

INGENIERÍA MECATRÓNICA



DI_CERO

DIEGO CERVANTES RODRÍGUEZ

INSTRUMENTACIÓN VIRTUAL

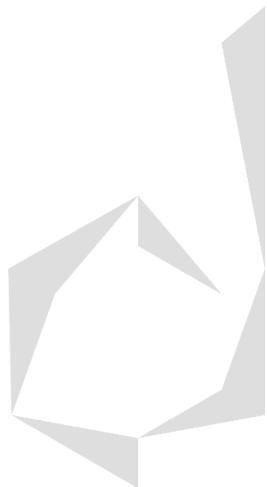
NI LABVIEW 2020 (32-BIT)

Data Socket: Conexión Remota de
VIs por Medio de Bloques

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: Conexión de VIs con el Protocolo Data Socket	10
Desarrollo del Programa: Transmisión y Recepción de Datos	10
DataSocket Server - Aplicación NI: Protocolo de Comunicación Entre Ordenadores.....	10
DataSocket Server: VI 1 - MASTER: Programa que Envía las Instrucciones a la otra VI.....	11
Block Diagram - Bucle While: Ejecución Continua del Programa	11
Block Diagram - Basic Function Generator: Simulación de un Generador de Funciones	12
Block Diagram - DataSocket Open: Abrir una Comunicación Inalámbrica por DataSocket	13
Block Diagram - DataSocket Write: Mandar Instrucciones o Transmitir Datos por DataSocket ...	14
Block Diagram - Bucle While - Shift Register: Registros de Memoria dentro de un Ciclo.....	16
Front Panel - Waveform Graph: Ventana que Muestra una Señal (Array)	19
Block Diagram - Wait Until Next ms Multiple: Temporizador en milisegundos.....	21
Block Diagram - DataSocket Close: Cerrar Comunicación Inalámbrica por DataSocket	23
DataSocket Server: VI 2 - SLAVE: Programa que Recibe las Instrucciones de la otra VI	24
Block Diagram - DataSocket Open: Abrir una Comunicación Inalámbrica por DataSocket	25
Block Diagram - DataSocket Close: Cerrar Comunicación Inalámbrica por DataSocket	25
Block Diagram - DataSocket Read: Recibir Instrucciones o Datos por DataSocket.....	25
Block Diagram - Bucle While: Ejecución Continua del Programa	26
Front Panel - Waveform Graph: Ventana que Muestra una Señal (Array)	28
Block Diagram - Wait Until Next ms Multiple: Temporizador en milisegundos.....	28
Ejecución del Programa: Transmisión y Recepción de Datos con el Data Socket Activado	31
DataSocket Server: VI 2 - SLAVE: Programa que Recibe las Instrucciones de la otra VI	31

DataSocket Server: VI 1 - MASTER: Programa que Envía las Instrucciones a la otra VI..... 31



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.

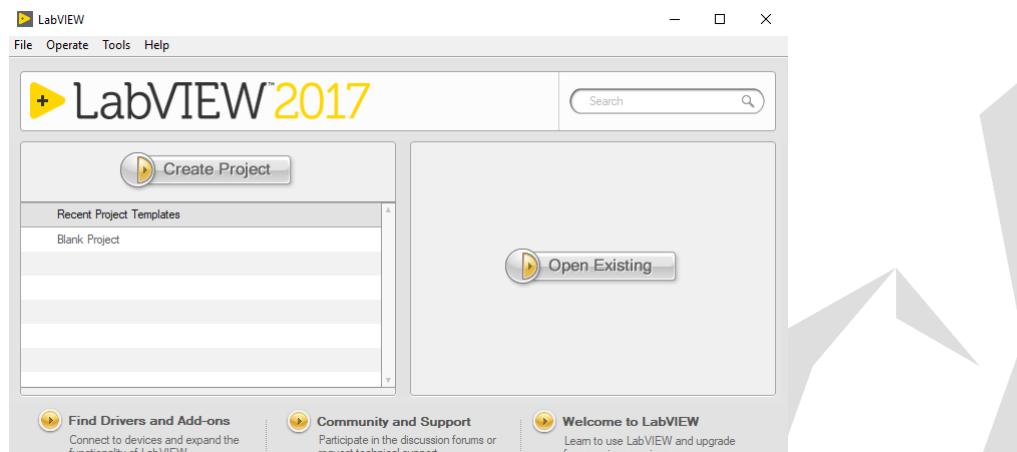
- **Instrumentación Tradicional:** El hardware es más usado, como por ejemplo con los circuitos integrados de un osciloscopio.
- **Instrumentación Virtual:** El software es el más utilizado y sus funciones son modulares, como lo es en una tarjeta de desarrollo de National Instruments.

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

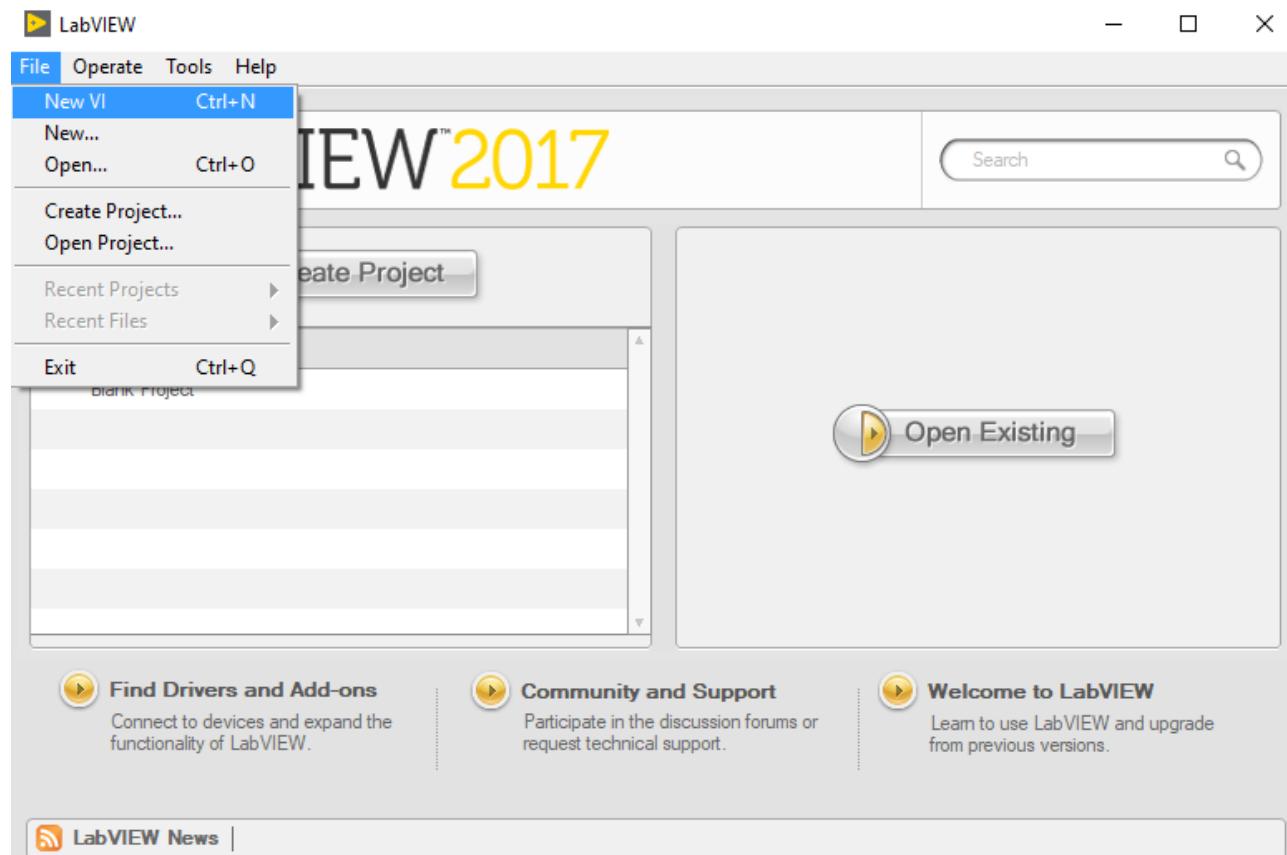


Introducción al Entorno de LabVIEW:

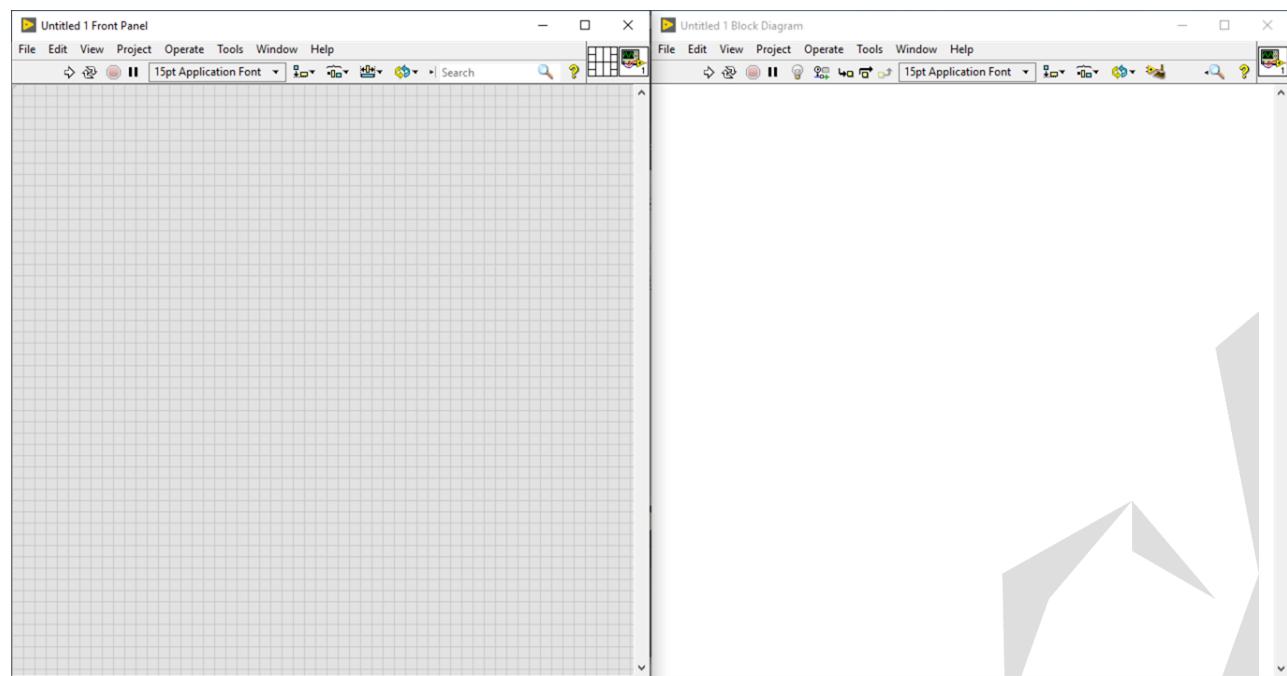
Un nuevo proyecto de LabView se abre por medio del botón de Create project que aparece inmediatamente cuando abra el programa.



VI se refiere a Virtual Instrument.

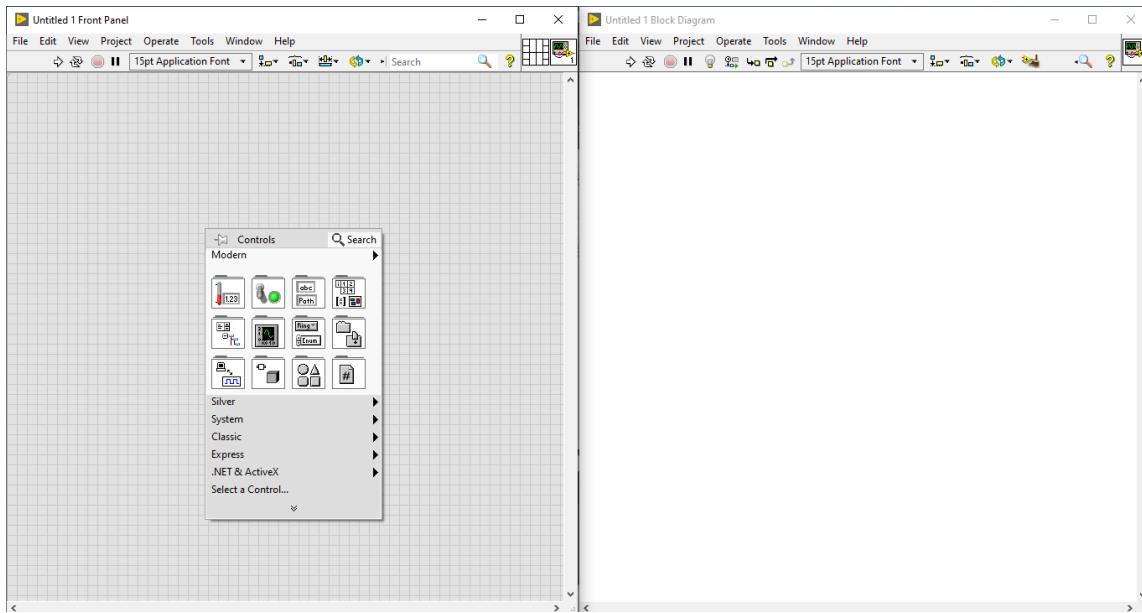


Al hacerlo me abrirá estas dos ventanas, en una de ellas se creará el programa con bloques (Ventana Block Diagram) y en la otra se verá la interfaz (Ventana Front Panel).



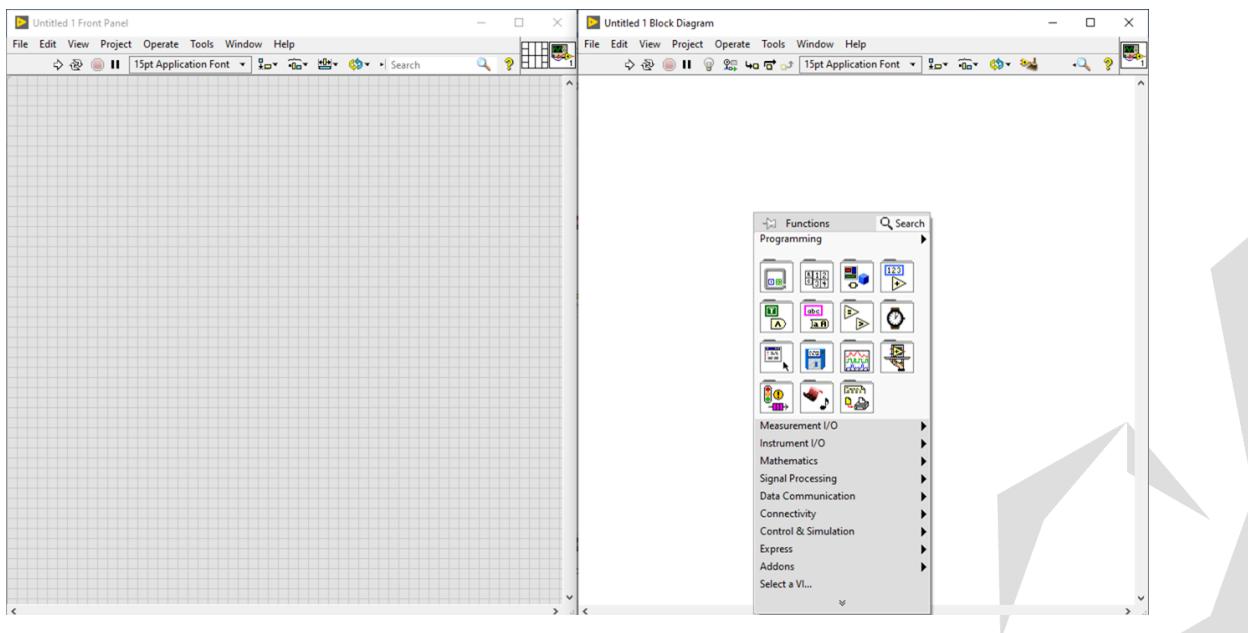
Front Panel: Ventana Gris con la Interfaz del Programa

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.



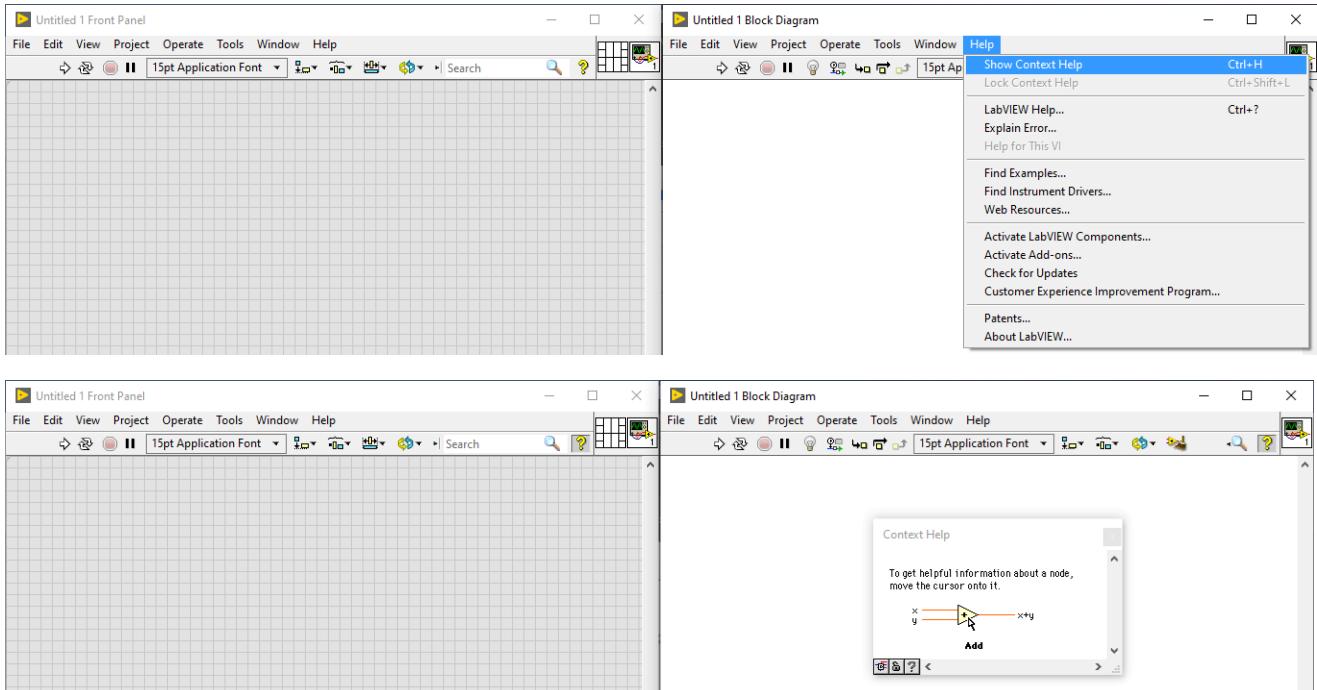
Block Diagram: Ventana Blanca con la Lógica del Programa (Bloques)

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.



