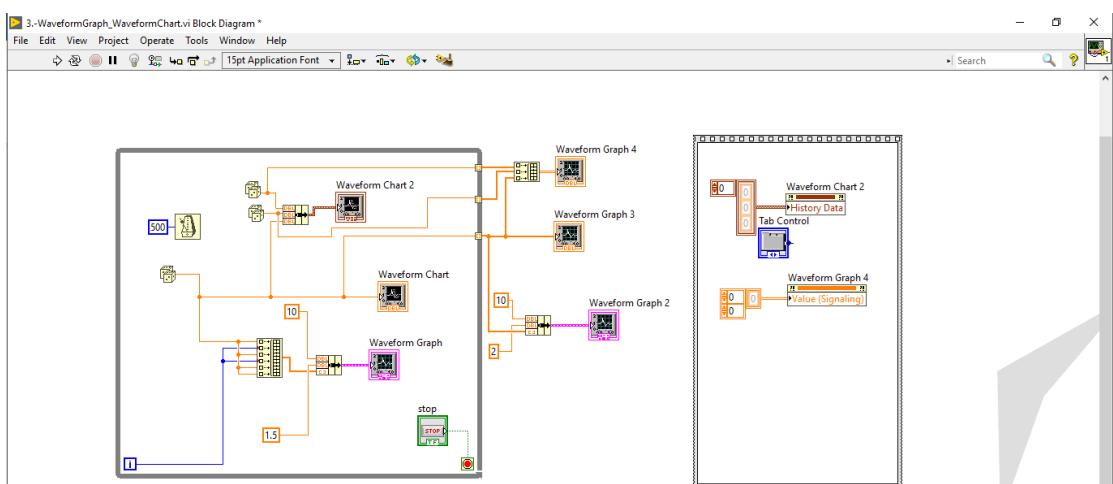
En la primera pestaña voy a meter las gráficas que tenga dentro del bucle while, a continuación se muestra una descripción de cada gráfica creada:

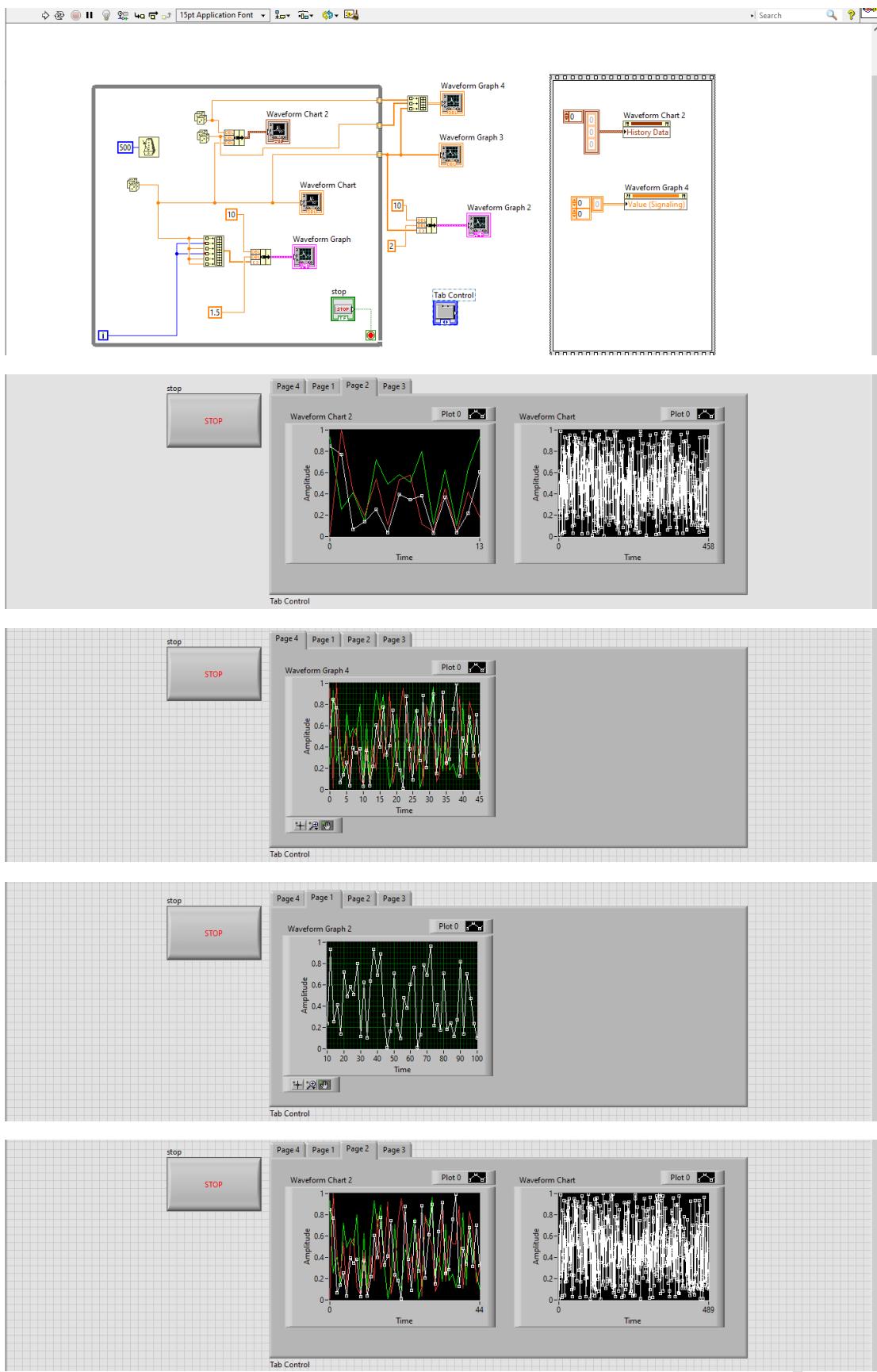
- **Waveform Chart (**Dynamic Data**)**: Gráfica creada con 1 solo valor numérico aleatorio dentro del bucle while.
- **Waveform Chart 2 (**Dynamic Data**)**: Gráfica creada con 3 valores numéricos aleatorios almacenados en un array intermedio, se encuentra todo dentro del bucle while.
- **Waveform Graph (**Arrays y Clusters**)**: Gráfica creada con 1 array de valores numéricos aleatorios y 2 constantes almacenadas en un array intermedio, se encuentra todo dentro del bucle while.
- **Waveform Graph 2 (**Arrays y Clusters**)**: Gráfica creada con 1 array de 1 túnel de valores numéricos aleatorios que sale del bucle while y 2 constantes creadas fuera del bucle while, todo almacenado en un array intermedio, se encuentra fuera del bucle while.
- **Waveform Graph 3 (**Arrays y Clusters**)**: Gráfica creada con 1 valor numérico aleatorio que sale por un túnel del bucle while, se encuentra fuera del bucle while.
- **Waveform Graph 4 (**Arrays y Clusters**)**: Gráfica creada con 1 array de 3 túneles de valores numéricos aleatorios que salen del bucle while, se encuentra fuera del bucle while.

