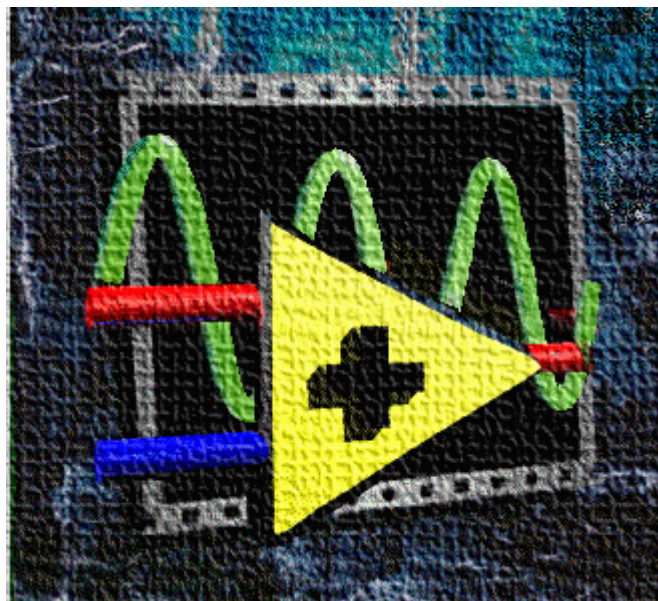


INTRODUCCIÓN A LA INSTRUMENTACIÓN VIRTUAL. PROGRAMACIÓN EN LABVIEW.



VERSIÓN 5.6

**Ignacio Moreno Velasco
Pedro L. Sánchez Ortega**

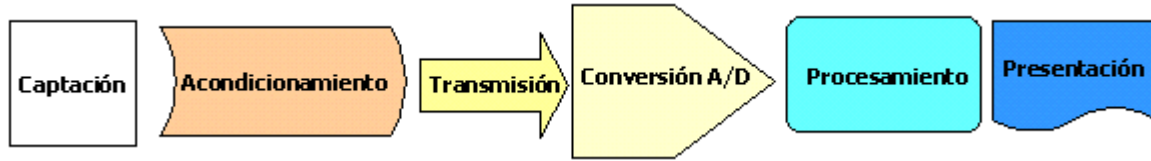
INDICE

1.- INTRODUCCIÓN A LOS SISTEMAS DE ADQUISICIÓN	5
Entorno Industrial	5
Entorno de laboratorio	5
2.- SOFTWARE	7
CONCEPTO DE INSTRUMENTO VIRTUAL	7
Estructura del software	7
Programa de aplicación	7
SOFTWARE CONTROLADOR DE DISPOSITIVO	8
3.- PRÁCTICAS DE INICIACIÓN A LABVIEW	9
Práctica 1ª: Realizar la suma de 2 números	10
Insertar los elementos en el panel frontal	10
Insertar la operación de suma:	10
Llamadas a la AYUDA:.....	10
Ejecución del programa	10
Sustitución de controles e indicadores.....	11
Práctica 2: Otras operaciones y controles	11
Introducir funciones de comparación:	12
Tipos de datos:	12
Concatenación de cadenas alfanuméricas frente a suma de números.	13
Práctica 3ª: adquisición y procesamiento básicos	14
Técnicas de depuración:.....	14
Creación de SubVI's	15
Práctica 4ª: Utilización de subVI's	16
Práctica 5ª: Secuencia While-loop	16
paso a	17
paso b	17
Práctica 6ª: registros de desplazamiento	17
Práctica 6ª: registros de desplazamiento	18
paso a	18
paso b.....	18
Práctica 7ª: Bucle For.	19
Práctica 8ª: MATRICES.	20
paso a	20
paso b.....	20
Práctica 9: tipos de gráficos	21
Práctica 10: Nudo fórmula	21
Práctica 11: Estructura CASE	22
Opción múltiple de la estructura case.....	22
4.- ADQUISICIÓN DE DATOS.	23
Conceptos básicos de adquisición de datos (a/d)	23
Velocidad de muestreo.....	23
Resolución	24
Rango	25
Práctica 12: Adquisición de datos	27
Paso A: uso de la función "AI Acquire waveform simu.VI"	27
Paso B: Escalado de tiempos.	27
Paso C: análisis en frecuencia.....	29
Paso D: Escalado de frecuencia.....	29
Ejercicios.....	30
Práctica 13: Ventanas de alisado	31
Práctica 14: Filtrado digital	32
14.1.- Promediado.....	32

14.2.- Filtros digitales	33
5.- BUS GPIB.	34
Introducción al bus GPIB.....	34
Historia	34
Características mecánicas y eléctricas	34
Funcionamiento	35
IEEE 488-2.....	37
HS488	37
Programación GPIB con LabVIEW	38
Conexión labview-dispositivo GPIB	38
Envío de comandos.....	38
Recepción de respuestas	39
Obtención de datos de la cadena de respuesta de un instrumento:.....	40
AplicacionesGPIB EN RED.....	41
Servidor remoto de comandos GPIB.....	41
Aplicaciones de distribución de señal en red.....	42

1.- INTRODUCCIÓN A LOS SISTEMAS DE ADQUISICIÓN

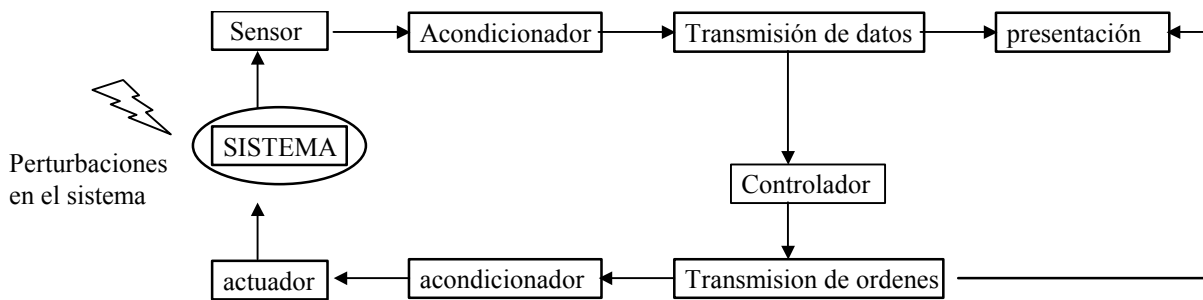
El esquema general de una cadena de medida cuya misión es la adquisición de datos puede ser el siguiente:



Este esquema general o alguna de sus variaciones lo encontraremos en múltiples entornos de los que cabe destacar dos: uno será el de los procesos industriales y el otro será el de ensayos y test que englobaremos bajo la denominación de entorno de laboratorio.

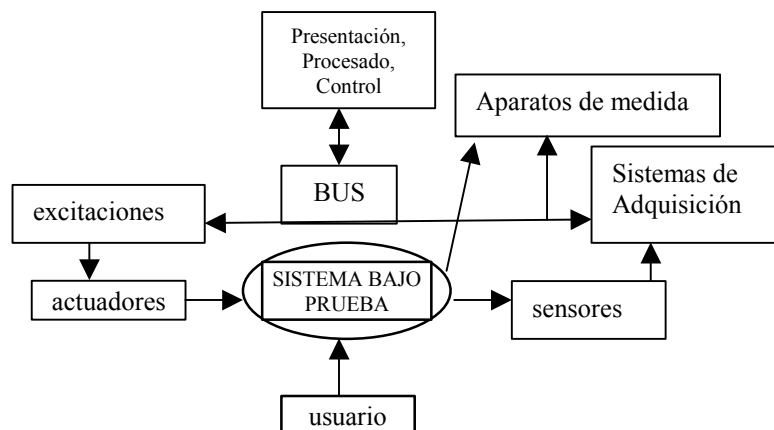
ENTORNO INDUSTRIAL

En un proceso industrial existirán varios sensores que suministran información, convenientemente acondicionada, al elemento controlador del sistema. El elemento controlador, que estará basado en algún microprocesador, recibirá la información de los sensores directamente o mediante un proceso de comunicación. Además de la presentación de dicha información en la forma deseada (generalmente gráfica) el elemento controlador dará las ordenes oportunas a los actuadores para mantener el proceso funcionando dentro de los márgnes previstos.



ENTORNO DE LABORATORIO

Para la instrumentación virtual o los procesos de laboratorio, la información puede venir dada no sólo por sensores, sino también por otros sistemas de medida (osciloscopios, multímetros, etc.) con capacidad de comunicación. Partiendo de la información recogida podemos cambiar las condiciones de la prueba, modificando parámetros de los aparatos (generadores de funciones, fuentes de alimentación) .



En un entorno como el descrito, la tendencia actual es que sea un software especializado quien se encargue del control del sistema, coordinando el funcionamiento de los distintos elementos.

Uno de estos programas software es LabView de la multinacional National Instruments. Labview permite recoger, analizar y monitorizar los datos dentro de un entorno de programación gráfico en el que se ensamblan objetos llamados instrumentos virtuales (Vis) para formar el programa de aplicación con el que interactuará el usuario y que se denomina **instrumento virtual**.

Además de lo que es la propia representación de los datos en los paneles interactivos que funcionan como si se tratara de instrumentación real, permite múltiples opciones de manejo de datos, como su almacenamiento en disco y compartirlos en red o con otras aplicaciones. La interacción con otras aplicaciones se podrá realizar mediante llamadas a librerías de enlace dinámico (DLL: Dinamic Link Library) e intercambio dinámico de datos (DDE: Dynamic Data Exchange) en modo local o mediante TCP/IP en conexiones remotas. Siempre buscando independencia de la plataforma en la que hayamos realizado nuestra aplicación.

La capacidad de comunicación con otros sistemas será una cualidad importante en cualquier equipo ó sistema. Además de la comunicación mediante interfaces comunes como el RS-232 o 485, podremos utilizar otros estándares más específicos de instrumentación como el IEEE-488 más conocido como GPIB , el VXI o en entornos industriales mas específicos el CAN.

2.- SOFTWARE

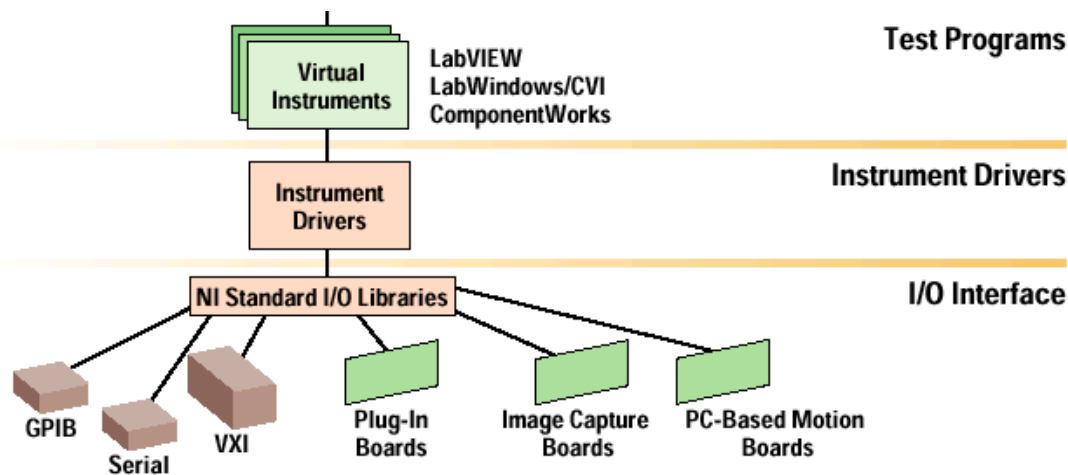
CONCEPTO DE INSTRUMENTO VIRTUAL

A diferencia de un instrumento real, que podemos tener en cualquier laboratorio o planta de procesos, y que queda perfectamente definido por unos mandos de control y unos elementos de representación, un instrumento virtual estará ligado al concepto de software. Este software se ejecutará en un ordenador que tendrá alojado unos elementos hardware concretos, tarjetas de adquisición de datos (analógicos y digitales), tarjetas de interfaz con los buses de instrumentación y unos canales de control también analógicos y digitales.

Nuestro instrumento virtual permitirá manejar ese hardware mediante una interfaz gráfica de usuario (IGU) que se asemejará al panel de mandos de los aparatos habituales (Osciloscopio, multímetro, etc.)

Mediante la representación en pantalla de los elementos gráficos de visualización y control que servirán de interfaz con el usuario, este observará los estados de las entradas seleccionadas en la pantalla e interactuará con las salidas directamente o mediante la ejecución de las rutinas que haya programado.

ESTRUCTURA DEL SOFTWARE



- Básicamente, el software se encargará de comunicar la interfaz de usuario del ordenador con el hardware de adquisición de datos dotando a la aplicación de la funcionalidad requerida.
- Podemos realizar una separación de las capas o partes del software: Programa de aplicación, controladores de dispositivo (drivers) y librerías de aplicación (API's).

PROGRAMA DE APLICACIÓN

El programa de aplicación, también llamado instrumento virtual, consta de dos partes: interfaz de usuario y funcionalidad de la aplicación:

IGU (Interfaz Gráfica de usuario)

- Permite la interacción de la aplicación con el usuario.
- Básicamente consta de controles e indicadores para visualización e introducción de datos.
- La mayoría de entornos de programación disponen de librerías de controles e indicadores creados que evitan una gran cantidad de trabajo al usuario.

Funcionalidad de la aplicación

- Una de las funciones básicas será la de obtener datos del hardware de forma transparente al usuario.

- La funcionalidad del programa incluye tratamiento de señal, control del flujo de programa, control de errores, etc...
- Puede implementarse en lenguajes basados en texto (Visual Basic, C++, LabWindows/CVI, etc.) o puede utilizar lenguaje gráfico como LabWiew, Snap Master, DasyLab, HP-VEE, Visual Designer de Burr Brown, etc..

Nuestro estudio se centrará en la programación bajo el entorno de programación gráfica LabView.

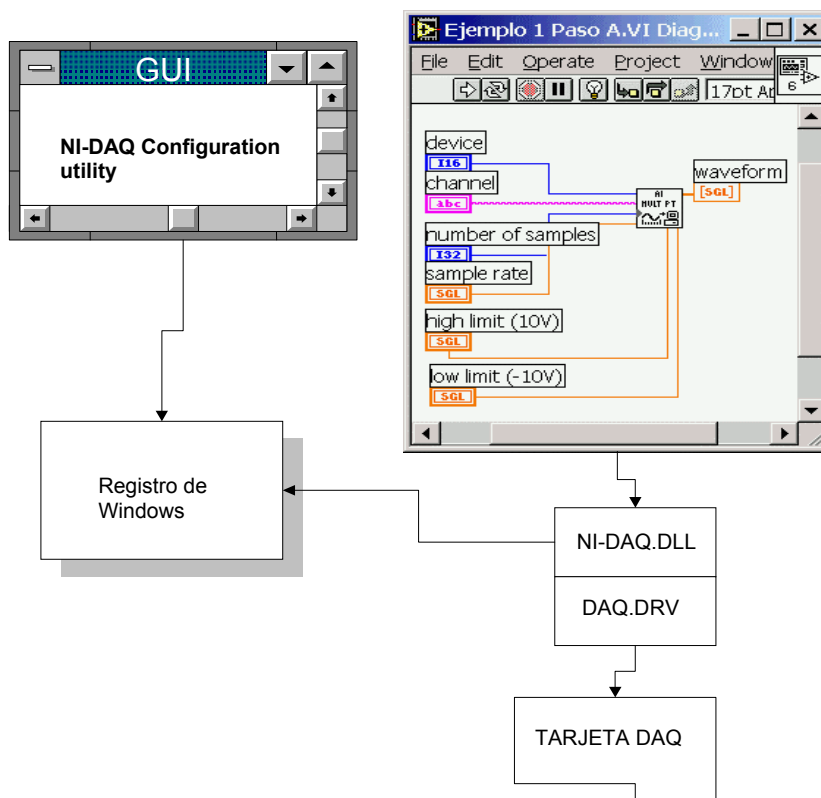
PROGRAMACIÓN GRÁFICA

Los procesos programables se definirán mediante un lenguaje gráfico en vez de un lenguaje orientado a líneas de código como estamos acostumbrados normalmente.