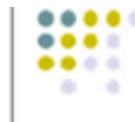
Front Panel o Block Diagram - Show Context Help: Descripción de Bloques

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:

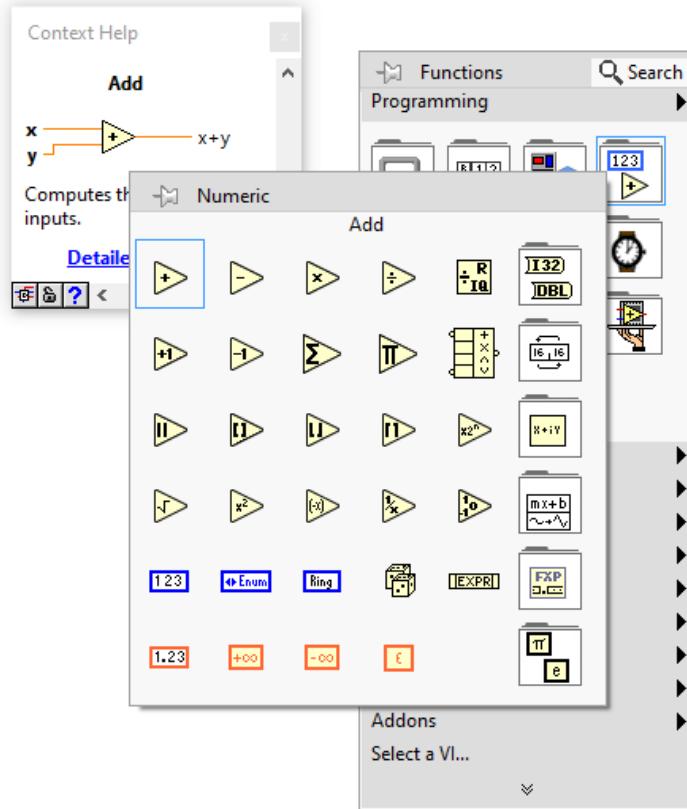
VIs Express, VIs y Funciones



- **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.

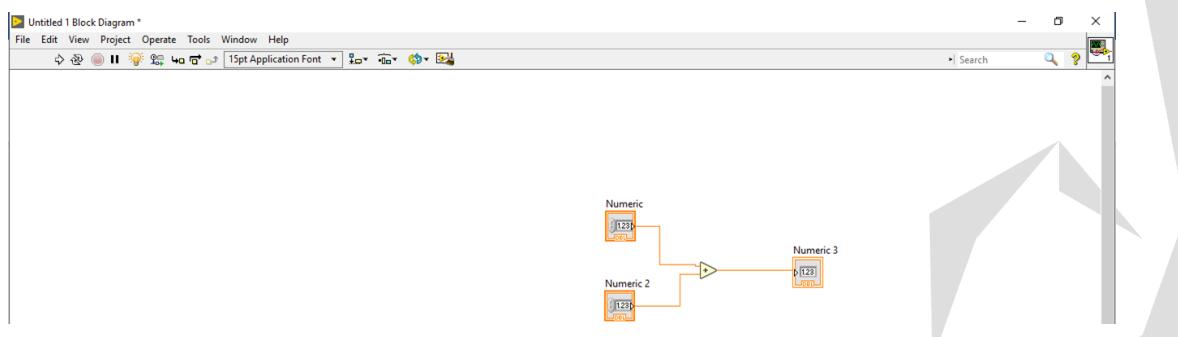


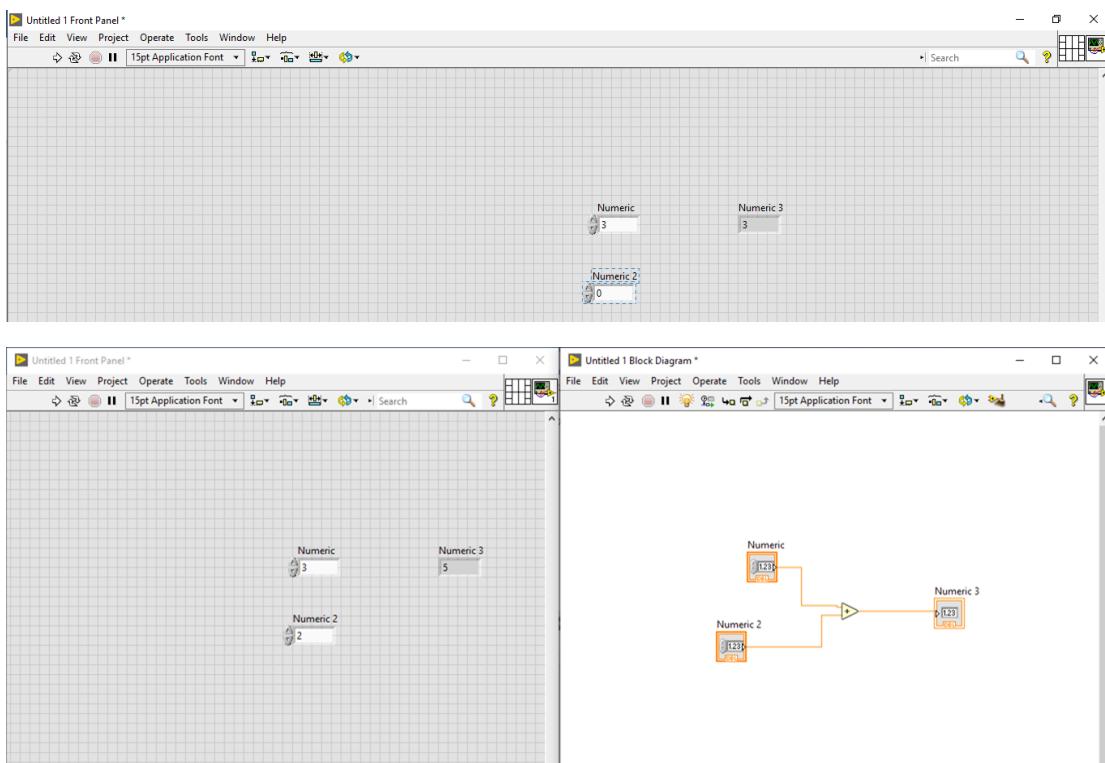
El programa es autocompilable, es decir que se corre por sí solo, por lo que si la flechita aparece rota es porque hay un error en el programa.



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

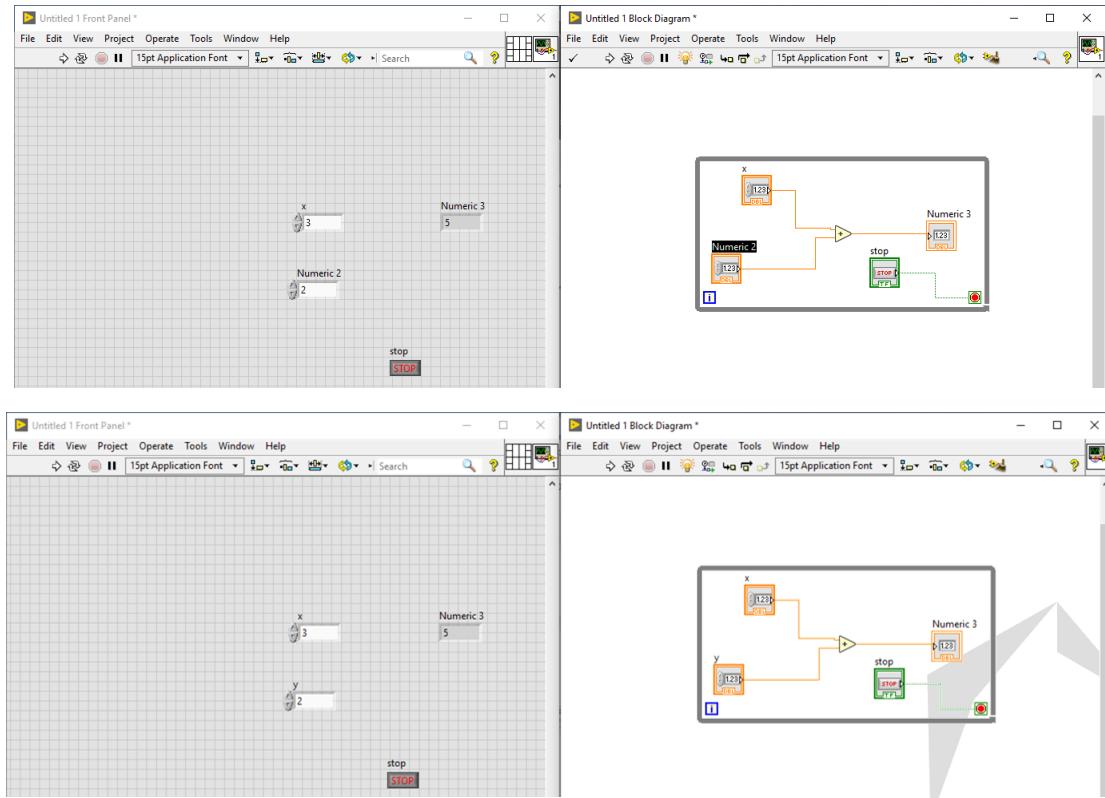
Al dar doble clic en el bloque de la pantalla blanca, me llevará al punto donde se encuentra el mismo bloque, pero en la pantalla gris.

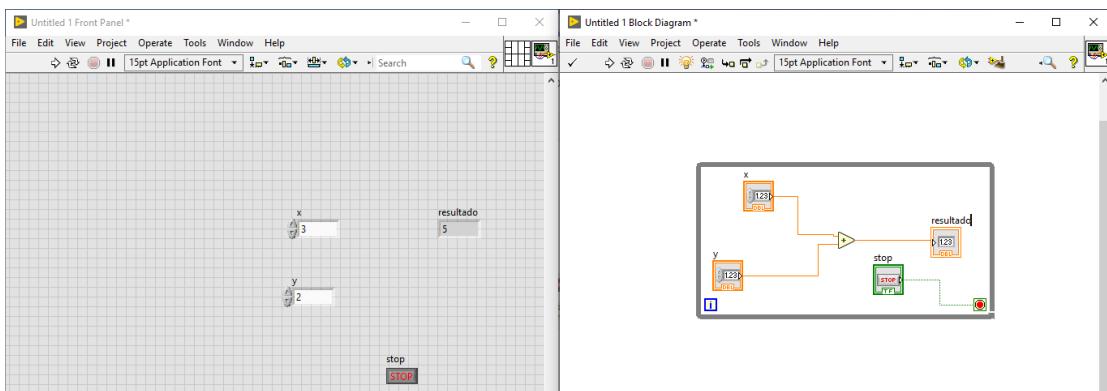




Block Diagram - Cambiar Nombre a los Bloques: Nombre de los elementos en el Front Panel

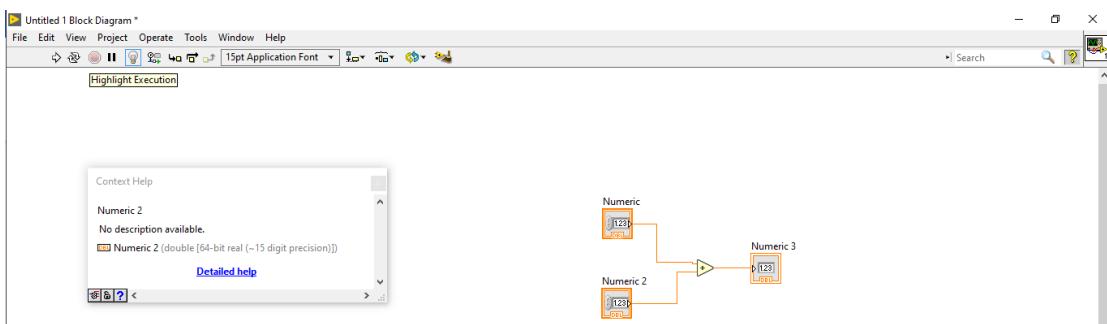
El nombre de los elementos de las interfaces se puede cambiar desde el Block Diagram, cambiándole literal el nombre a los bloques.





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.

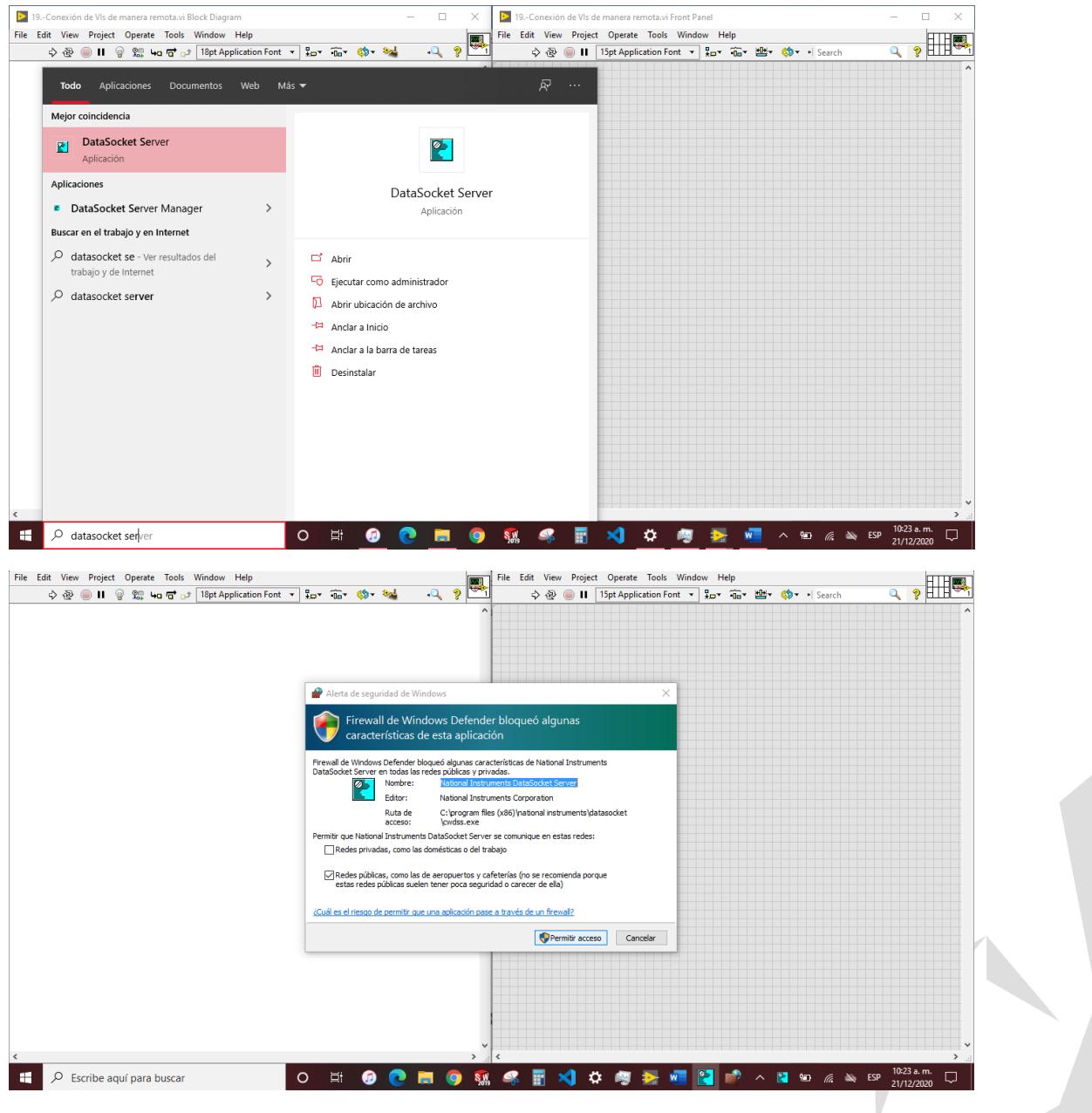
Programa: Conexión de VIs con el Protocolo Data Socket

Se conectarán dos VIs (programas de LabVIEW) distintas por medio del protocolo Data Socket.

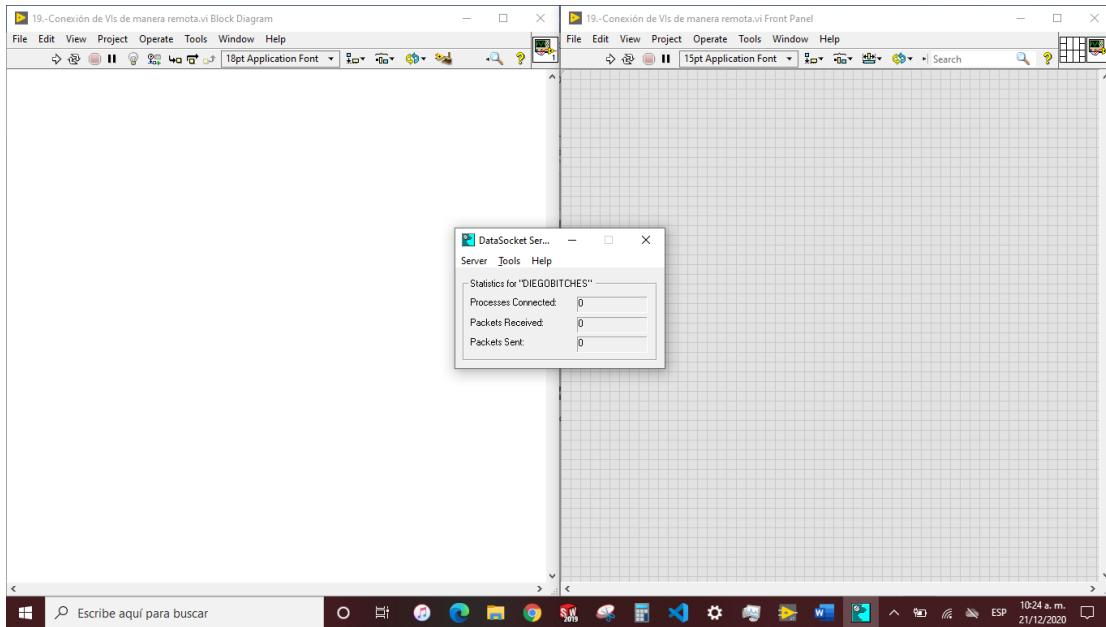
Desarrollo del Programa: Transmisión y Recepción de Datos

DataSocket Server - Aplicación NI: Protocolo de Comunicación Entre Ordenadores

Vamos a utilizar la herramienta de **DataSocket Server** perteneciente a National Instruments que es la empresa creadora de LabVIEW para activar el protocolo de Data Socket en nuestro ordenador, que es utilizado para comunicar dos computadoras, aunque en este caso se utilizará a usar la misma computadora y su servidor local.