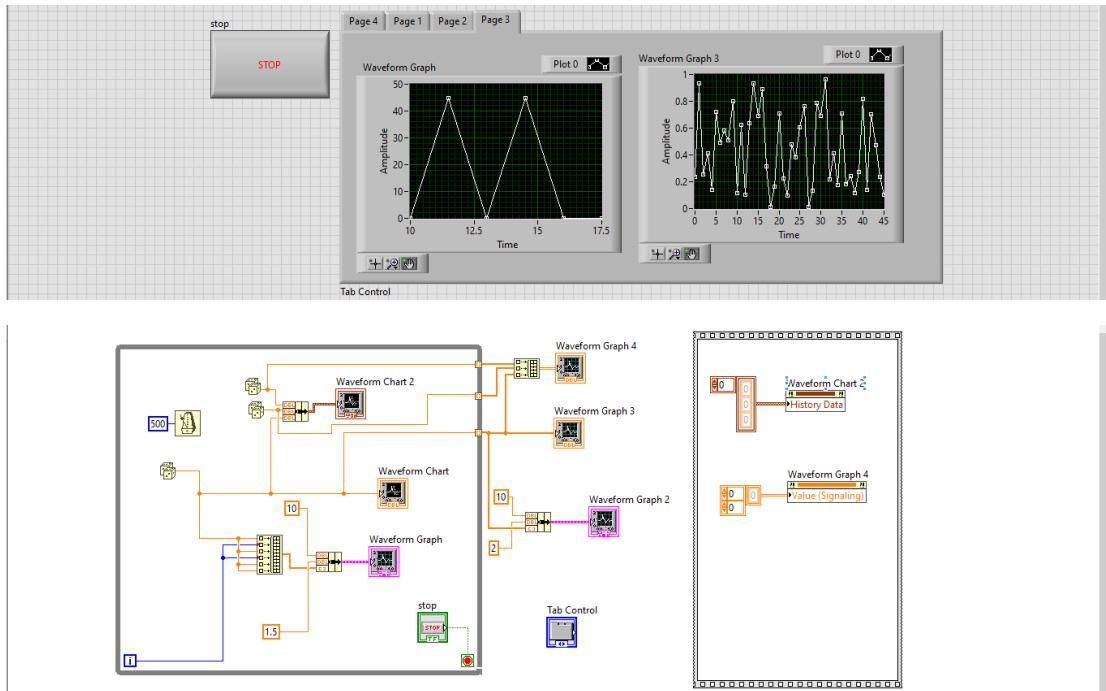




Esto en la ventana del Block Diagram se ve como un bloque de Tab control que no hace nada.

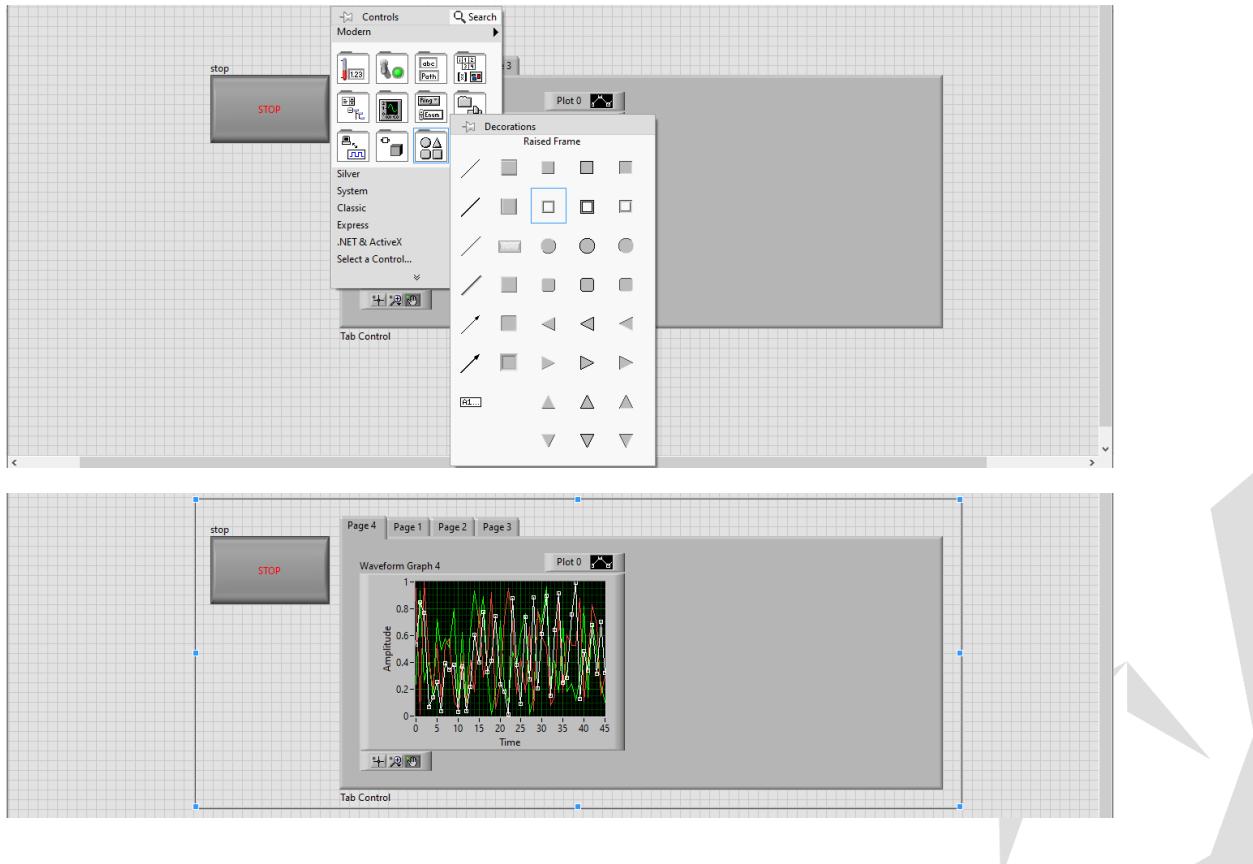


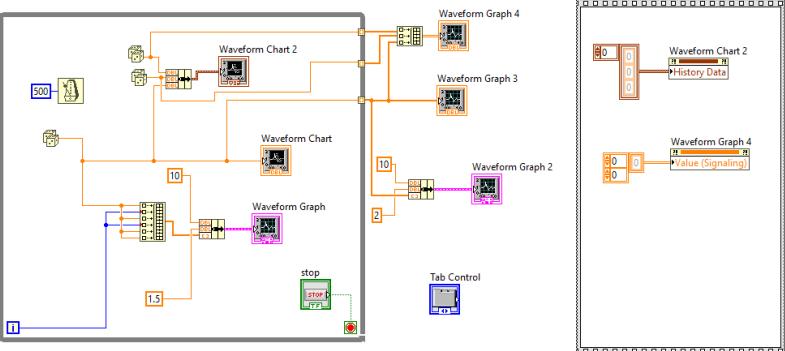




