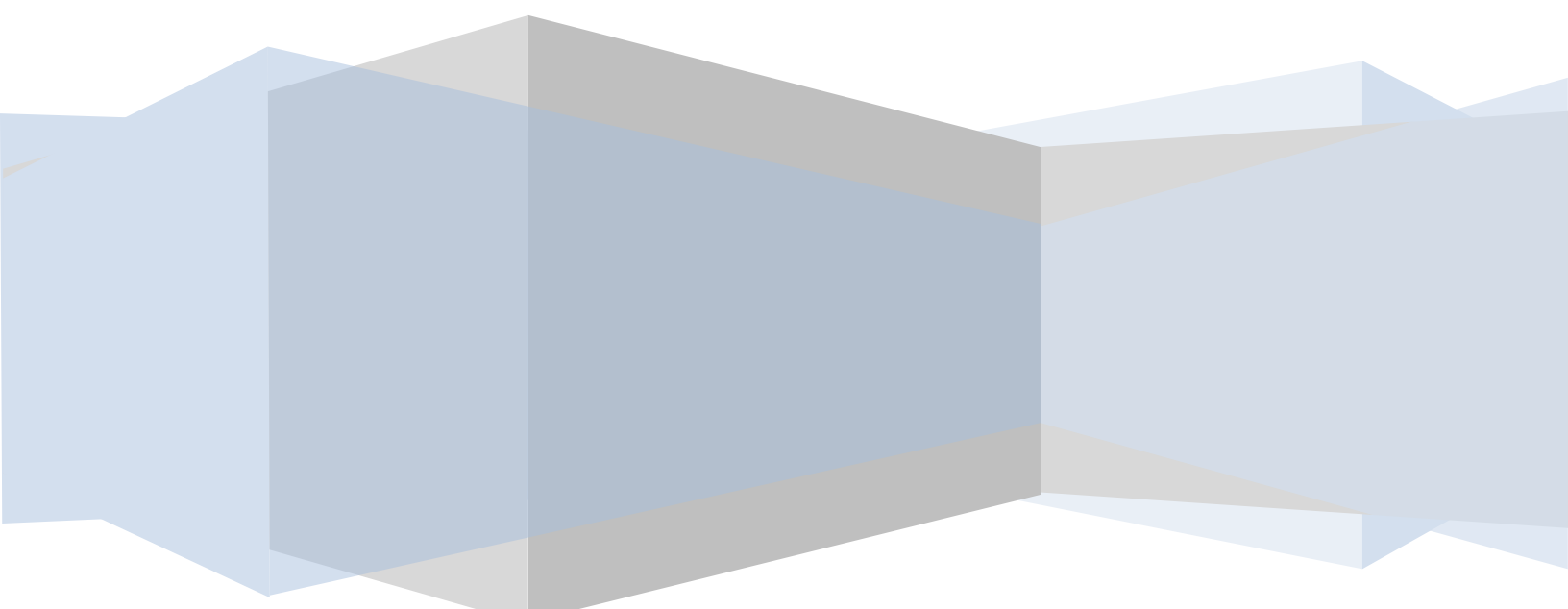




*Universidad de la Frontera
Facultad de Ingeniería, Ciencias y Administración
Departamento de Ingeniería Eléctrica*



GUIA DE INICIACION EN LABVIEW 7.1



**JUAN ALBORNOZ CARRASCO
PABLO LAGOS NORAMBUENA**

2008

CONTENIDO

1	
CONTENIDO	2
1.1 ¿QUÉ ES LABVIEW?	4
1.2 ¿QUIÉNES LO USAN?	5
1.3 SOBRE ESTA GUÍA DE INICIACIÓN EN LABVIEW	6
1.4 ENTORNO DE LABVIEW	6
1.4.1 Panel Frontal y Diagrama de Bloques	7
2.1 PALETAS DE FUNCIONES Y CONTROLES/INDICADORES	11
2.1.1 Paleta de Funciones (Function Palette)	12
2.1.2 Paleta de controles (Controls palette)	14
2.1.3 Controles e Indicadores	16
2.1.4 Terminales	18
2.1.5 Alambres	18
2.1.6 Highlight Execution	19
2.1.7 Tipos de Datos en LabVIEW	20

3.1	ESTRUCTURAS	22
3.1.1	Case Structure	23
3.1.2	Sequence Structure	24
3.1.3	For Loop	25
3.1.4	While Loop	26
3.1.5	Formula Node	27
4.1	TIPOS DE VARIABLES	29
4.1.1	Locales	29
4.1.2	Globales	30
5.1	GRAFICADORES	31
5.1.1	Waveform Charts	32
5.1.2	Waveform Graph	33
5.1.3	XY Graph	33
5.1.4	Intensity Graph e Intensity Chart	33
6.1	SUB VI	33
6.2	BUSCANDO ERRORES DE SINTAXIS	39
6.3	ATAJOS DE TECLADO EN LABVIEW	40

1.1 ¿QUÉ ES LABVIEW?.

LabVIEW es el acrónimo de Laboratory Virtual Instrument Engineering Workbench. Es un lenguaje y a la vez un entorno de programación gráfica en el que se pueden crear aplicaciones de una forma rápida y sencilla.

National Instruments es la empresa desarrolladora y propietaria de LabVIEW, comenzó en 1976 en Austin, Texas y sus primeros productos eran dispositivos para el bus de instrumentación GPIB. En abril de 1983 comenzó el desarrollo de lo que sería su producto estrella: LabVIEW, que vería la luz en octubre de 1986 con el lanzamiento de LabVIEW 1.0 para Macintosh (los ordenadores más populares en aquella época que ya disponían de interfaz gráfica) y en 1990 la versión 2.0. Para Windows habría que esperar hasta septiembre de 1992.

LabVIEW es un revolucionario ambiente de desarrollo gráfico con funciones integradas para realizar adquisición de datos, control de instrumentos, análisis de mediciones y presentaciones de datos. LabVIEW da la flexibilidad de un poderoso ambiente de programación sin la complejidad de los ambientes tradicionales.

A diferencia de los lenguajes de propósito general, LabVIEW provee funcionalidad específica para que pueda acelerar el desarrollo de aplicaciones de medición, control y automatización.

LabVIEW le entrega herramientas poderosas para crear aplicaciones sin líneas de texto de código. Con LabVIEW usted coloca objetos ya contruidos para rápidamente crear interfases de usuario. Después usted especifica la funcionalidad del sistema armando diagramas de bloques.

Además usted puede conectar con otras aplicaciones y compartir datos a través de ActiveX, la Web, DLLs, librerías compartidas, SQL, TCP/IP, XML, OPC y otros.

Con LabVIEW puede desarrollar sistemas que cumplan con sus requerimientos de desempeño a través de las plataformas incluyendo Windows, Macintosh, UNIX o sistemas de tiempo real.

Además LabVIEW trabaja con más de 1000 librerías de instrumentos de cientos de fabricantes, y muchos fabricantes de dispositivos de medida incluyen también herramientas de LabVIEW con sus productos.

Utilizando un sistema basado en LabVIEW, tiene acceso a sistemas de instrumentación completos con costos mucho más bajos que un único instrumento comercial. National Instruments también asegura que los programas que desarrolla hoy pueden migrar para aprovechar las tecnologías del futuro.

LabVIEW tiene extensas capacidades de adquisición, análisis y presentación disponibles en un sólo paquete, de tal forma que se puede crear una solución completa de manera única en la plataforma que ha elegido. Con LabVIEW puede publicar sus aplicaciones de datos en la Web muy fácilmente o conectarse a otras aplicaciones a través de una variedad de tecnologías estándar, como TCP/IP, DLLs y ActiveX.

1.2 ¿QUIÉNES LO USAN?.

Ingenieros, científicos y técnicos de todo el mundo utilizan LabVIEW para desarrollar soluciones que respondan a sus exigentes aplicaciones. LabVIEW es un revolucionario entorno gráfico de desarrollo para adquisición de datos, control de instrumentos, análisis de medidas y

presentación de datos. LabVIEW le da la flexibilidad de un potente lenguaje de programación sin la complejidad típica asociada a estos.

1.3 SOBRE ESTA GUÍA DE INICIACIÓN EN LABVIEW.

Este manual ha sido elaborado con el fin que los estudiantes tengan información necesaria para aprender a programar bajo el lenguaje de programación LabVIEW. Es importante aclarar que el manual es de carácter básico, esta hecho para introducir al estudiante sobre la programación de este lenguaje. El manual esta basado en la versión 7.1.

1.4 ENTORNO DE LABVIEW.

LabVIEW es una herramienta de programación gráfica. Originalmente este programa estaba orientado a aplicaciones de control de instrumentos electrónicos usadas en el desarrollo de sistemas de instrumentación, lo que se conoce como instrumentación virtual.

Por este motivo los programas creados en LabVIEW se guardarán en ficheros llamados VI y con la misma extensión, que significa instrumento virtual (Virtual Instruments). También relacionado con este concepto se da nombre a sus dos ventanas principales: un instrumento real tendrá un Panel Frontal donde estarán sus botones, pantallas, etc. y una circuitería interna. En LabVIEW estas partes reciben el nombre de Panel Frontal y Diagrama de Bloques respectivamente.

- ❖ Panel Frontal, es la parte que verá el usuario, suele tener fondo gris.
- ❖ Diagrama de Bloques, es donde se realizará la programación y suele tener fondo blanco.

1.4.1 Panel Frontal y Diagrama de Bloques.

El Panel Frontal y el Diagrama de Bloques están conectados a través de los terminales (elementos que sirven como entradas o salidas de datos). De la misma forma que un indicador luminoso de la carátula de un instrumento está representado como un diodo en la circuitería interna, en un programa en LabVIEW ese mismo indicador luminoso estará representado en el Diagrama de Bloques como una salida de tipo booleano sobre el que escribirá un valor.

A continuación se presentan las pantallas típicas y además describe la utilización de los botones que están en la parte superior tanto del Panel Frontal como del Diagrama de Bloques.