En este tipo de programación las funciones son bloques que se interconectan entre sí, intercambiando la información.

SOFTWARE CONTROLADOR DE DISPOSITIVO

El acceso al hardware ya no se realiza mediante llamadas directas a sus registros, si no que los fabricantes proporcionan una capa intermedia que aísla al programador de detalles hardware. Esta capa intermedia facilita la comunicación entre el hardware y nuestro entorno de programación. Suele implementarse mediante DLLs, por lo que se necesita una versión específica para cada sistema operativo. (p. ej. Win16 y Win32).



Todas las tarjetas ofrecen estas librerías como complemento software. Es tan importante la documentación como la variedad y flexibilidad de las librerías.

P. EJ. NI-DAQ DE NATIONAL INSTRUMENTS

- *nidaq32.dll*: Ocupa más de 2 MB y contiene cientos de funciones para el manejo de tarjetas de NI.
- *nidaqcfg.dll*: Librería para la configuración de los dispositivos conectados.

3.- PRÁCTICAS DE INICIACIÓN A LABVIEW

Las prácticas de que consta esta introducción a la programación con LabVIEW y los objetivos perseguidos son:

INTRODUCCIÓN A LA PROGRAMACIÓN GRÁFICA CON LABVIEW

Practica 1:

- Realización de la suma de 2 números.
- Contacto con el entorno de trabajo, panel de control y zona de diagrama.
- Concepto de visualizador y control.
- Llamadas a la ayuda en línea para las funciones.
- Ejecución única y ejecución cíclica.

Practica 2:

- Cambio de los controladores numéricos por otros gráficos.
- Tipos de datos
- Suma de cadenas de caracteres.

Práctica 3:

- Ejecución en modo depuración.
- Concepto de instrumento virtual (VI)
- Creación de instrumentos virtuales.

Práctica 4

- Incorporación de subVI
- Jerarquía de programa.

Práctica 5

- Secuencia While-Loop
- Gráficas

Práctica 6

- Registro de desplazamiento
- 'Bundle'

Práctica 7

- Bucles 'For'

Práctica 8

Arrays de elementos

Práctica 9

- Gráficas Chart y graph

Práctica 10

- Nodo fórmula

Práctica 11

- Estructuras 'case'

ADQUISICIÓN DE DATOS

Práctica 12

- Simulación de adquisición de datos
- Escalado de tiempos
- Gráficas en función de la frecuencia
- Analizador de espectro

Práctica 13

- Filtrado

PRÁCTICA 1ª: REALIZAR LA SUMA DE 2 NÚMEROS

INSERTAR LOS ELEMENTOS EN EL PANEL FRONTAL.

Para ello en el panel frontal insertaremos el elemento **Controls/Numeric/DigitalControl**. de la paleta "Functions"
Lo haremos 3 veces, una para el elemento A, otra para el B. Para el tercero, que será la suma de ambos A+B, insertaremos un indicador en vez de un control, **Controls/Numeric/DigitalIndicator**.

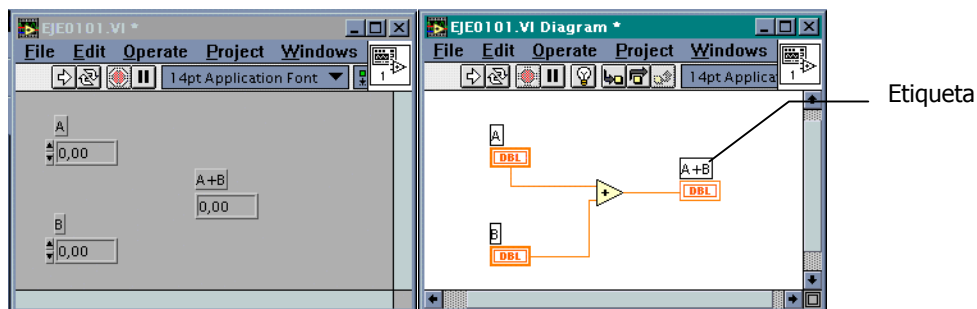
- A medida que los vayamos insertando rellenaremos la casilla de etiqueta para cada uno de ellos.
- Durante la ejecución, aumentaremos el valor de A y B mediante el dedo de la barra de herramientas
- Para mover uno de los elementos insertados podemos hacerlo mediante la flecha de la barra de herramientas, arrastrando el objeto y soltándolo en la posición deseada.

INSERTAR LA OPERACIÓN DE SUMA:

En la ventana de diagrama seleccionar **FUNCTIONS/Numeric/Add**

Distinguiremos entre control e indicador en la parte del diagrama de bloques

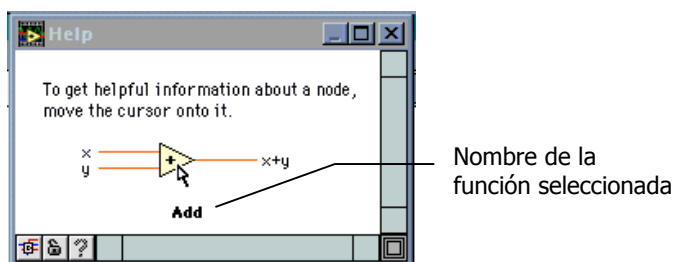
Las uniones que relacionan los elementos con la operación se hacen mediante el elemento **carrete de hilo** de la paleta de herramientas



LLAMADAS A LA AYUDA:

Mediante **CONTROL+H** o seleccionando la opción **Show Help** del menú **HELP**

Situándonos sobre cualquier elemento nos informará de su utilidad y que conexiones necesita.



EJECUCIÓN DEL PROGRAMA

Para ver el resultado de nuestro programa ejecutamos el programa pulsando con el ratón sobre el botón . Esto ejecutará el programa una sola vez. Si cambiamos los valores de los controles digitales no veremos el resultado correcto hasta que lo pulsemos de nuevo

Si pulsamos el botón el programa se ejecutará continuamente, por lo que si cambiamos los valores de los controles el resultado se refrescaba instantáneamente.

Pulsando sobre los botones de abortar o pausa, , respectivamente, podremos detener la ejecución definitiva o temporalmente. Para salir de la pausa volveremos a pulsar sobre ese botón.

Pulsar sobre **CONTROL+B** borra de nuestro diagrama las uniones defectuosas realizadas mediante el carrete de hilo, porque no llevan a ningún sitio o porque unan elementos no relacionables.

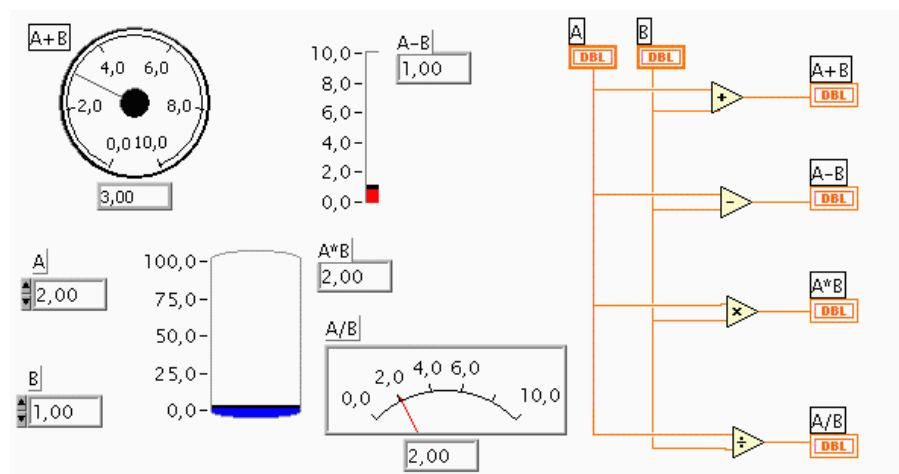
SUSTITUCIÓN DE CONTROLES E INDICADORES

Sustituiremos o reemplazaremos los controles existentes por otros diferentes, como por ejemplo por **CONTROLS/NUMERIC/Horizontal Pointer Slide**.

Obsérvese que solamente cambiamos la parte correspondiente a la interfaz de usuario, no a la funcionalidad.

PRÁCTICA 2: OTRAS OPERACIONES Y CONTROLES

Partiendo de lo aprendido en la práctica 1, realizar las operaciones de suma, resta, multiplicación y división de las entradas A y B utilizando como salidas para los resultados distintos visualizadores, tanque, agujas, etc.

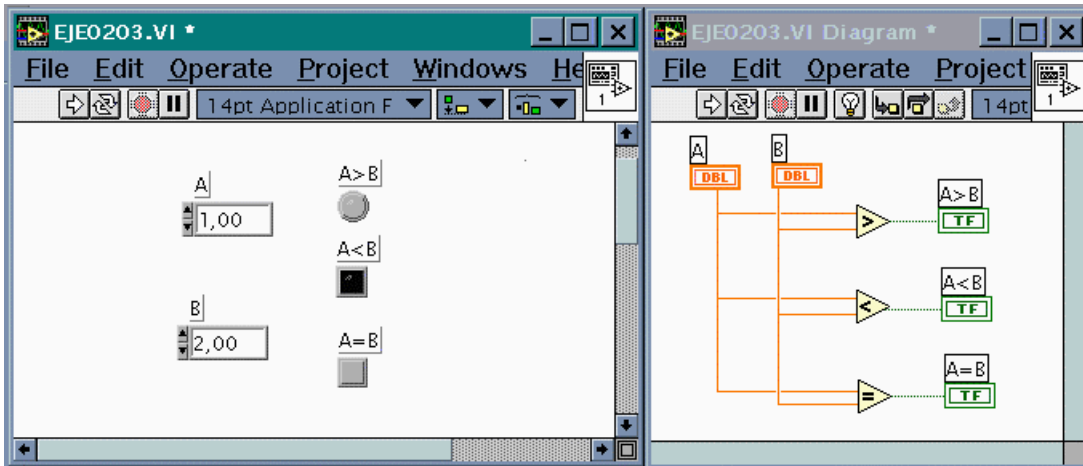


Para cambiar la escala del depósito al valor 100, basta con sobrescribir el valor máximo con la herramienta de escritura


Podemos realizar el cambio entre las distintas herramientas, 'flecha', 'mano', 'carrete de hilo' con la pulsación del tabulador y el espaciador.

INTRODUCIR FUNCIONES DE COMPARACIÓN:

Las funciones de comparación se encuentran en **FUNTIONS/COMPARISON/GREATER**.
 Introducir Leds como resultado de las comparaciones, **CONTROLS/BOOLEAN**.



Cambio de colores: podemos modificar las propiedades de color de la mayoría de los elementos del panel de control con la herramienta **pince!**

La alineación de las partes de un diagrama y del panel de control se realiza mediante las listas desplegables  de la barra de herramientas.

TIPOS DE DATOS:

Existen 12 representaciones para los controles o indicadores digitales:

- Precisión simple: 32 bits (SGL).
- Precisión doble: 64 bits (DBL).
- Precisión extendida (EXT): números de coma flotante.
- Número entero con signo (I8) de tipo byte (8 bits).
- Número entero sin signo (U8) de tipo byte (8 bits).
- Número entero con signo (I16) de tipo palabra (16 bits).
- Número entero sin signo (U16) de tipo palabra (16 bits).
- Número entero con signo (I32) de tipo entero extendido (32 bits).
- Número entero sin signo (U32) de tipo entero extendido (32 bits).
- Complejos de precisión simple (CSG).
- Complejos de precisión doble (CDB).
- Complejos de precisión extendida (CXT): números complejos de coma flotante.

Los límites, máximo y mínimo, dependen del tipo ; un entero con signo (8 bits) estará entre los valores de -128 a 127.

Precisión:	Simple	Doble	Extendida
Número positivo máximo	3,4 E38	1,7 E308	1,1 E4932
Número positivo mínimo	1,5 E-45	5,0 E-324	1,9 E-4951
Número negativo mínimo	-1,5 E-45	-5,0 E-324	-1,9 E-4951
Número negativo máximo	-3,4 E38	-1,7 E308	-1,1 E4932

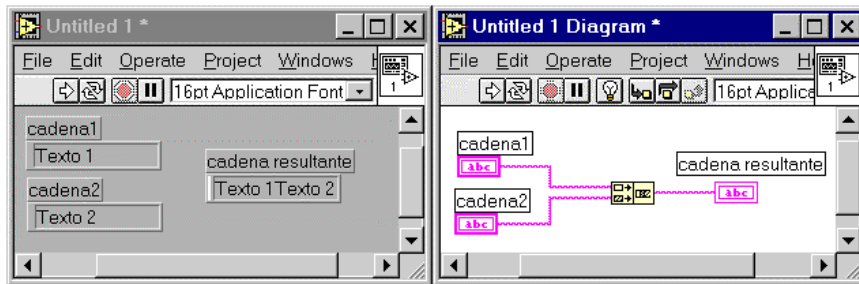
Para realizar el cambio de la precisión de un dato invocaremos el menú contextual pulsando el botón derecho cuando estemos sobre él y seleccionando el submenú de **REPRESENTATION** del menú emergente.

CONCATENACIÓN DE CADENAS ALFANUMÉRICAS FRENTE A SUMA DE NÚMEROS.

La función equivalente a la suma de números es la concatenación de caracteres, que da por resultado una única cadena formada por otras simples.

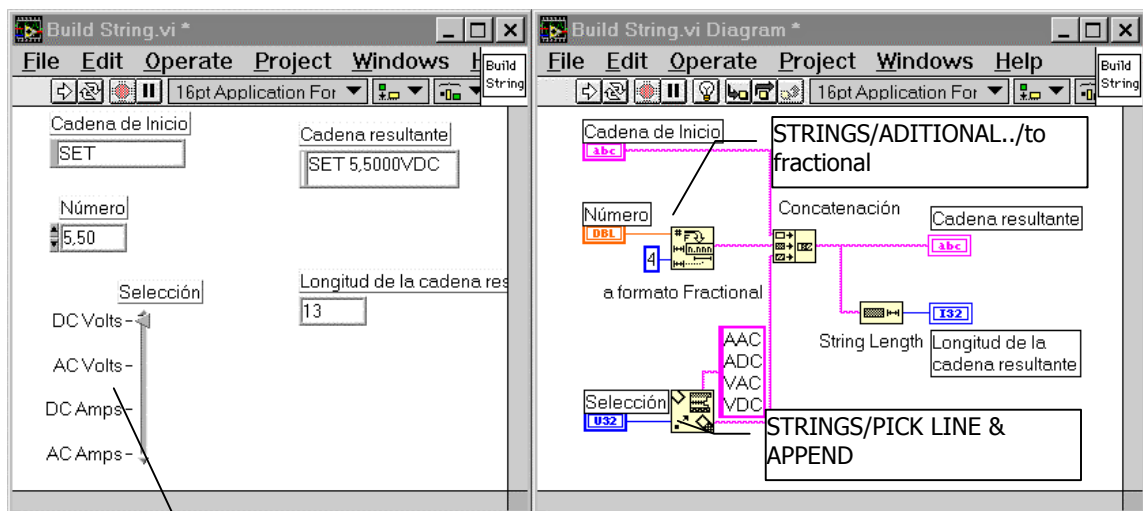
Paso A

La concatenación básica corresponde con este ejemplo, donde usamos la función **Funcions/strings/concatenate strings**



Paso B

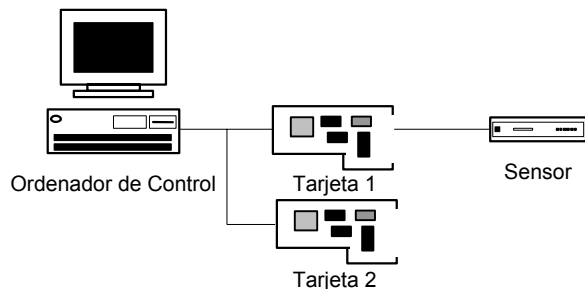
En el ejemplo siguiente concatenamos varios elementos; algunos de los cuales son el resultado de la conversión de números a cadenas de caracteres.



