Universidad Nacional Autónoma de México

Facultad de Ingeniería

SISTEMA DE INSTRUMENTACIÓN BASADO EN PC

TESIS PROFESIONAL

QUE PARA OBTENER EL TÍTULO DE

INGENIERO ELÉCTRICO ELECTRÓNICO

PRESENTA

JUAN ANTONIO JIMÉNEZ MARTÍNEZ

DIR. ING. GLORIA MATA HERNÁNDEZ

MÉXICO, D.F. 2005

A mis padres
Juan Antonio Jiménez Villagrana
y Felipa Martínez Ángeles
por el apoyo brindado en el
transcurso de todo este tiempo.

A mis hermanos
Ricardo Jiménez Martínez
y Yedid Jiménez Martínez.

A la familia Contreras Villeda
de manera especial al
Ing. Alejandro Contreras González

A mis Sinodales
Ing. Gloria Mata Hernández
M. en I. Ricardo Garibay Jiménez
M. en I. Juan Manuel Gómez González
M. en I. Serafín Castañeda Cedeño
y al Ing. Sabino Ortega Monjaras
por el valioso tiempo dedicado
a la revisión de este trabajo.

A la Universidad Nacional Autónoma de México.

Índice General

1 Introducción **7**

2 Sistema General de Medición **11**

2.1 Introducción 12

2.2 Etapa detectora de variables 14

2.2.1 Transducción resistiva 15

2.2.2 Transducción capacitiva 16

2.2.3 Transducción inductiva 17

2.2.4 Transducción magnética 18

2.2.5 Transducción por voltaje, corriente y frecuencia 19

2.3 Etapa acondicionadora 21

2.4 Etapa de despliegue de información 27

2.5 Interfase GPIB 28

2.5.1 Protocolo 29

2.5.2 Comandos 32

3 Herramienta de desarrollo **37**

3.1 Ambiente de desarrollo de HP VEE 40

3.1.1 Operaciones en el ambiente de desarrollo 42

3.1.2 Objetos 45

3.1.3 Reglas de propagación en los objetos 47

3.1.4 Depuración de un programa 48

3.2 Creación de programas de prueba 50

3.3 Lectura y escritura de datos 55

3.4 Formas de controlar instrumentos 61

3.4.1 Utilización del INSTRUMENT MANAGER 61

3.4.2 Objeto DIRECT I/O 63

3.4.3 PANEL DRIVERS 66

3.4.4 COMPONENT DRIVER 67

3.5 Análisis y despliegue de datos 69
3.6 Funciones en HP VEE 73
3.6.1 Objetos de usuario 73
3.6.2 Funciones de usuario 74
3.6.3 Librerías 76

4 Aplicaciones **77**
4.1 Descripción general 78
4.2 Toma de muestras de una población 80
4.3 Medición de temperatura 89
4.4 Adquisición de datos de transductores 99
4.5 Ajuste de curvas 113
4.6 Conversión de variables eléctricas 117
4.7 Osciloscopio 124
4.8 Integración con el lenguaje C 128
4.9 Generación de funciones 135
4.10 Operaciones entre funciones 140
4.11 Puerto serie 149

Conclusiones 154

A Comandos SCPI **156**

B Hoja Técnica **165**

Bibliografía 167

Índice de Figuras

2.1 Sistema tradicional de medición 12
2.2 Sistema de medición basado en PC 13
2.3 Buffer 21
2.4 Amplificador de Instrumentación 22
2.5 Puente de Wheatstone 23
2.6 Filtro paso bajas 24
2.7 Curva de transferencia para un transductor que muestra una relación no lineal entre la temperatura y el voltaje de salida . 25
2.8 Señal del transductor linealizada con nivel de DC. 26
2.9 Conectores IEEE488 29
2.10 Diferentes topologías para GPIB 30
2.11 Diagrama de tiempo del bus IEEE-488 31

3.1 Ejemplo de programa. 40
3.2 Ambiente de desarrollo de HP VEE 41
3.3 Estructura general de la barra del objeto. 43
3.4 Estructura general de objetos y sus terminales. 45
3.5 Vista de panel. 46
3.6 Estructura general del objeto como ícono. 46
3.7 Orden de los eventos de un objeto. 47
3.8 Programa para hacer filtrado. 50
3.9 Panel de usuario del filtro promediador. 51
3.10 Objeto TO FILE. 55
3.11 Caja de diálogo de una transacción de E/S. 56
3.12 Formato de una transacción 57
3.13 Manejador de instrumentos. 61
3.14 Objetos direct I/O del multímetro, generador de señales y osciloscopio respectivamente. 63
3.15 Menú parcial del objeto DIRECT I/O. 64
3.16 Objeto DIRECT I/O en vista abierta. 65

3.17 Panel driver para un osciloscopio. 66
3.18 Component driver para un osciloscopio. 68
3.19 Contextos entre objetos de usuario. 74