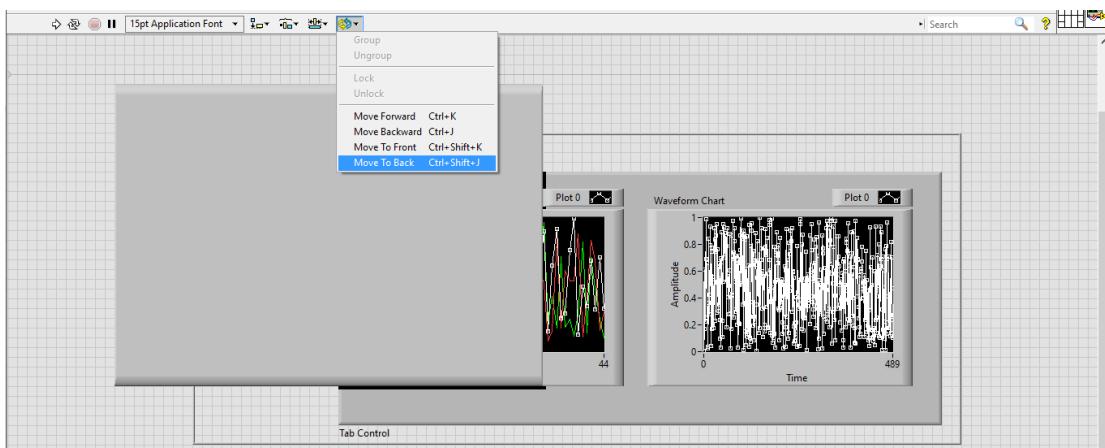
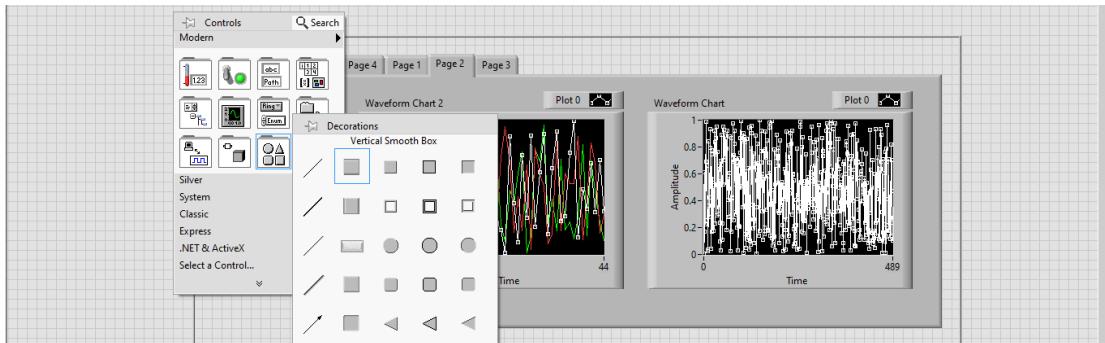
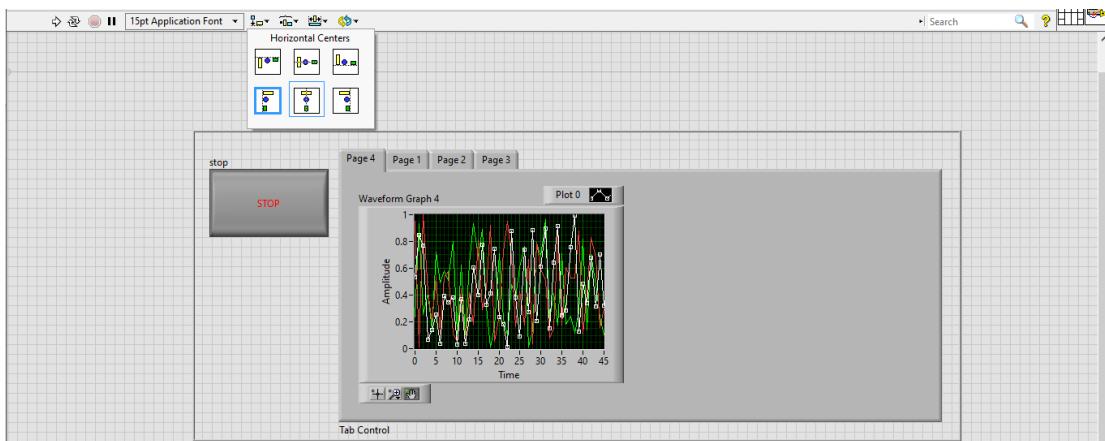
Front Panel - Decorations: Elementos de Diseño de Interfaz, no afectan al Block Diagram