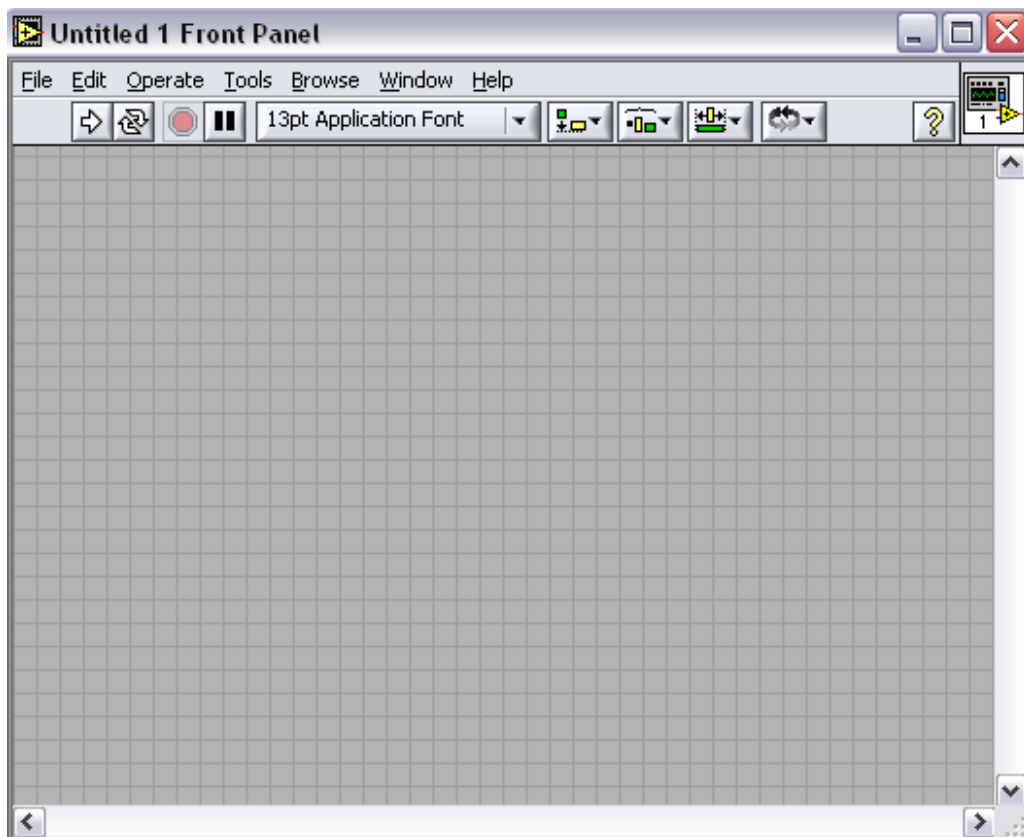


Figura 1 Presentación Panel Frontal.

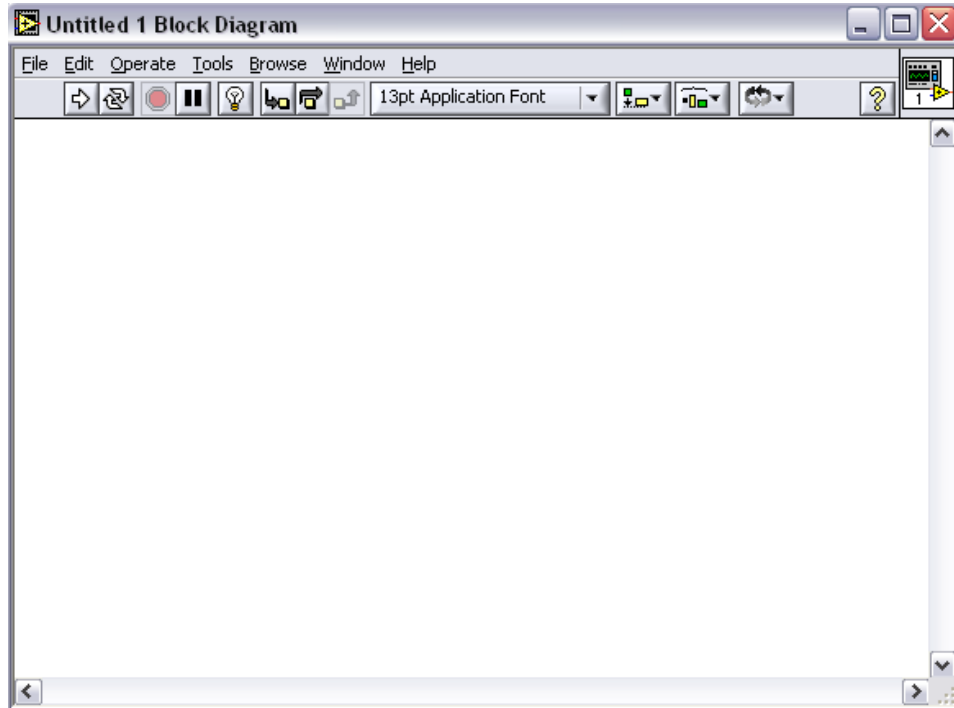


Figura 2 Presentación Diagrama de Bloques.

En la parte superior de estas ventanas se sitúa una barra con varias herramientas. En el Diagrama de Bloques esta barra tiene algunas opciones más.



Figura 3 Paleta principal del Diagrama de Bloques.



El primer grupo de herramientas sirve para controlar la ejecución de un programa en LabVIEW. El primer botón indica si hay errores en el programa (flecha rota), y cuando no los hay (flecha completa), ejecuta una vez el programa. El segundo botón ejecuta de forma continua el programa, como regla general este botón no debe usarse, en su lugar se empleará un bucle en el programa. El tercer botón aborta la ejecución y el cuarto permite realizar una pausa.



El segundo grupo de botones sirve para ayudar a su depuración. El primer botón es *Highlight Execution*, una de las herramientas más útiles para depurar, ralentiza la ejecución permitiendo ver el camino que siguen los datos en el programa. Los tres siguientes se utilizan para ejecutar el programa paso a paso.



El menú desplegable permite variar tamaños, colores y estilos de textos, es recomendable usar los formatos predefinidos como *Application Font* o *System Font*. El siguiente grupo se usa para alinear, distribuir, controlar el tamaño, agrupar y ordenar objetos.



En el lateral derecho tanto del Panel Frontal como del Diagrama de Bloques aparece el icono que representa al VI.



También existe una pequeña, pero muy necesaria paleta llamada Tools Palette. La paleta de herramientas permite crear, modificar y depurar Vis utilizando sus diferentes opciones. Si la paleta de herramientas no aparece puede activarla seleccionando en cualquiera de los dos paneles en: Windows → Show Tools Palette.

A continuación se detallan las utilidades de cada uno de los iconos que aparecen dentro de la paleta de herramienta. (Tools Pallets)



Herramienta de Selección Automática. Al habilitarse la selección automática de herramienta, cuando se mueve el cursor sobre los diferentes objetos en el panel frontal o diagrama de bloques, LabVIEW selecciona automáticamente la herramienta correspondiente de la paleta.

Cada icono de la paleta cambia el comportamiento del cursor en LabVIEW, con lo que se puede posicionar, operar y editar las diferentes tareas de los VIs.



Herramienta de operación. Permite cambiar los valores de los elementos en el panel frontal y permite modificar textos existentes.



Herramienta de posicionamiento. Permite seleccionar, mover o redimensionar objetos.



Herramienta de etiquetado. Permite modificar etiquetas, nombres de variables y modificar y cambiar las propiedades de los textos.



Herramienta de cableado. Permite realizar las conexiones entre diferentes bloques en el diagrama.



Herramienta de menú. Permite desplegar un menú con diferentes opciones en cada uno de los objetos, esta acción se realiza también al presionar click derecho sobre un elemento.



Herramienta de deslizamiento. Permite deslizarse a través de una ventana sin utilizar las barras de desplazamiento.



Herramienta de puntos de detención. Permite definir “puntos de parada” en una aplicación, de tal manera que la aplicación termine cuando se llegue allí.



Herramienta de pruebas. Permite colocar puntos de prueba en una aplicación. Los puntos de prueba permiten ver información de los valores calculados.



Herramienta de copiado de color. Copia colores que aparecen en la ventana activa para ser usados en otros sitios.



Herramienta de color. Permite y cambiar el color de los objetos.