4.1 Esquema de conexión PC–Instrumento (HP34401A). 80
4.2 Panel de usuario para el muestreador. 81
4.3 Panel para tipo de medición y num. de muestras. 82
4.4 Implementación panel para tipo de medición y num. de muestras. 83
4.5 Implementación del muestreador. 84
4.6 Para iniciar el muestreo. 85
4.7 Para tomar muestras. 85
4.8 Sobrepaso de muestras. 85
4.9 Objeto alloc real en vista abierta. 86
4.10 Panel Driver del multímetro digital. 86
4.11 Caja de diálogo para guardar las lecturas a un archivo. 87
4.12 Conexión del Termopar tipo J 89
4.13 Conexión del Termistor NTC 89
4.14 Conexión del LM35DZ 90
4.15 Panel de usuario de selección de transductores de temperatura. 90
4.16 Programa para los tres transductores. 93
4.17 Objeto tipo Radio Buttons para selección de transductor. . . 94
4.18 Selector del tipo de transductor. 94
4.19 Objetos para inicializar la medición. 95
4.20 Proceso de medición con el objeto Direct I/O. 96
4.21 Caja de diálogo indicando el guardado del archivo. 96
4.22 Objetos para inicializar archivo de guardado de datos. 98
4.23 Introducción del valor de la variable independiente 99
4.24 Esquema de conexión PC–HP34401A. 99
4.25 Objeto de asignación para almacenado de valores. 100
4.26 Programa MUESTREADOR_MODULOS.VEE. 101
4.27 Panel de la aplicación MUESTREADOR_MODULOS.VEE. . . . 102
4.28 Esquema de conexión PC–dos multímetros. 103
4.29 Gráfica Módulo G22 (desplazamiento). 104
4.30 Gráfica Módulo G25 (fuerza). 105
4.31 Gráfica Módulo G24 y MIL27(presión). 106
4.32 Gráfica Módulo G27 (LVDT). 107
4.33 comportamiento del fototransistor. 110
4.34 comportamiento del fotodiodo. 111
4.35 comportamiento de fotoresistencia. 111

4.36 Programa para controlar dos multímetros. 112
4.37 Implementación del programa de ajuste. 113
4.38 Caja de diálogo para selección de un archivo. 114
4.39 Vista abierta objeto File Name Selection. 114
4.40 Objeto para hacer la regresión de los datos. 114
4.41 Panel del programa de ajuste. 115
4.42 Vista abierta objeto From File. 116
4.43 Esquema de conexión para visualizador. 117
4.44 Panel del vizualizador de cantidades eléctricas a físicas 117
4.45 Programa VISUALIZACION_VARIABLES.VEE 123
4.46 Esquema de conexión PC–HP54603B. 124
4.47 Panel del programa para manejar el HP54603B. 124
4.48 Implementación del programa para manejar el HP54603B. . . 127
4.49 Objeto para importar una librería. 131
4.50 Objeto Call para llamar a la función dentro de la DLL. . . . 133
4.51 Uso de una función compilada desde el objeto Formula. . . . 133
4.52 Dos librerías con sus funciones respectivas. 134
4.53 Esquema de conexión PC–Instrumento (HP33120A). 135
4.54 Panel de usuario del programa de demostración del HP33120A 136
4.55 Programa de demostración del HP33120A 138
4.56 Captura de señal SINC con la aplicación OSCILOSCOPIO.VEE. 139
4.57 Esquema de conexión PC—Generador—Osciloscopio. 140
4.58 Panel de usuario del generador arbitrario de funciones 141
4.59 Implementación del generador arbitrario de funciones 143
4.60 Parámetros de inicialización 146
4.61 Resta de dos señales cuadrada menos senoidal. 148
4.62 Esquema de conexión PC–Instrumento (HP33120A). 149
4.63 Utilería **IO Config** para agregar la interfase serie. 151
4.64 Parámetros de comunicación del puerto serie. 152
4.65 Programa para manejar el instrumento mediante el puerto serie. 153
4.66 Ruta de la utilería **I_O config** para agregar la interfase serie. 153

A.1 Modelo de un instrumento programable 157
A.2 Arbol parcial del comando MEASure 157

Índice de Tablas

2.7 Codigos IEEE488 36

3.1 Formatos para la transacción READ TEXT. 58
3.1 (continua ...) 59
3.1 (continua ...) 60
3.2 Tipos de datos 69
3.2 (continua ...) 70
3.3 Tipos de desplegadores 71
3.3 (continua ...) 72

4.1 Sentencias SCPI, aplicación TEMPERATURA.VEE. 91
4.2 Equivalencias entre tipos de datos 131
4.3 Correspondencia entre pines del puerto serie de la PC y el Instrumento 152

Capítulo 1

Introducción

La inestabilidad de patrónes físicos por envejecimiento de materiales o por variaciones de temperatura llevó a la evolución de estos mismos (como el metro y kilogramo patrón) a la predicción de su comportamiento por medio modelos matemáticos que residen dentro de una computadora digital, los cuales son generados a través de experimentos y correlaciones entre mediciones ópticas y de tiempo. Se puede decir que estos patrones son instrumentos virtuales porque no existen físicamente.