La herramienta de **DataSocket Server** nos va a activar el protocolo Data Socket Server que activará en la computadora un servidor local específico.



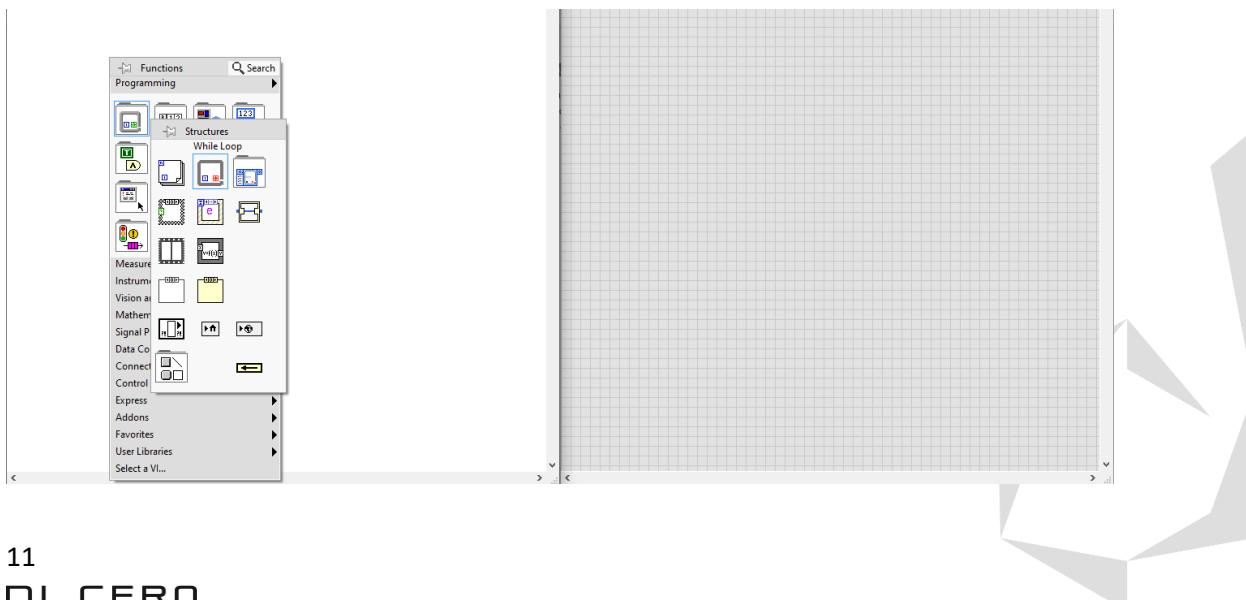
Esto no cambia la configuración del servidor local en mi computadora, solo determina el protocolo por medio del cual accedemos al servidor local de la computadora, en este caso usaremos el DSTP (Data Socket Transport Protocol), la dirección IP del servidor local en la computadora es 127.0.0.1 o localhost.

Para este programa entonces crearé 2 VIs diferentes, donde una VI obedecerá a la otra:

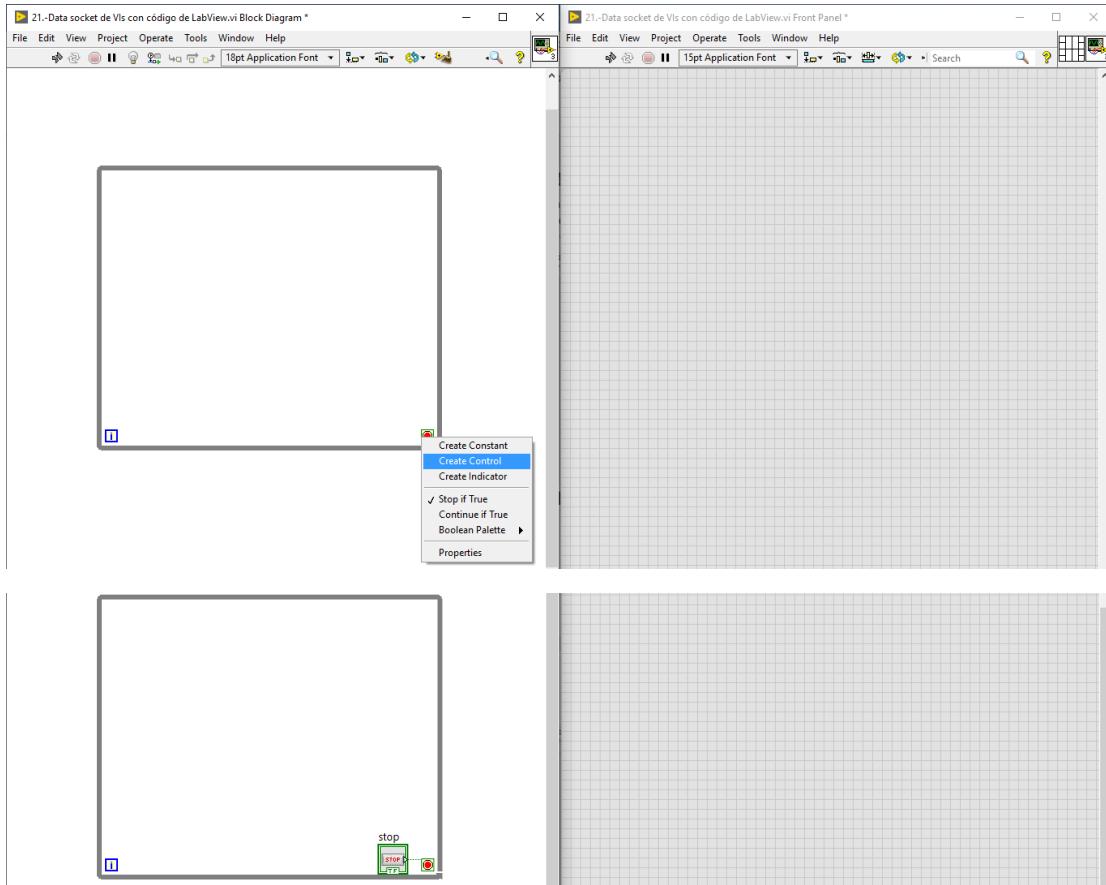
DataSocket Server: VI 1 - MASTER: Programa que Envía las Instrucciones a la otra VI

Block Diagram - Bucle While: Ejecución Continua del Programa

El ciclo while hace que el programa se ejecute hasta que dé clic en el botón de STOP, por eso todo el diagrama de bloques que tengo actualmente lo voy a encerrar en un ciclo while para que esté ejecutando de manera continua.

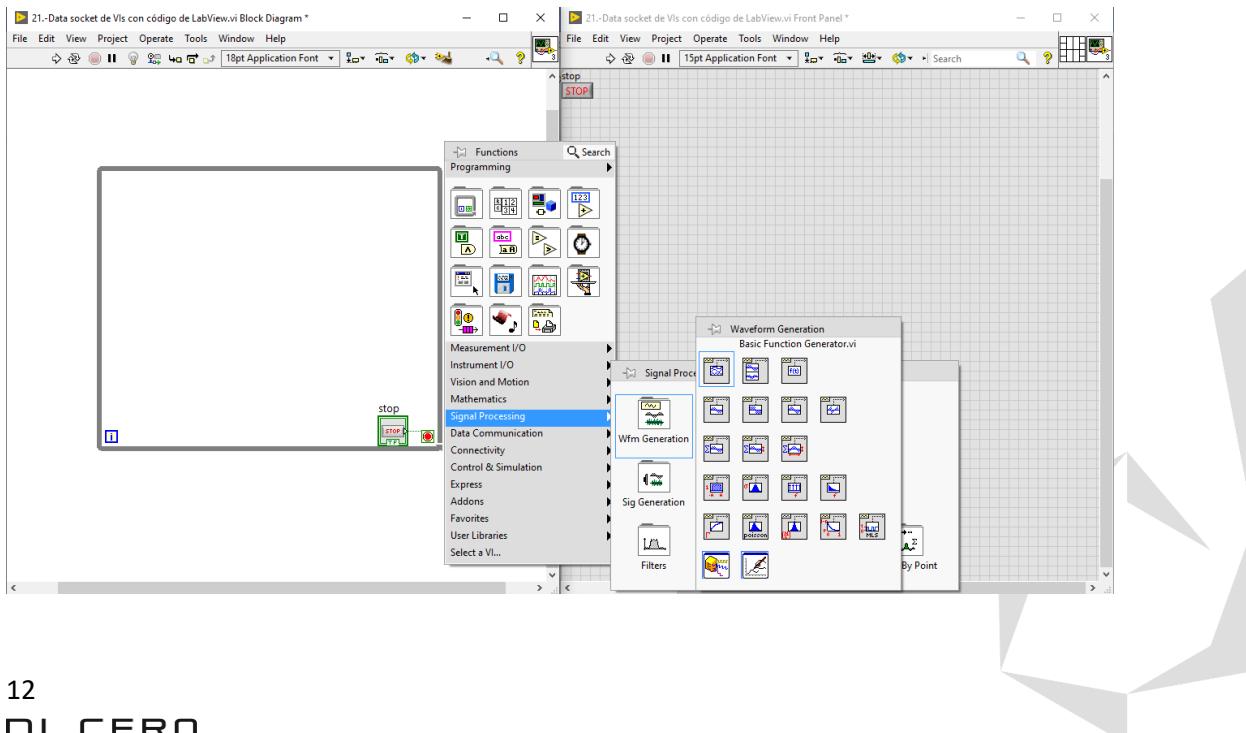


Crear un Control para un Bloque: Clic derecho en el bloque → Create Control.

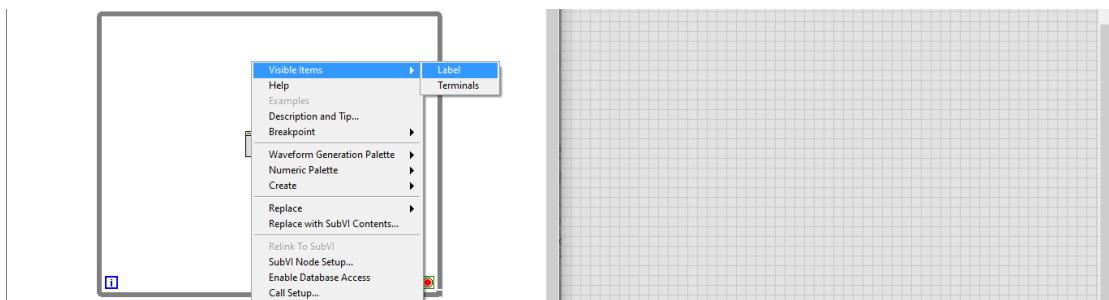


Block Diagram - Basic Function Generator: Simulación de un Generador de Funciones

Simulación de un generador de funciones que puede crear varios tipos de señales virtuales en LabVIEW.

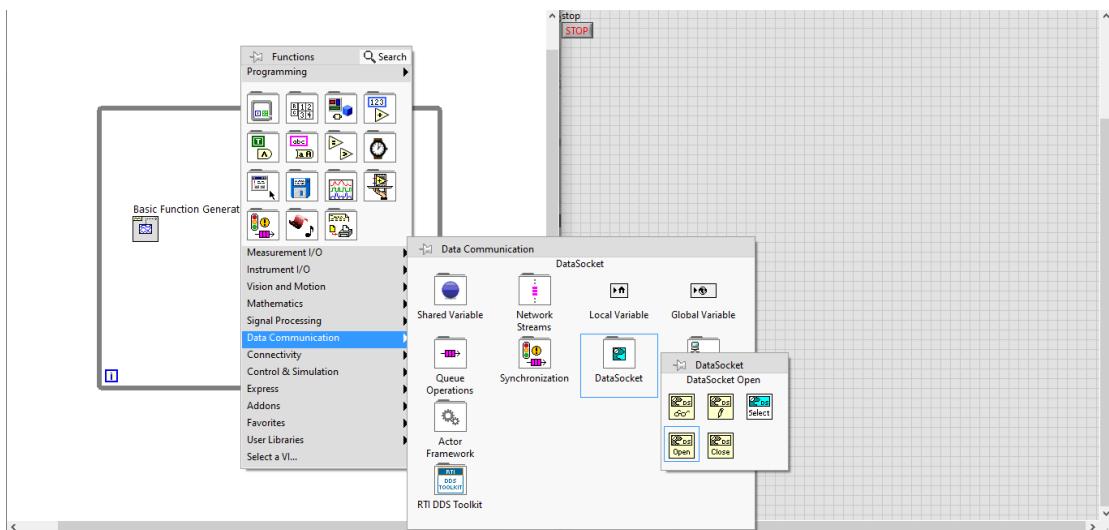


Mostrar nombre del bloque: Clic derecho → Visible Items → Label.

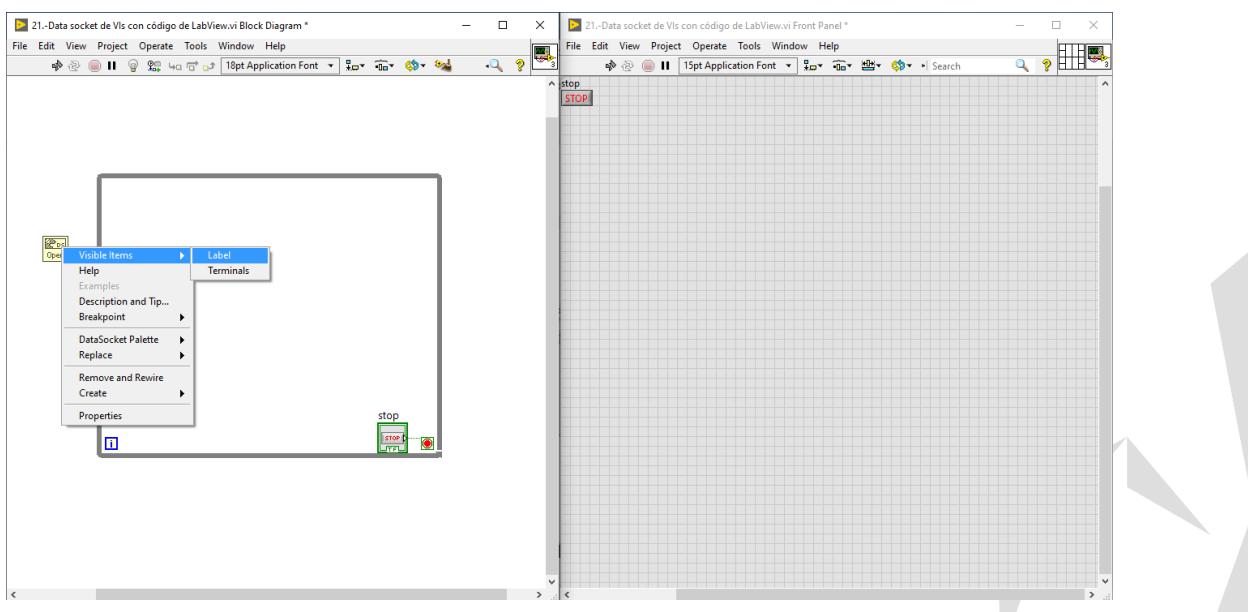


Block Diagram - DataSocket Open: Abrir una Comunicación Inalámbrica por DataSocket

El bloque de DataSocket Open sirve para abrir una comunicación inalámbrica y remota entre VIs.

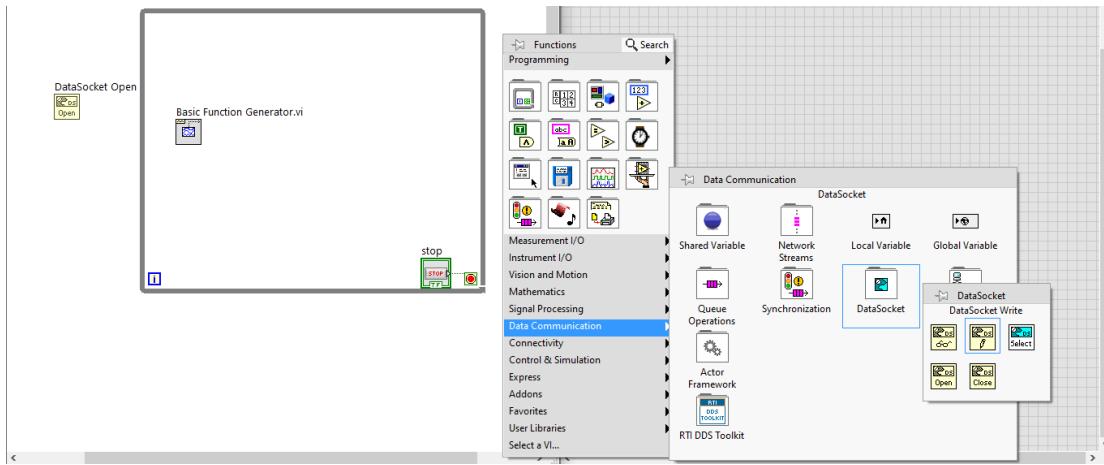


Mostrar nombre del bloque: Clic derecho → Visible Items → Label.