2.1 PALETAS DE FUNCIONES Y CONTROLES/INDICADORES.

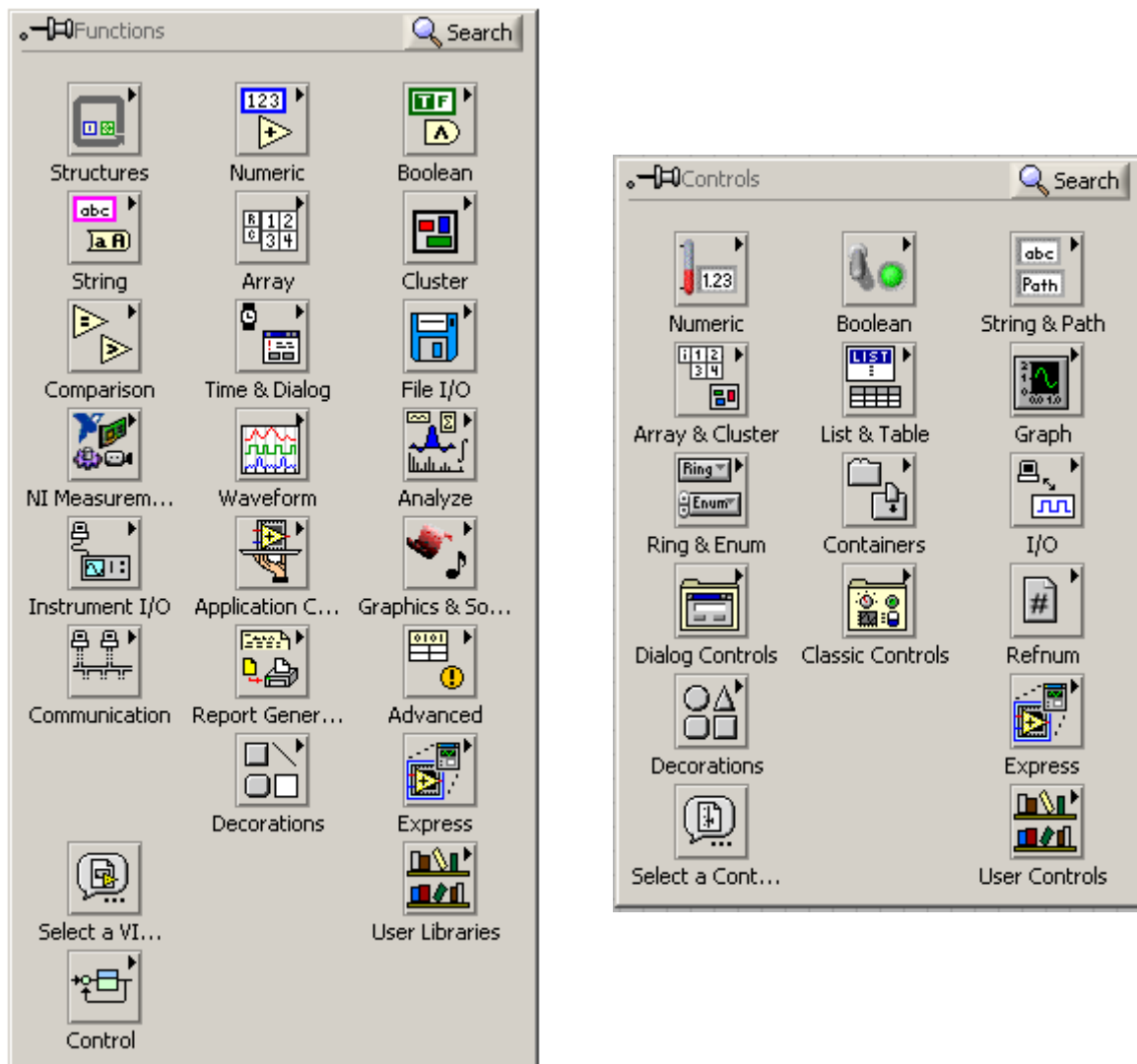
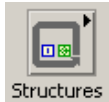


Figura 4 A la izquierda la paleta de Funciones, y ala derecha la paleta de Controles.

2.1.1 Paleta de Funciones (Function Palette).

La *paleta de funciones* contiene todos los objetos que se emplean en la implementación del programa del VI, ya sean *funciones* aritméticas, de entrada/salida de señales, entrada/salida de datos a fichero, adquisición de señales, temporización de la ejecución del programa, etc.



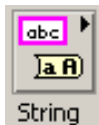
Structures, muestra las *estructuras* de control del programa, junto con las variables locales y globales.



Numeric, muestra *funciones* aritméticas y constantes numéricas.



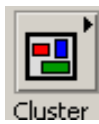
Boolean, muestra *funciones* y constantes lógicas.



String, muestra *funciones* para manipular cadenas de caracteres, así como constantes de caracteres.



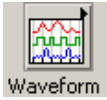
Array, contiene *funciones* útiles para procesar datos en forma de vectores, así como constantes de vectores.



Cluster, contiene *funciones* útiles para procesar datos procedentes de gráficas y destinados a ser representados en ellas, así como las correspondientes constantes.



Comparison, muestra *funciones* que sirven para comparar números, valores booleanos o cadenas de caracteres.



Waveform

Waveform, contiene *funciones* que permiten construir formas de ondas, incluyendo sus valores, canales. Extrae y edita información de una waveform.



Time & Dialog

Time & Dialog, contiene *funciones* para trabajar con cuadros de diálogo, introducir contadores y retardos, etc.



File I/O

File I/O, muestra *funciones* para operar con ficheros.



Instrument I/O

Instrument I/O, muestra un submenú de *VI*s, que facilita la comunicación con instrumentos periféricos que siguen la norma ANSI/IEEE 488.2-1987, y el control del puerto serie.



Analyze

Analyze, contiene un submenú en el que se puede elegir entre una amplia gama de *funciones* matemáticas de análisis.



Application C...

Application control, contiene varias *funciones* que regulan el funcionamiento de la propia aplicación en ejecución.



Communication

Communication, muestra diversas *funciones* que sirven para comunicar varios ordenadores entre sí, o para permitir la comunicación entre distintos programas.



NI Measurem...

NI Measurement, contiene *funciones* que permiten trabajar con tarjetas u otros dispositivos adquirentes de datos.



Report Generation, contiene variadas funciones para crear historiales de datos.



Advanced, contiene diversos submenús que permiten el control de la ayuda de los VIs, manipulación de datos, procesado de eventos, control de la memoria, empleo de programas ejecutables o incluidos en librerías DLL, etc.

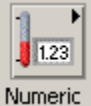











User Libraries, muestra las librerías definidas por el usuario, se pueden cargar en *C:\Archivos de programa\National Instruments\LabVIEW 7.1\user.lib*




2.1.2 Paleta de controles (Controls palette).

Se utiliza únicamente en el *panel frontal*. Contiene todos los controles e indicadores que se emplearán para crear la interfaz del VI con el usuario.

Tabla 1 Controles e Indicadores disponibles en el Panel Frontal.

 <p>Numeric</p>	<p><i>Numeric</i>, para la introducción y visualización de cantidades numéricas</p>
 <p>Boolean</p>	<p><i>Boolean</i>, para la entrada y visualización de valores booleanos.</p>

 <p>String & Path</p>	<p><i>String & Path</i>, para la entrada y visualización de texto. Path permite conocer el directorio en el que se encuentra cierto texto procesado.</p>
 <p>Array & Cluster</p>	<p><i>Array & Cluster</i>, para agrupar elementos de otros indicadores u controles.</p>
 <p>List & Table</p>	<p><i>List & Table</i>, para visualizar y/o seleccionar una lista de opciones y tablas.</p>
 <p>Graph</p>	<p><i>Graph</i>, para representar gráficamente los datos. Controles e indicadores de gráficas. Pueden ser gráficas de barrido, graficas XY, o de tonos de colores.</p>
 <p>Ring & Enum</p>	<p><i>Ring & Enum</i>, para gestión de archivos.</p>
 <p>Containers</p>	<p><i>Containers</i>, entre otras cosas posee controles ActiveX que permiten transferir datos y programas de unas aplicaciones a otras dentro de Windows.</p>
 <p>I/O</p>	<p>I/O, posee diversos componentes creados por National Instrument para Hardware de la misma compañía.</p>
 <p>Decorations</p>	<p><i>Decorations</i>, para introducir decoraciones en el <i>panel frontal</i>. No visualizan datos.</p>

 Select a Cont...	<i>Select a Control</i> , para seleccionar cualquier control.
 User Controls	<i>User Controls</i> , para elegir un control creado por el propio usuario.
 Classic Controls	<i>Classic Controls</i> , para visualizar los mismos controles e indicadores descritos anteriormente, pero con un formato más clásico.

2.1.3 Controles e Indicadores.

Un control es un objeto que usted puede colocar en su panel frontal para entrar datos a un VI interactivamente o en un subVI. Un indicador es un objeto que usted coloca en su panel frontal para desplegar información (salidas). Los controles e indicadores en G son similares a las entradas y salidas, respectivamente, en lenguajes tradicionales. La mecánica de programación consta en colocar controles e indicadores en el panel frontal y alambrándolos entonces a funciones o VI's en el diagrama de bloques.

Cada vez que el programador crea un nuevo control o indicador en el panel frontal, LabVIEW crea el terminal correspondiente en el diagrama de bloques.

Uno de los controles mas frecuentes es el botón *stop*. Este es posible editarlo para diferentes acciones mecánicas que se describen en las líneas siguientes. Para seleccionar el modo de operación posicionarse sobre el botón stop, luego presionar el click derecho del mouse, se verá el siguiente menú.

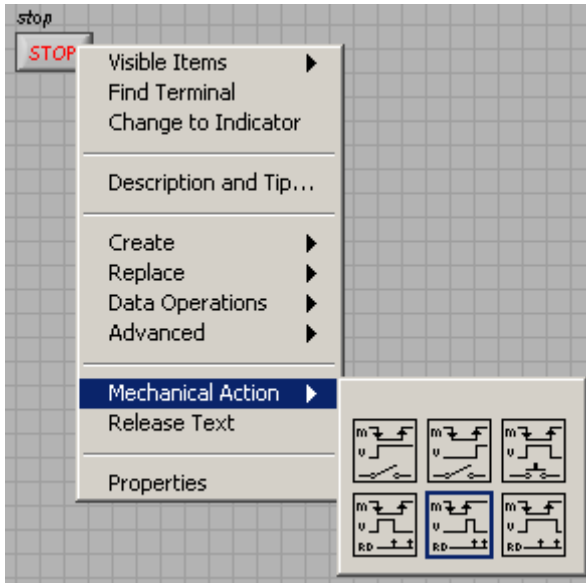


Figura 5 Acciones Mecánicas disponibles para stop button.



Switch When Pressed Cambia el valor cuando el mouse es presionado, el nuevo valor se mantiene.



Switch When Released Cambia el valor cuando el mouse se deja de presionar, el nuevo valor se mantiene.



Switch Until Released Cambia el valor cuando el mouse es presionado, y vuelve al valor original cuando el mouse se suelta.



Latch When Pressed Cambia un ciclo después de ser presionado y vuelve al valor original en un nuevo ciclo



Latch When Released Cambia un ciclo después de ser liberado y vuelve al valor original en el siguiente ciclo



Latch Until Released Cambia el valor apenas el mouse es presionado y vuelve al valor original un ciclo después de que el mouse es liberado

El mismo menú se despliega para *Push Button*, *Rocker*, *Slide Switch*, *Vert Rocker*, *Horizontal Toggle Switch*, *Vert Toggle Switch*, *Ok Button*, *Cancel Button*, etc. Todos estos se encuentran en el panel frontal, en el menú *Boolean*.