Los instrumentos virtuales existen como alternativa al equipo tradicional, ya que disponen de funciones complejas y además, son herramientas que incrementan la productividad en el laboratorio a nivel académico y profesional.

Un beneficio de la instrumentación virtual, así como también el hecho de que en la actualidad se tenga la disponibilidad de computadoras poderosas que debido al avance rápido de la tecnología están en constante evolución, es que los datos adquiridos y guardados están en un formato listo para un análisis posterior en un medio digital. La adquisición de datos en el laboratorio es mejorada grandemente y en el ámbito académico para los estudiantes se les presenta lo último de la tecnología mientras la experiencia en los equipos clásicos se mantiene vigente.

Otro beneficio es que estos sistemas permiten hacer las mediciones de manera transparente para que los usuarios no tengan que preocuparse por la manera en como se adquieren los datos ya que el software se encarga de esta tarea. Esto es especialmente útil dentro del ambiente industrial cuando se desarrolla un prototipo inicial ya que se invierte gran cantidad de tiempo en la etapa de corrección y sobre todo en la etapa de verificación del diseño mediante mediciones.

Un sistema de instrumentación se compone en términos generales de tres etapas: la etapa detectora de variables, de acondicionamiento, y de despliegue de información. La etapa detectora de variables esta integrada por elementos llamados transductores que permiten convertir la variable física de interés (ya sea fuerza, temperatura, presión, etc.) a un equivalente en magnitud pero en una variable eléctrica, la cual es posible manipular mediante la etapa de acondicionamiento para obtener una señal proporcional de corriente, voltaje o frecuencia. Los sistemas tradicionales de instrumentación para el despliegue de información utilizan dispositivos como pantallas de cristal líquido, pantallas de rayos catódicos (CRT), indicadores analógicos, mientras que el despliegue de información en un sistema de instrumentación basado en PC se hace en la pantalla de la misma.

Los sistemas de instrumentación tradicionales resultan ser caros debido a que para cada variable que se necesite visualizar o medir se requiere de un instrumento específico. Un sistema de instrumentación basado en PC se compone de los tres bloques mencionados anteriormente, los cuales realizan funciones que se llevan a cabo con hardware y/o software. Los dos primeros básicamente tienen las mismas funciones, y para el despliegue de información se utiliza la PC, la cual no solo realiza esta función, sino que se aprovecha su capacidad para llevar a cabo funciones de procesamiento y análisis de datos, interfases gráficas de usuario en diversos formatos y manejo de instrumentos. Para un sistema de instrumentación basado en PC el software juega un papel muy importante ya que es la herramienta para manejar los instrumentos y manejar la adquisición de los datos, además, debe ser lo mas funcional y versatil posible.

En el caso específico de este trabajo, el despliegue y procesamiento de la información se realizará utilizando la herramienta *HP VEE*[1], la cual está optimizada para construir aplicaciones de medición y prueba, especialmente interfases gráficas para usuarios, permite adquirir y analizar datos así como manejar diversos instrumentos de laboratorio. La presentación de datos puede ser tan sencilla como las hojas de cálculo, o hasta complicadas rutinas que simplifican el desarrollo de procedimientos de prueba mediante la simulación de paneles de control, en los que se puede operar y programar instrumentos, observar en detalle la respuesta de un sistema o el espectro de una señal.

En contraste con los sistemas tradicionales, y dependiendo del tipo de interfase (en este caso es GPIB), un sistema de instrumentación basado en PC permite tener mas de un dispositivo de medición, con lo que el costo

[1]Hewlett-Packard Visual Engineering Enviroment

del mismo se logra abatir, especialmente cuando se requieren medir diversas variables, además de que los datos pueden ser almacenados para su posterior análisis y procesamiento. *HP VEE* es una herramienta de propósito general, y en este caso será aprovechada en el laboratorio de Medición e Instrumentación ya que se cuenta con los elementos para desarrollar y diseñar diferentes aplicaciones.

El sistema de instrumentación basado en PC es una alternativa a un sistema convencional ya que es mas funcional, mas flexible, y permite tener un mejor control de las mediciones.

Por lo anterior, el objetivo de este trabajo de Tesis es diseñar y desarrollar diversas aplicaciones, sustentadas en los sistemas de instrumentación basados en PC, utilizando la herramienta de desarrollo *HP VEE* y los instrumentos HP34401A (multímetro digital), HP33120A (generador de funciones) y HP54603B (osciloscopio digital), para que sirvan de referencia, no solo para el laboratorio de Medición e Instrumentación, sino también en aplicaciones a nivel profesional.