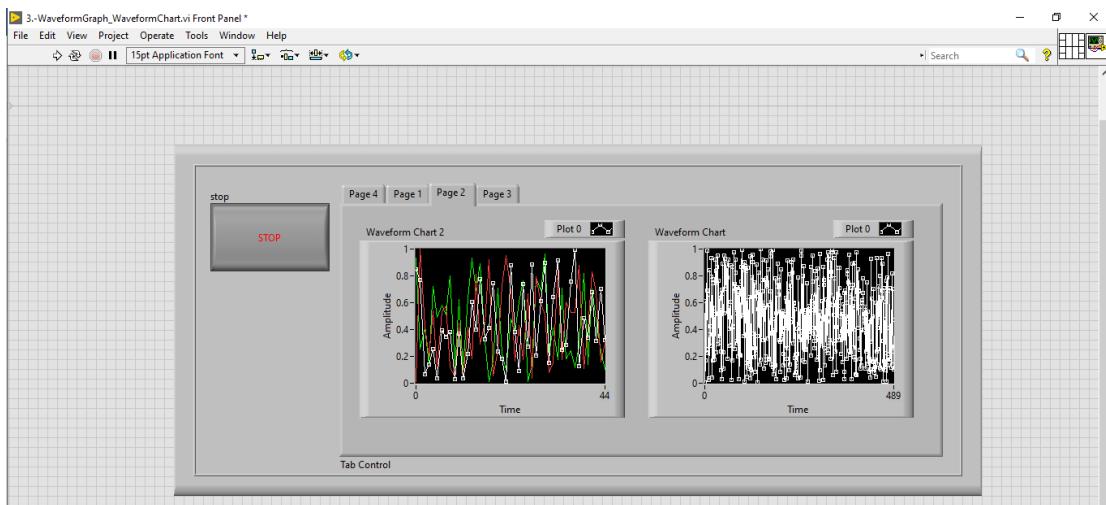
Además, puedo agregar más elementos de diseño, que no se verán en la pestaña de bloques porque no están haciendo nada, como círculos, rectángulos, etc.