2.1.4 Terminales.

Los terminales son regiones de un VI a través de las cuales pasan los datos. Los terminales son análogos a los parámetros en programación basada en texto. Es importante que el programador alambree correctamente los terminales de una función o VI. Puede verse el conector del icono para hacer la instalación eléctrica más fácil. Para hacer esto, accese el menú emergente de la función y elija Show ➔Terminals.



Figura 6 Función *Multiply* con sus tres terminales.

2.1.5 Alambres.

Un alambre es el camino de los datos entre los nodos. Los alambres están coloreados según el tipo de dato que llevan. Los alambres azules llevan enteros, las naranjas llevan números en punto flotante, los alambres verdes llevan Booleanos, y los alambres rosas llevan cadenas de caracteres. Para alambrear de un terminal a otro, escoja la herramienta de la Instalación eléctrica y pulse el botón en el primer terminal, mueva la herramienta al segundo terminal, y pulse el botón. No importa en qué terminal se empiece.

Cuando la herramienta de la Instalación eléctrica está encima de un terminal, el área terminal titila entre negro y blanco, para indicar que pulsando el botón conecta el alambre a ese terminal. No sujete el botón del mouse mientras esté moviendo la herramienta de la Instalación eléctrica de un terminal a otro.

Para facilitar la tarea de conexión de todos los terminales, en el menú “*Help*” puede elegirse la opción “*Show Help*” o CTRL + H, con lo que al colocar el cursor del ratón sobre un elemento aparece una ventana con información relativa a éste (parámetros de entrada y salida).

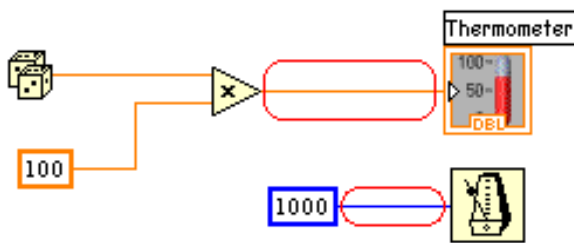


Figura 7 Representación de dos tipos de cables/datos distintos.

En la figura 7 se encerraron en un marco rojo las líneas naranja y azul, esto debido a que LabVIEW maneja distintos tipos de datos que son representados por colores. Mas adelante se detallarán todos lo tipos de datos que se pueden utilizar en este software.

2.1.6 Highlight Execution.

Anteriormente se comentó la utilización de una herramienta muy usada en la programación de LabVIEW, es el *Highlight Execution*, representada por un icono con forma de ampolleta de luz en la barra de herramientas. Cuando esta opción está activada la ejecución se ralentiza y se puede ver el fluir de los datos por el Diagrama de Bloques.













	Tipo de dato	Bits de almacenamiento	Dígitos decimales (Aprox.)	Límites	Color
	Precisión extendida Coma flotante	128	33/15 (20)	6,48e-4966 a 1,19e+4932 -6,48e-4966 a -1,19e+4932	Naranja
	Precisión doble Coma flotante	64	15	4,94e-324 a 1,79e+308 -4,94e-324 a -1,79e+308	Naranja
	Precisión simple coma flotante	32	6	1,40e-45 a 3,40e+38 -1,40e-45 a -3,40e+38	Naranja
	Largo Entero con signo	32	9	-2.147.483.648 a 2.147.483.647	Azul
	Palabra Entero con signo	16	4	-32.768 a 32.767	Azul
	Byte Entero con signo	8	2	-128 a 127	Azul
	Largo Entero sin signo	32	9	0 a 4.294.967.295	Azul
	Palabra Entero sin signo	16	4	0 a 65.535	Azul
	Byte Entero sin signo	8	2	0 a 255	Azul
	Precisión Extendida Complejo	256	33/15	Igual que EXT para cada parte (real e imaginario)	Naranja
	Precisión doble Complejo	128	15	Igual que DBL para cada parte (real e imaginario)	Naranja
	Precisión simple Complejo	64	6	Igual que SGL para cada parte (real e imaginario)	Naranja

Figura 10 Tipos de datos en LabVIEW.







	Variable Booleana
	Número de referencia (refnum)
	Variant (datos con tipo de dato incluido)
	Dato polimórfico (función que admite varios tipos de datos)
	Nombre I/O (nombres de canal I/O, recurso VISA...)
	Gráficos

Figura 11 Otros tipos de datos en LabVIEW.

3.1 ESTRUCTURAS.

En la *paleta de funciones* la primera opción es la de las *estructuras*. Éstas controlan el flujo del programa, bien sea mediante la secuenciación de acciones, ejecución de bucles, etc.

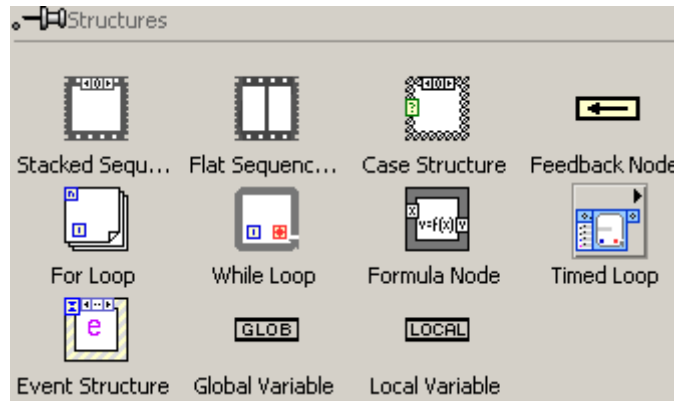


Figura 12 Paleta de Structure.

Las estructuras se comportan como cualquier otro nodo en el diagrama de bloques, ejecutando automáticamente lo que está programado en su interior una vez tiene disponibles los datos de entrada, y una vez ejecutadas las instrucciones requeridas, suministran los correspondientes valores a los cables unidos a sus salidas. Sin embargo, cada estructura ejecuta su *subdiagrama* de acuerdo con las reglas específicas que rigen su comportamiento, y que se especifican a continuación.

Un *subdiagrama* es una colección de nodos, cables y terminales situados en el interior del rectángulo que constituye la estructura. El *For Loop* y el *While Loop* únicamente tienen un subdiagrama. El *Case Structure* y el *Sequence Structure*, sin embargo, pueden tener múltiples subdiagramas superpuestos como si se tratara de cartas en una baraja, por lo que en el diagrama de bloques únicamente será posible visualizar uno a la vez. Los subdiagramas se construyen del mismo modo que el

resto del programa Las siguientes estructuras se hallan disponibles en el lenguaje G.

3.1.1 Case Structure.

Al igual que otras estructuras posee varios *subdiagramas*, que se superponen como si de una baraja de cartas se tratara. En la parte superior del subdiagrama aparece el identificador del que se está representando en pantalla. A ambos lados de este identificador aparecen unas flechas que permiten pasar de un *subdiagrama* a otro.

En este caso el identificador es un valor que selecciona el subdiagrama que se debe ejecutar en cada momento.

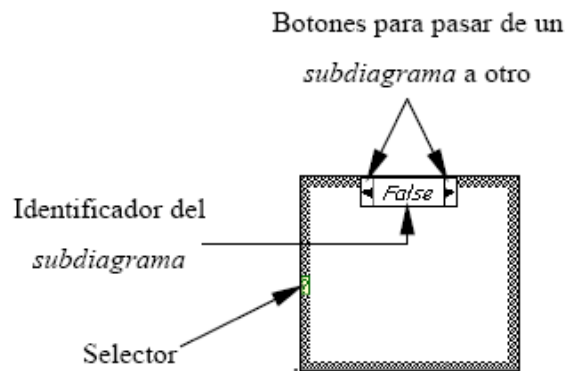


Figura 13 Case Structure.

La estructura *Case* tiene al menos dos *subdiagramas* (*True* y *False*). Únicamente se ejecutará el contenido de uno de ellos, dependiendo del valor de lo que se conecte al *selector*.

3.1.2 Sequence Structure.

Este tipo de estructuras presenta varios *subdiagramas*, superpuestos como en una baraja de cartas, de modo que únicamente se puede visualizar una en pantalla.

También poseen un identificador del *subdiagrama* mostrado en su parte superior, con posibilidad de avanzar o retroceder a otros *subdiagramas* gracias a las flechas situadas a ambos lados del mismo. Estos subdiagramas se insertan pulsando el botón derecho del ratón sobre el borde de la estructura, seleccionando la opción *Add Frame After*.

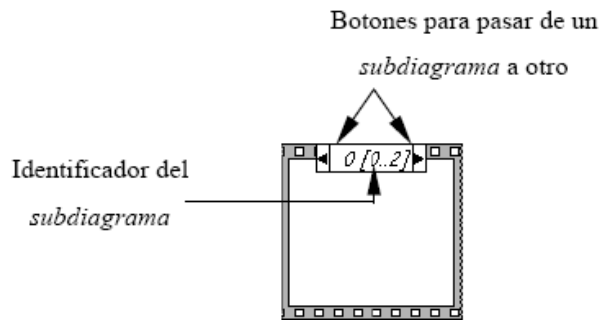


Figura 14 Sequence Structure.

Esta estructura secuencia la ejecución del programa. Primero ejecutará el *subdiagrama* del *frame* nº0 (círculo rojo) que ejecutará valores aleatorios entre 0 y 1, después se ejecutará el *frame* nº 1 que entregará el resultado (salida), y así sucesivamente. Para pasar datos de una hoja a otra se pulsará el botón derecho del ratón sobre el borde de la estructura, seleccionando la opción *Add sequence local*, se agregaran etiquetas similares a las encerradas en los círculos azules.