Este trabajo esta organizado de la siguiente manera:

- En el capítulo 1 "Introducción" se describe de manera específica el objetivo de este trabajo.
- En el capítulo 2 "Sistema General de Medición" se hace la descripción general del sistema de medición, las etapas que lo componen y se incluye una sección donde se trata de manera breve la interfase GPIB, su protocolo y sus comandos.
- En el capítulo 3 "Herramienta de desarrollo" se describe ampliamente el uso y configuración de la herrameinta HP VEE para la programación de los instrumentos.
- En el capítulo 4 "Aplicaciones" se presentan las mismas con el propósito de mostrar la amplia gama de posibilidades de desarrollo que tiene la herramienta HP VEE.
- En el capítulo 5 "Conclusiones" se presentan las mismas como resultado de este trabajo.
- En el Apendice A "Comandos SCPI" se incluyen los comandos SCPI más comunes utilizados para manejar los instrumentos.

- En el Apendice B "Hoja técnica" se incluye la hoja técnica de un sensor de temperatura utilizado en la aplicación TEMPERATURA.VEE en la sección 4.3.

Capítulo 2

Sistema General de Medición

2.1 Introducción 12
2.2 Etapa detectora de variables 14
2.2.1 Transducción resistiva 15
2.2.2 Transducción capacitiva 16
2.2.3 Transducción inductiva 17
2.2.4 Transducción magnética 18
2.2.5 Transducción por voltaje, corriente y frecuencia . . 19
2.3 Etapa acondicionadora 21
2.4 Etapa de despliegue de información 27
2.5 Interfase GPIB 28
2.5.1 Protocolo 29
2.5.2 Comandos 32

2.1 Introducción

El sistema de medición mas sencillo consiste en un dispositivo que detecta la variable física y que visualiza o registra su valor en un medio adecuado.

Si se desea indicar el valor medido en un lugar situado a cierta distancia del punto de medida es preciso introducir un medio de transmisión de información entre el dispositivo de medida y el dispositivo de visualización (o almacenamiento). Determinados sistemas mecánicos utilizan un cable acoplado a engranes (velocímetro de un automovil) u otro enlace mecánico, los sistemas neumáticos utilizan tubos portadores de aire o se utilizan convertidores de presión a corriente y viceversa. Sin embargo, los sistemas de medición remotos más populares son los electrónicos.

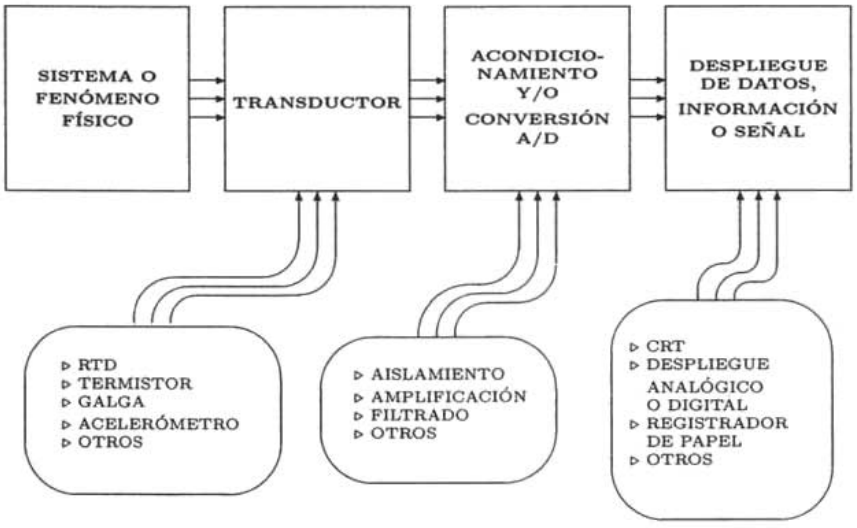


Figura 2.1: Sistema tradicional de medición

En la figura 2.1 la variable se detecta mediante un transductor cuya salida se acondiciona a un valor proporcional a la magnitud de entrada para su presentación y/o graficación. Este es el esquema tradicional del sistema de medición donde para cada variable a medir se necesita un dispositivo con los mismos bloques.

El sistema basado en PC de la figura 2.2 tiene los mismos bloques de la figura 2.1 pero la diferencia es la forma de presentación y el procesamiento de las magnitudes adquiridas. La cantidad de variables físicas que se pueden estar detectando casi al mismo tiempo aunque de manera secuencial pueden ser múltiples y la interfase que funciona de enlace entre la parte que mide

y la PC es quien gobierna el sistema, además de que los datos que se requieran almacenar en la PC pueden ser aprovechados de distintas maneras y conforme a las necesidades del momento.