Para realizar un selector como el de la imagen debemos usar la propiedad **TEXT LABEL** en el menú contextual del selector. Una vez hecho esto, podremos añadir nuevos textos al selector, pulsando en el menú contextual que aparece en su display sobre las opciones **ADD ITEM BEFORE** ó **ADD ITEM AFTER**.

PRÁCTICA 3ª: ADQUISICIÓN Y PROCESAMIENTO BÁSICOS

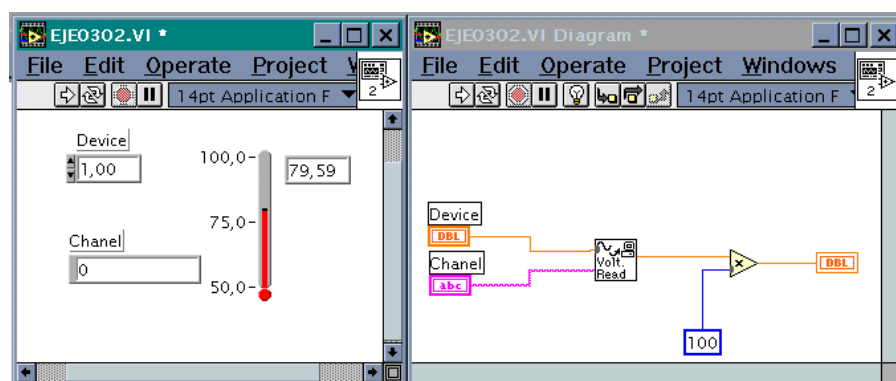
La adquisición aunque simulada correspondería con un esquema hardware donde un sensor está conectado a uno de los canales de una de las tarjetas de adquisición de datos disponibles en el ordenador. El esquema sería el siguiente:



Insertar la función : **FUNCTIONS/TUTORIAL/Demo voltaje Read.VI**

Este VI solo funciona si seleccionamos los valores **0** ó **1** en el control **Device** y el control **Chanel** dando error en cualquier otro caso. (i.e. Device 0 = Tarjeta 1, Device 1 = Tarjeta 2).

TÉCNICAS DE DEPURACIÓN:



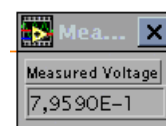
Propuesto: Dibujar la curva de calibración del sensor.

Seleccionando el botón de la ventana de diagrama se ejecutará la aplicación en modo depuración.

Podemos insertar una **punta de prueba** en cualquier cable del diagrama para visualizar el valor en dicho punto.

- Para crear esta punta de prueba, seleccionamos la herramienta del menú de herramientas, pulsando con el ratón en aquellos puntos de cable donde deseemos saber el valor de la variable.

- En el momento de ejecución nos aparecerá en el diagrama una ventana con el valor.



Recordemos que para obtener ayuda sobre una determinada función podemos activar la ayuda. Se la llama mediante CONTROL-H, y al situarnos sobre un elemento nos da la información referida a él, conexiones y tipos de datos que utiliza.

CREACIÓN DE SUBVI'S

Son el equivalente a las subrutinas en los lenguajes de programación basados en texto. Con ellos conseguimos una programación mejor estructurada y por tanto más legible así como evitar la repetición de código. La diferencia respecto a las tradicionales subrutinas es que un subVI puede ejecutarse de forma autónoma sin necesidad de que esté incluido en un VI.

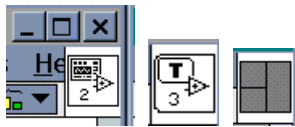
Al hacer doble click sobre un subVI se abre el panel de control correspondiente a dicho subprograma.

Para salvar un VI como fichero lo haremos mediante las distintas opciones del menú **FILE, SAVE** y **SAVE AS**. Debemos poner nosotros la extensión **.VI** ya que por defecto el programa no añade extensión alguna.

Crear un subVI

Partiendo de un VI como el del ultimo ejemplo, crearemos un subVI para utilizarlo posteriormente en otras partes de nuestros VI's. Tendremos que definir unas entradas y unas salidas para posteriormente efectuar las conexiones en el diagrama.

Haciendo doble click (doble pulsación) sobre el icono de la parte superior derecha del panel de control podemos editar el dibujo que identificará a nuestro subVI.



Si una vez modificado y dejado como definitivo, pulsamos con el botón derecho y elegimos la opción **Show conector** veremos la disposición de conexiones de nuestro subVI.

Pulsar el botón derecho para ver los distintos tipos de **patterns** (plantillas de conexión) y elegir el que se adapte a nuestras entradas y salidas; normalmente las entradas estarán a la izquierda y las salidas a la derecha.

En la ventana panel de control seleccionar la posición dentro del **pattern** y con el carrete de hilo el control correspondiente de entrada o salida. Si esta seleccionado como entrada o salida habrá cambiado del blanco al gris en el pattern elegido por nosotros.

Una vez realizadas las conexiones y modificaciones del icono podemos guardarlas.

PRÁCTICA 4ª: UTILIZACIÓN DE SUBVI'S

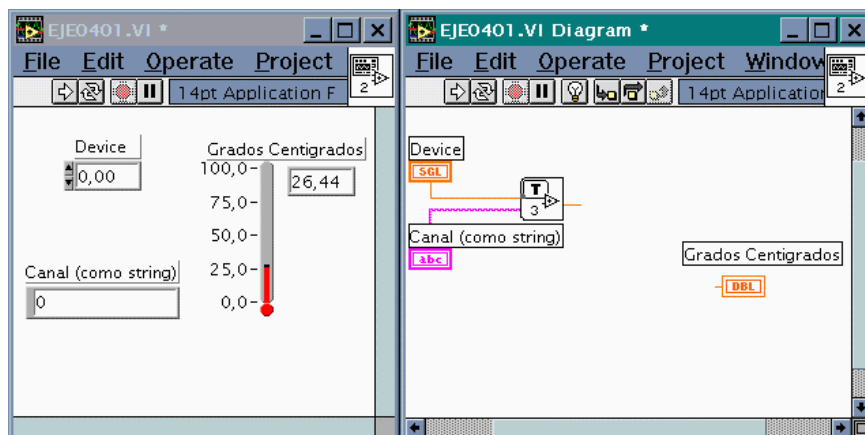
Partiendo del subVI que acabamos de crear y que muestra la temperatura en grados Fahrenheit, realizar un VI que visualice la temperatura en grados centígrados.

Introducir el subVI creado en la práctica 3ª

Lo haremos mediante el botón de la paleta de funciones '**Select a VI...**'. Lo seleccionaremos de la lista de ficheros y lo pegaremos en la ventana **diagrama** de nuestro instrumento virtual.

Completar el diagrama de bloques teniendo en cuenta que:

$$T^a (^{\circ}C) = [T^a (^{\circ}F) - 32] * 5/9$$




OPCIONAL:
Añadir un indicador de grados Kelvin
 $^{\circ}K = ^{\circ}C + 273$

OPCIONAL:
Crear un subVI con tres salidas: $^{\circ}C$, $^{\circ}F$ y K.

En vez de crear manualmente un control para cada entrada y un indicador para cada salida, es posible realizar esta tarea automáticamente:

- Mediante el carrito de hilo, pulsar con el botón derecho sobre la entrada o salida y elegir la opción **create control** o **create indicator**.
- Según el caso el programa crea automáticamente el tipo necesitado incluso con su etiqueta.

Si no hemos puesto la etiqueta a algún elemento deberemos seleccionar el elemento con el botón derecho y elegir la opción **show label**, una vez en pantalla introducir el valor deseado.

Siguiendo la ejecución del programa en modo de depuración  podemos ver como las funciones esperan de izquierda a derecha a que se vayan generando los datos que necesitan para completar la operación.

Un SubVI puede llamar a múltiples VI's dando lugar a una jerarquía que podemos visualizar gráficamente. Para ello existe la opción **show VI hierarchy** del menú **Project**.

PRÁCTICA 5ª: SECUENCIA WHILE-LOOP

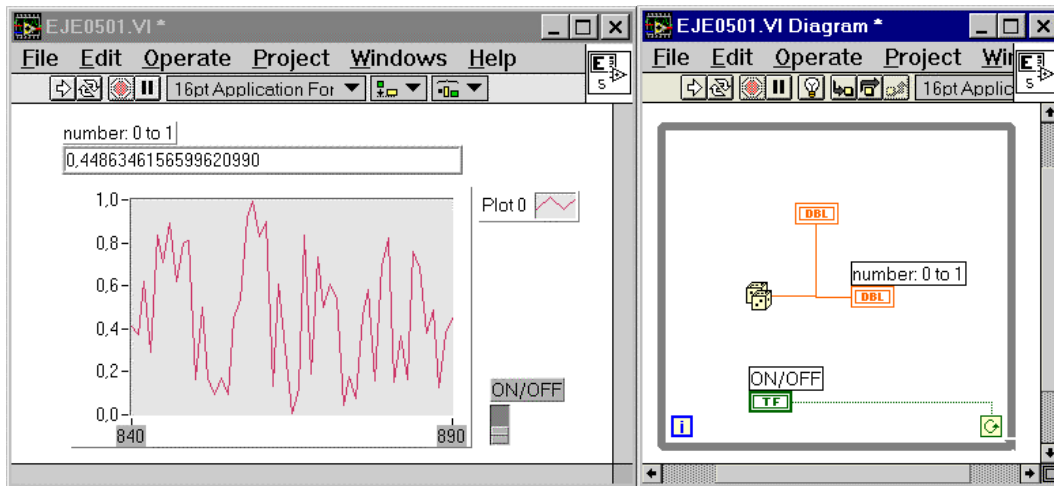
La secuencia **While-loop** permite repetir las acciones que situemos dentro del bucle hasta que deje de cumplirse una condición que nosotros establezcamos.

PASO A

Como ejemplo generaremos un número aleatorio entre 0 y 1 mediante la función **FUNTIONS/NUMERIC/Ramdom Number**. El resultado lo representaremos mediante una gráfica en el panel de control: **CONTROL/GRAPH/Wave form Chart**.

La ejecución debe parar cuando el usuario pulse un botón **ON/OFF**.

Para ello, situaremos la función dentro de un bucle While que se ejecutará como mínimo 1 vez hasta que la condición procedente del botón sea **FALSE**.

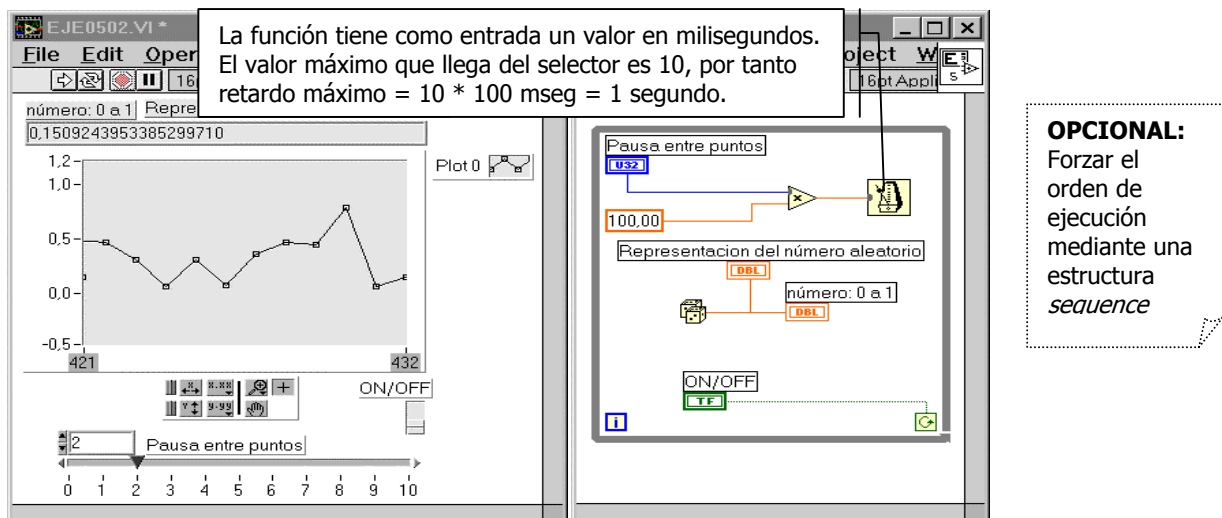


El número de decimales que aparecen en el visualizador puede ser modificado mediante la propiedad asociada **FORMAT & PRECISION** de su menú contextual.

- Diferenciaremos entre la ejecución única del programa, que se detendrá cuando pongamos en **OFF** el botón asociado al control del bucle y la parada en la ejecución mediante el botón de **STOP**.

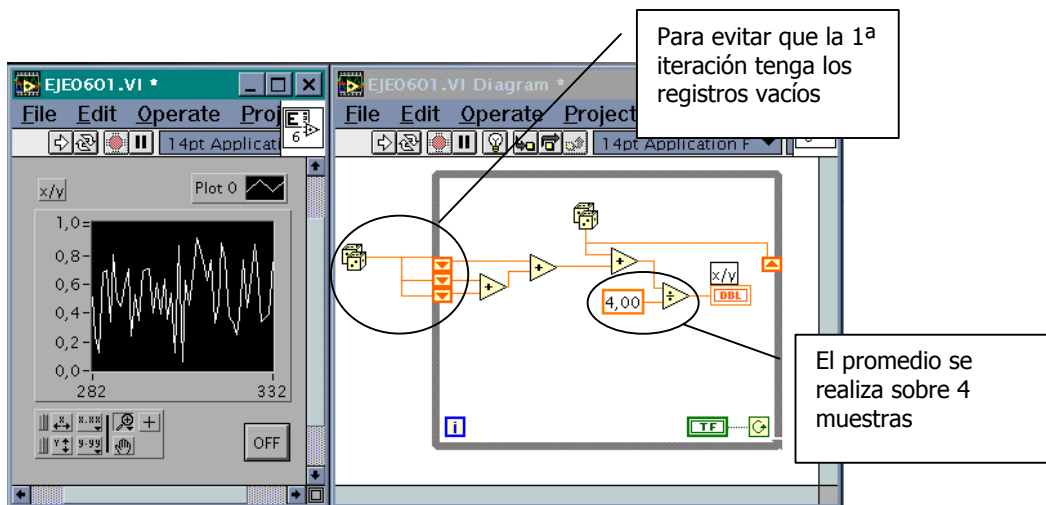
PASO B

Podemos incluir un retardo entre la generación de uno y otro punto mediante la función **FUNTIONS/Time & Dialog/Wait Until Next ms Multiple**, tal y como puede verse en la siguiente figura:



PRÁCTICA 6ª: REGISTROS DE DESPLAZAMIENTO

Veremos la aplicación de los registros de desplazamiento (*Shift registers*) en el cálculo de promedios.




PASO A

Añadir los registros de desplazamiento:

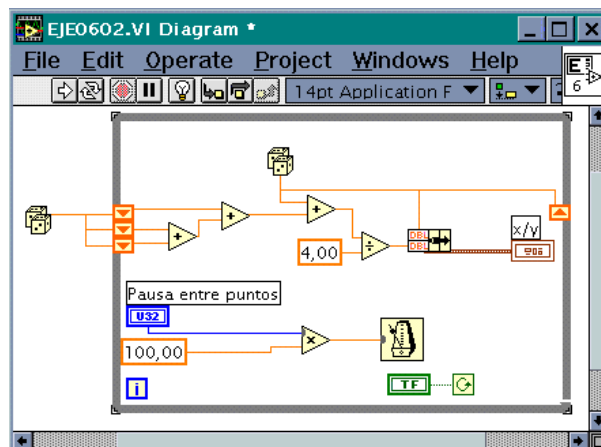
En la zona izquierda del bucle pinchar con el ratón y pulsar el botón derecho; seleccionar **Add Shift Register**. Después en la zona derecha pinchar y seleccionar **Add Element**

Para comprender mejor el efecto del registro de desplazamiento activaremos el modo de depuración, para ver como evolucionan los valores.