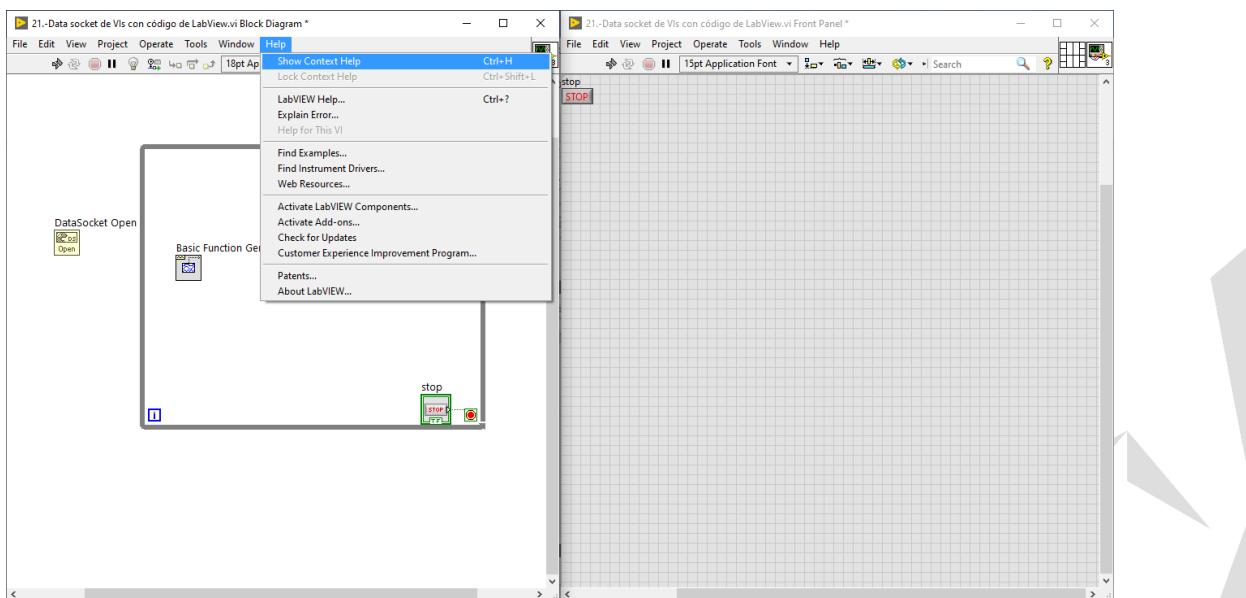
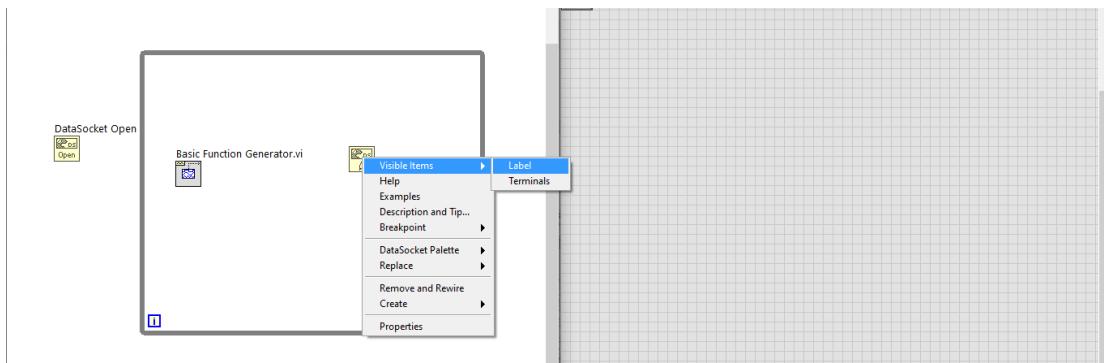


Block Diagram - DataSocket Write: Mandar Instrucciones o Transmitir Datos por DataSocket

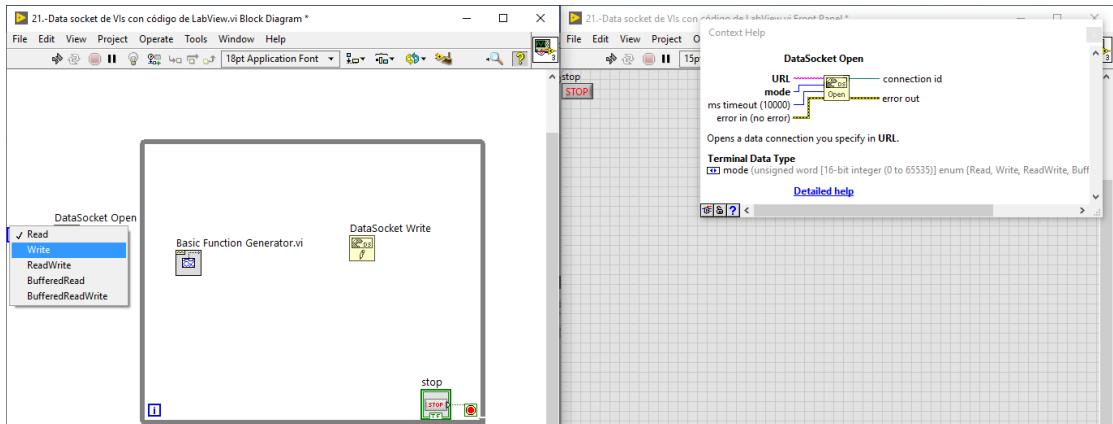
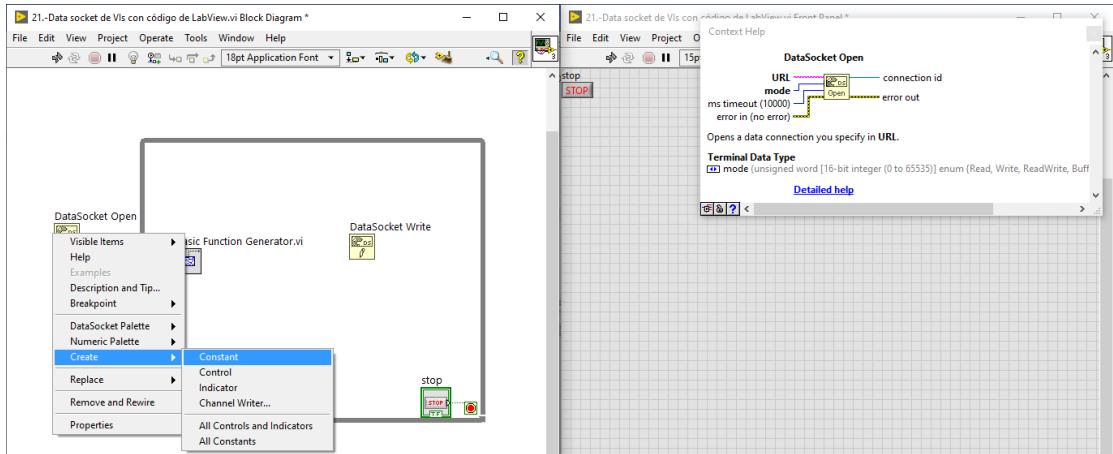
El bloque de DataSocket Write sirve para mandar instrucciones o datos de forma inalámbrica y remota entre VIs, donde uno será el que mande las instrucciones o datos (MASTER) y el otro u otros serán los que los reciban (SLAVE).



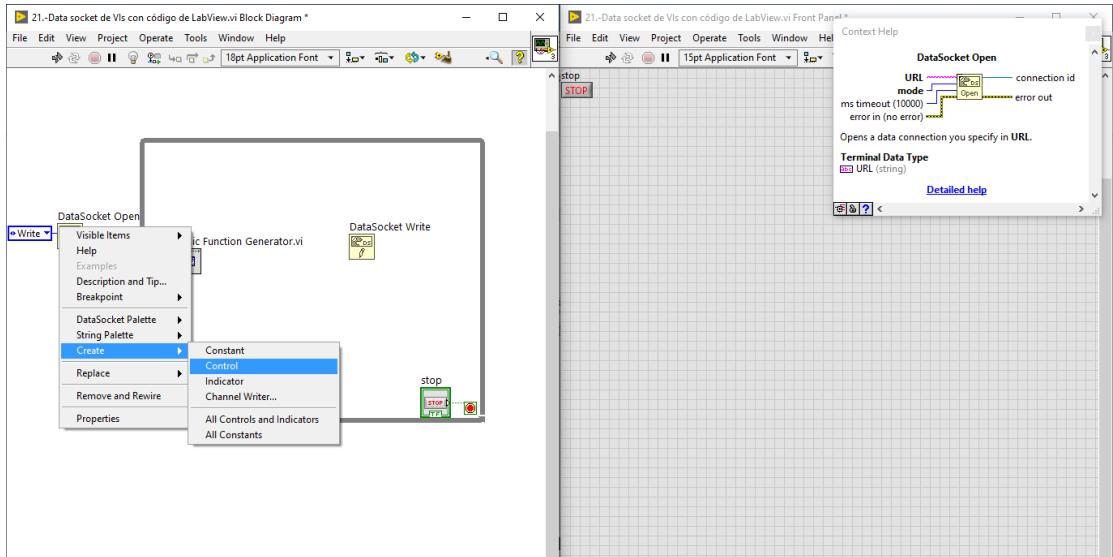
Mostrar nombre del bloque: Clic derecho → Visible Items → Label.



Crear una Constante para un Bloque: Clic derecho en la terminal del bloque de interés → Create → Constant. Esta constante sirve para elegir si la comunicación se quiere hacer de lectura, escritura, lectura y escritura, etc.

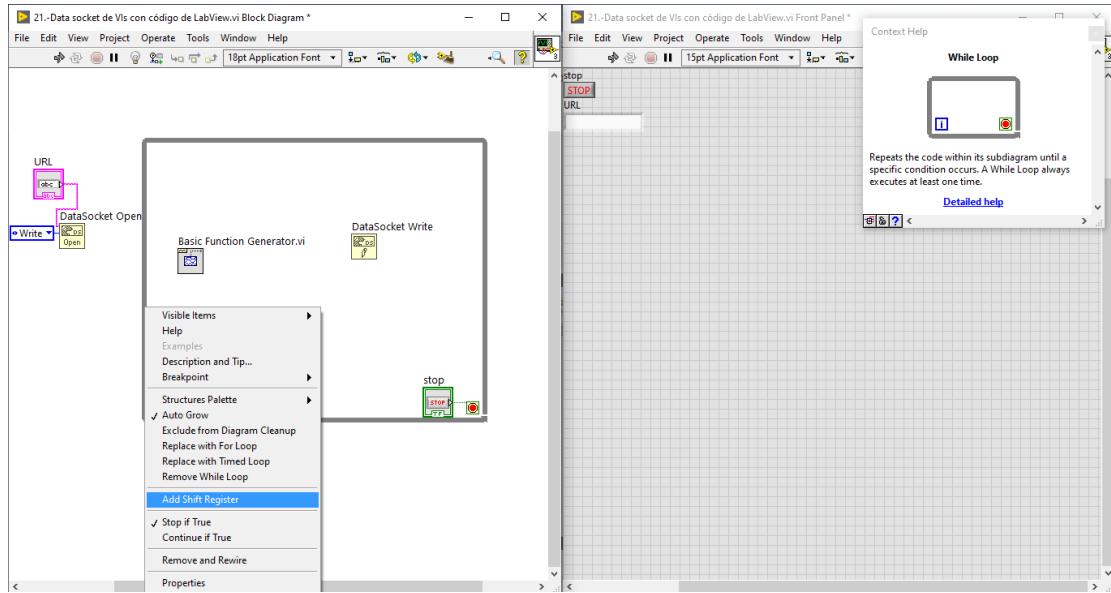


Crear un Control para un Bloque: Clic derecho en la terminal del bloque de interés → Create → Control. Este control permite ingresar una URL por medio de la interfaz del Front Panel.

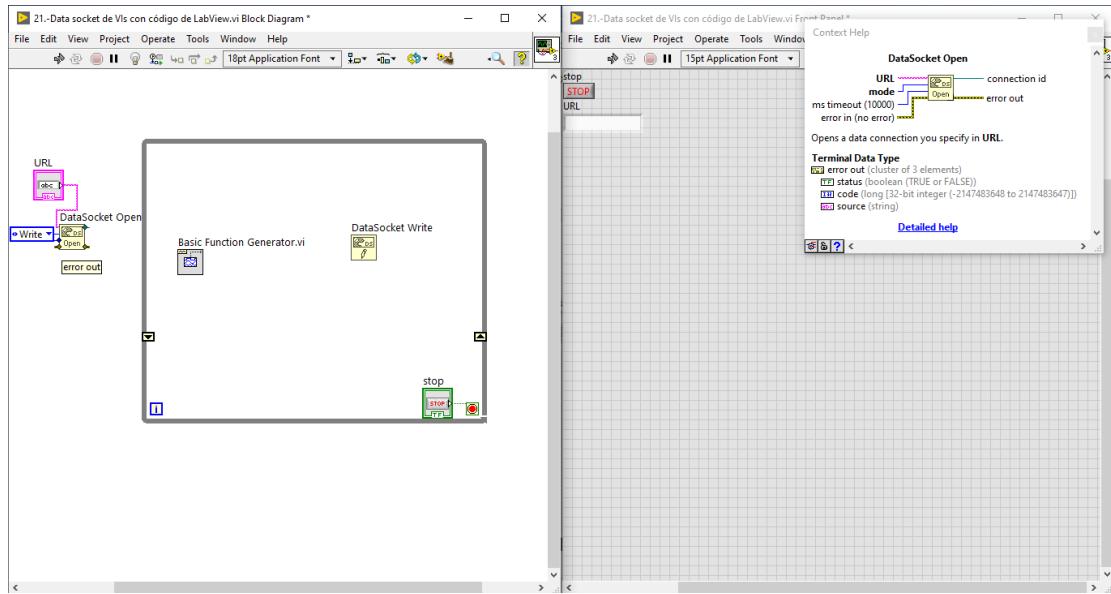


Block Diagram - Bucle While - Shift Register: Registros de Memoria dentro de un Ciclo

Ahora vamos a dar clic derecho al bucle y seleccionar la opción de Add Shift Register, esto es para que almacenemos en una dirección de memoria temporal algún dato generado en un ciclo for o while.

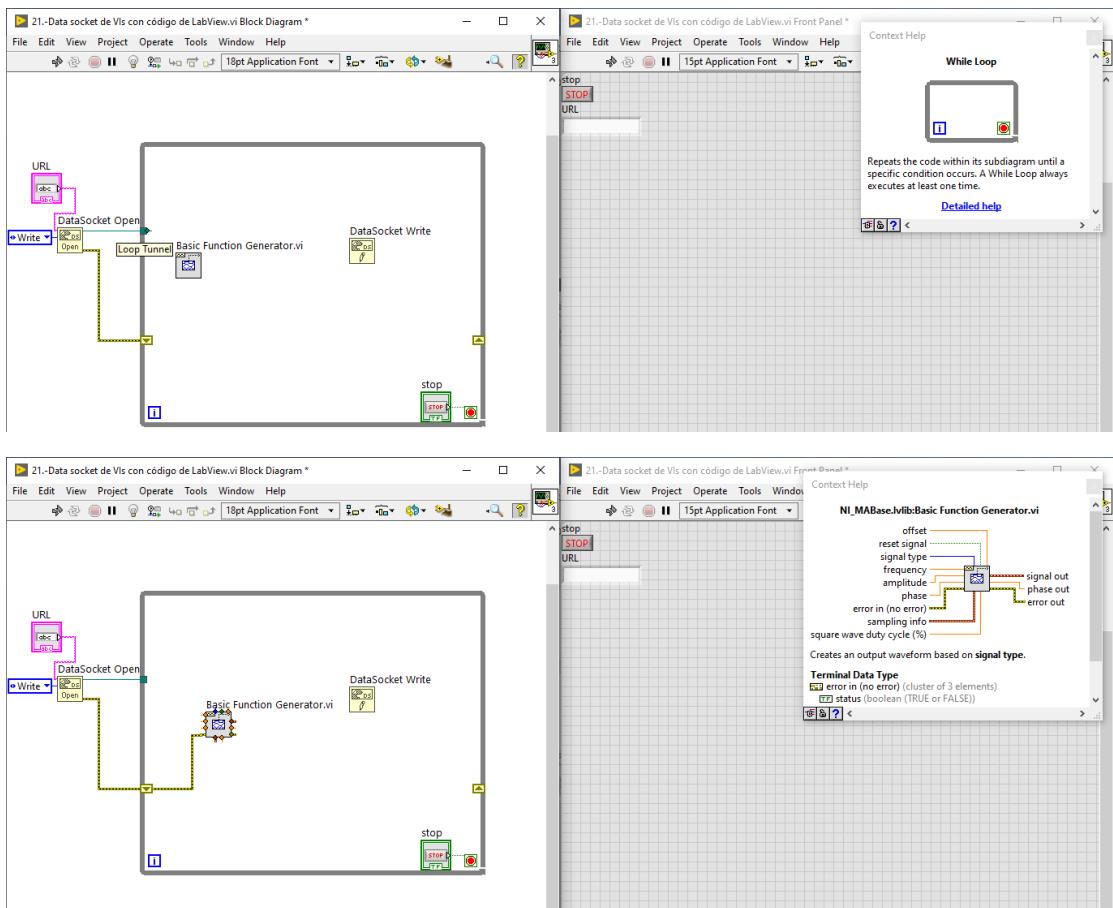


Cuando el Shift Register se muestra de color negro es porque no se le ha asignado un dato.