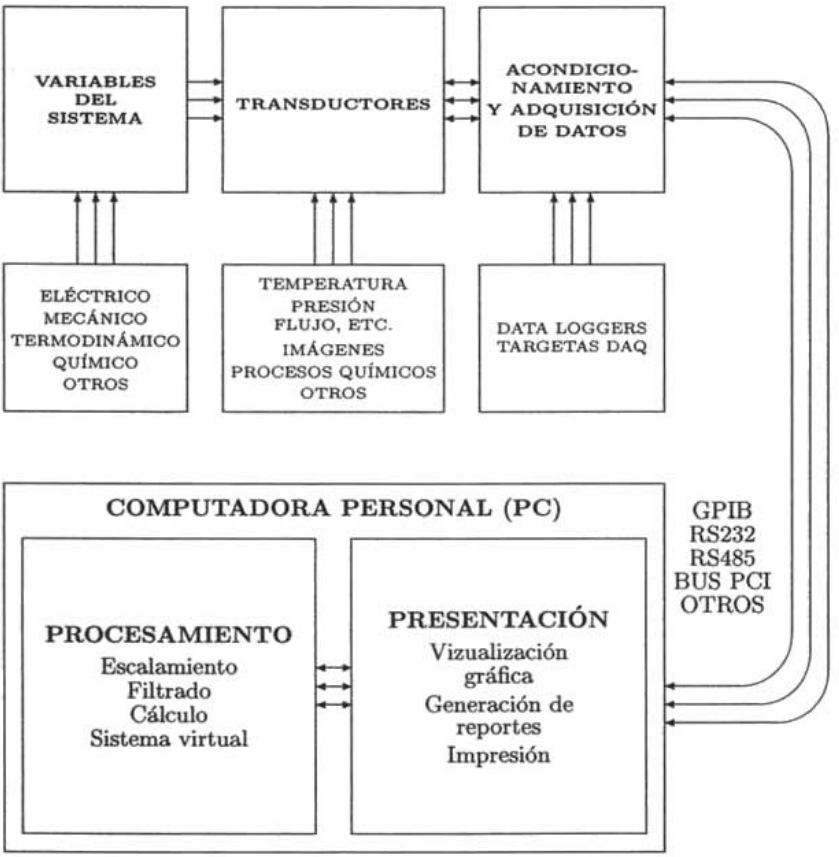


Figura 2.2: Sistema de medición basado en PC

2.2 Etapa detectora de variables

La etapa de detección de variables físicas involucra hacer uso de elementos transductores para cada magnitud en cuestión. Se denomina *transductor* (ref[10]) a todo dispositivo que es capaz de convertir la energía de una variable física (ya sea mecánica, térmica, magnética, eléctrica, óptica y nuclear) en otra forma de energía distinta a la original. En la práctica, los transductores mas comunes son los que tienen en su salida una forma de energía eléctrica (ya sea señal de voltaje continuo, corriente o frecuencia). En la etapa de transducción siempre se extrae cierta cantidad de energía del sistema donde se esta midiendo, por lo que es importante asegurar que esto no lo perturba o lo perturba en lo más mínimo posible de tal manera que no altere la magnitud de las variables involucradas.

Según el aporte de energía, los transductores se pueden dividir en *activos* y *pasivos*. En los transductores pasivos (como los potenciómetros), la energía de salida de las señal procede en su mayor parte de una fuente de energía auxiliar (de excitación). En los transductores activos la energía de salida es suministrada en su totalidad por el sistema (como el termopar).

Existen una gran variedad de transductores para casi cuaquier tipo de variable física con salida de tipo eléctrica, pero todos ellos se incluyen en cualquiera de los tipos de transducción como: la resistiva, capacitiva, inductiva, magnética, voltaje, corriente y frecuencia.

2.2.1 Transducción resistiva

Este es uno de los principios de transducción mas comunes. La resistencia es una propiedad que puede ser modificada por distintos métodos como el deslizamiento de un cursor en un reostato, aplicando un esfuerzo mecánico, variando la intensidad de luz, cambiando la temperatura, etc. en un dispositivo dado. En la siguiente tabla se muestran algunos de los transductores que se basan en este principio.

Dispositivo	**Funcionamiento**	**Aplicación**
Potenciómetro	El posicionamiento de un cursor por medio de una fuerza externa varia la resistencia de un potenciómetro	Presión, desplazamiento
Galga extensométrica	La resistencia de un alambre o semiconductor cambia según la elongación o compresión debida a los esfuerzos aplicados externamente	Fuerza, par, desplazamiento
Medidor de alambre caliente o medidor pirani	La resistencia de un elemento caliente varia enfriandolo con flujo de gas	Flujo de gas, presión de gas
Termometro de resistencia (RTD)	La resistividad de un metal (platino, niquel, germanio) es positiva en función de la temperatura, lo que causa un cambio en la resistencia del alambre.	Temperatura, calor radiante
Termistor	La resistencia de la mezcla de ciertos óxidos (niquel, manganeso,hierro, cobalto, etc.) de metal desciende con el incremento de la temperatura	Temperatura
Higrómetro	La resistencia de una cinta conductiva se altera con el contenido de humedad	Humedad relativa
Celda fotoconductiva	La resistencia de una celda se modifica con la luz incidente.	Relevador fotosensible, luz