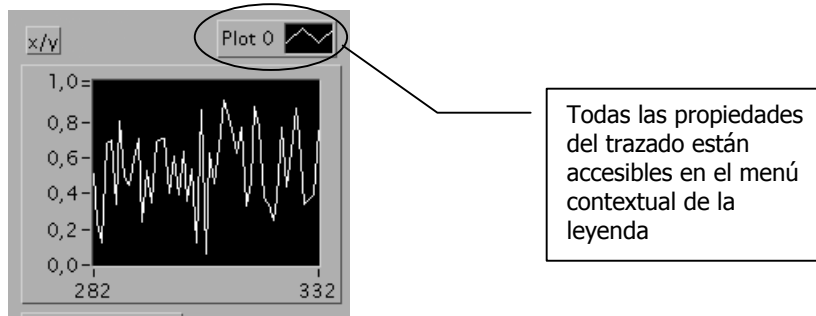
Es importante reseñar la diferencia entre parar un programa correctamente, una vez acabada las acciones contenidas en un bucle o abortar la ejecución mediante el botón  sin ningún control sobre las acciones que han sido ejecutadas y las que han quedado pendientes.

PASO B

Para visualizar simultáneamente la grafica correspondiente a la media y al valor original; lo haremos mediante la opción **FUNCTIONS/CLUSTER/BUNDLE**.

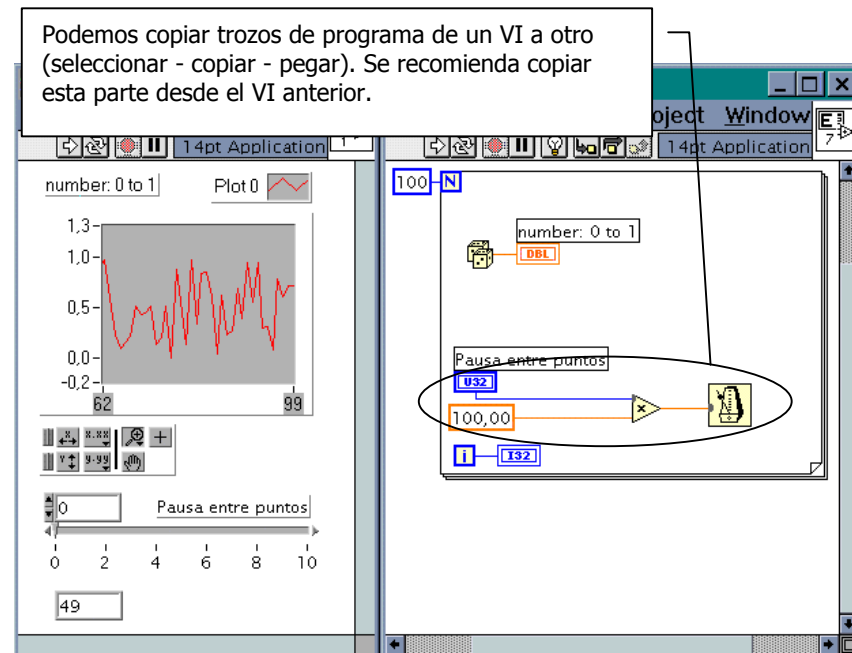


Cambiar el color, grosor y tipo de trazo de las gráficas:



PRÁCTICA 7ª: BUCLE FOR.

Un bucle For (**For-loop**) permite la ejecución del código que situemos dentro del bucle un número de veces predeterminada.



Contestar a las siguientes preguntas:

- ¿Cuál es el rango de valores de $N = 0$ a 99 ó de 0 a 100 ?
- ¿Aumenta i y después tira el dado, o primero tira el dado y después aumenta i ?

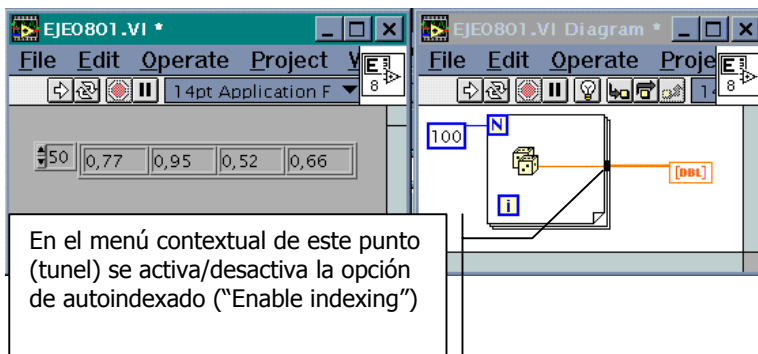
PRÁCTICA 8ª: MATRICES.

PASO A

Construiremos una matriz que contendrá las 100 tiradas del dado. Para ello, en el panel de control:

- Primero, insertar un visualizador de tipo *array*: **CONTROL/ARRAY & CLUSTER/Array**.
- Después debe introducirse un visualizador numérico dentro del control *array* (soltándolo encima).

Los elementos del array se almacenarán en el túnel (punto negro en el marco de la estructura for-loop) si tenemos habilitada la opción de autoindexado, sino sólo el último dato será pasado al exterior del bucle.



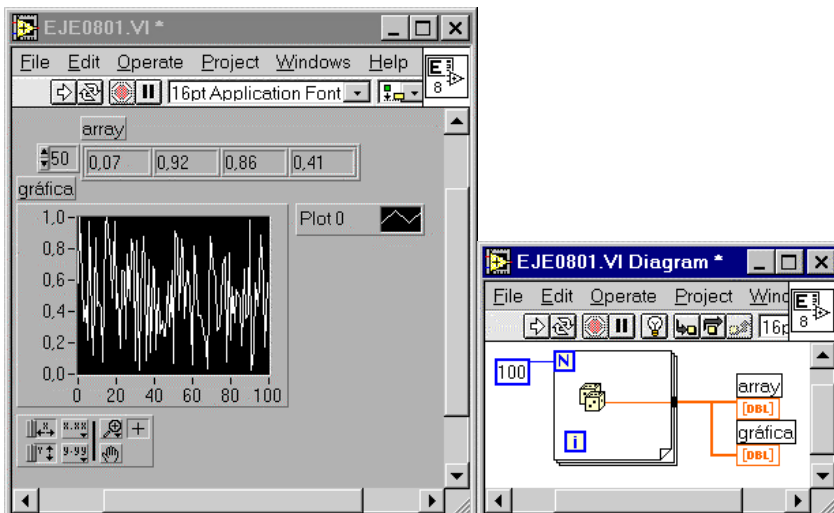
OPCIONAL:

Realizar el promedio de los 100 puntos mediante un registro de desplazamiento.

Ojo, debe inicializarse el registro a cero.

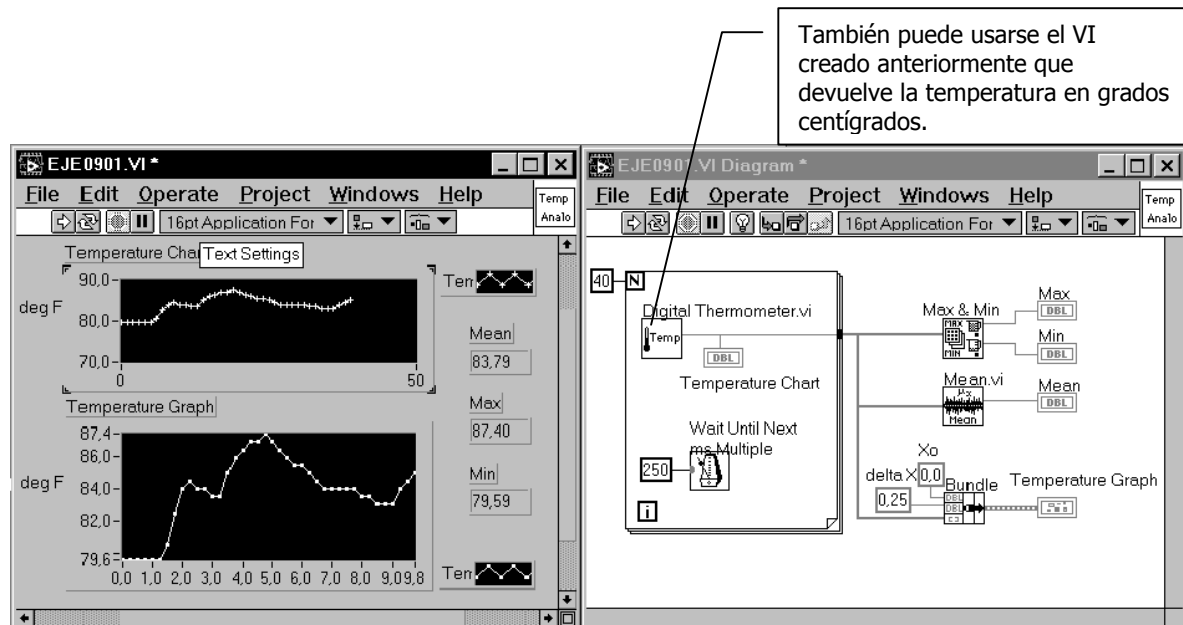
PASO B

Añadir un gráfico **CONTROLS/GRAPH/Wave form Graph** para la visualización gráfica del array. Quedará como se muestra en el panel y diagrama siguientes:



PRÁCTICA 9: TIPOS DE GRÁFICOS

Comparar la diferencia entre los gráficos **CHART** y los **GRAPH**, calculando la media, máximo y mínimo de las temperaturas generadas aleatoriamente por el subVI denominado **Digital Thermometer.vi** de la librería **Tutorial**.



- Observar la diferencia en el grosor del cable del array que sale del bucle FOR.
- Las funciones para encontrar el máximo y el mínimo del array se encuentran en **FUNTIONS/ARRAY/Array Max & min**
- Para calcular la media lo haremos mediante la función **FUNTIONS/ANALISIS/Probability & Statistics/Mean**
- Para hacer el **BUNDLE** mas grande de forma que permita conectar más elementos, situamos el puntero en alguna de las esquinas y estiramos hacia abajo.

Tener en cuenta que incluimos 2 tipos de gráficos, *Graph* y *Chart*. Para dar sentido temporal al eje x de la gráfica se debe realizar un bundle con tres elementos: la matriz de puntos, su separación en el tiempo y el origen. Como hemos fijado pausas de 250 ms entre muestras entonces delta X es 0'25 (segundos). Xo establece el origen del eje x, en este caso 0.

- ¿Cuál es la frecuencia de muestreo?

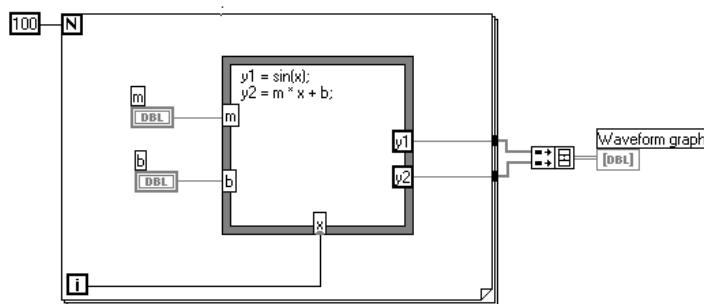
PRÁCTICA 10: NUDO FÓRMULA.

Permite incorporar a un VI fórmulas matemáticas definidas por el usuario.

Lo insertamos con **FUNTIONS/STRUCTURES/FORMULA NODO**

Para añadir entradas o salidas lo haremos pulsando con el botón derecho del ratón en el marco del nodo fórmula y seleccionando **add input** o **add output**.

Las fórmulas incluidas (todas las que necesitemos para calcular las salidas) deben terminar en punto y coma (;). Los resultados de cada fórmula dan lugar a una matriz unidimensional. La unión de esas dos matrices da lugar a una matriz de 2 dimensiones. Esa conversión se realiza mediante la función **FUNTIONS/ARRAY/BUILD ARRAY**



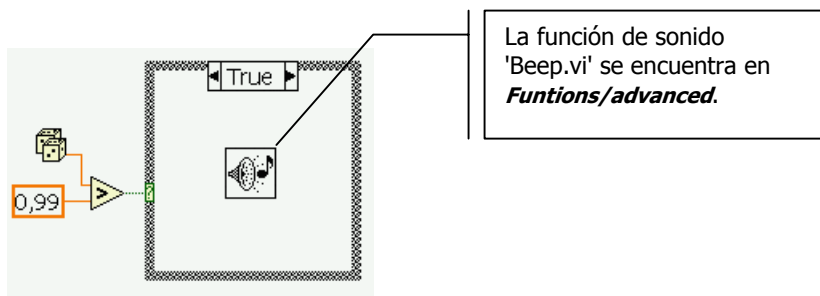
OPCIONAL:
Hallar el valor RMS de la función senoidal mediante el VI *Functions/analysis/probability and statistics/RMS*

- Modificar el programa de forma que el usuario pueda variar la amplitud de la senoide.

Un ejemplo mas completo y vistoso del uso de la estructura **formula-node** puede verse en la zona de ejemplos, *Examples/General/Graphs/gengraph.llb* en el instrumento virtual llamado **bouncing cube.vi**

PRÁCTICA 11: ESTRUCTURA CASE

Esta estructura de programación permite que se ejecute una porción de código u otra según se cumpla o no una condición impuesta por el usuario.



Para evitar que pite de forma continua, cambiar la constante de comparación a un valor **0,999**

OPCIÓN MÚLTIPLE DE LA ESTRUCTURA CASE.

Aunque en el ejemplo sólo existen el caso verdadero y falso, pueden ponerse tantos casos como se necesiten:

- Para añadir un caso más, ya sea antes o después deberemos seleccionar pulsando con el boton derecho del ratón en el borde **Add case before** ó **Add case After** respectivamente.
- Evidentemente la entrada al selector ya no podrá ser binaria, sino que deberá corresponder a un número entero.

4.- ADQUISICIÓN DE DATOS.

CONCEPTOS BÁSICOS DE ADQUISICIÓN DE DATOS (A/D)

VELOCIDAD DE MUESTREO

Teorema de Nyquist

Establece que para poder reconstruir una señal muestreada, la velocidad de muestreo f_s debe ser al menos el doble de la mayor de las componentes de la señal muestreada:

$$f_s \geq 2 * f_{max}$$

En la práctica se toma al menos entre 3 y 5 veces mayor que la frecuencia máxima. Como explicaremos en el próximo punto, esto evita que las frecuencias superiores cercanas a la máxima produzcan aliasing.

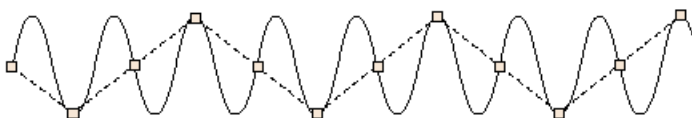
Ej.: Las señales de audio recogidas por un micrófono y convertidas a señal eléctrica tienen (idealmente) componentes de hasta 20 KHz. Para digitalizar esta información y poder reconstruirla después se debe muestrear a una velocidad mayor de 40 Kmuestras/s. (Los Compact Disc lo hacen a la velocidad normalizada de 44,8 Kmuestras/s)

Aliasing (solapamiento)

EN EL DOMINIO DEL TIEMPO:



Señal muestreada a frecuencia superior a la de Nyquist.