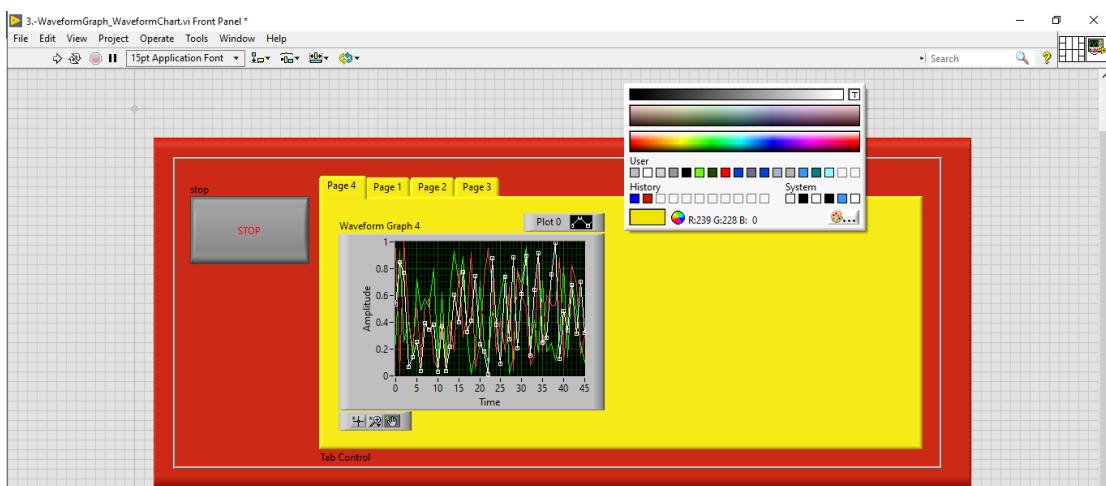
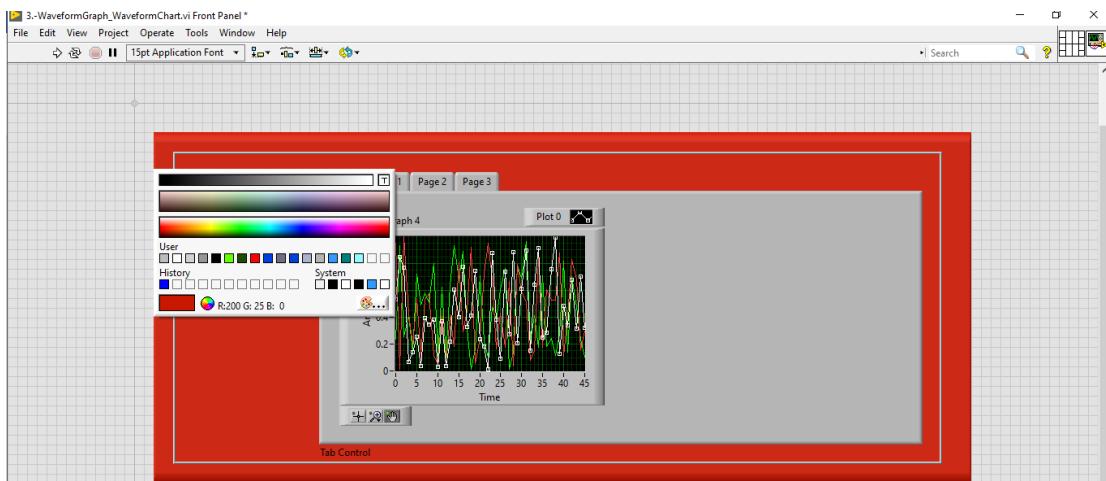
Para acomodar la interfaz de usuario vamos a hacer lo siguiente.