2.2.2 Transducción capacitiva

Para este tipo de transducción se utiliza el capacitor de placas paralelas como base. Su principio de operación es el almacenamiento de carga eléctrica en el campo eléctrico formado por la parte interna de las placas. En este tipo de capacitor se pueden variar los parámetros de distancia y área para poder modificar el valor de la capacitancia.

$$C = \epsilon_0 \epsilon_r \frac{(n-1)A}{d} \tag{2.1}$$

C = capacitancia (F)
ϵ_0 = permitividad del vacio,8.85×10^{-11},F/m
ϵ_r = permitividad relativa (adimencional)
n = número de platos
A = área del plato,m^2
d = distancia entre las superficies del plato

En la siguiente tabla se dan algunos ejemplos de transductores de este tipo.

Dispositivo	**Funcionamiento**	**Aplicación**
Medidor de presión de capacitancia variable	Una fuerza aplicada externamente varia la distancia entre dos placas paralelas	Desplazamiento, presión
Microfono de capacitor	La presión del sonido altera la capacitancia entre una placa fija y un diafragma movil	Voz, música, ruido
Medidor dieléctrico	La capacitancia varia por cambios en el dieléctrico	Nivel de líquidos, espesor

2.2.3 Transducción inductiva

Este tipo de transducción ocurre cuando la variable a detectar cambia la inductancia una bobina. Un método común para cambiar la inductancia es mover un núcleo magnético dentro de una bobina polarizada.

Dispositivo	**Funcionamiento**	**Aplicación**
Transformador diferencial de variación lineal (LVDT)	El voltaje diferencial entre dos devanados secundarios de un transformador varia al mover el núcleo magnético por medio de una fuerza aplicada desde el exterior	Presión, fuerza, desplazamiento, posición
Medidor de corriente parásita	La inductancia de una bobina se altera por la proximidad de una placa con corrientes parásitas inducidas	Desplazamiento, espesor
Medidor de magnetrostricción	Las propiedades magnéticas cambian por presión y esfuerzos	Fuerza, presión, sonido
De inductancia variable	El desplazamiento de un núcleo móvil dentro de una bobina (alimentada con CA) aumenta la inductancia de forma casi proporcional a la porción metálica del núcleo contenido dentro de la bobina	Presión, desplazamiento
Tensión inducida	Se aprovecha la conductividad eléctrica del fluido para inducir un voltaje por medio de unos electrodos a ambos lados del conductor y un campo magnético constante que sea perpendicular al fluido	Flujo y velocidad de flujo
Generador de bobina móvil	El movimiento de una bobina en un campo magnético genera un voltaje (Ley de Faraday)	Velocidad, vibración

2.2.4 Transducción magnética

El Efecto Hall es el pricipio para este tipo de transducción, ocurre cuando un conductor con una corriente que fluye a través de el se coloca dentro de un campo magnético de tal manera que el campo eléctrico del conductor y el campo magnético tengan un ángulo recto entre si, cuando esto sucede, un voltaje aparece a través del conductor en una dirección perpendicular a la corriente y al campo magnético. Un cambio en el campo magnético produce un cambio en el voltaje de Hall, este voltaje puede ser usado para determinar la fuerza del campo magnético o para determinar la corriente en el conductor.

Dispositivo	**Funcionamiento**	**Aplicación**
Detector por efecto Hall	Generación de un voltaje a traves de una placa semiconductora (germanio) cuando un flujo magnético interactua con una corriente	Flujo magnético, corriente
De reluctancia variable	El circuito magnético se alimenta con una fuerza magnetomotriz constante (imán permanente o electroimán) que crea un campo magnético dentro del cual se mueve una armadura de material magnético, al cambiar la posición de la armadura varia la reluctancia y por lo tanto el flujo.	Presión, desplazamiento, vibración, posición
Tacómetro de pulsos	Un disco con una saliente de iman permanente cercano a una cabeza detectora fija (una bobina con núcleo magnético) que genera un voltaje en sus terminales debido al movimiento de la saliente del disco.	Velocidad angular, desplazamiento

2.2.5 Transducción por voltaje, corriente y frecuencia

Este tipo de transducción se lleva a cabo con elementos que obtienen la totalidad de energia necesaria para funcionar del sistema en cuestión y tienen en su salida una señal de corriente voltaje o frecuencia según sea el tipo de dispositivo y el principio de funcionamiento, este tipo de transductores se puede considerar en los de tipo pasivos o de autogeneración.