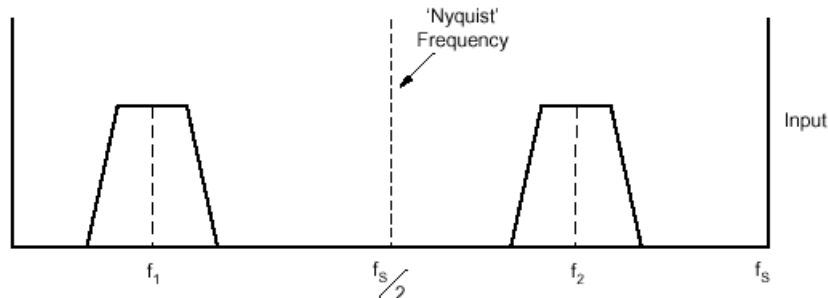


Señal muestreada a frecuencia inferior a la de Nyquist, aparece el *aliasing*: Al reconstruir la señal se obtiene otra frecuencia inferior.

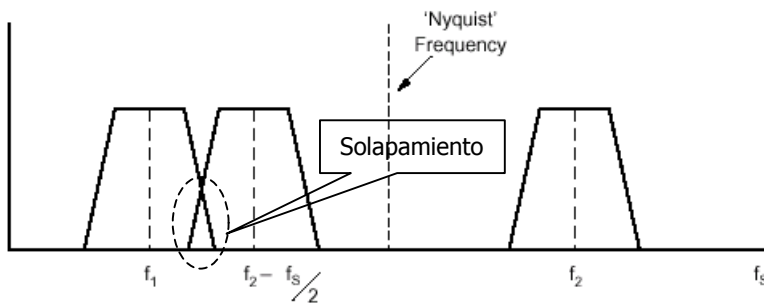
En la imagen anterior, si la señal original fuera de 3 KHz, ¿Qué frecuencia observaríamos tras reconstruir la señal muestreada?

EN EL DOMINIO DE LA FRECUENCIA:

Supongamos una señal de interés f_1 y otra interferente f_2 como pueden verse en la siguiente figura:



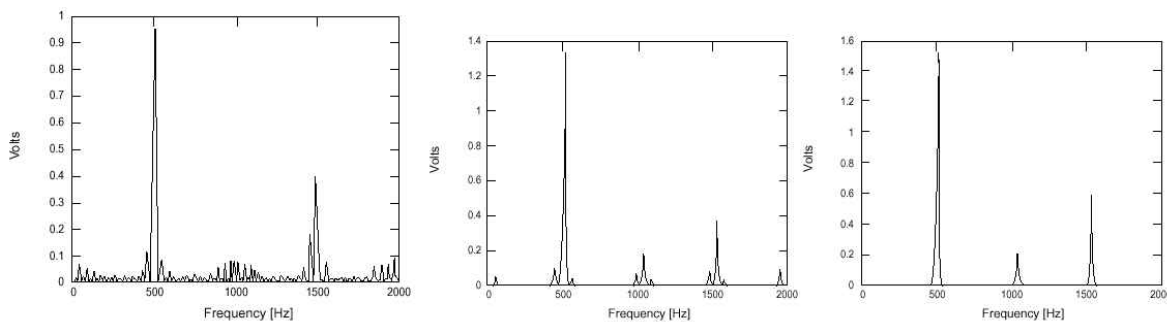
Cuando una señal de frecuencia f_x se muestra a una velocidad f_s , su espectro aparece reflejado en $f_x - f_s/2$ como puede apreciarse en la siguiente figura:



En este caso, al muestrear la señal f_x , el espectro reflejado $f_x - f_s/2$ queda a la izquierda del origen de frecuencias, por lo que no hay problema. Sin embargo al muestrear la señal f_2 su espectro reflejado $f_2 - f_s/2$, se solapa con f_1 confundiendo con él. De esta forma, la señal f_2 interfiere con la señal útil f_1 .

Ejemplo: señal cuadrada de 500 Hz:

Únicamente nos fijaremos en un ancho de banda limitado a 2 KHz, por lo que idealmente, además de la fundamental sólo debería existir un armónico a 1500 Hz.



- En la **primera imagen** la señal es muestreada a $f_s=4\text{KHz}$ y no se ha filtrado. Observamos *aliasing* de los armónicos superiores.
- En la **segunda imagen**, $f_s=4\text{KHz}$ y previamente al muestreo se realizó un filtrado paso-bajo con frecuencia de corte de 2 KHz que ha eliminado muchos de los picos debidos al *aliasing*.
- En la **tercera imagen**, además del filtrado se ha subido $f_s=8\text{KHz}$ eliminando los picos. El armónico de 1000 Hz es debido a la imperfección de la señal cuadrada.

El filtrado debe ser previo al muestreo (i.e. filtrado analógico). Después del muestreo, el *aliasing* no puede eliminarse; la señal *alias* no puede separarse de la señal útil mediante ningún tipo de filtro, pues cae en su ancho de banda.

RESOLUCIÓN

Se refiere al incremento mínimo de tensión detectable, que coincidirá con el valor del bit menos significativo (**LSB**). Sin embargo, en tarjetas de adquisición de datos la resolución suele expresarse como el número de bits del convertor A/D.

Una resolución de 12 bits de un convertor A/D indica que es capaz de representar 4096 combinaciones binarias, es decir, 1 parte entre 4096 ($2^{12}=4096$).

LSB (LEAST SIGNIFICANT BIT)

Cuando hablamos de convertidores A/D, el valor del bit menos significativo recibe el nombre de LSB.

$$1LSB = \frac{\text{Rango}}{n^{\circ} \text{ cuentas}} = \frac{\text{Rango}}{2^{n^{\circ} \text{ bits}}} \text{ (Voltios)}$$

Si tenemos en cuenta que antes del convertidor A/D casi siempre hay un amplificador, debemos incluir su ganancia en el cálculo del **LSB**.

$$1LSB = \frac{\text{Rango}}{\text{Ganancia} \cdot 2^{n^{\circ} \text{ bits}}}$$

donde el n° de bits se refiere al convertidor A/D, y el rango a la diferencia entre el valor máximo y mínimo de tensión admitido en la entrada.

P.ej. 12 bits → 10V en 4096 niveles → 1 LSB = 2'44 mV (Ganancia=1)

➤ **Hallar la resolución de un convertidor de 16 bits**

RANGO

Como puede verse en la siguiente tabla, el rango de entrada viene determinado por la ganancia seleccionada.

Tabla 1: Configuraciones de entrada para la serie PCI E de National Instruments

Range Configuration	Gain	Actual Input Range	Precision ¹
0 to +10 V	1.0	0 to +10 V	152.59 μV
	2.0	0 to +5 V	76.29 μV
	5.0 ²	0 to +2 V	30.52 μV
	10.0	0 to +1 V	15.26 μV
	20.0 ²	0 to +500 mV	7.63 μV
	50.0 ²	0 to +200 mV	3.05 μV
	100.0	0 to 100 mV	1.53 μV
-10 to +10 V	1.0	-10 to +10 V	305.18 μV
	2.0	-5 to +5 V	152.59 μV
	5.0 ²	-2 to +2 V	61.04 μV
	10.0	-1 to +1 V	30.52 μV
	20.0 ²	-500 to +500 mV	15.26 μV
	50.0 ²	-200 to +200 mV	6.10 μV
	100.0	-100 to +100 mV	3.05 μV

¹ The value of 1 LSB of the 16-bit ADC; that is, the voltage increment corresponding to a change of one count in the ADC 16-bit count

² Not available on the PCI-MIO-16XE-50

Note: See Appendix A, Specifications, for absolute maximum ratings.

Observar en la tabla que:

- Donde dice **precision** debería decir **resolution**. La precisión máxima coincide con el valor de 1/2 LSB, pero no siempre es así debido a tensiones de offset, ruido, derivas térmicas, etc.
- El producto **Ganancia * Rango de entrada** es constante: **Rango configurado = Ganancia * Rango de entrada**.

Configuración

El rango de las señales de entrada puede ser **UNIPOLAR** o **BIPOLAR**

- UNIPOLAR: el rango de tensión de entrada está entre 0 y un valor positivo
- BIPOLAR: el rango de tensión de entrada está entre un valor negativo y un valor positivo

Se puede programar la polaridad y el rango de forma que cada canal tenga una configuración propia.

Hay que seleccionar el rango (Polaridad y ganancia) de forma que se ajuste al máximo al rango de la señal a medir obteniendo así la MAYOR RESOLUCIÓN posible. (1 LSB = $\frac{\text{Rango}}{\text{Ganancia} \cdot 2^N}$ Voltios)

Rango dinámico

Expresa la diferencia máxima de magnitud que puede haber entre dos señales de entrada de forma que ambas puedan medirse. Suele expresarse en decibelios (**dB**).

Ejemplo: DAQ 12 bits, 10 V

Máx. tensión medible: 10 V

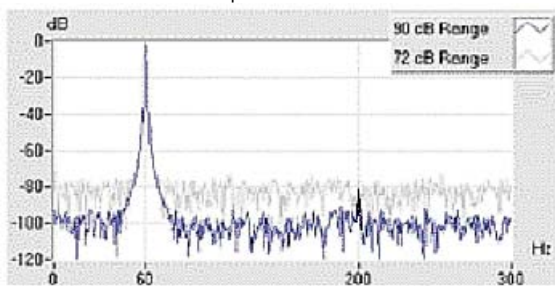
Mín. tensión medible: 1 LSB = $10\text{V}/2^{12} = 2'44$ mV

Rango dinámico = $20 \log_{10} (10 \text{ V} / [10\text{V}/2^{12}]) = 20 \log_{10} (2^{12}) = 72$ dB

- Cada bit de resolución añade 6 dB (teóricos) al rango dinámico ($20 \log_{10} 2 = 6$)

Ejemplo: máquina rotativa

En una máquina rotativa, hay una frecuencia que determina la velocidad de rotación del eje y otras componentes pequeñas que pueden delatar un deterioro de los cojinetes. La relación entre las componentes pequeñas y la principal determina el mínimo rango dinámico. Las componentes pequeñas aumentan con el uso, de forma que un mayor rango dinámico del sistema de medida puede significar una detección más temprana.



En la figura puede verse:

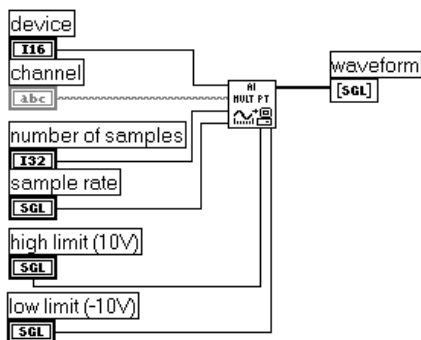
- En trazo claro con un rango dinámico de 72 dB (12 bits) una pequeña componente debida a una vibración de 200 Hz pasa desapercibida.
 - En trazo oscuro con un rango dinámico de 90 dB (15 bits) se puede observar dicha componente.
-

PRÁCTICA 12: ADQUISICIÓN DE DATOS

FUNCIONES RELACIONADAS CON LA ADQUISICIÓN.

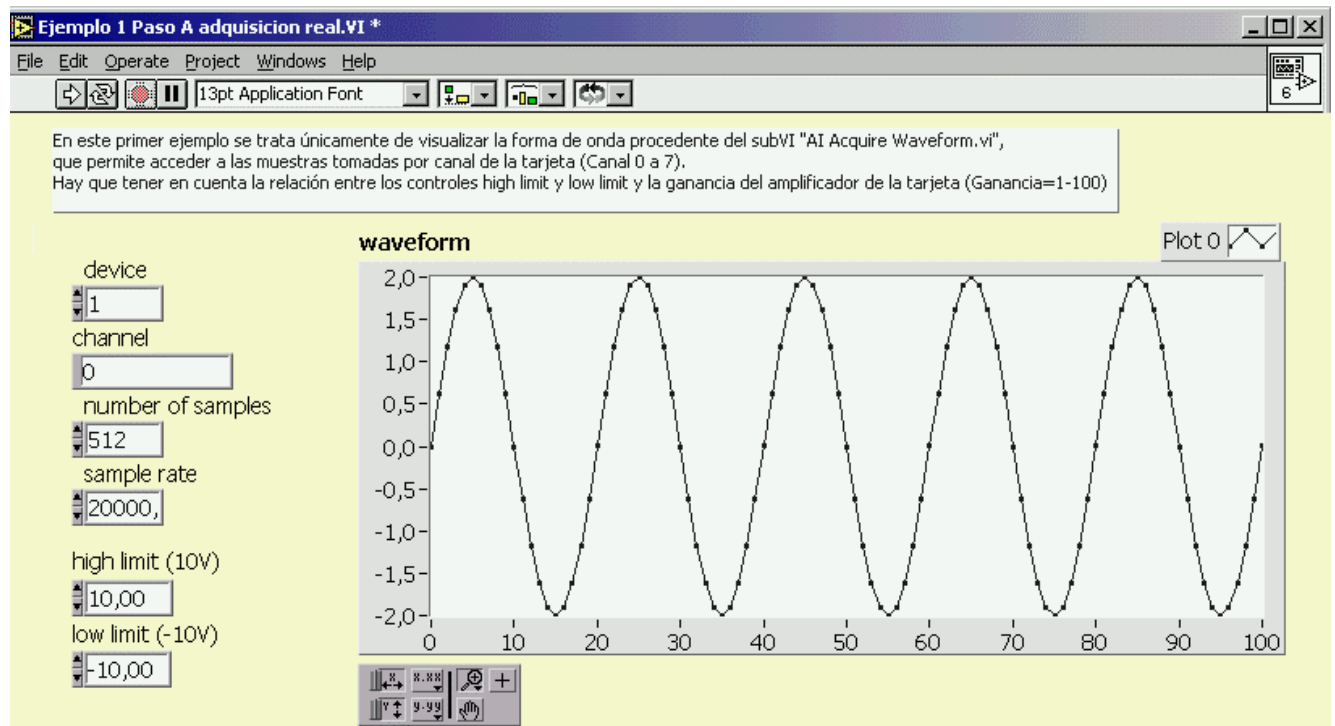
LabVIEW posee una biblioteca completa para el acceso al hardware de adquisición de datos. Esta biblioteca está subdividida en funciones *básicas* y *avanzadas* según su complejidad.

PASO A: USO DE LA FUNCIÓN "AI ACQUIRE WAVEFORM SIMU.VI"



La función necesaria es:

Data Acquisition/Analog Input/AI Acquire waveform.VI.

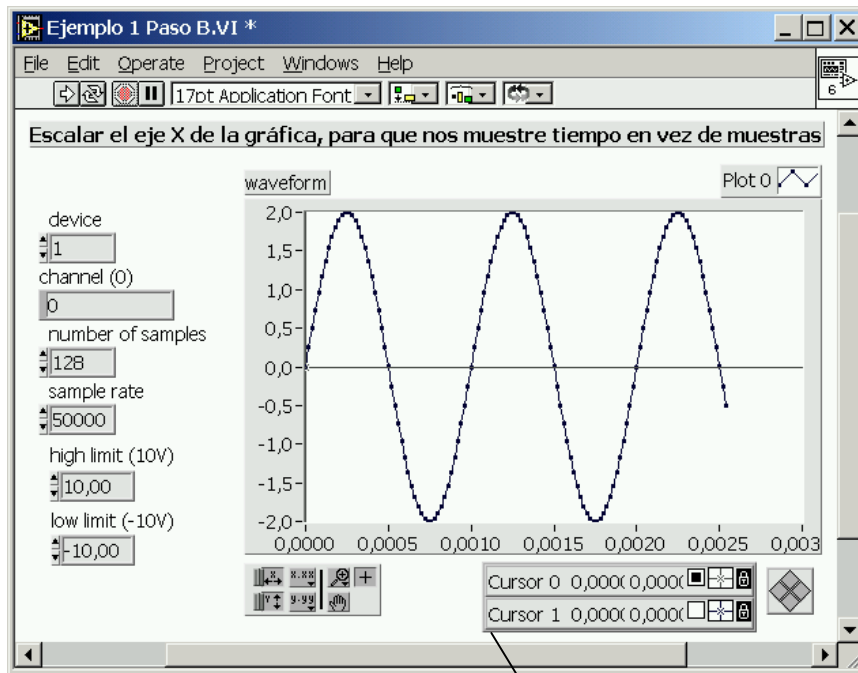


En la imagen se observa la señal obtenida si conectáramos, por ejemplo un generador de funciones al canal 0 de la tarjeta 1.