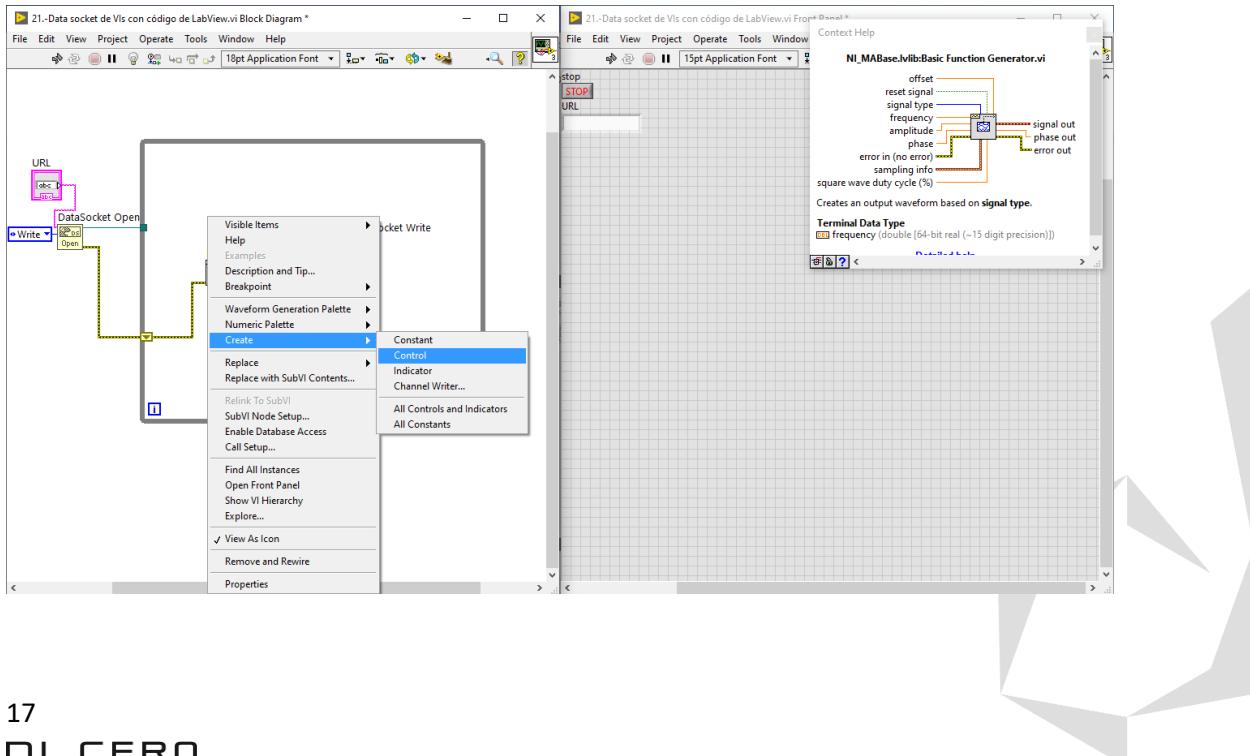


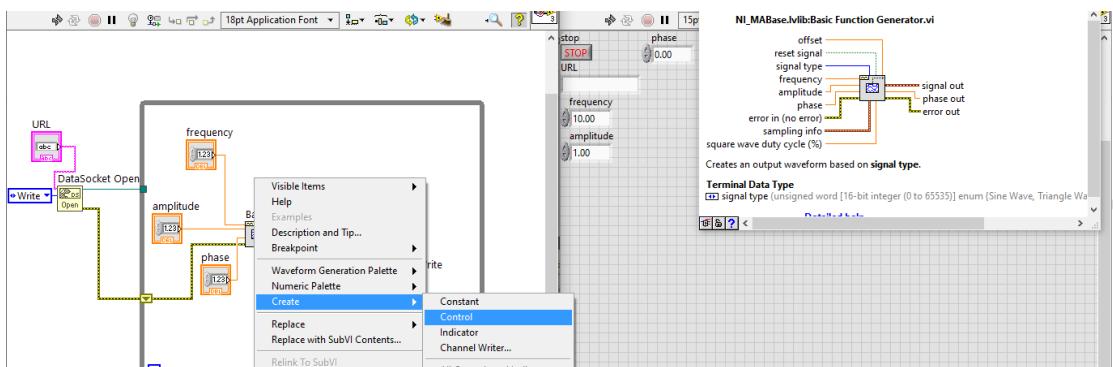
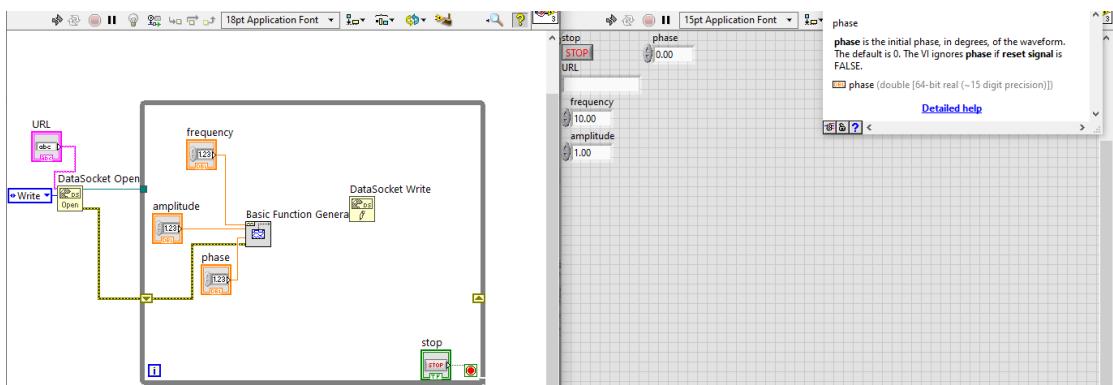
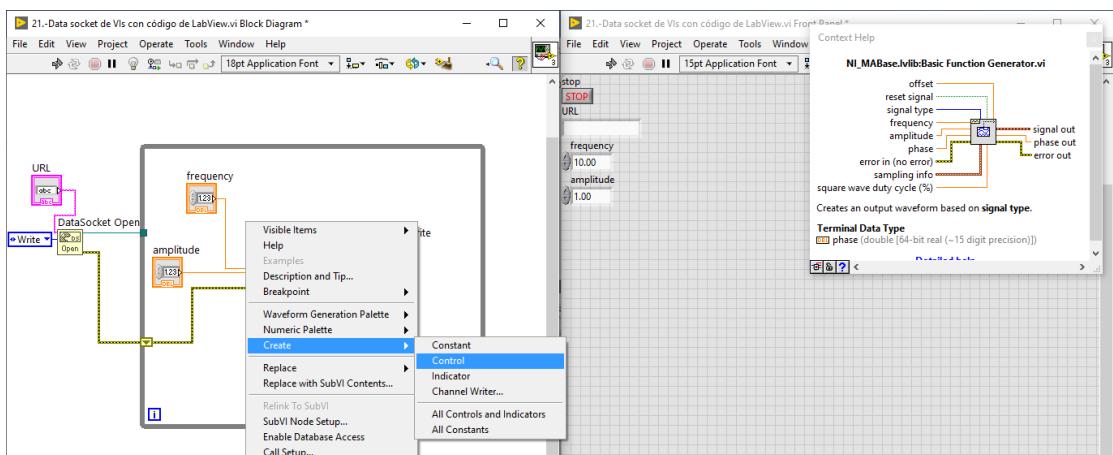
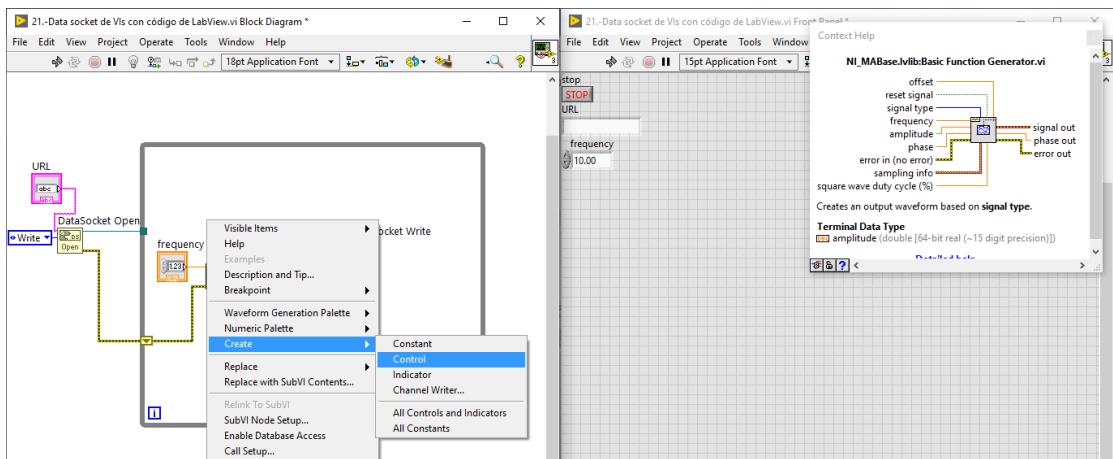
Si lo extiendo puedo ver las direcciones de memoria a donde se van a estar asignando los valores, cuando un dato entre al primer registro (posición de arriba hacia abajo) del Shift Register, si tenía un dato almacenado ahí, lo va a mover a la siguiente posición (hacia abajo) y así se estarán moviendo y moviendo los datos en los registros del Shift Register mientras vayan entrando más y más datos a la memoria, hasta que el último dato ya no se pueda mover a otra posición, entonces se borrará ese dato.

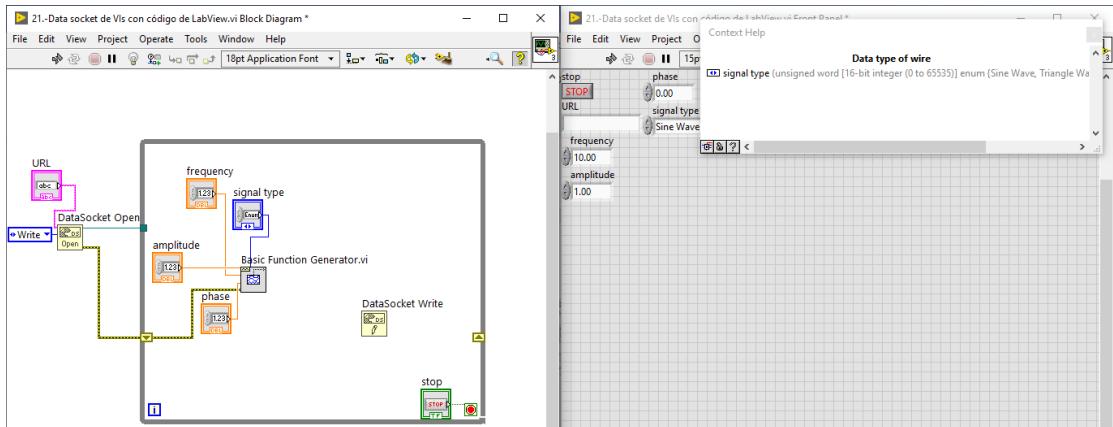
En este caso a los registros de corrimiento (los de la izquierda) va a entrar la terminal del error proveniente del bloque de DataSocket Open por medio de un túnel.



Crear un Control para un Bloque: Clic derecho en la terminal del bloque de interés → Create → Control. Estos controles permitirán introducir la amplitud, frecuencia, fase y tipo de la señal creada por medio del generador de funciones en LabVIEW, que originará una señal virtual.

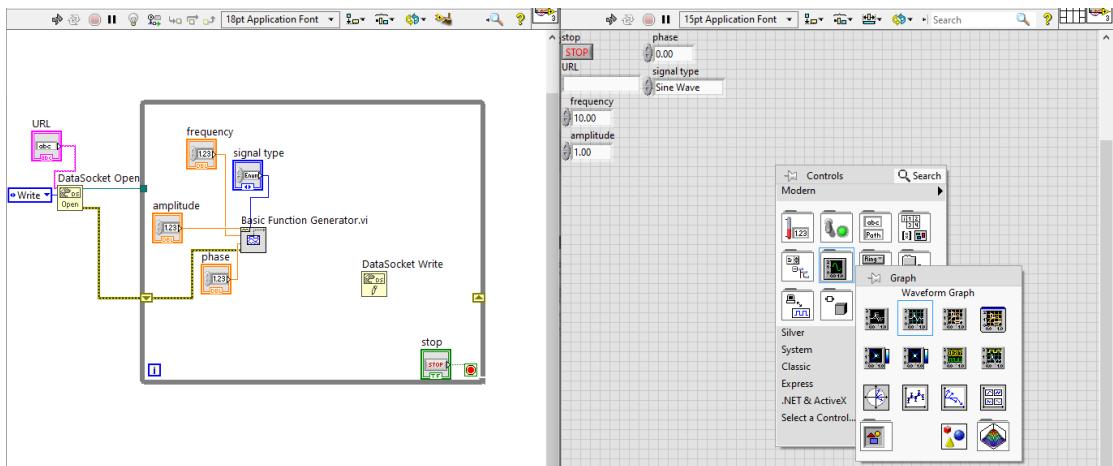




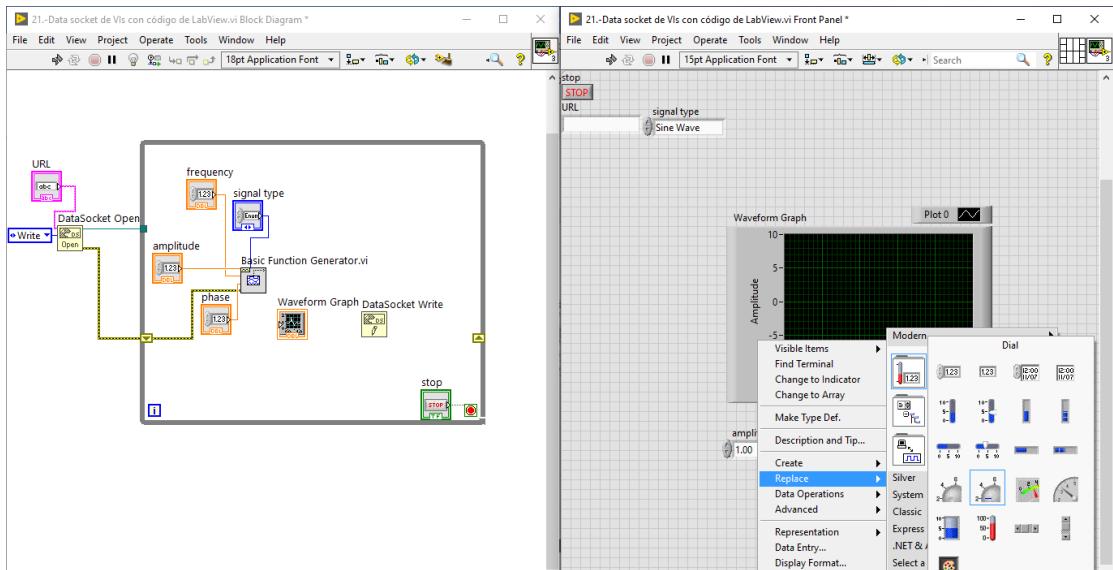


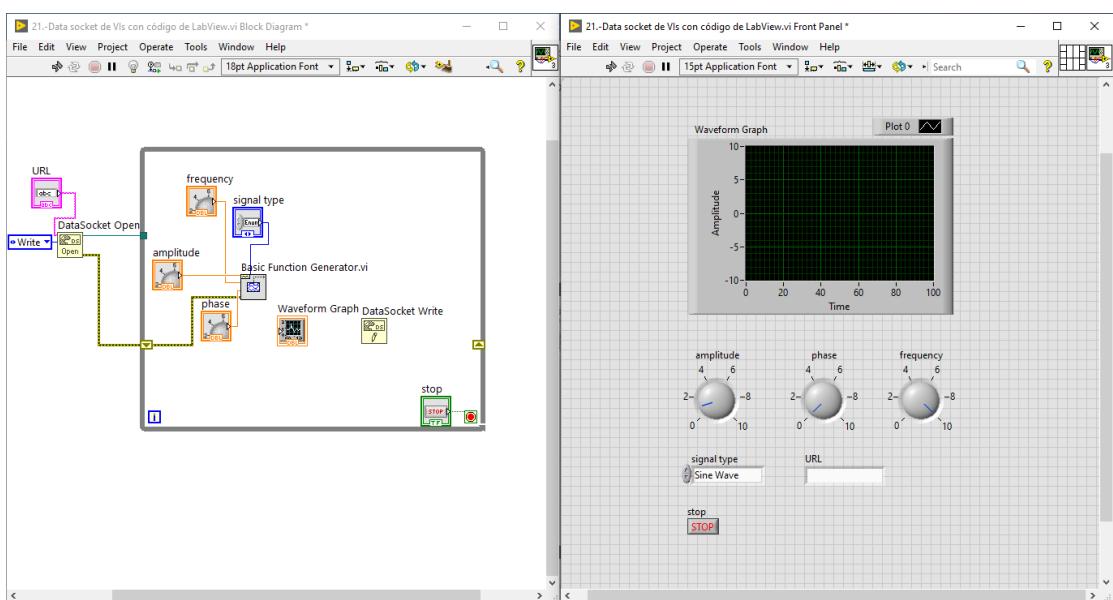
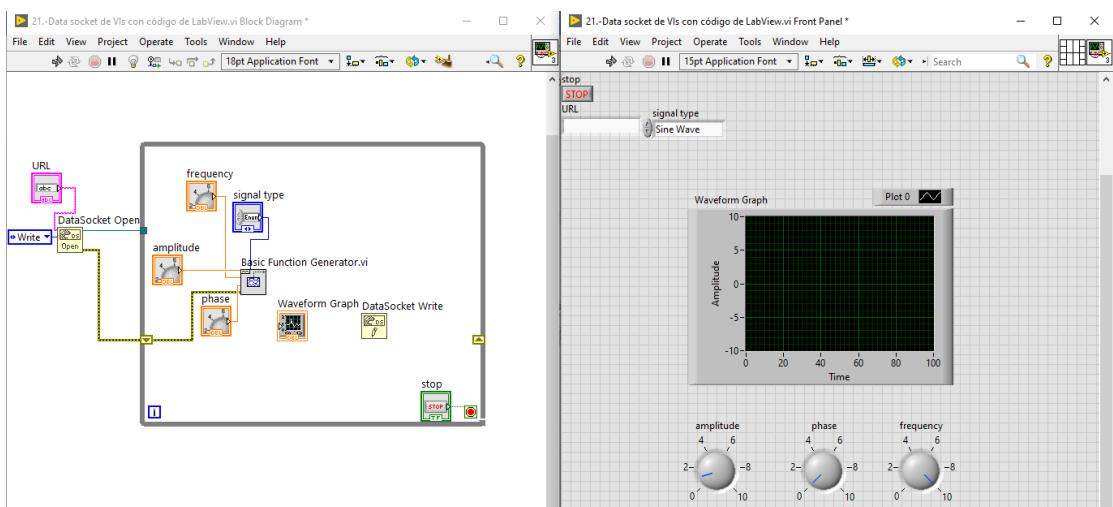
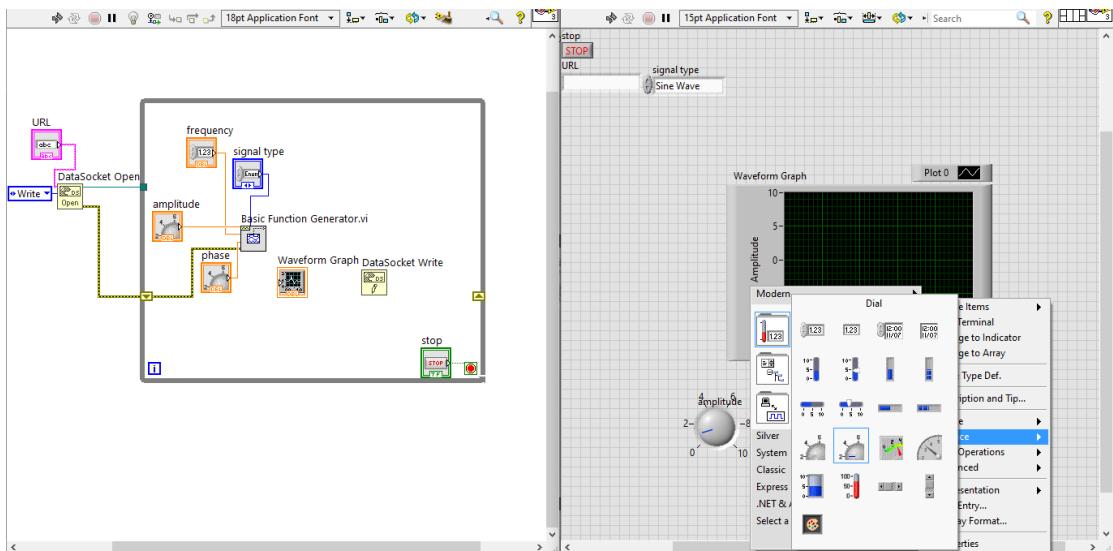
Front Panel - Waveform Graph: Ventana que Muestra una Señal (Array)

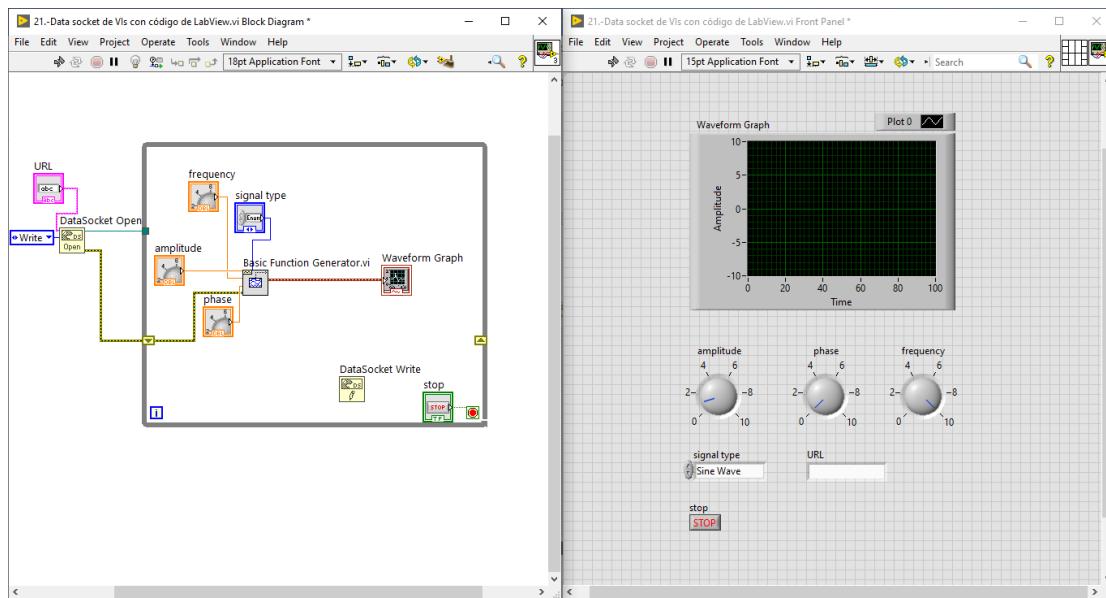
El Waveform Graph muestra gráficas de tipo Array, las cuales son cualquier tipo de grupos de números.