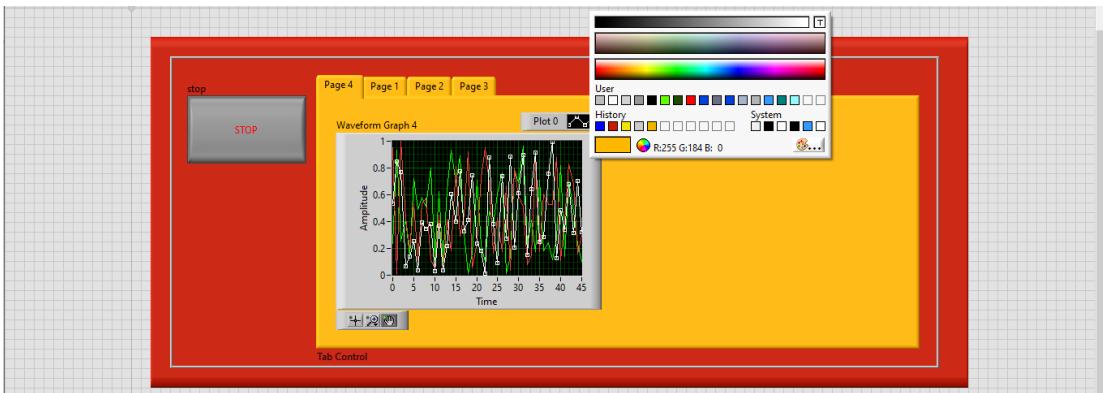
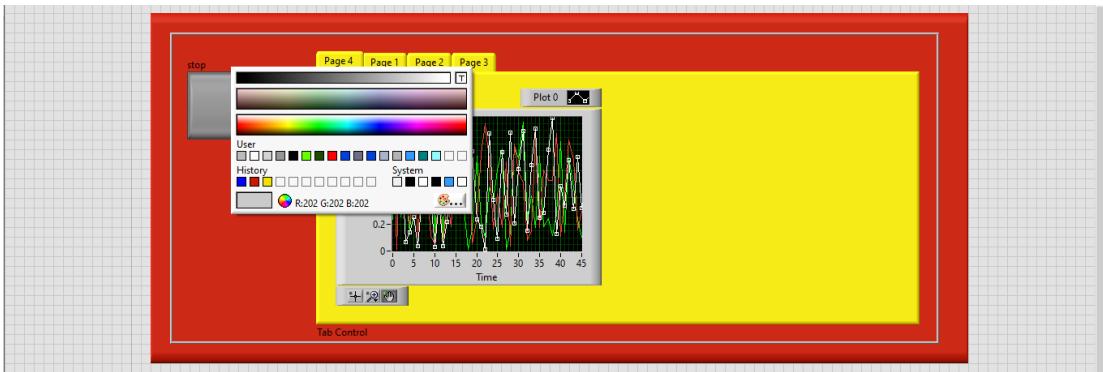




Front Panel - SHIFT+Clic Derecho: Pluma para Colorear la Interfaz

Para que le pueda dar color, debo dar clic en SHIFT + clic derecho y saldrá una plumita que me permitirá colorear mi interfaz gráfica. Ya que aparezca la brochita debo dar clic derecho otra vez para elegir que color le quiero dar a mi elemento.





Para hacer que se deje de mostrar la plumita debo dar clic en SHIFT + clic derecho otra vez.