PASO B: ESCALADO DE TIEMPOS.

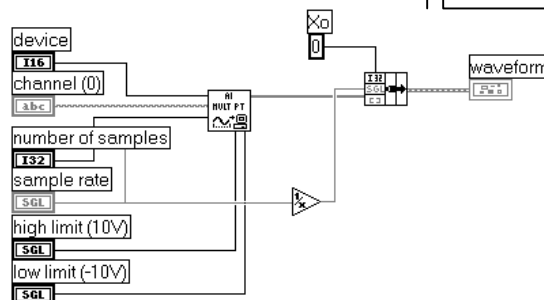
En esta segunda parte del ejercicio aprenderemos a escalar el eje X de la gráfica, para que nos muestre tiempo en vez de muestras.

Utilizar los cursores para determinar los tiempos y frecuencias de la señal visualizada



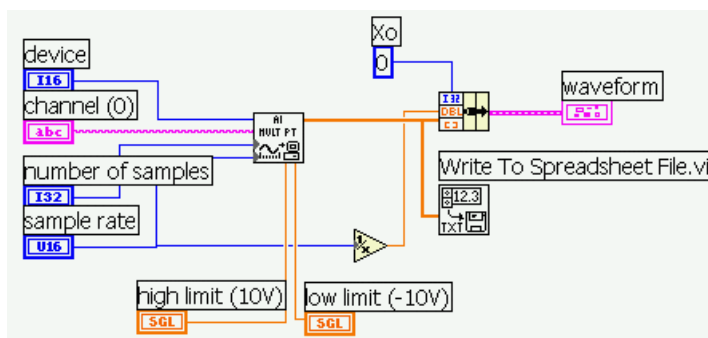
Las opciones de cursores se obtienen del menú contextual de la gráfica.

Ejemplo 1 Paso B



OPCIONAL:

Guardar los datos adquiridos en un fichero con formato de hoja de cálculo:



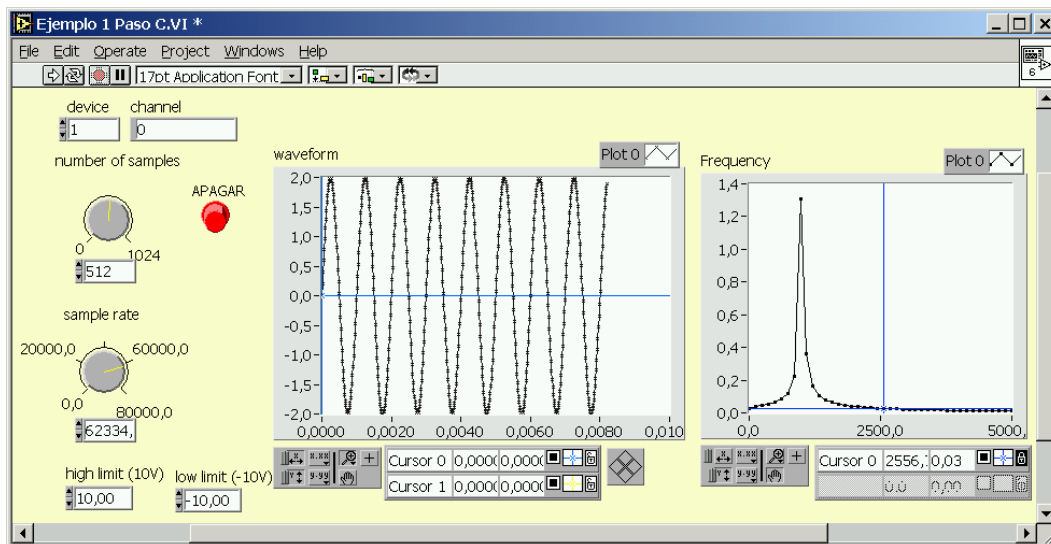
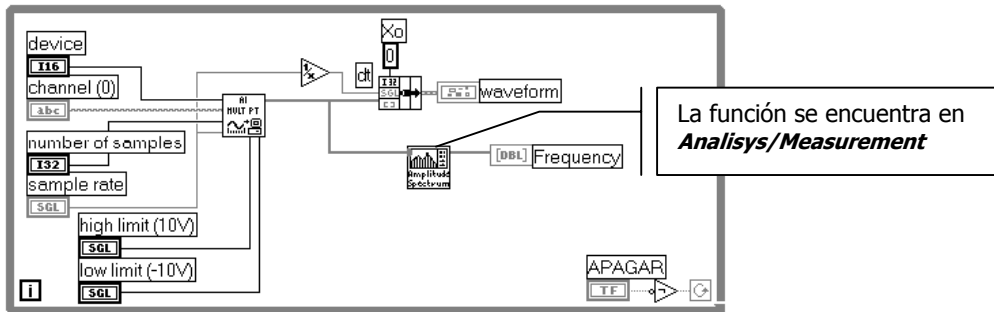
El VI que hemos añadido se encuentra en el grupo de funciones **File I/O**

PASO C: ANÁLISIS EN FRECUENCIA

En esta tercera parte del ejercicio realizaremos las siguientes tareas:

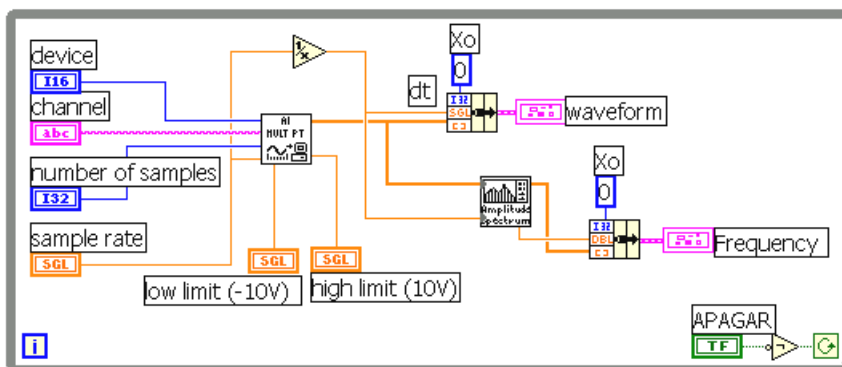
- Intercalar un subVI de análisis que realiza el espectro de amplitud. (Librería: **Analisis/Measurement**)
- Meter el grueso del programa dentro de un bucle While controlado por un botón de paro (Acción mecánica **Latch When Released**)
- Cambiar el número de muestras y la velocidad de muestreo (sample rate) mediante controles tipo potenciómetro observando su efecto sobre la señal visualizada.
- Observar el tipo de acción mecánica para el botón **APAGAR**.

Ejemplo 1 Paso C



PASO D: ESCALADO DE FRECUENCIA

De forma análoga a lo que se hizo para el eje de tiempos, hay que escalar el eje X de la gráfica del espectro para que muestre la frecuencia en Hz. Observar las diferencias respecto al anterior diagrama:



EJERCICIOS

MEDIDA MEDIANTE CURSORES

1. Generar una señal cuadrada de 100 Hz.
2. Comprobar mediante los cursores de la gráfica temporal que la frecuencia de la señal se corresponde con la que aparece en el espectro.
3. De la misma forma medir la amplitud

	amplitud	frecuencia
Medida en el generador de funciones		
Medida (tiempo)		
Medida con el VI (espectro)		
Error absoluto		
Error relativo		

RESOLUCIÓN ESPECTRAL

Se trata de averiguar de forma experimental la relación que define la resolución espectral Δf , para lo cual:

1. Seleccionar una señal senoidal de p.ej. 100 Hz
2. Dejar f_s constante (cumpliendo Nyquist) y variar el nº de muestras N (P. ej. 128, 256, 512). ¿Qué ocurre con la separación entre los puntos del espectro, Δf ?

fseñal	fmuestreo	Nº de muestras	Δf (aumenta ó disminuye)
100 Hz		128	
		256	
		512	
		1024	

3. Dejar el nº de muestras constante (P. ej. 256) y variar f_s cumpliendo Nyquist. ¿Qué ocurre con el espectro?.

fseñal	Nº de muestras	fmuestreo	Δf (aumenta ó disminuye)
100 Hz	256		

4. A partir de los datos recogidos establecer la ecuación que define Δf en función de f_s y N . ¿Cuál de las tres expresiones es la correcta?

$$\Delta f = N / f_s \qquad \Delta f = f_s / N \qquad \Delta f = f_s * N$$

5. Comprobar numérica y experimentalmente la relación hallada para alguno de los casos anteriores:

fseñal	Nº de muestras	fmuestreo	Δf

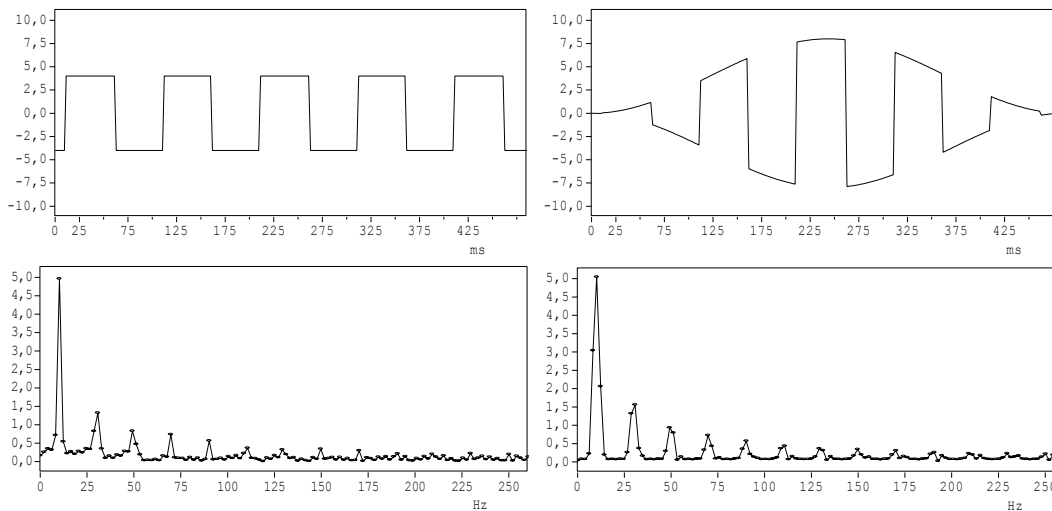
ALIASING

Seguir los pasos siguientes y rellenar la tabla:

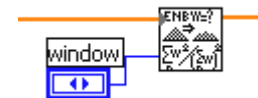
Señal senoidal de 0'5 V@2000 Hz			
<ol style="list-style-type: none"> 1. Inyectar una señal senoidal de 0'5 V@2000 Hz en el canal 0 de la tarjeta. 2. Configurar la ganancia para obtener la máxima resolución. 3. Muestrear la señal respetando el teorema de Nyquist. 4. Medir la frecuencia de la señal (armónico único), comprobando que sea la esperada. 5. Configurar el VI de forma que pueda rellenarse la tabla. 	Ganancia		
	Frec. muestreo		Armónico medido (Hz)
	$f_s = f_{Nyquist}$		
	$f_s > f_{Nyquist}$		
$f_s < f_{Nyquist}$			
CONCLUSIONES:			

PRÁCTICA 13: VENTANAS DE ALISADO

Con este ejercicio, trataremos de ver el efecto de realizar la FFT sobre un número **no** entero de periodos de la señal muestreada. Este efecto puede paliarse mediante el uso de ventanas de alisado previas a la FFT.



1. Insertar el VI **Functions/Analysis/Measurement/Scaled Time Domain Window** después de la adquisición y antes de la gráfica.
2. Generar una señal senoidal de 500 Hz.
3. Muestrear a 5 kS/s
4. Ajustar el nº de muestras para ver exactamente 3 periodos de señal.
5. Observar el espectro de amplitud, especialmente los alrededores del armónico fundamental.
6. Ajustar el nº de muestras que permita ver exactamente 2'5 periodos de señal.
7. Observar el espectro de amplitud, especialmente los alrededores del armónico fundamental.
8. Activar la ventana tipo "Hamming" y repetir los pasos anteriores.
9. Conclusiones:



¿Qué sucede cuando el número de muestras por periodo es un número entero?.
¿Qué sucede cuando el número de muestras por periodo NO es un número entero?.

PRÁCTICA 14: FILTRADO DIGITAL

14.1.- PROMEDIADO

El proceso más evidente de filtrado es el promediado. La relación Señal/Ruido que se obtiene mediante esta técnica es:

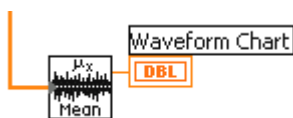
$$\frac{N_{out}}{N_{in}} = \frac{1}{\sqrt{n^{\circ} \text{ muestras}}}$$

Por ejemplo un promediado de 4 muestras reduce a la mitad la potencia de ruido en la salida. Observar que para reducir a la cuarta parte dicha relacion tendríamos que usar un promediado de 16 muestras

ELIMINAR INTERFERENCIA DE LA RED ELÉCTRICA MEDIANTE PROMEDIADO

Cuando tenemos una señal cuasi continua a la que se ha añadido una interferencia como la de la red eléctrica, podemos aprovechar que el valor medio de una senoide es cero para eliminarla. Si tomamos un n° de muestras tal que completen un ciclo de la senoide, su valor medio será cero. Se trata de modificar nuestro VI para que realice dicha tarea:

1. Fijar una velocidad de muestreo suficiente para la interferencia de la red eléctrica.
2. Verificar mediante el VI anterior que la frecuencia se corresponde con la esperada.
3. Calcular el n° de puntos del promediado que debemos realizar para eliminar dicha interferencia.
4. Modificar el VI anterior para que realice este cometido añadiendo los siguientes elementos:



5. Fijar el n° de muestras del VI al n° de puntos calculado, y comprobar el resultado.
6. Rellenar la siguiente tabla con los datos obtenidos.

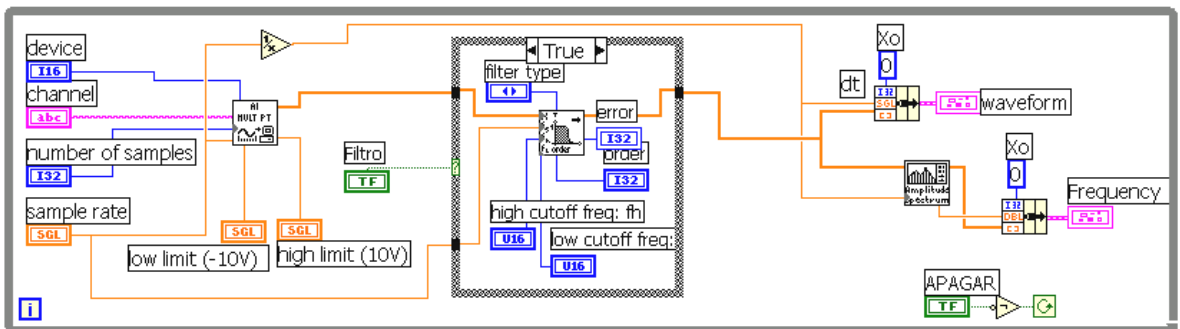
Frec. muestreo	$N = \frac{\dot{} \dots \dots \dots ?}{\dot{} \dots \dots \dots ?}$	Observaciones
$f_1 =$		
$f_2 =$		

14.2.- FILTROS DIGITALES

AÑADIR FILTRADO DIGITAL DE BUTTERWORTH

Vamos a intercalar un filtro entre el array de señal y las gráficas de forma que podamos filtrar varios armónicos de una señal cuadrada (distorsionar). Para ello:

1. Crear una estructura **case** que permita al usuario elegir si desea filtrar o no la señal mediante un botón.
2. Dentro del caso **True** incorporar un filtro Butterworth (se encuentra en el grupo de funciones **Analysis/Filters/Butterworth filter.vi**)
3. Crear automáticamente los controles necesarios para configurar el filtro: **tipo, orden, frecuencia superior de corte, frecuencia inferior de corte**. Debemos obtener algo parecido a la siguiente figura.