Reemplazar un Control por una Perilla: Clic derecho en el control → Replace → Dial. Podemos cambiar el control por una perilla en el Front Panel.

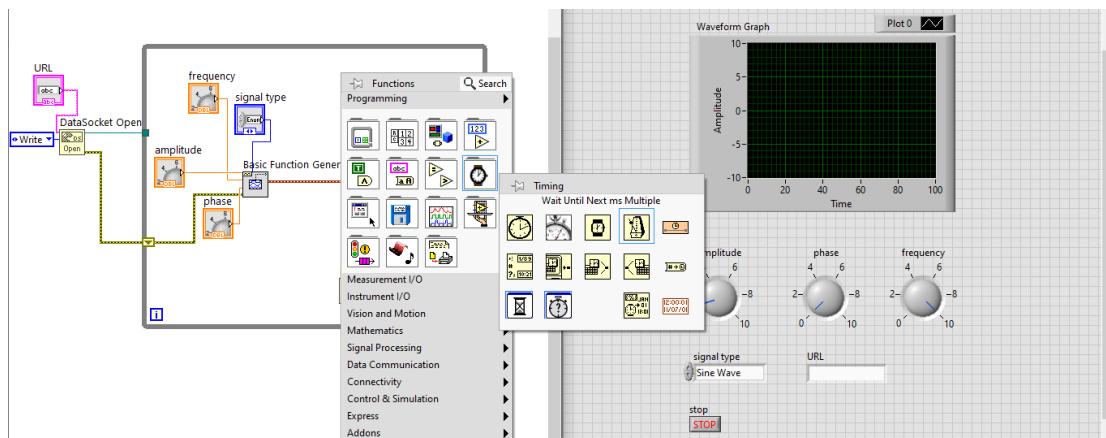




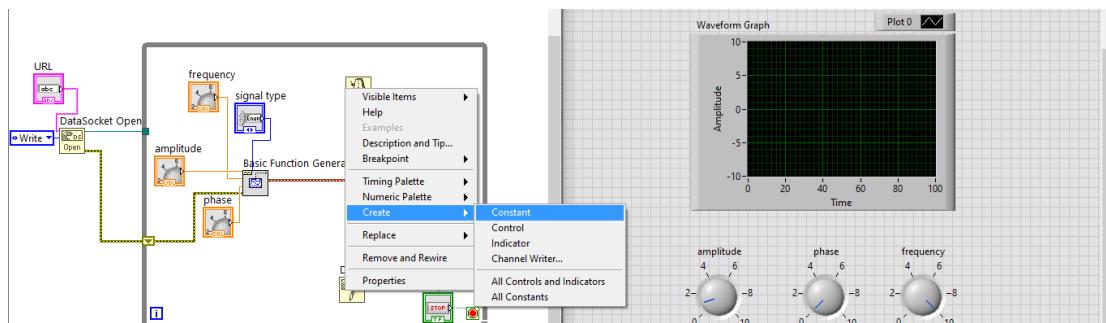


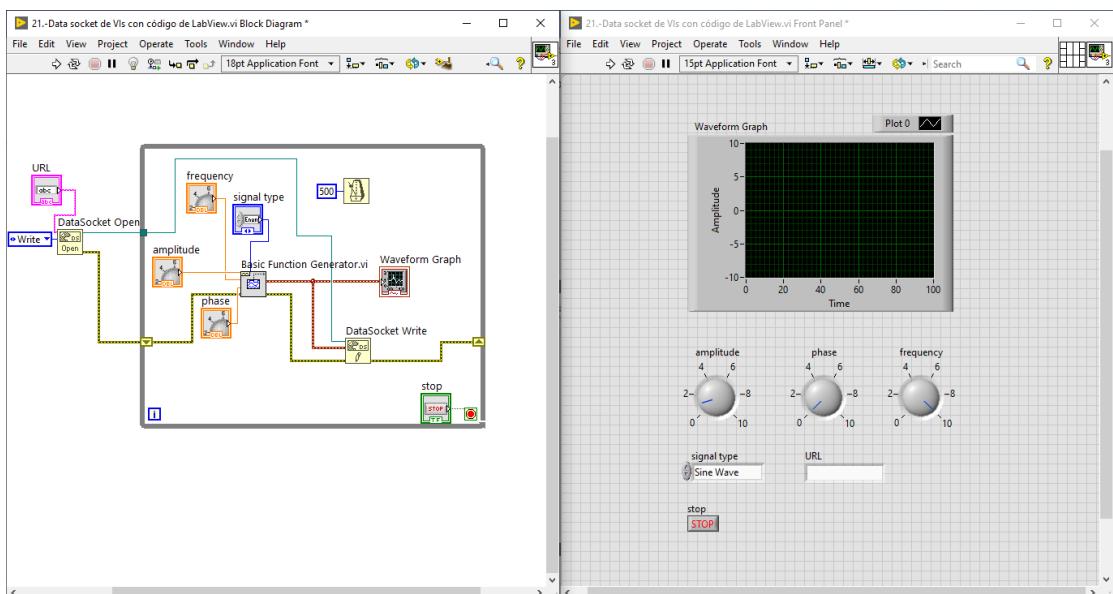
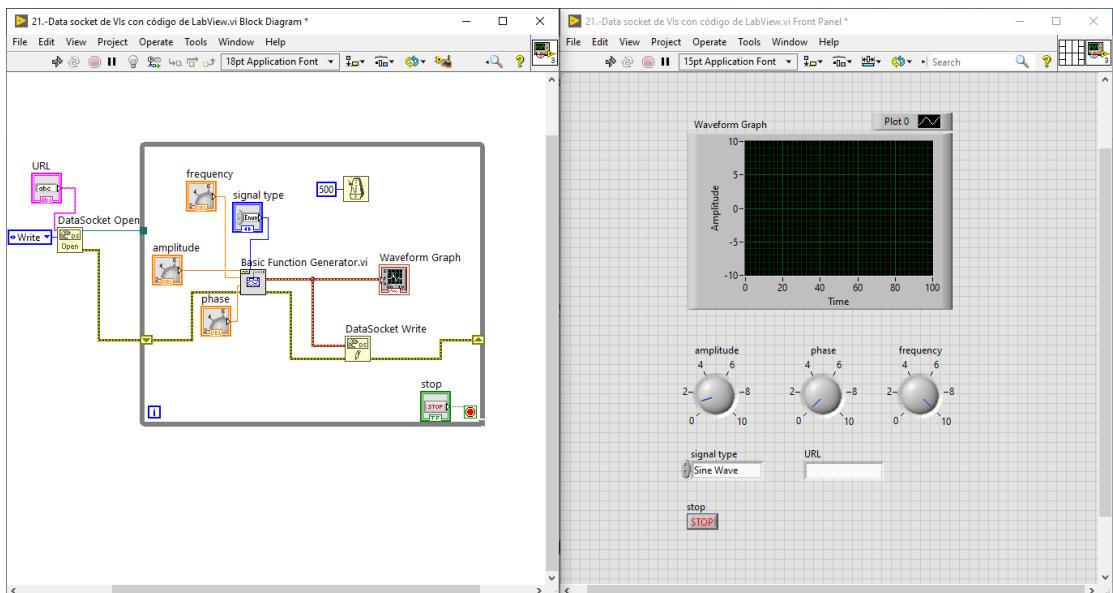
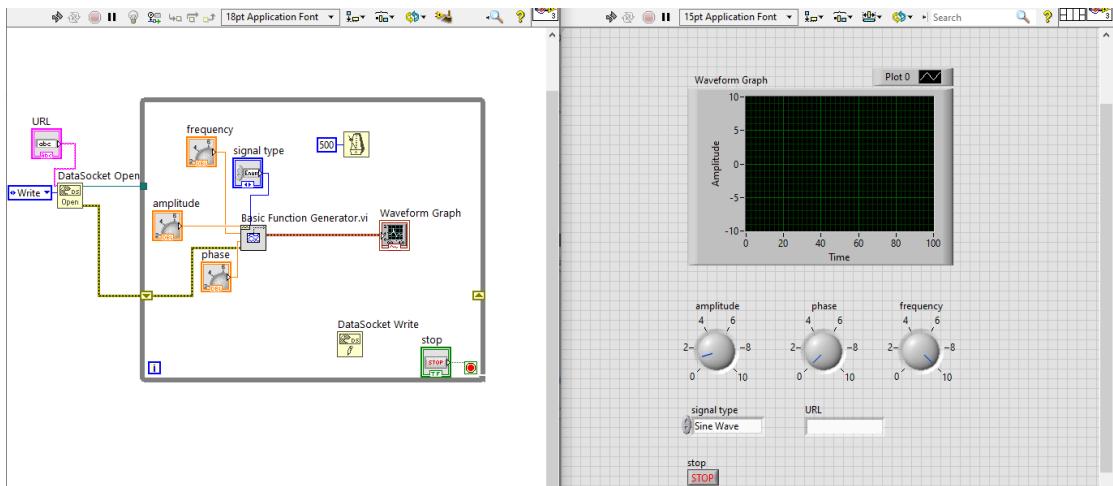
Block Diagram - Wait Until Next ms Multiple: Temporizador en milisegundos

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.



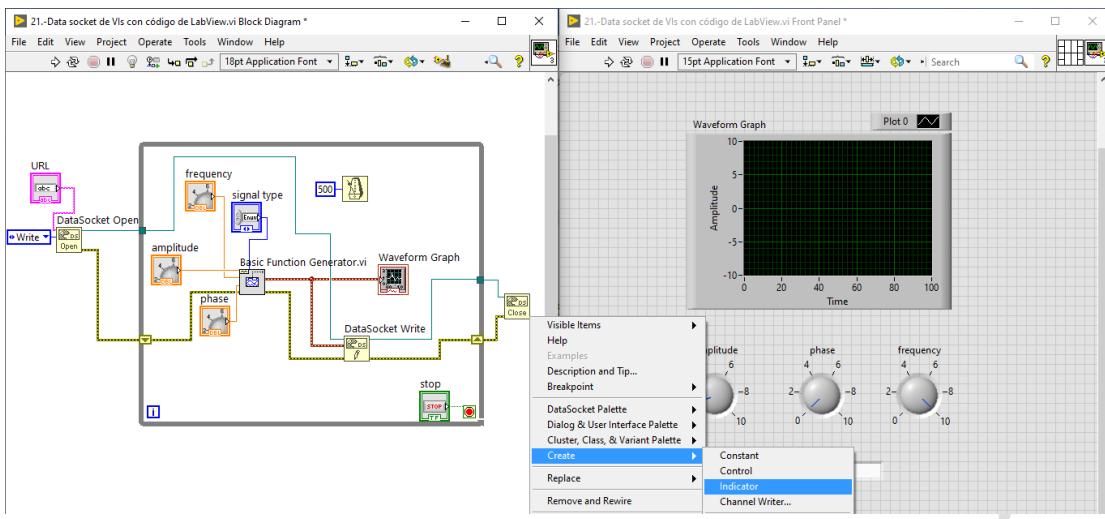
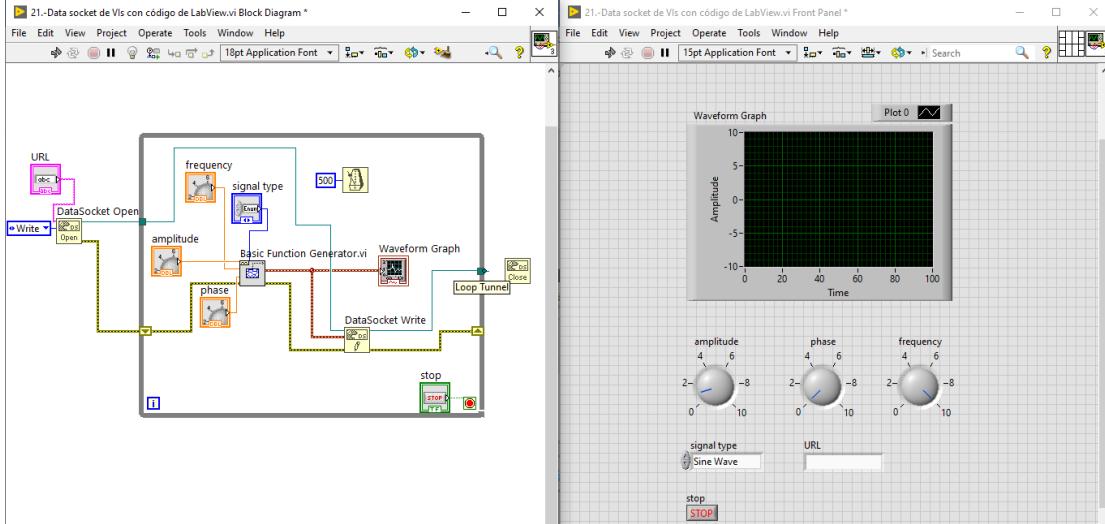
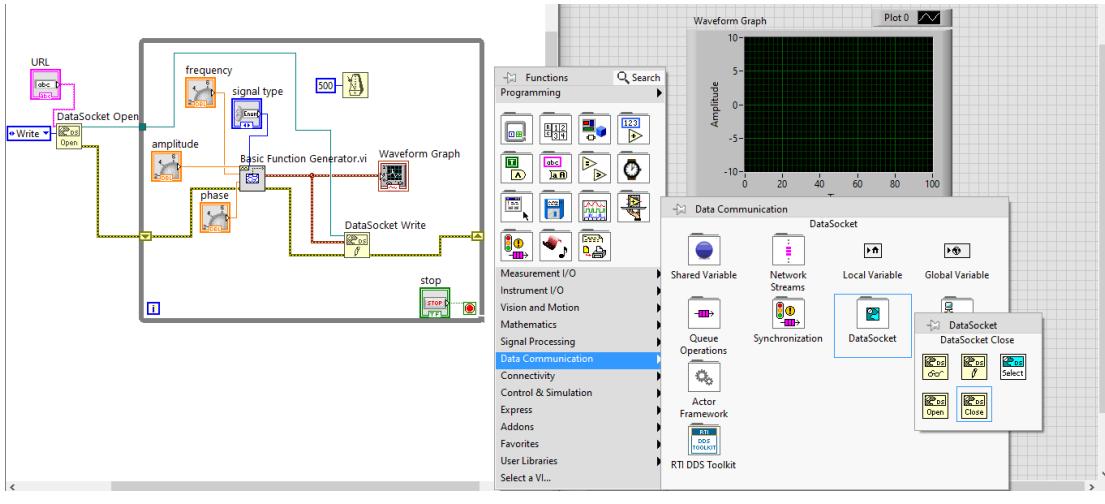
Crear una Constante para un Bloque: Clic derecho en la terminal del bloque de interés → Create → Constant. Esta constante sirve para indicar el tiempo de retraso en milisegundos.



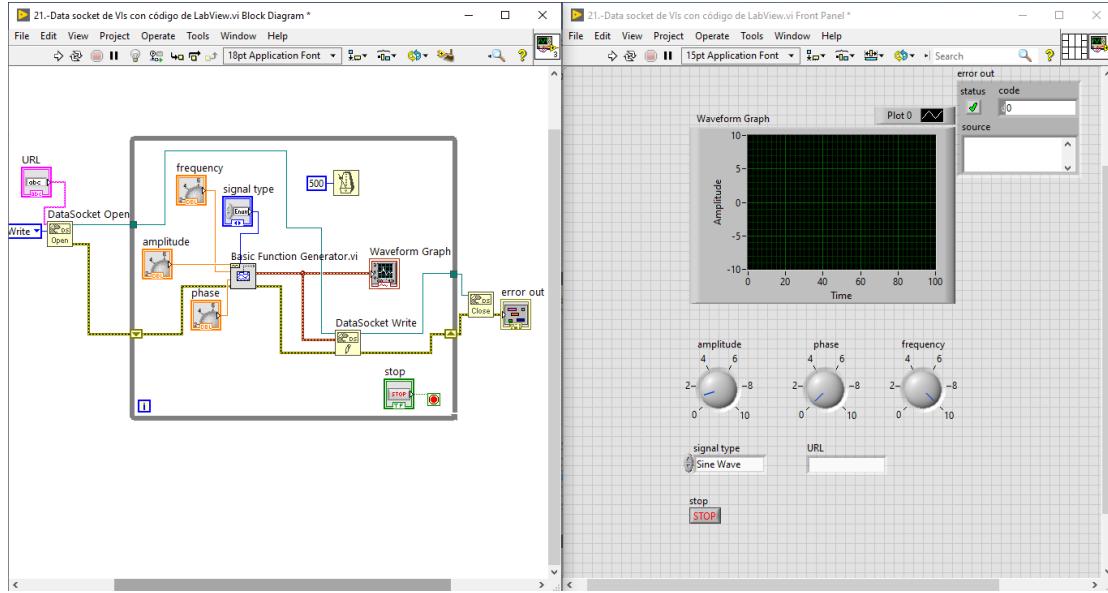


Block Diagram - DataSocket Close: Cerrar Comunicación Inalámbrica por DataSocket

El bloque de DataSocket Close sirve para cerrar una comunicación inalámbrica y remota entre VIs previamente abierta con el bloque de DataSocket Open.