Dispositivo	**Funcionamiento**	**Aplicación**
Cámara de ionización	Se induce un flujo de electrones mediante la ionización de un gas debido a la radiación radiactiva	Conteo de partículas, radiación
Celda fotoemisiva	Hay una emisión de electrones debida a la radiación incidente en una superficie fotoemisiva	luz y radiación
Tubo fotomultiplicador	La emisión de electrones secundarios es debida a la radiación incidente sobre un cátodo fotosensible	luz y radiación, relevadores fotosensibles
Termopar y termopila	Se genera una fem por el calentamiento de la unión de dos metales diferentes o semiconductores	Temperatura, flujo de calor, radiación
Detector piezoeléctrico	Son materiales cristalinos (cuarzo y titanato de bario) que al deformarse fisicamente por la acción de una presión generan una señal eléctrica, cuando el cristal se deforma como consecuencia de la presión aplicada, se induce en su superficie un voltaje proporcional a la presión y al espesor del cristal.	Sonido, vibración, aceleración, cambios de presión
Celda fotovoltaíca	Se genera un voltaje en un dispositivo de unión semiconductora cuando la energía radiante estimula la celda	Medidor de luz, celda solar
Tacogenerador	Generador de voltaje de DC proporcional a la velocidad angular o de AC proporcional a la frecuencia	Velocidad angular

Fotodiodo, fototransistor	La juntura en el fotodiodo esta expuesta a la luz y la corriente a traves de el aumenta de manera casi proporcional a la intensidad luminosa, en el fototransistor la base es la que esta expuesta y sucede lo mismo que con el fotodiodo solo que la corriente que maneja es mayor.	Similar a la celda fotovoltaíca
Transductor de presión con salida de frecuencia	Tienen cristales de cuarzo como sensores, la frecuencia de salida varía de acuerdo al fabricante pero esta en el intervalo de 10 a 100kHz y para medir desde 0 hasta 2000 psi	Su uso en ambiente industrial incluye soluciones salinas, gases, aceites,alcohol, acidos, combustible
Transductor de luz con salida de frecuencia	Contiene un arreglo de fotodiodos que dependiendo del fabricante puede llegar hasta 100 elementos o mas acoplados a un convertidor corriente—frecuencia, la salida es un tren de pulsos proporcional a la intensidad de la luz que incide en el arreglo de fotodiodos. Dependiendo de la irradiancia la salida de frecuencia varia desde 0.01 Hz hasta 1000 kHz	Ajuste automático del enfoque de cámaras de video y fotografía, detección de la posición de objetos, detección de transición luz—obscuridad

2.3 Etapa acondicionadora

Una vez que se ha detectado la variable de interés es necesario hacerle una modificación conveniente con el fin de prepararla para que pueda ser utilizada por el instrumento, esto mediante un acondicionador de señal .

Los acondicionadores de señal son elementos del sistema de medición que ofrecen, a partir de la señal de salida del transductor, una señal preparada que permite un procesamiento posterior mediante un equipo digital (convertidor A/D, tarjeta adquisidora de datos) o instrumento estándar. Consisten normalmente en circuitos integrados o elementos pasivos (resistencia, inductor, capacitor) que ofrecen las siguientes funciones: acoplamiento de impedancias, amplificación, filtrado, linealización, entre otras.

Acoplamiento de impedancia

El acoplamiento de impedancia (buffer, figura 2.3) es probablemente el circuito de acondicionamiento más básico, se utiliza para cambiar la impedancia entre la señal generada por el transductor y el sistema de medición que va a utilizar dicha señal, con el propósito de aumentar la impedancia de entrada (disminuir la cantidad de energía del transductor hacia el sistema de medición) y hacer baja la impedancia de salida (para que la energía requerida por el sistema de medición no sea tomada directamente del transductor).

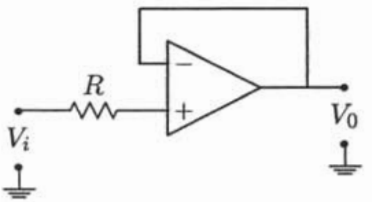


Figura 2.3: Buffer

Amplificador de Instrumentación

Tal véz el tipo de acondicionamiento mas frecuente es un cambiador de amplitud o nivel. Un cambio en la amplitud de la señal requiere de una atenuación o amplificación de la señal de entrada. La atenuación es generalmente llevada a cabo mediante el empleo de una red divisora de voltaje o de manera similar que la amplificación, la cual requiere de dispositivos activos como

transistores o amplificadores operacionales. En la práctica es común utilizar la atenuación y la amplificación para propósitos de acondicionamiento en muchos equipos electrónicos de prueba, tales como multímetros y osciloscopios con el objetivo de manejar intervalos de medición múltiples.