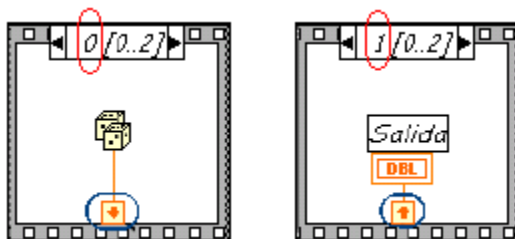


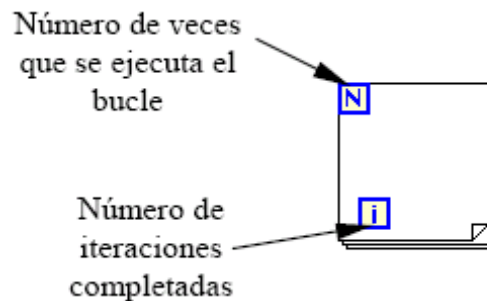
Figura 15 Paso del frame 0 a la 1.

Las dos *Sequence Structure* mostradas arriba en realidad son solo una pero que posee dos frame (0 y 1 encerrados en un círculo rojo).

3.1.3 For Loop.

Es el equivalente al bucle *for* en los lenguajes de programación convencionales. Ejecuta el código dispuesto en su interior un número determinado de veces.

Figura 16 For Loop.



Para pasar valores de una iteración a otra se emplean los llamados *shift registers*. Para crear uno, se pulsará el botón derecho del ratón mientras éste se halla situado sobre el borde del bucle, seleccionando la opción *Add Shift Register*. El shift register consta de dos terminales, situados en los bordes laterales del bloque. El terminal izquierdo almacena el valor obtenido en la iteración anterior. El terminal derecho guardará el dato correspondiente a la iteración en ejecución. Dicho dato aparecerá, por tanto, en el terminal izquierdo durante la iteración posterior. La siguiente figura esquematiza lo anterior.

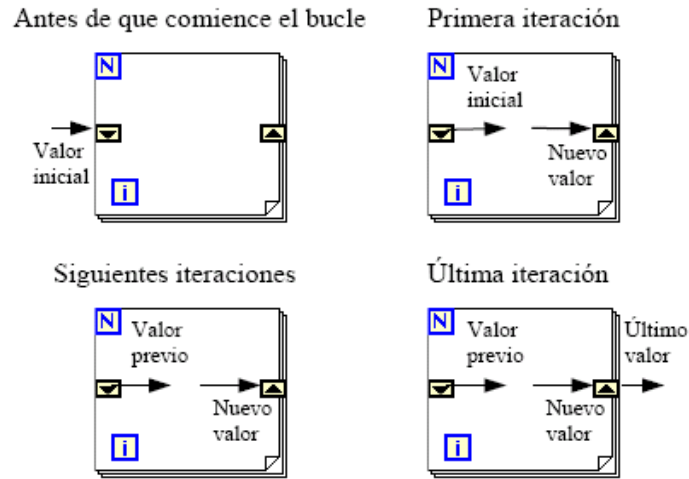


Figura 17 Secuencia de Iteraciones, de la inicial a la última.

Se puede configurar un *shift register* para memorizar valores de varias iteraciones previas. Para ello, con el ratón situado sobre el terminal izquierdo del *shift register* se pulsará el botón derecho, seleccionando a continuación la opción *Add Element*.

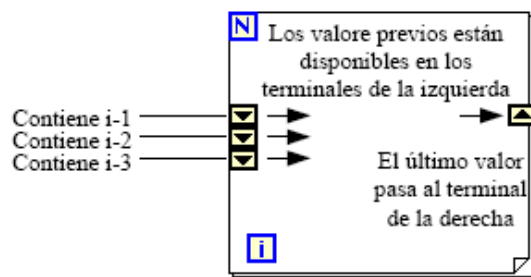


Figura 18 For Loop que permite memorizar tres iteraciones.

3.1.4 While Loop.

Es el equivalente al bucle *while* empleado en los lenguajes convencionales de programación. Su funcionamiento es similar al del bucle *for*.

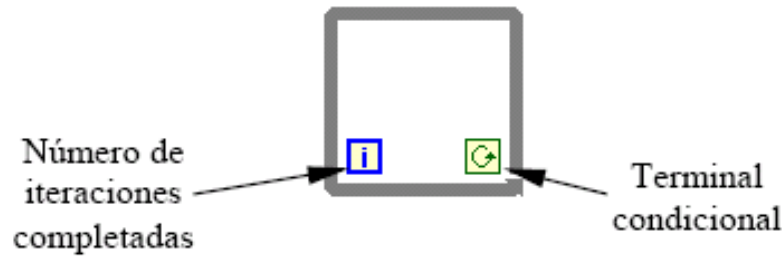


Figura 19 While Loop.

El programa comprueba el valor de lo que se halle conectado al terminal condicional para finalizar el bucle. Por lo tanto, el bucle siempre se ejecuta al menos una vez. Con esta estructura también se pueden emplear los *shift registers* para tener disponibles los datos obtenidos en iteraciones anteriores (es decir, para memorizar valores obtenidos), su empleo es análogo al de los bucles *for*, por lo que se omitirá su explicación.

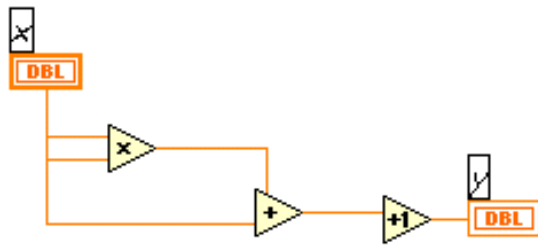
3.1.5 Formula Node.

La estructura denominada *Formula Node* se emplea para introducir en el diagrama de bloques fórmulas de un modo directo. Resulta de gran utilidad cuando la ecuación tiene muchas variables o es relativamente compleja. Por ejemplo, se desea implementar la ecuación:

$$y = x^2 + x + 1$$

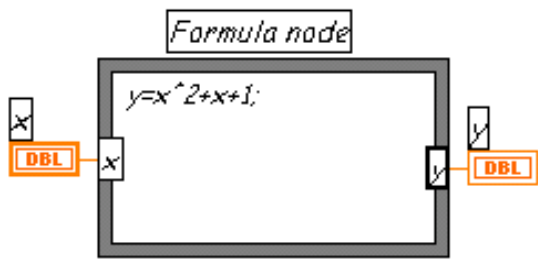
Existen dos formas de resolver esta ecuación, según la figura 20a y la 20b, (la segunda es más fácil de programar que la primera) usando LabVIEW. Veamos la primera opción.

a)



En esta opción se utilizan funciones aritméticas que se pueden ubicar en el diagrama de bloques, en la paleta de funciones, en el menú *Numeric*.

b)



La segunda opción utiliza un *Formula Node* en el que se puede escribir directamente la ecuación a calcular.

Figura 20 Comparación de dos métodos para resolver lo misma ecuación.

X es un *Numeric Control*, que básicamente permite cambiar a voluntad el dato ingresado, en este caso este dato es un entero. Y es un *Numeric Indicator*, este muestra el resultado del cálculo. Ambos se visualizan en el panel frontal, y se encuentran ingresando al menú *Numeric*.

4.1 TIPOS DE VARIABLES.

4.1.1 Locales.

Son variables asociadas a algún control o indicador dentro de un programa VI en el cual son usadas. Cuando se escribe en una de éstas, el contenido del indicador o control cambia.

El uso de estas variables facilita la visualización en el diagrama cuando se va a acceder varias veces a un mismo dato, puesto que evita llenar de cables conductores la pantalla en el diagrama de bloques, o colmar de indicadores o controles numéricos, que conduzcan el valor desde el control al lugar requerido.

Las variables locales solo son entendidas por el programa VI que las posee, ninguna subrutina entiende el contenido de éstas, ni ningún programa VI diferente, es decir, son exclusivas del VI en el que fueron creadas.

Para obtener una de estas variables, se busca en el menú de *Structures*, *Local Variable*. Una vez creada, se presiona sobre el botón izquierdo del mouse, apareciendo inmediatamente un menú (ver figura 21) que permite elegir el control o indicador que deseo utilizar como variable local, en este caso el control a escoger es *stop*.

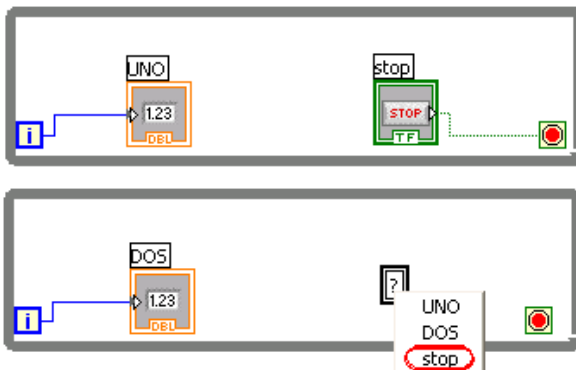


Figura 21 Selección del stop para la variable local.

Si se desea leer un valor de la variable seleccionar en el menú *change to read*. El icono de la variable mostrará las paredes laterales más gruesas que en el caso de la variable a la que se escribe. Para escribir seleccionar en el menú *change to write*.

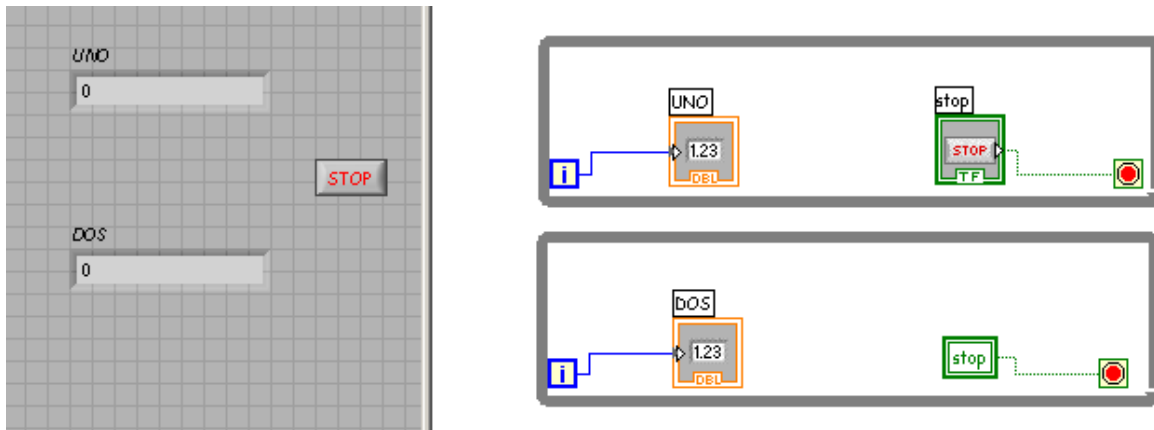


Figura 22 Utilización de una variable local stop.