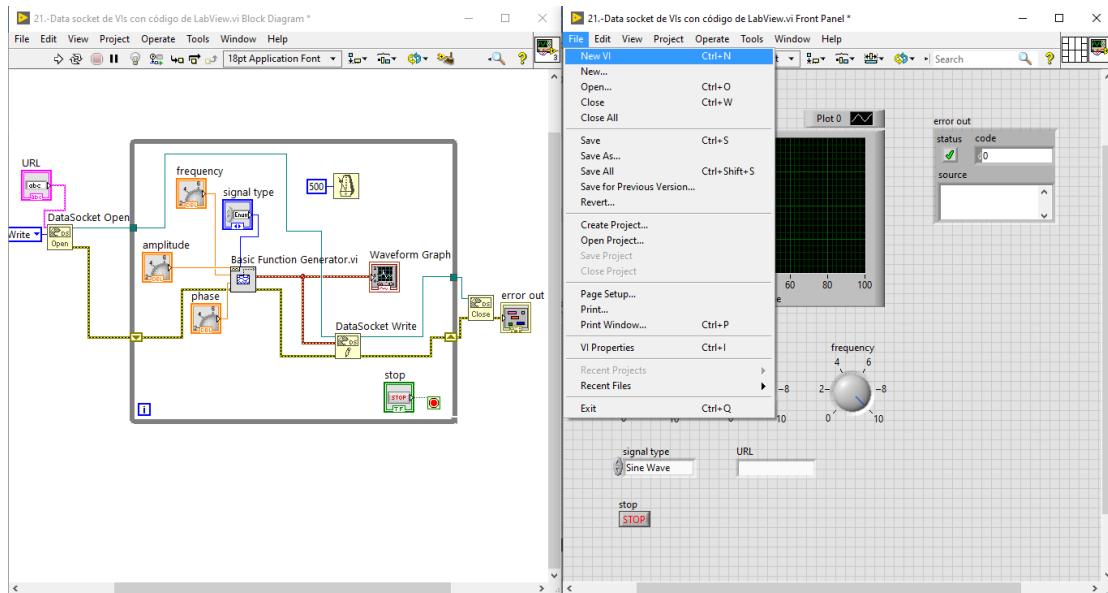


Crear un Indicador para un Bloque: Clic derecho en la terminal del bloque de interés → Create → Indicator. Con esto se crea un elemento de la interfaz donde sea visible el error al ejecutar la conexión por medio del DataSocket.



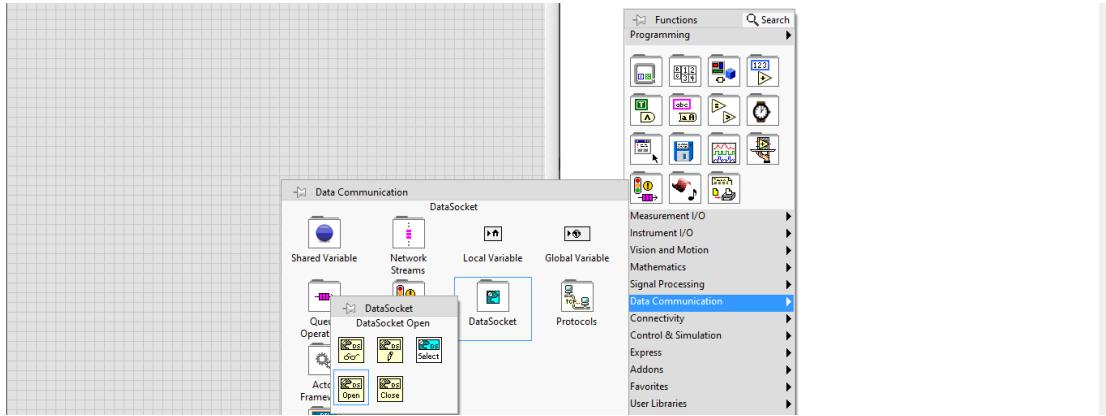
DataSocket Server: VI 2 - SLAVE: Programa que Recibe las Instrucciones de la otra VI

Se crea una VI nueva que servirá como SLAVE, por lo tanto, esta recibirá los datos o instrucciones transmitidas por el VI MASTER, es importante mencionar que cuando se utiliza el protocolo de comunicación de DataSockets en LabVIEW, se puede tener varios SLAVE y un MASTER trabajando a la vez de forma inalámbrica.



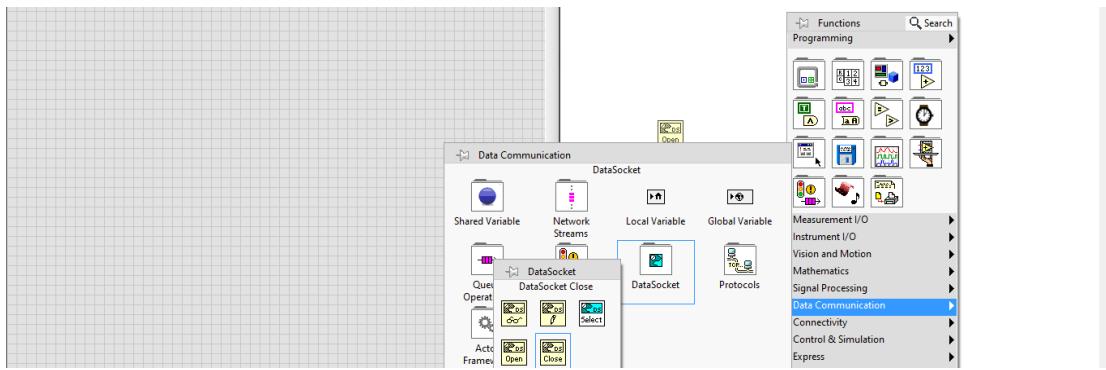
Block Diagram - DataSocket Open: Abrir una Comunicación Inalámbrica por DataSocket

El bloque de DataSocket Open sirve para abrir una comunicación inalámbrica y remota entre VIs.



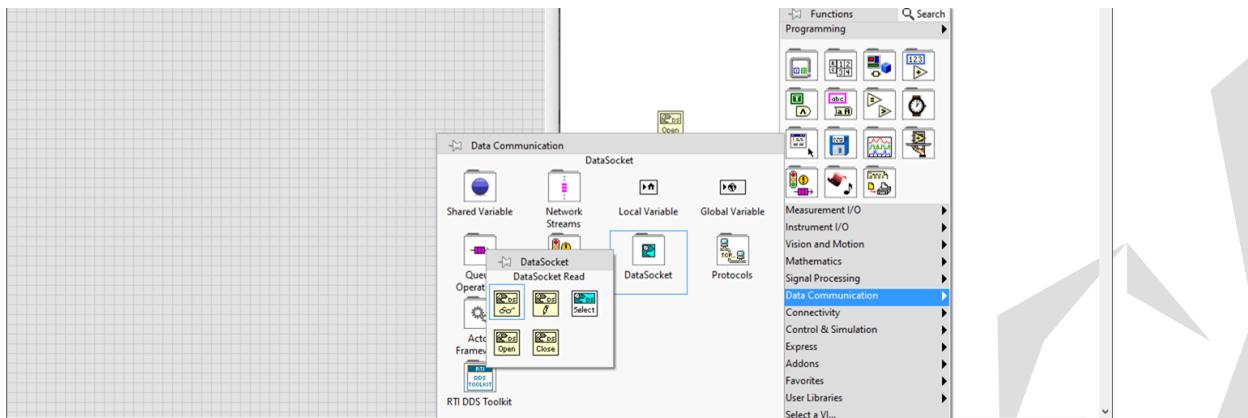
Block Diagram - DataSocket Close: Cerrar Comunicación Inalámbrica por DataSocket

El bloque de DataSocket Close sirve para cerrar una comunicación inalámbrica y remota entre VIs previamente abierta con el bloque de DataSocket Open.



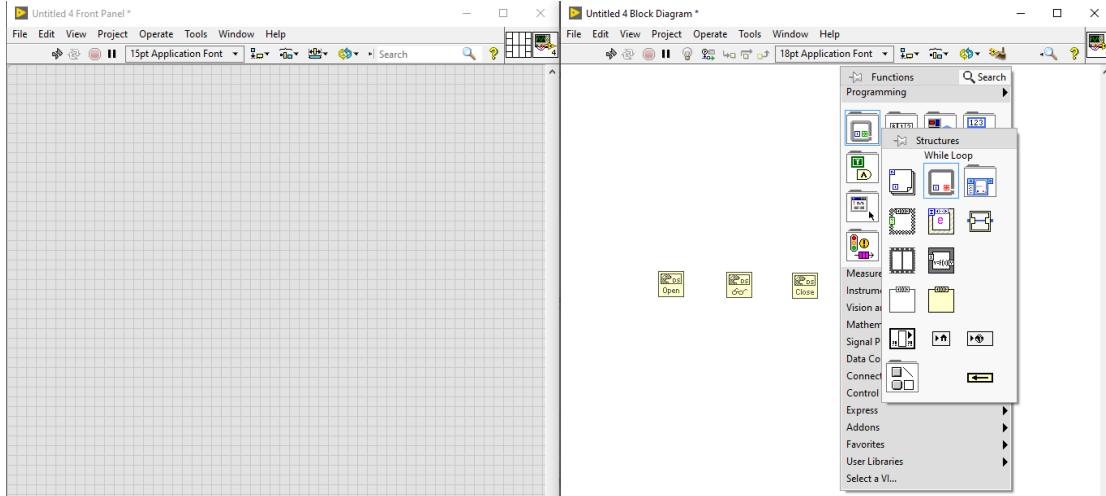
Block Diagram - DataSocket Read: Recibir Instrucciones o Datos por DataSocket

El bloque de DataSocket Write sirve para recibir instrucciones o datos de forma inalámbrica y remota entre VIs, donde uno será el que mande las instrucciones o datos (MASTER) y el otro u otros serán los que los reciben (SLAVE).

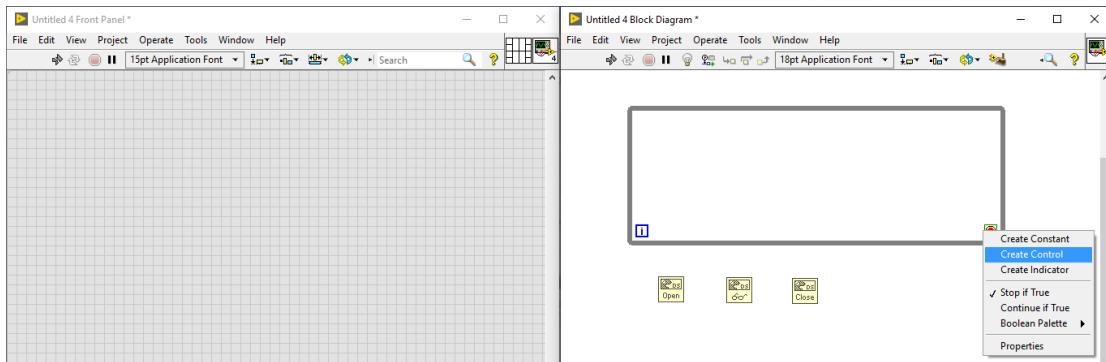


Block Diagram - Bucle While: Ejecución Continua del Programa

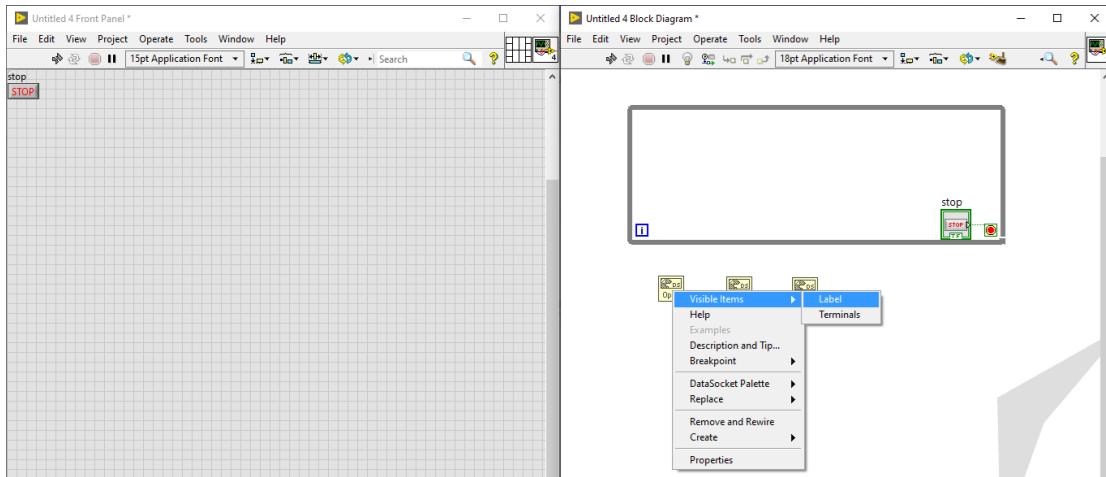
El ciclo while hace que el programa se ejecute hasta que dé clic en el botón de STOP, por eso todo el diagrama de bloques que tengo actualmente lo voy a encerrar en un ciclo while para que esté ejecutando de manera continua.

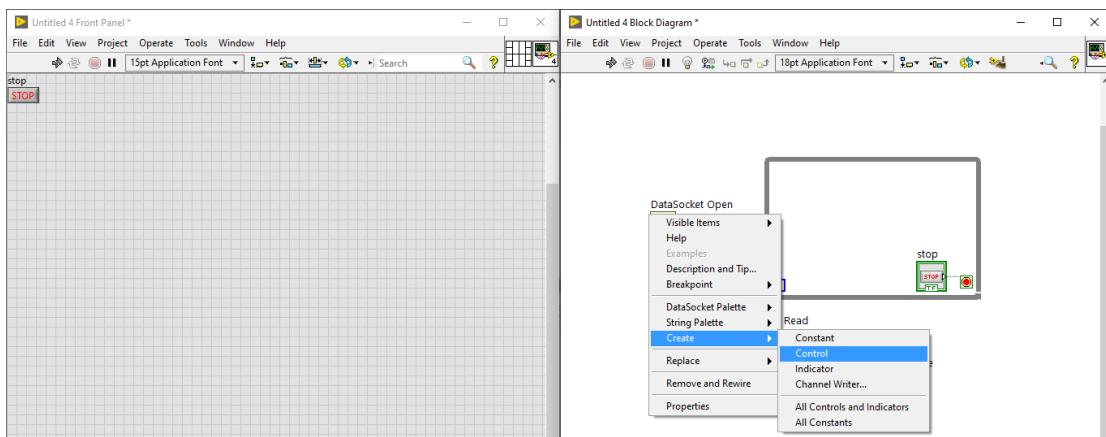


Crear un Control para un Bloque: Clic derecho en el bloque → Create Control.

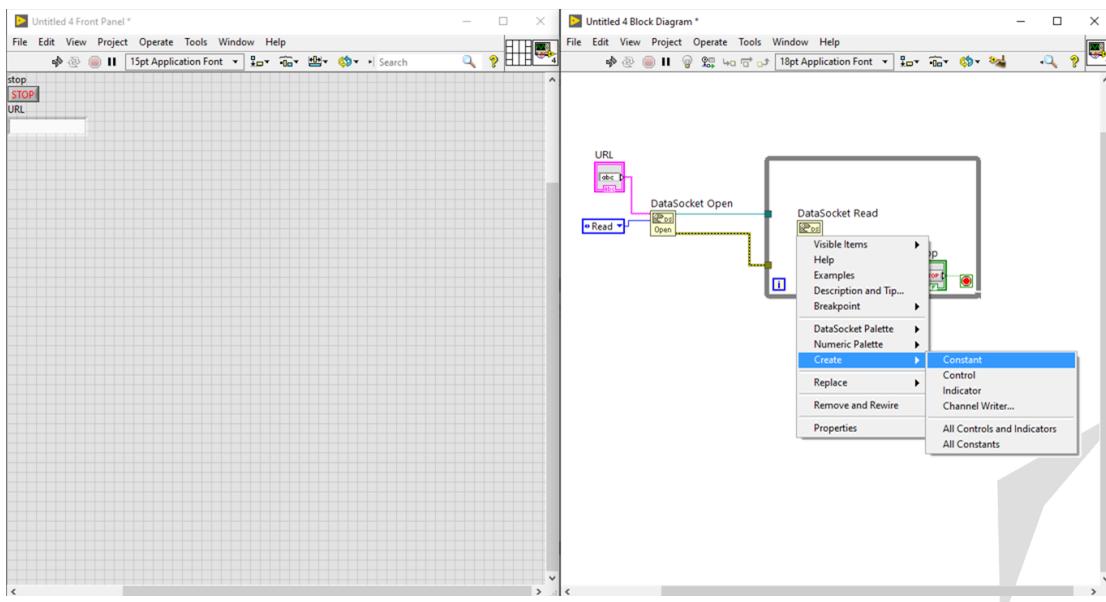
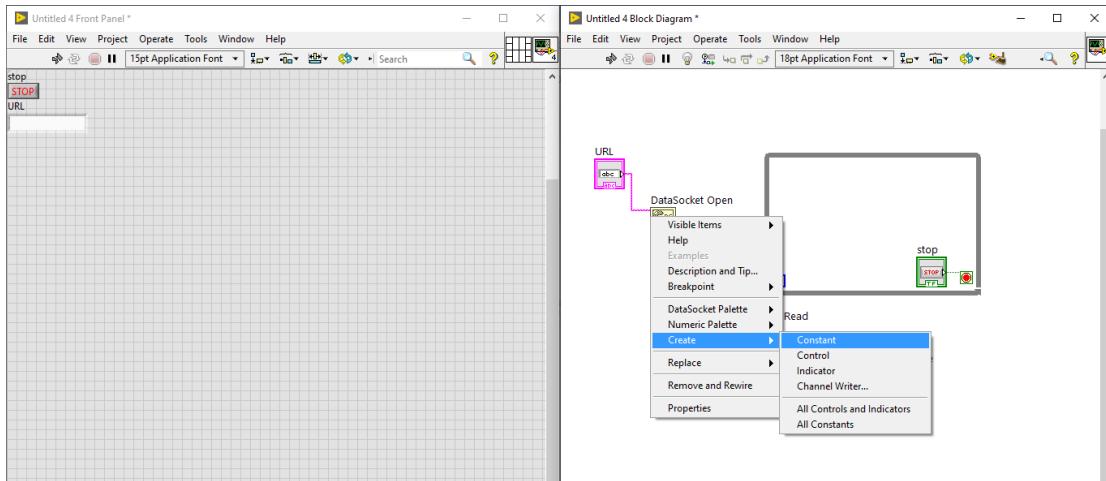


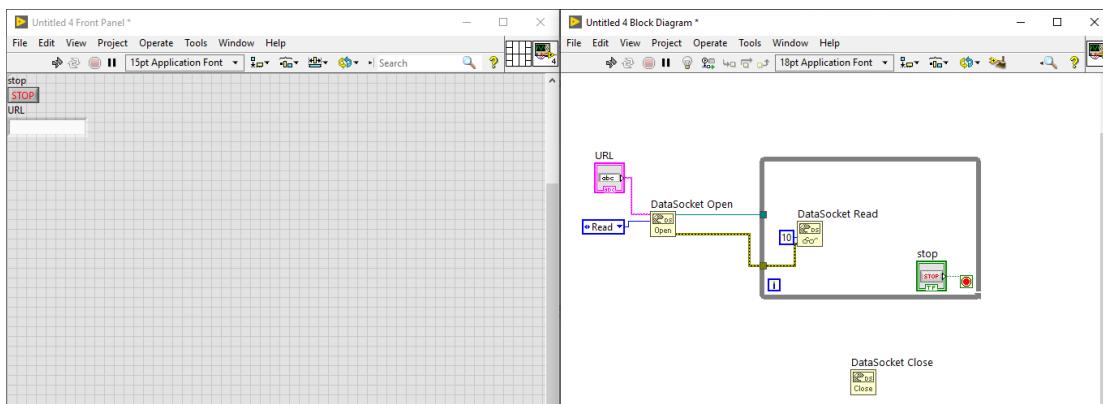
Mostrar nombre del bloque: Clic derecho → Visible Items → Label.