4. Generar una señal cuadrada de 1 kHz.
5. Aislar el 1^{er} armónico mediante el filtro dispuesto.

Orden del filtro	Nº de muestras	fmuestreo	Δf	f teórica del 1 ^{er} armónico	f medida del 1 ^{er} armónico

En el filtro debe cumplirse $0 < f_{corte} \leq f_s/2$, de lo contrario dará un error. Si, según Nyquist sólo pueden muestrearse señales por debajo de $f_s/2$, el filtro sólo funcionará hasta esa frecuencia.

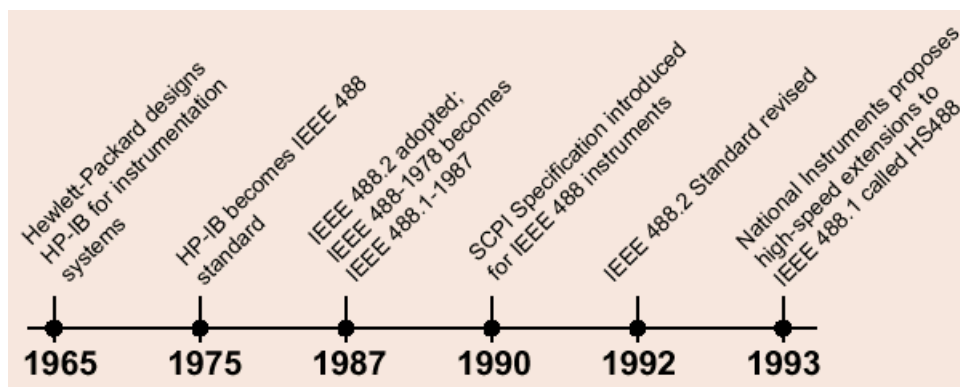
Describir la influencia del orden del filtro en los resultados obtenidos:

5.- BUS GPIB.

INTRODUCCIÓN AL BUS GPIB

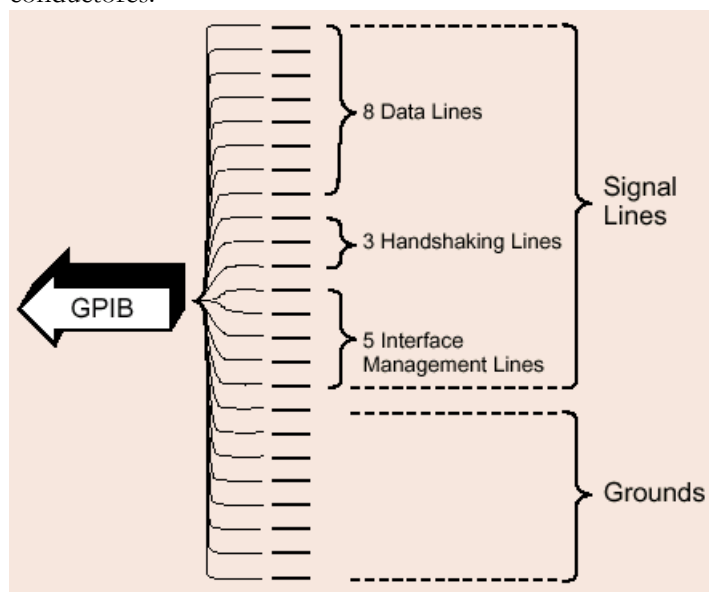
HISTORIA

- En 1965 la empresa Hewlett Packard crea la interfaz de bus HP-IB (Hewlett Packard Interface Bus) para conectar instrumentos programables a sus ordenadores.
- Debido a sus ventajas se populariza de tal forma que en 1975 se estandariza como IEEE 488-1975 (Aunque es más conocido como GPIB -General Purpose Interface Bus-)
- En 1987 evoluciona dando paso al ANSI/IEEE 488.2, por lo que el anterior IEEE 488-1975 pasa a denominarse ANSI/IEEE 488.1

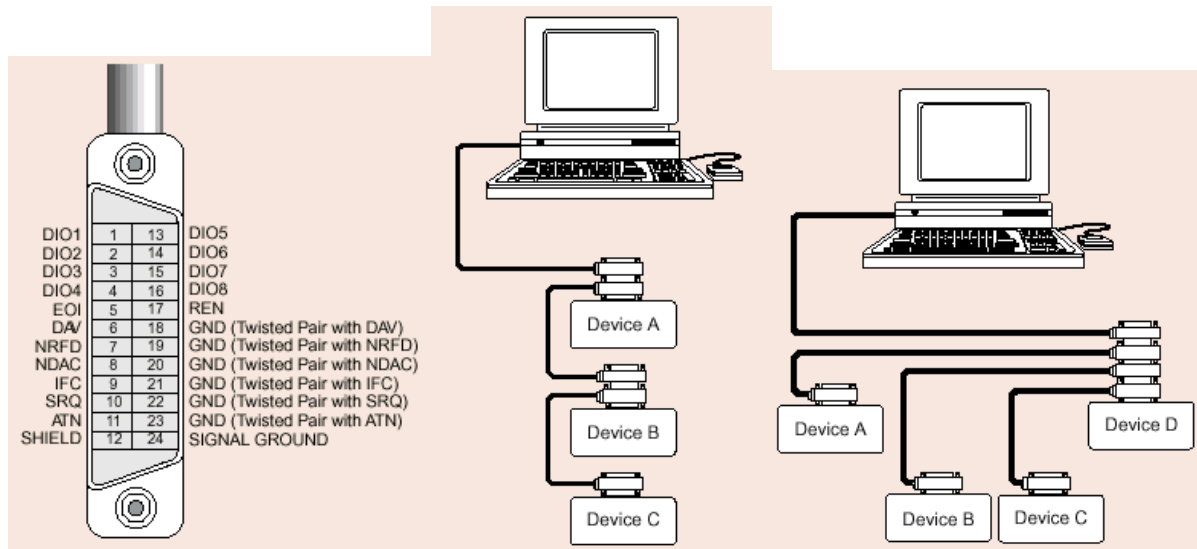


CARACTERÍSTICAS MECÁNICAS Y ELÉCTRICAS

Se trata de un bus paralelo, formado por 8 líneas de datos, 3 líneas de protocolo, 5 de gestión, 7 de masa y 1 apantallamiento. Tenemos por tanto un cable de 24 conductores.



Permite la conexión de dispositivos en estrella, o en línea:



- Utiliza niveles TTL y lógica negada.
- Velocidades de transferencia de hasta 1 MB/s.
- Hasta 15 dispositivos conectados al bus.
- Separación máxima de 4 m entre dos dispositivos cualesquiera del bus. Longitud total del cable de hasta 20 m

FUNCIONAMIENTO

Los dispositivos conectados al bus se comunican entre ellos mediante mensajes

Tipos de mensajes

- **Dependientes del dispositivo:** Llamados datos a secas, contienen información específica de los dispositivos como pueden ser resultados de una medida, estado de un aparato, instrucciones de configuración, etc.
- **De gestión:** También llamados mensajes de comando, realizan funciones como inicializar el bus, direccionar dispositivos, etc.

Roles de los dispositivos

Los dispositivos conectados al bus pueden adoptar uno o varios de los roles definidos: Hablante, escuchador y controlador.

ESCUCHADOR (LISTENER)

Capaz de recibir datos de la interfaz cuando esté direccionado (habilitado) por el controlador. Puede haber hasta 14 escuchadores activos simultáneamente en el bus. Dispositivos escuchadores pueden ser por ejemplo una impresora, un generador de funciones, etc..

HABLANTE (TALKER)

Un hablante, cuando se le ha direccionado, envía mensajes a uno o varios escuchadores que reciben los datos. Por ejemplo un osciloscopio puede actuar como hablante y como escuchador. Sólo puede haber un hablante activo sobre la interfaz.

CONTROLADOR

Gestiona el flujo de información en el bus enviando comandos a todos los dispositivos. El controlador es capaz de direccionar (habilitar) a un hablante que quiera enviar un mensaje a varios escuchadores, permitiendo una operación de transferencia de datos. Él mismo puede ejercer de emisor o receptor.

Se podría configurar un bus sin dispositivo controlador y en el que hubiera un dispositivo con capacidad únicamente de hablante y varios con capacidad únicamente de escucha.

La función de controlador la suele realizar un dispositivo conectado al ordenador (Tarjeta ISA, PCI, PCMCIA, etc.) que habitualmente tiene además capacidad de hablante y escuchador.

En sistemas con varios controladores sólo uno puede estar actuando como tal. El controlador activo puede pasar el mando del bus a otro controlador que se halle inactivo.

Lineas según su función

LINEAS DE DATOS

Son 8 y se denominan **DIO1-DIO8**

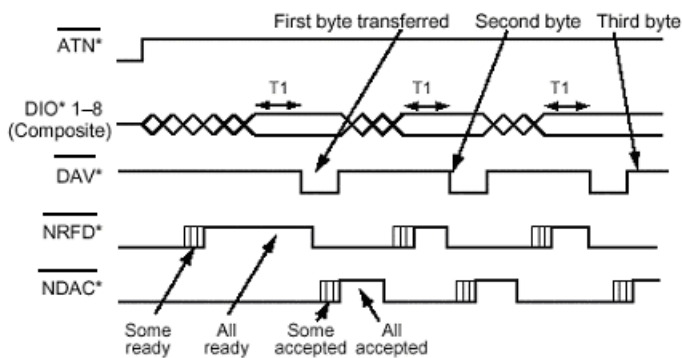
- El estado de la línea de gestión **ATN** (se verá más adelante) determina si la información presente en el bus de datos corresponde a un dato o a un comando.
- Todos los comandos y la mayoría de datos utilizan el código ASCII de 7 bits, de forma que la línea **DIO8** se utiliza como paridad.

LINEAS DE PROTOCOLO

Estas tres líneas controlan de forma asíncrona la transferencia de bytes de mensajes entre dispositivos, garantizando que se realiza sin errores.

- **DAV** (Data Valid): Es activada por el controlador cuando envía comandos y por el hablante cuando envía mensajes de datos. Indica cuando la señales de datos son estables (válidas) de forma que puedan ser interpretadas de forma fiable por los escuchadores.
- **NRFD** (Not Ready For Data): Un escuchador activa esta línea cuando no está preparado para la recepción de datos. Tendrá valor *falso* cuando todos los receptores direccionados estén listos para recibir datos. Se realiza la función **OR cableada** de las salidas **NRFD** de lo escuchadores direccionados. Siendo necesario que todos estén preparados (todas las salidas **NRFD** a *falso*) para que la línea **NRFD** se ponga a *falso*.
- **NDAC** (No Data Accepted): En estado *verdadero* indica que algún escuchador direccionado no ha aceptado todavía los datos enviados. Un estado *falso* indica que todos los receptores activos han aceptado los datos. Nuevamente se realiza la función **OR cableada** de todas las salidas **NDAC** de los receptores activos.

En la siguiente figura puede verse la secuencia de control de transferencia de datos o **handshake**.



LINEAS DE GESTIÓN

Estas cinco líneas gestionan el flujo de datos e información del bus.

- **ATN** (ATtention): El controlador activa **ATN** a *verdadero* cuando usa las líneas de datos para enviar comandos y la pone a *falso* para permitir que un hablante envíe datos.
- **IFC** (Interface Clear): El controlador activa esta línea para inicializar el bus interrumpiendo el proceso que se estaba realizando. Se deshabilita al hablante y a los escuchadores activos, quedando todos inactivos. El controlador asume el mando del bus. Todos los dispositivos deben responder a esta línea en cualquier instante.
- **REN** (Remote Enabled): El controlador activa esta línea para poner los dispositivos direccionados en modo de programación remota. Cuando no está activada los dispositivos se encuentran en modo de control local.
- **SQR** (Service Request): Cualquier dispositivo puede activar esta línea para pedir servicio al controlador.

- **EOI** (End Or Identify): Tiene dos funciones; el hablante activa esta línea para indicar el último byte de un dato a los receptores activos. Con ATN a "1", la activación de esta línea indica que el controlador realiza un sondeo paralelo.

IEEE 488-2

- Viene a paliar los problemas que arrastra el estándar desde el veterano IEEE 488.1.
- Se mantiene la compatibilidad con IEEE 488.1, pero los beneficios del nuevo estándar sólo se obtienen cuando se tiene un sistema totalmente compatible IEEE 488.2
- Normaliza el lenguaje de programación de los dispositivos incorporando el estándar SCPI.

SCPI (Standar Commands for Programmable Instruments)

Una de las cosas que no define el estándar IEEE 488.1 es la estructura y sintaxis del lenguaje que se usa para programar los dispositivos, lo que provoca que p. ej. un programa elaborado para un osciloscopio no funcione con un osciloscopio de otra marca. Esto aumenta el trabajo de los programadores y por tanto los costes de desarrollo y mantenimiento. Podemos comparar el problema con el caso de las líneas telefónicas: puedo establecer una comunicación con China por que las líneas son compatibles, pero si no conozco el idioma, no lograré que el hablante y el escuchador se entiendan. SCPI viene a solventar esa dificultad idiomática entre instrumentos definiendo un estructura y sintaxis común del lenguaje.

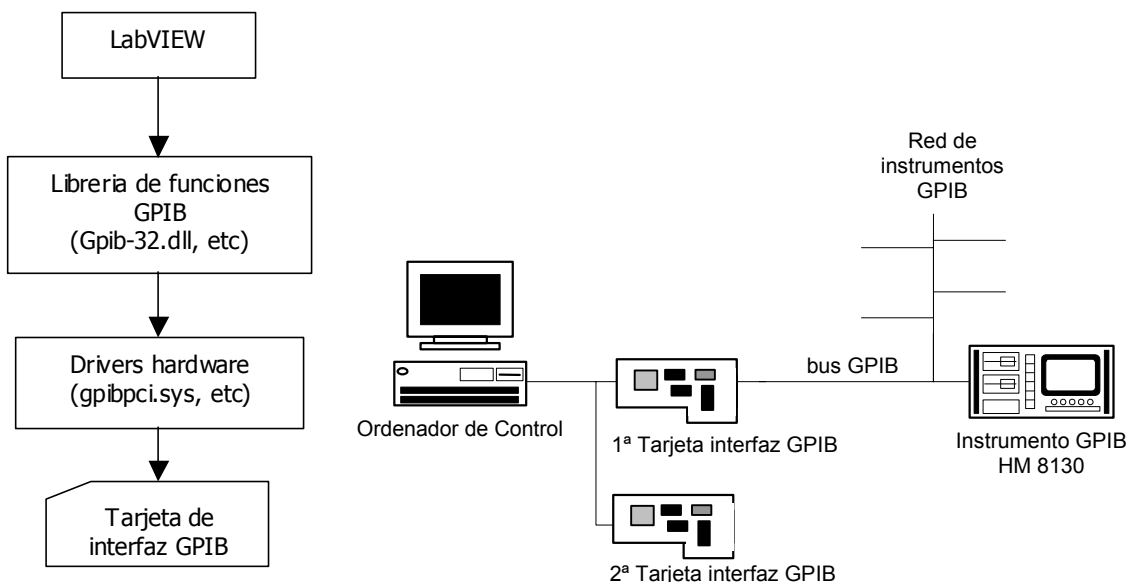
HS488

Es una propuesta de la compañía National Instruments para la mejora de las velocidades de transmisión de la norma 488. Entre dos dispositivos compatibles HS488 separados por 2 metros de cable se pueden alcanzar de hasta 8 MB/s. En un sistema con 15 dispositivos y 15 metros de cable la velocidad puede alcanzar 1'5 MB/s

PROGRAMACIÓN GPIB CON LABVIEW

CONEXIÓN LABVIEW-DISPOSITIVO GPIB

Las siguientes figuras ilustran la conexión que permite controlar un instrumento GPIB mediante el software LabVIEW cargado en nuestro ordenador.