4.1.2 Globales.

La diferencia con las variables locales radica en que estas pueden ser entendidas por cualquier programa y/o subrutina VI, y pueden ser actualizadas por los mismos. Estas se almacenan en un archivo diferente de extensión **.GLB** que consta únicamente de un panel frontal donde se encuentran todas las variables asociadas a dicho archivo, lo que quiere decir que en un archivo se pueden guardar numerosas variables.

Para crear una variable global, se selecciona del menú de *Structures*. Luego con el menú se da la orden de abrir el panel frontal de esta variable, y allí se colocan todos los indicadores y controles que almacenan los datos deseados. Posteriormente se graba como cualquier otro programa VI, pero con extensión DBL.

Con la variable creada solo es seleccionar en el pop-up menú del icono de la variable con *select item*, el valor al que se lee o escribe.

Para colocar en el diagrama otra variable global del mismo dato, ahora se hace por medio de la opción VI, en el menú de funciones, tal como si se fuera a usar una subrutina ya creada.

5.1 GRAFICADORES.

LabView cuenta con algunos controles o indicadores que presentan gráficas de los datos obtenidos en el programa. Estos se encuentran en el submenú Graph en el menú de controles. Para cada uno se pueden configurar muchos parámetros como escala de la gráfica, auto escala, color de las líneas, número de líneas en una gráfica, presentación de letreros, paletas de control, indicadores, etc.

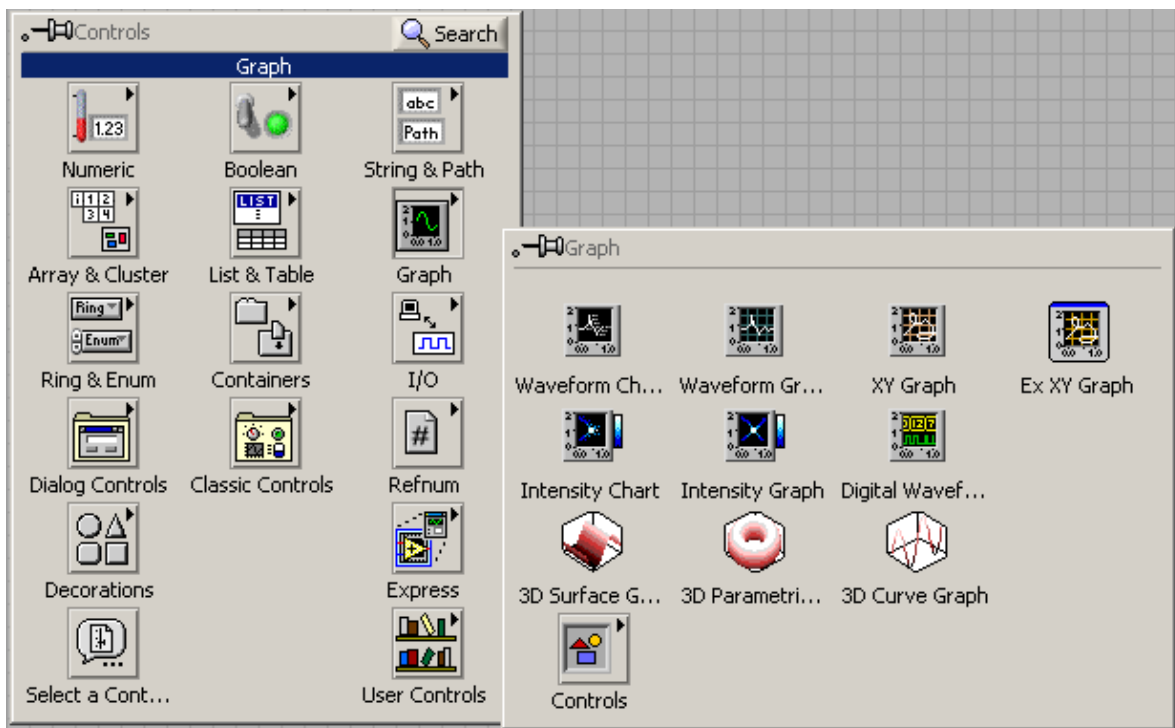


Figura 23 Paleta de Graph dentro del Panel Frontal.

Entre los graficadores más utilizados se encuentran:

5.1.1 Waveform Charts.

Permite entrar datos de a un número, o en una matriz de una dimensión. Grafica dando automáticamente la secuencia en el eje X, simplemente por el dato que sigue al anterior, es decir, por pasos. El eje X siempre corresponde a tiempo. Si se grafican dos o más datos se deben entrar con los dos o mas cables al cluster, como se muestra en la figura 24 (marco azul). Si se desea entrar todos los datos al mismo tiempo para llenar un buffer de la gráfica, hacerlo como una matriz, y si se desea graficar llenando buffers de varias graficas simultáneamente, entrar los datos como una matriz de clusters.

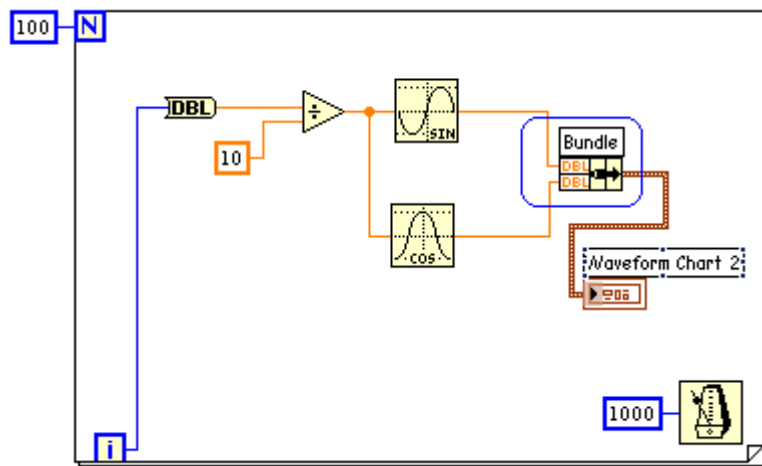


Figura 24 Utilización de un Waveform Chart.

La función *sin* y *cos* se encuentran en el diagrama de bloques, en el menú *Numeric*, en el sub menú *Trigonometric*. La función *DBL* se encuentra en el mismo menú, pero en el sub-menú *Conversion*, básicamente esta

función permite convertir lo que in en dato *DBL* en la out. El cluster utilizado se llama *Bundle* y se ubica en el interior del menú *Cluster*.

5.1.2 Waveform Graph.

De comportamiento similar a la Waveform Chart, pero con esta ya se puede definir la escala en el tiempo, a los valores deseados, mientras que en la anterior la escala en X es propiamente de pasos, más que tiempo.

5.1.3 XY Graph.

En ésta se entran los datos por pares ordenados en una matriz bidimensional, o una matriz de clusters de dos datos cada uno X, Y. Permite graficar funciones matemáticas, círculos, etc., dando una secuencia de puntos, X, Y.

5.1.4 Intensity Graph e Intensity Chart.

Para graficar planos de diferentes colores, para matrices de dos dimensiones, donde los valores contenidos corresponden a un color.

6.1 SUB VI.

Los programas creados en LabVIEW se llaman VIs (Virtual Instrument). En muchas ocasiones un programa será de un tamaño tal que habrá que separarlo en varios VIs o habrá alguna sección de código que convenga reutilizarla varias veces. Un VI puede contener a otro de forma que el segundo sería un subVI del primero, el concepto es equivalente a las funciones de un lenguaje tradicional. Por ejemplo, el

siguiente programa es posible reducirlo notablemente, basta con solamente crear subVI.

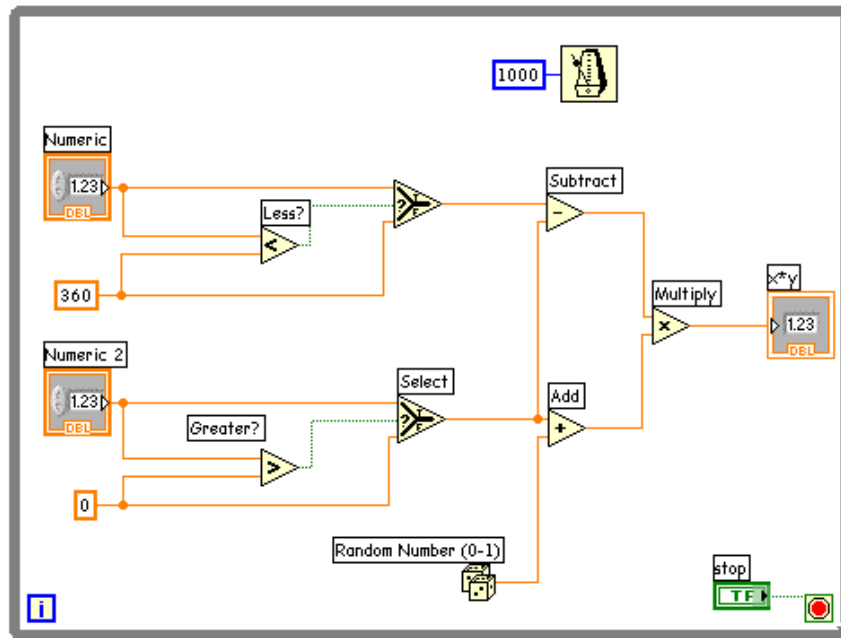


Figura 25 Programa que utiliza diferentes funciones.