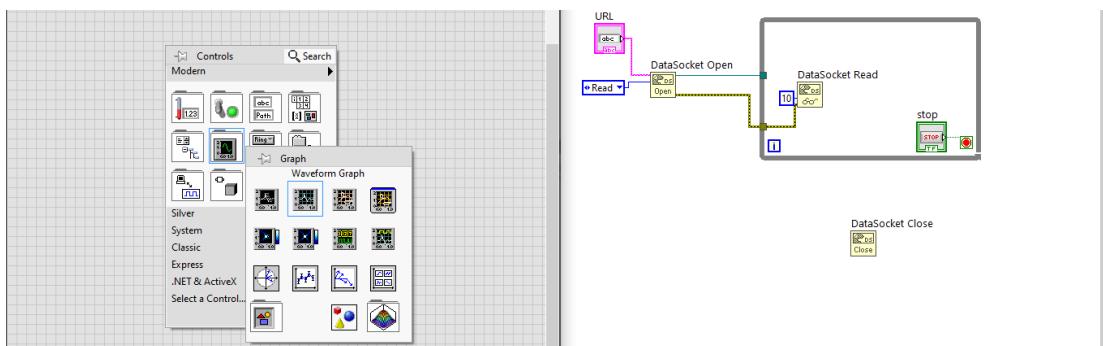
Crear una Constante para un Bloque: Clic derecho en la terminal del bloque de interés → Create → Constant. Esta constante en el Bloque DataSocket Open sirve para elegir si la comunicación se quiere hacer de lectura, escritura, lectura y escritura, etc. Y en el Bloque DataSocket Read para indicar cuantos elementos va a leer.





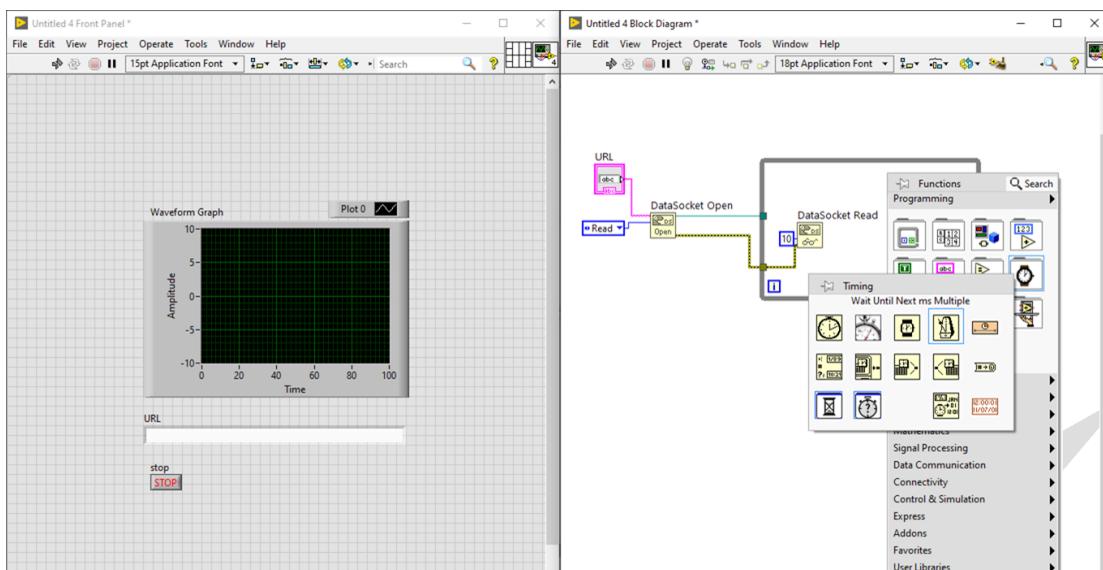
Front Panel - Waveform Graph: Ventana que Muestra una Señal (Array)

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

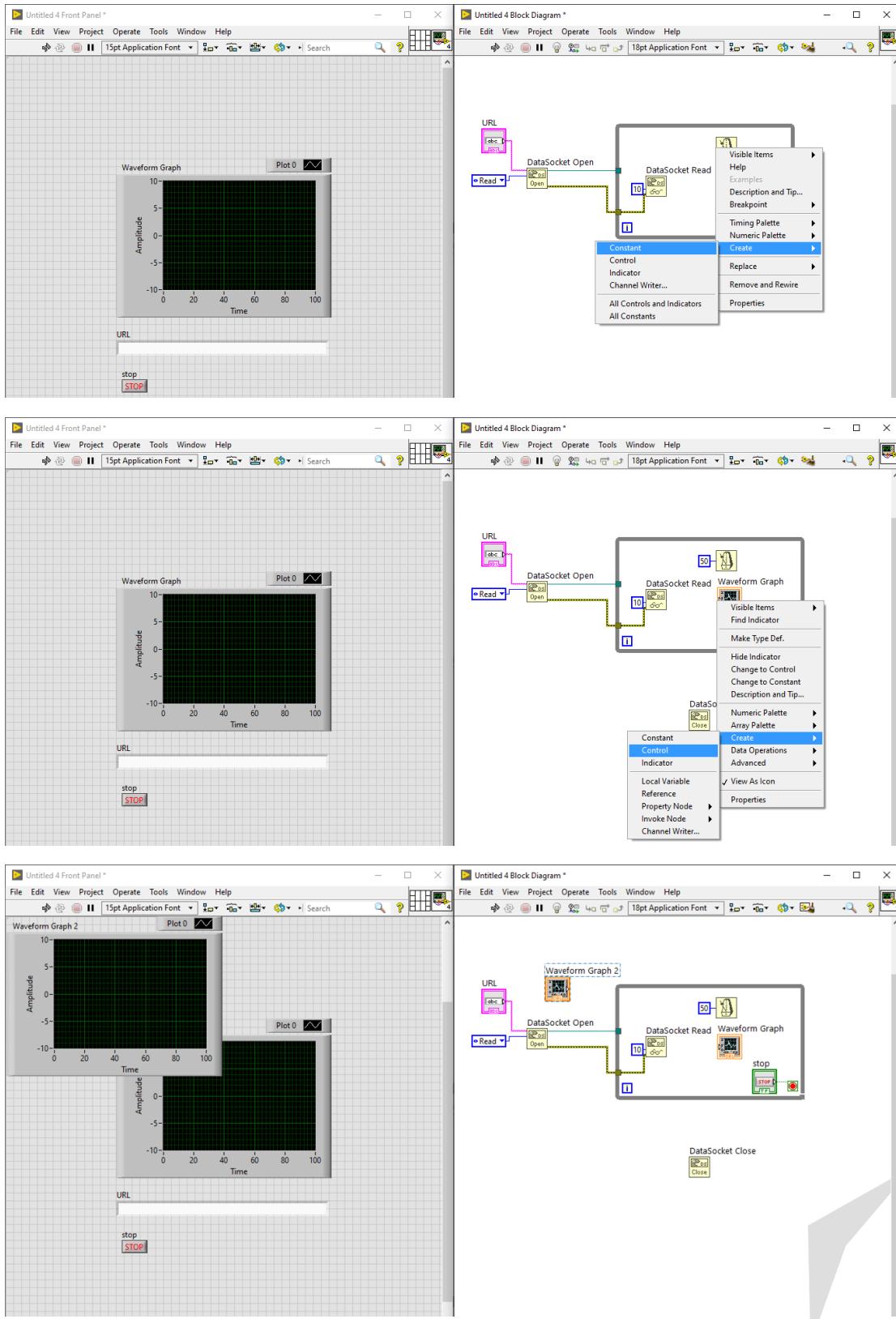


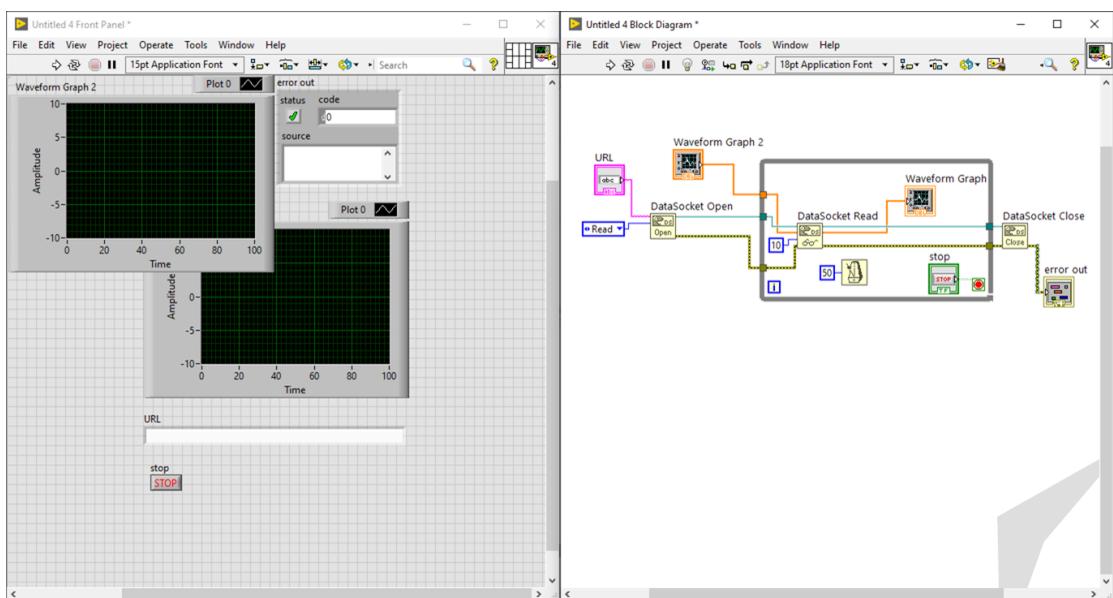
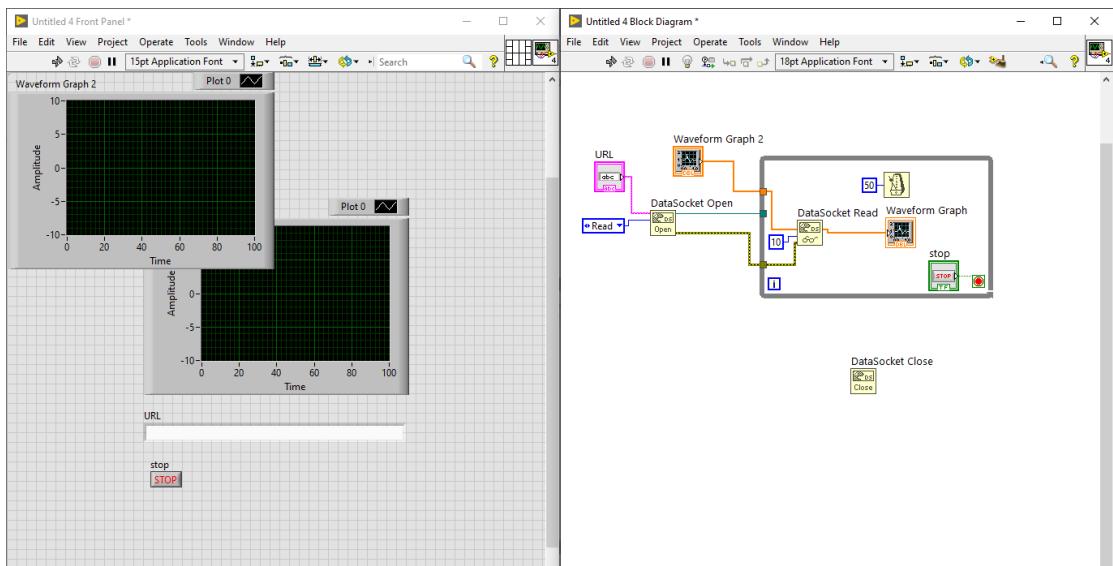
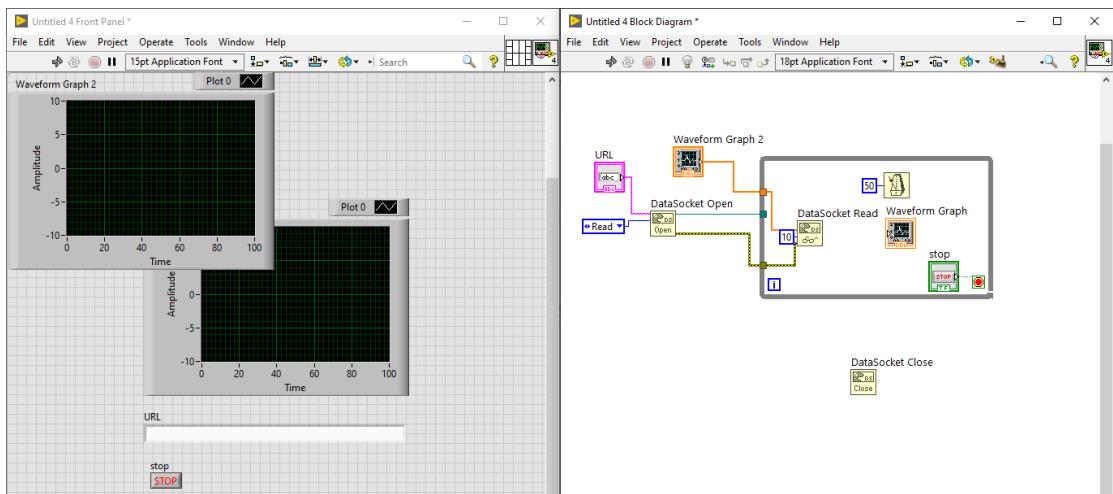
Block Diagram - Wait Until Next ms Multiple: Temporizador en milisegundos

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.



Crear una Constante para un Bloque: Clic derecho en la terminal del bloque de interés → Create → Constant. Esta constante sirve para indicar el tiempo de retraso en milisegundos.

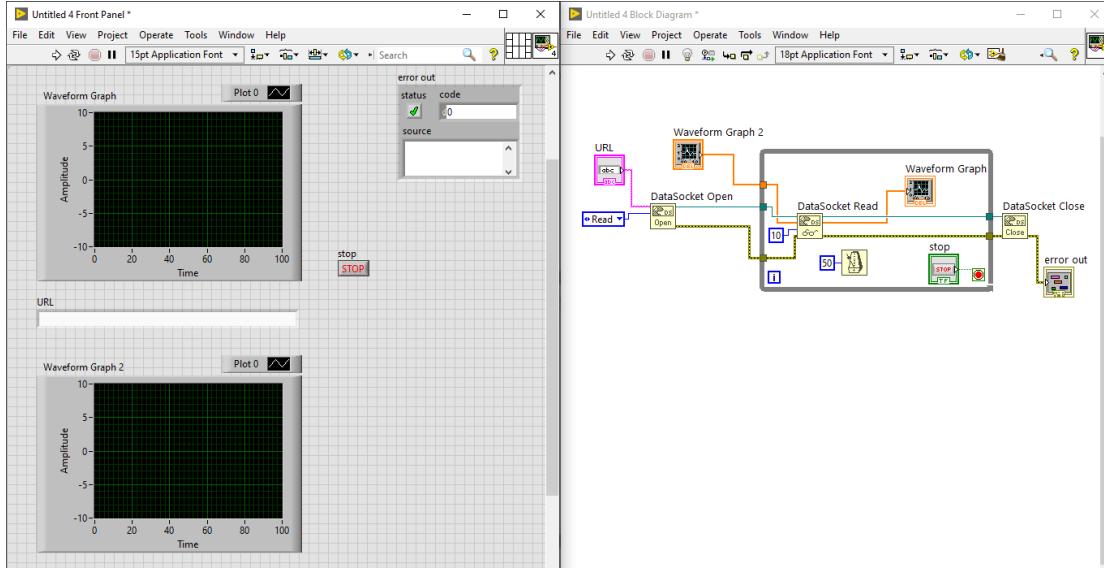




Ejecución del Programa: Transmisión y Recepción de Datos con el Data Socket Activado

DataSocket Server: VI 2 - SLAVE: Programa que Recibe las Instrucciones de la otra VI

La interfaz del VI SLAVE se encuentra a la izquierda.



DataSocket Server: VI 1 - MASTER: Programa que Envía las Instrucciones a la otra VI

La interfaz del VI MASTER se encuentra a la derecha.

Cuando se ejecute el programa se podrá seleccionar la amplitud, fase, frecuencia, tipo de señal del generador de funciones y URL de conexión con el protocolo DataSocket en ambos VIs (MASTER y SLAVE), además de ver en los Waveform Graph los gráficos de Arrays en tiempo real actualizándose.

Es importante mencionar que en las URL se debe colocar la dirección del Data Socket Transfer Protocol:

dstp://ip_home/cualquier_nombre_de_señal_opcional