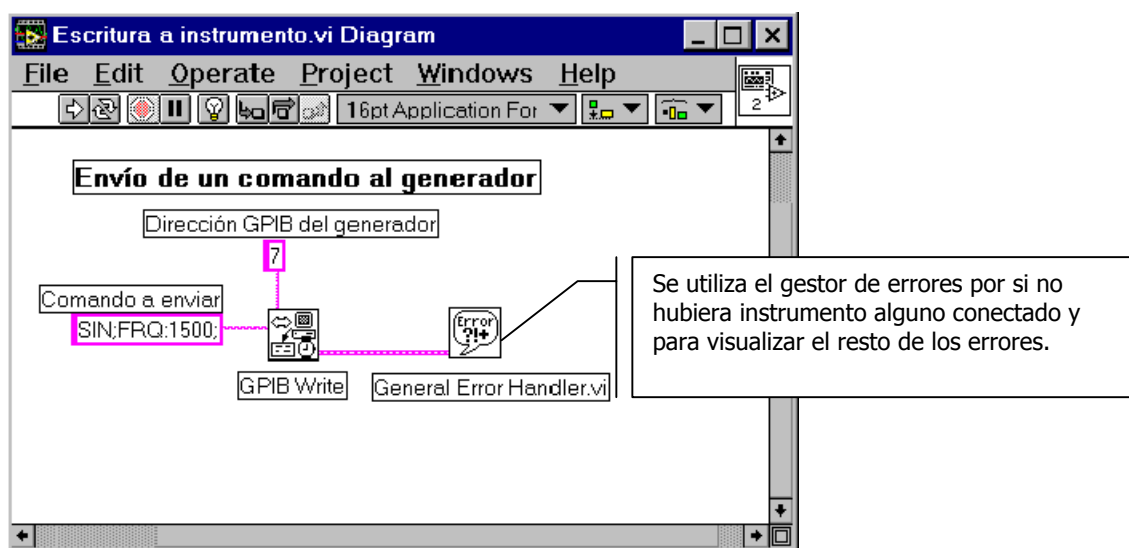


ENVÍO DE COMANDOS

En este apartado veremos como enviar ordenes por el bus GPIB al generador de funciones Hameg HM-8130 mediante LabView.

Las funciones relativas al bus GPIB se encuentran en *Functions/Instrument I-O/GPIB*

Debemos conocer previamente la dirección del bus asignada a nuestro aparato. Podemos verla directamente según la configuración de los microinterruptores que se encuentran en su parte trasera. Además nos lo muestra en su display cada vez que encendemos el aparato.



Entre los comandos que admite el generador de funciones HM-8130 tenemos por ejemplo:

- SIN** Genera a la salida la onda senoidal de los valores seleccionados
- OTO** Desactiva y **OT1** activa la salida de señal.
- OFO** Desactiva la tensión de Offset en la salida y **OF1** la activa.
- DFR** Visualiza la frecuencia.
- DAM** Visualiza la amplitud
- LK1** Bloquea la botonera del generador para impedir su manejo en modo local. **LK0** lo desbloquea.

Otros comandos que precisan adjuntar un dato son por ejemplo:

- FRQ:dato** Sitúa la frecuencia de la señal en el valor de 'dato' (Hz).
- AMP:dato** Sitúa la amplitud de salida según 'dato' (Voltios).
- OFS:dato** Sitúa la tensión de Offset al valor expresado por 'dato' (Voltios).

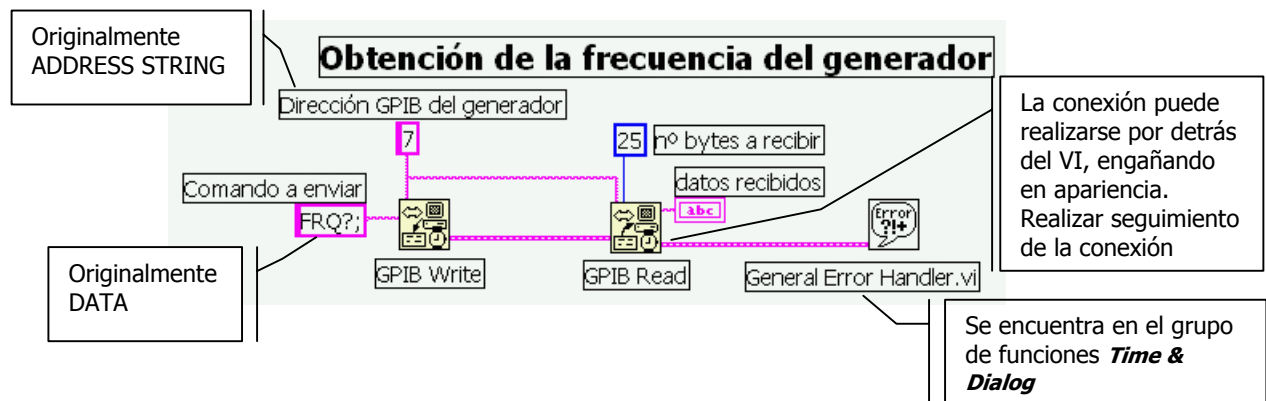
Los comandos del generador pueden concatenarse para formar una única cadena de comandos y enviarse a la vez. Cada comando debe acabar con punto y coma “;”. Por ejemplo:

LK1;TRM;RMP;FRQ:0;AMP:2,000E-2;OT1;OFO;

RECEPCIÓN DE RESPUESTAS

La comunicación con los instrumentos implica la recepción de respuestas tras el envío de comandos de interrogación. Por ejemplo para saber la frecuencia actual de la señal generada.

El subVI **General Error Handler** mostrará en pantalla los errores producidos en la comunicación por el bus en caso de producirse. y asegura (al unir la salida de la acción de escritura con la de lectura) la correcta secuencia; no hay ninguna duda de que primero se producirá la escritura del comando de interrogación, en espera de la respuesta y posteriormente la lectura de la misma.

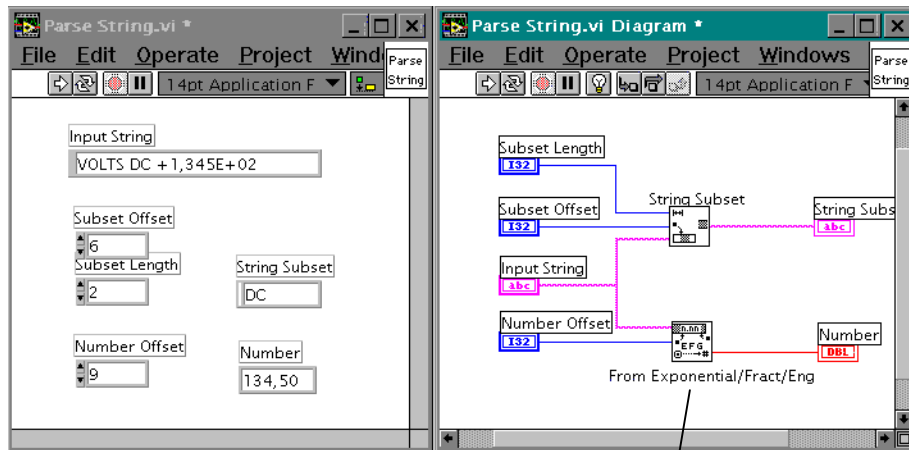


Entre los comandos de interrogación que ofrecen respuesta por parte del generador de funciones están:

- FRQ?** Devuelve la frecuencia actual
- AMP?** Voltaje actual a la salida
- OFS?** Tensión de Offset.
- ID?** Identificación del aparato.
- VER?** Versión del equipo.
- STA?** Estado del equipo.

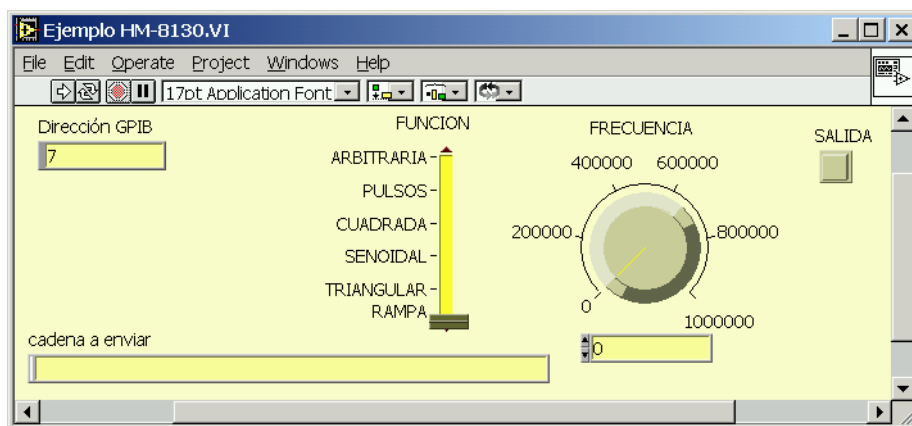
OBTENCIÓN DE DATOS DE LA CADENA DE RESPUESTA DE UN INSTRUMENTO:

El instrumento devolverá como respuesta a nuestra interrogación una cadena de caracteres que deberemos tratar para obtener la información de interés. Para ello trocaremos la respuesta, en cadenas de caracteres y números, para su posterior utilización.

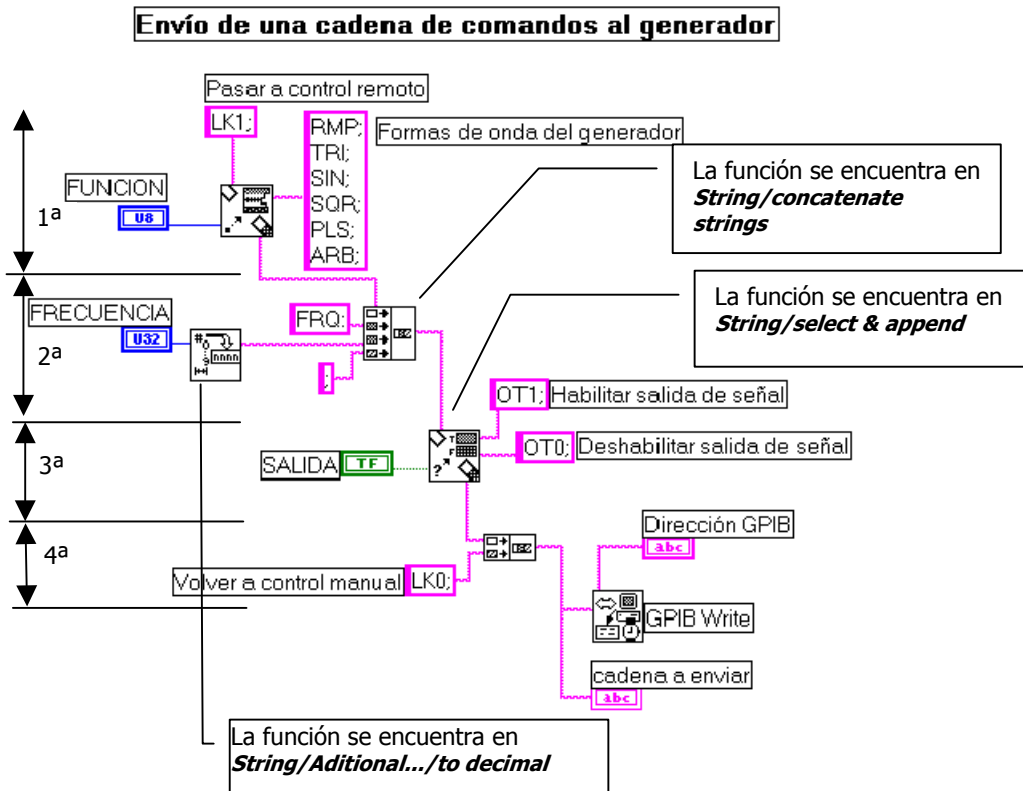


La función se encuentra en
FUNCTIONS/strings/additional string to number

En general es interesante para el usuario que la interfaz de nuestro VI se parezca lo más posible al panel frontal del instrumento real. Colocaremos mandos para todas las funciones que necesitemos controlar.



Por razones didácticas, hemos añadido un visualizador de la cadena que se enviará al instrumento. El diagrama de bloques que debemos implementar es el siguiente:



Realizar el programa por partes, visualizando el resultado de cada una de ellas (de 1ª a 4ª) según se incorporan nuevas funciones. Por último añadir el sub VI de escritura en el bus GPIB.

APLICACIONES GPIB EN RED

Podemos ver otras aplicaciones como son las relativas al control de instrumentación no directamente conectada a nuestro ordenador.

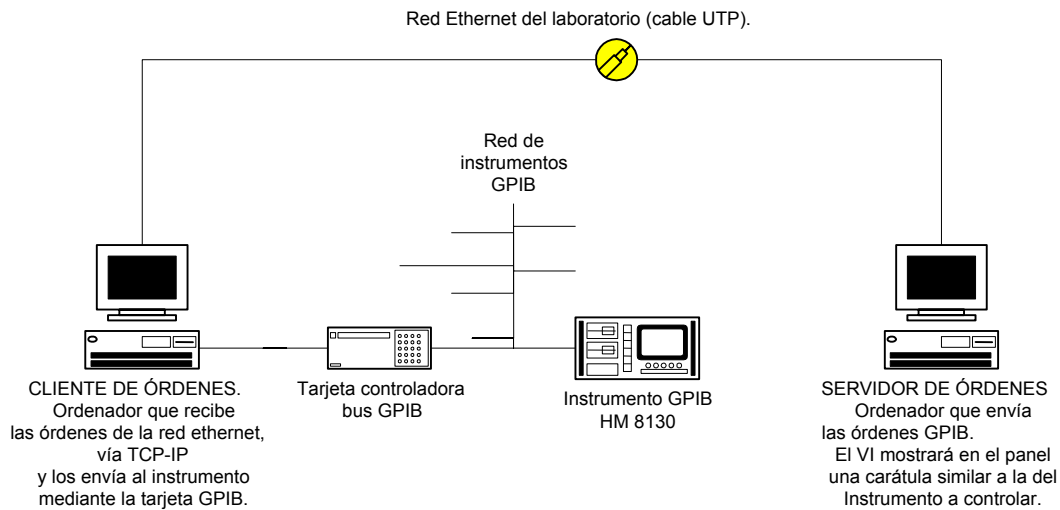
SERVIDOR REMOTO DE COMANDOS GPIB

Por ejemplo es posible controlar un dispositivo GPIB desde cualquier puesto de una red de ordenadores. Un ordenador contiene la tarjeta de interfaz GPIB, pero el control lo podemos realizar desde otro ordenador, que sería el que contenga el programa de aplicación.

Los datos llegarían al aparato a controlar mediante el ordenador que tiene la tarjeta de bus, que actuaría como un servidor de datos; estos serían enviados por el ordenador que tiene el programa de control y que actuaría como un cliente.

Este intercambio de datos se hace mediante la tecnología cliente-servidor dentro de las redes de ordenadores, y permite la conexión remota, incluso vía telefónica entre distintos equipos de distintas características siempre y cuando cumplan con el protocolo de comunicación.

Conexión para manejar el generador de funciones con comandos GPIB desde otro PC. El ordenador que no tiene la tarjeta controladora es el que genera los comandos de control. Ambos ordenadores se comunican mediante la red ethernet en el protocolo TCP-IP.



APLICACIONES DE DISTRIBUCIÓN DE SEÑAL EN RED

Cliente de la adquisición de señal.

Uno de los ordenadores, el que tiene la tarjeta de adquisición, manda a los otros, mediante TCP-IP por la red ethernet, la señal capturada.