En el ejemplo de la figura 25 se puede ver un programa que tiene como entradas Límite superior y Límite inferior, estas entradas se limitan de forma programada a 360 y 0 respectivamente mediante las funciones de comparación *Less?*, *Greater?* y *Select*. A las salidas de las funciones de comparación se obtendrá un valor *true* si la comparación es cierta y *false* en caso contrario. *Select* funciona como un multiplexor: a su salida estará el valor de la entrada superior (T) si la señal de selección (?) es cierta y el de la entrada inferior (F) si es falsa. Por otra parte, *Random Number (0-1)* genera un número aleatorio entre cero y uno. *Less?*, *Greater?* y *Select* se encuentran en el menú *Comparison*, ubicable en la paleta de funciones, en el diagrama de bloques. *Random Number (0-1)* es ubicable en el menú *Numeric*.

La parte central del programa resta las dos entradas y el resultado lo multiplica por la suma del límite inferior y el número aleatorio. Con esto se consigue generar un número aleatorio que será visto en el *Indicador Numeric*.

En el ejemplo anterior puede ser deseable hacer una función para la generación del número aleatorio entre dos límites, es decir, hacer que esa parte del código sea un VI distinto, de esta forma podrá ser usado en otras ocasiones. La forma más sencilla de conseguir esto es seleccionando la parte deseada del Diagrama de Bloques e ir a *Edit* → *Create SubVI*. Al hacerlo el código seleccionado será sustituido por el icono de un VI, con un doble clic sobre este icono se accederá al código de este subVI.

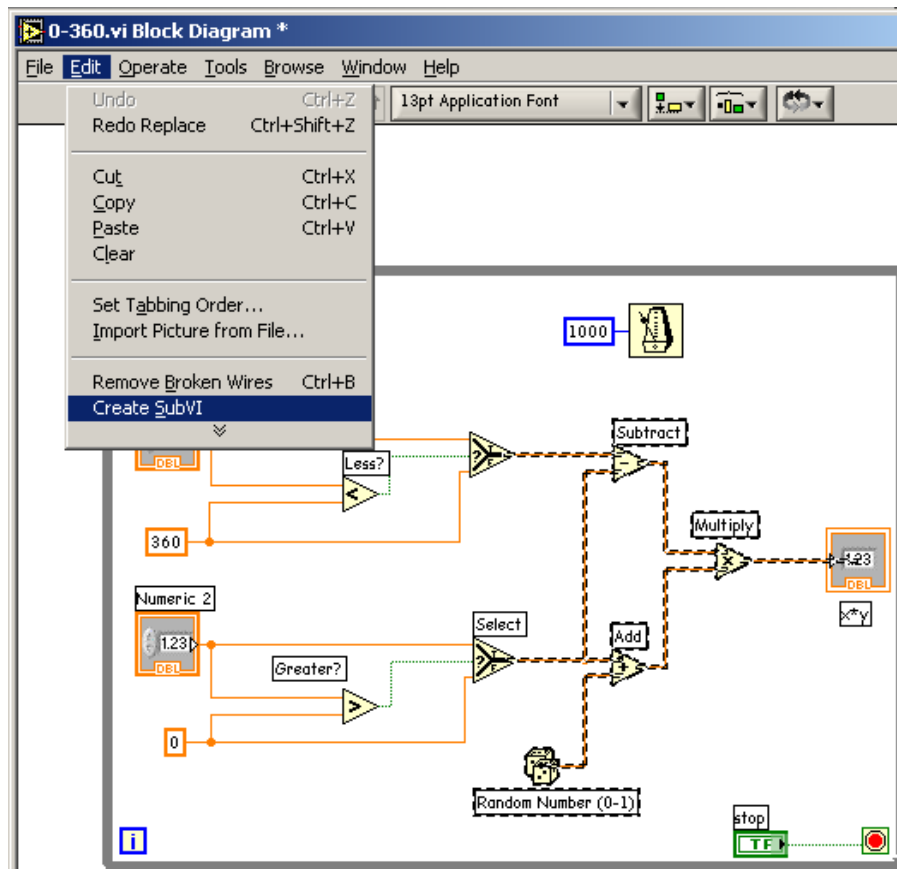


Figura 26 Creación del área seleccionada en Sub VI.

Otra forma de crear un Sub VI es definiendo de forma manual su interfaz, es decir, la forma en que se realizarán las conexiones cuando se use como sub VI dentro de otro VI. El primer paso será guardar el VI, después situarse en su Panel Frontal y hacer clic con el botón secundario del ratón sobre el icono del VI (parte superior derecha) para desplegar su menú contextual, como se puede ver en la figura 27.

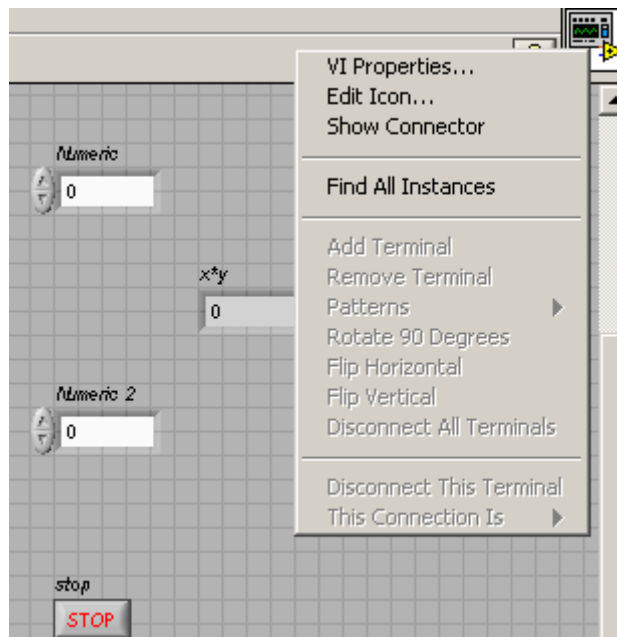


Figura 27 Otra opción para crear un VI en subVI.

En *Edit Icon...* es utilizado para cambiar la presentación original del icono de cualquier VI. *Show Connector* es utilizado para agregar o eliminar terminales, los terminales son los lugares donde se conectarán los cables cuando se use como subVI. Al presionar la opción *Show Connector* se muestra el icono en blanco indicando que no tiene ningún conector de entrada o salida asignado, para asignarle contactos se pincha sobre el

icono con el botón derecho del Mouse, se selecciona la opción Patterns, según muestra la figura 28.

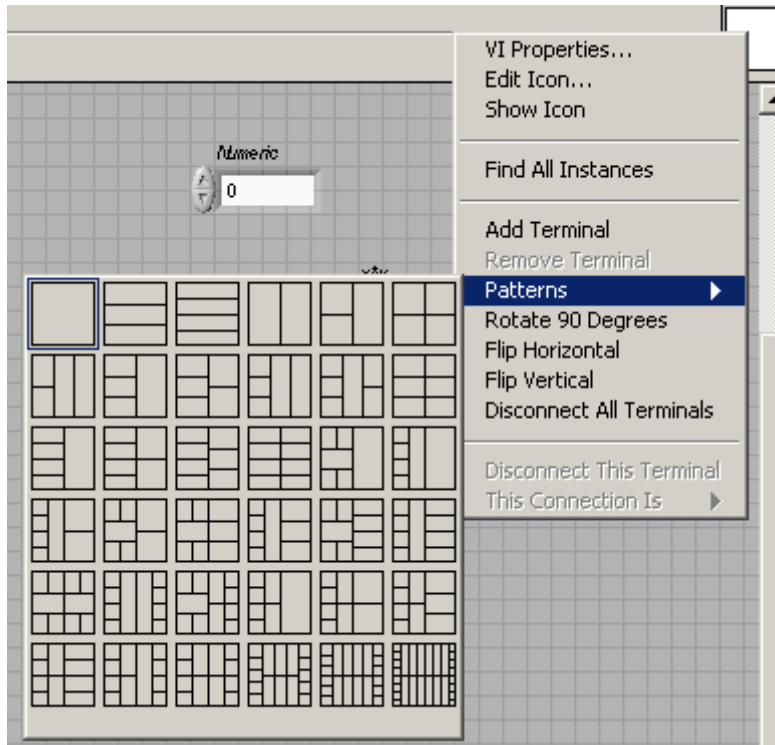


Figura 28 Selección de contactos, según sea necesario.

La rejilla de conexión define las entradas y salidas que se pueden conectar al VI al usarlo como subVI. La rejilla de conexión recibe datos en los terminales de entrada y los pasa al código del diagrama de bloques a través de los controles del panel frontal y recibe los resultados en sus salidas a través de los indicadores del panel frontal.

La rejilla de conexión se presenta como un modelo de conectores, se puede elegir otro modelo si es necesario. Cuenta con un terminal por cada control o indicador del panel frontal, se pueden asignar hasta 28 terminales. Si se piensa que van a haber cambios que requieran entradas o salidas, se pueden colocar terminales extra sin asignar.

Nota: no es aconsejable asignar más de 16 terminales, ya que se reduce mucho su funcionalidad y conectividad.

Finalmente, en este punto, para asignar un control o indicador a un terminal se debe seleccionar la herramienta *Connect Wire* y hacer clic en el terminal y en el control o indicador asociado del Panel Frontal, en ese momento el terminal se coloreará indicando el tipo de datos.

En la figura 26 se presentó un VI en el que se seleccionó cierta parte para crearlo como un subVI, veamos entonces como queda el diagrama de bloques (marco rojo) según figura 29. Se presenta además el Panel frontal del subVI.

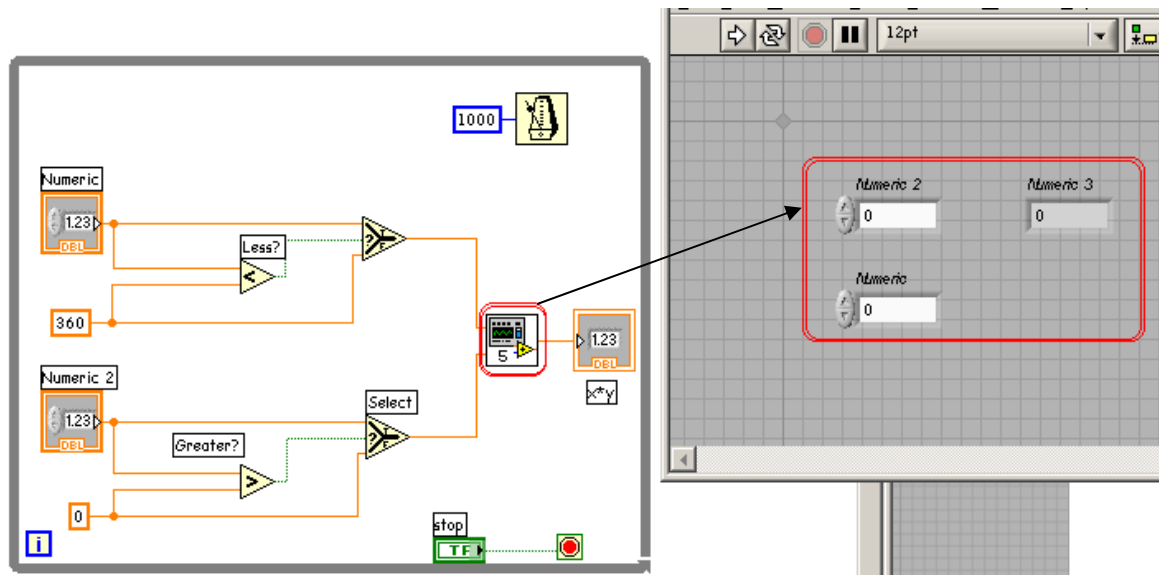


Figura 29 Presentación final del diagrama de bloques, con el subVI.

Para editar el icono se selecciona con el botón derecho del mouse en el icono del panel frontal y se selecciona **'Edit Icon'**. En este editor se puede dibujar el icono deseado

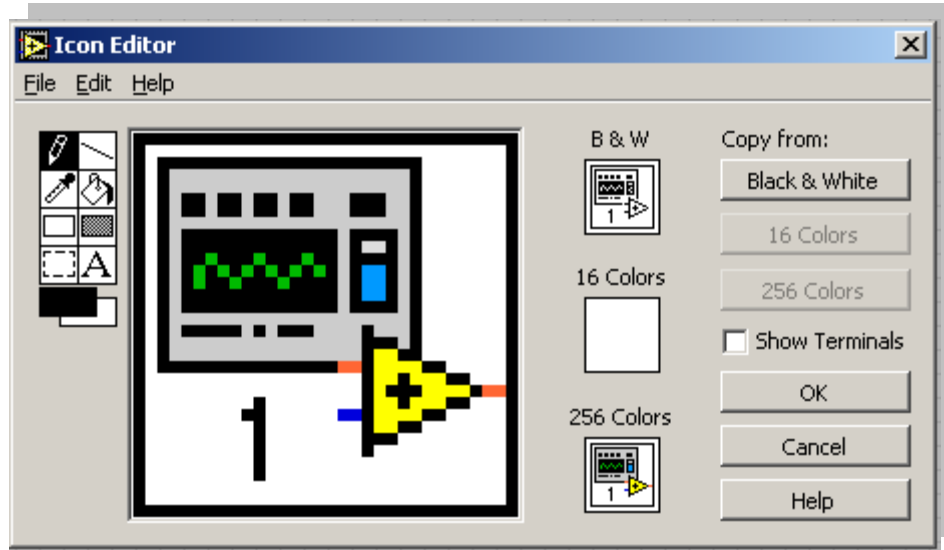


Figura 30 Opciones de edición de íconos.

6.2 BUSCANDO ERRORES DE SINTAXIS.

Un VI no puede compilarse o correr si la función de ejecución normal está rota. Esto pasa siempre que no existan todas las conexiones (alambres) hechas o se conectan variables de distintos tipos.

Si al terminar de alambrear todas las funciones en el diagrama de bloque el VI sigue roto (ver figura 31), debe buscarse el error.



Figura 31 Flecha rota que indica la presencia de algún error de programación.

Para listar los errores, pulse con el botón izquierdo del Mouse sobre la flecha rota, LabVIEW abrirá un cuadro de diálogo como el que se muestra en la siguiente ilustración.

Utilizando el mismo ejemplo mostrado en la figura 25 para generar voluntariamente un error. En este caso el error existe porque la salida de la función *Less?* no está conectada a entrada *s* de la función *Select* (marco azul). En el Error List se detallan la lista de error (es).

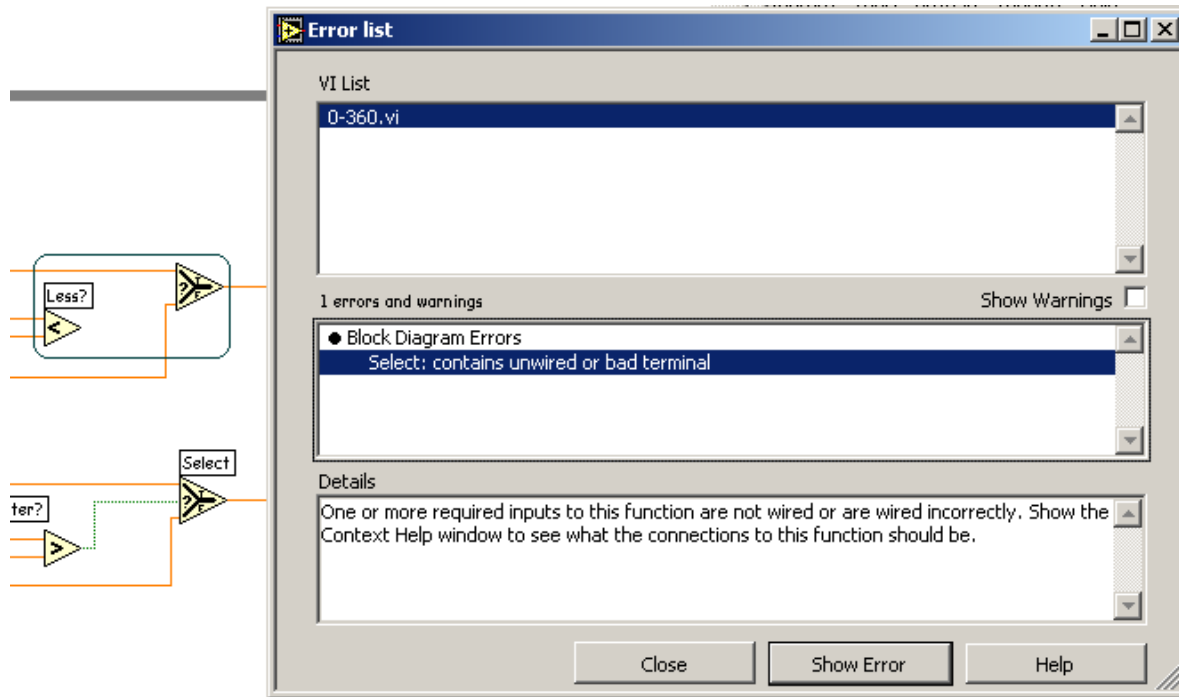


Figura 32 Detección de fallas en la programación de LabVIEW.

6.3 ATAJS DE TECLADO EN LABVIEW.

A continuación se muestra la Tabla 2 con los atajos de teclado más útiles de LabVIEW. Pueden personalizarse en Tools → Options → Menú Shortcuts.

Tabla 2 Atajo de teclado en LabVIEW.

Combinación de Teclas	Función
CTRL + R	Ejecuta el programa
CTRL + .	Aborta la Ejecución
CTRL + E	Intercambia pantalla entre el Panel Frontal y el Diagrama de Bloque.
CTRL + B	Elimina los hilos rotos
CTRL + H	Muestra o oculta la ayuda contextual
CTRL + ?	Muestra la ayuda.
CTRL + C	Copia objetos seleccionados.
CTRL + X	Elimina objetos seleccionados.
CTRL + V	Pega los objetos seleccionados.
CTRL + Z	Deshace la última edición.
CTRL + S	Guarda el VI.
TAB	Pone en activo la selección automática del Tools Pallettes.
CTRL + arrastrar	Selecciona objetos.
SHIFT + arrastrar	Mueve los objetos seleccionados en una dirección.

6.4 MENÚS DE LABVIEW.

La programación en LabVIEW obliga a utilizar con frecuencia los diferentes menús. La barra de menús de la parte superior de la ventana de un VI contiene diversos menús pulldown (desplegables). Cuando hacemos clic sobre un ítem o elemento de esta barra, aparece un menú por debajo de ella. Dicho menú contiene elementos comunes a otras aplicaciones Windows, como Open (Abrir), Save (Guardar) y Paste (Pegar), y muchas otras particulares de LabVIEW.



Figura 33 Barra de menú de LabVIEW.

- ❖ *File (Archivo):* Sus opciones se usan básicamente para abrir, cerrar, guardar, imprimir VIs. Desde la versión 8.0 se ha incluido ítems básicos para las operaciones de proyectos, tales como abrir, cerrar, guardar e imprimir proyectos.
- ❖ *Edit (Edición):* Se usa principalmente para organizar el panel frontal y el diagrama desbloques y establecer nuestras preferencias.
- ❖ *Operate (Función):* Sus comandos sirven para ejecutar el VI.
- ❖ *Tools (Herramientas):* Esta se utiliza para realizar operaciones como la publicación de páginas web, enlaces con otros programas NI, etc.
- ❖ *Browse:* Indica la jeraquia que tiene un VI, además muestra los subVI que tiene un VI.
- ❖ *Window:* Lo utilizamos para desplegar las paletas, para movernos entre las ventanas de LabVIEW, etc.