El circuito amplificador que se utiliza es el amplificador de instrumentación (figura 2.4), este es un circuito que tiene alta impedancia de entrada, alto rechazo a modo común (CMRR[1]), ganancia estable que se puede variar con una sola resistencia y sin que se contrapongan ganancia-ancho de banda (como sucede con un amplificador operacional que a mayor frecuencia la ganancia disminuye de manera dramática), tensión y corrientes offset bajas y con poca inestabilidad, e impedancia de salida también baja. Si todas las resistencias de la figura 2.4 son iguales a R con exepción de R_3, entonces la ganancia es igual a:

$$\frac{V_{sal}}{e_1 - e_2} = 1 + 2\left(\frac{R}{R_3}\right) \tag{2.2}$$

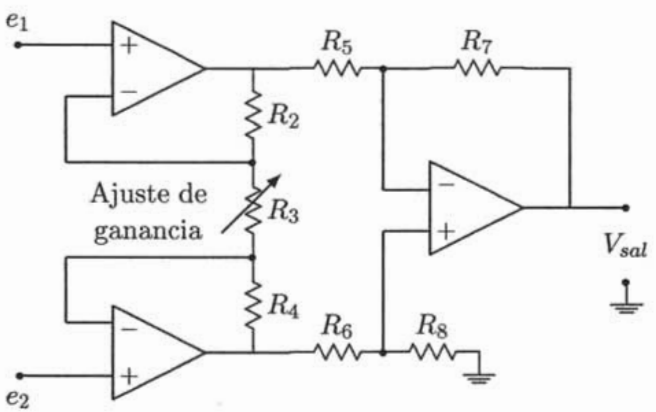


Figura 2.4: Amplificador de Instrumentación

Puente de Wheatstone

El acondicionamiento de señales frecuentemente requiere convertir la variación de un parámetro eléctrico en la variación proporcional de otro parámetro. Por ejemplo, muchos transductores tienen una indicación de cambio en alguna variable detectada por un cambio en su resistencia. Este cambio de

[1]Que rechaza las señales que son comunes a ambas entradas del circuito, como ruido ambiental, el más común es el de la línea de suministro de energía.

resistencia es convertido a un voltaje o corriente proporcional usando el transductor en un circuito puente (figura 2.5) o un circuito de amplificación.

Los circuitos de puente proporcionan un camino conveniente de convertir el cambio de resistencia a un cambio proporcional de voltaje. Estos son redes pasivas que son usadas para medir impedancias por comparación de potenciales. Cuando V_A iguala V_B en la figura 2.5, se dice que el puente esta balanceado. Cuando esto sucede no hay flujo de corriente hacia el detector D[2]. Por lo tanto, un puente balanceado indica una condición nula que sirve de referencia para hacer una calibración del mismo con respecto a un valor deseado. Cuando un transductor es usado en un circuito puente, su resistencia es generalmente igual a alguna de las resistencias que forman el puente en alguna condición deseada, tal como a una temperatura deseada, presión o intensidad de luz. Esto produce una condición nula. El puente se hace desbalanceado si el parámetro físico al cual el transductor es sensible esta cambiando de valor. Esto causa una diferencia de potencial entre los puntos A y B.

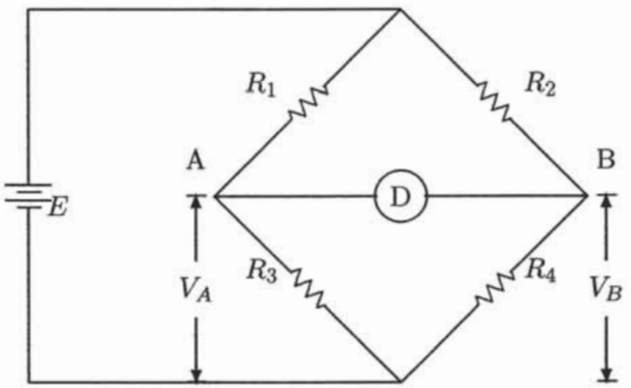


Figura 2.5: Puente de Wheatstone

Por ejemplo, si se decidiera que en el puente de la figura 2.5 $R1$ fuera la que indicara la variación de la temperatura, esta en realidad sería el transductor temperatura-voltaje.

[2]El detector hace la función de acondicionador ya que alguna de las resistencias dentro del puente es el transductor

Filtrado

En los ambientes industriales o simplemente en los laboratorios en los cuales los sistemas de adquisición de datos son instalados siempre hay tendencia de que las señales de ruido se introduzcan en el mismo. Estas señales indeseables son debidas frecuentemente a la línea de suministro de energía con frecuencias de 60 Hz, o a transitorios causados por cargas inductivas tales como el arranque de motores eléctricos. Tales interferencias pueden causar un error apreciable en la magnitud y frecuencia de la señal o señales de entrada, pero se reducen de manera significativa mediante el uso de un filtro (ver figura 2.6). Un filtro es un circuito que permite el paso de cierta banda de frecuencias, mientras atenua las señales de otras frecuencias. Los filtros pueden ser construidos como filtros pasivos usando solo resistores, inductores y capacitores, o filtros activos usando amplificadores operacionales con ganancia y retroalimentación.

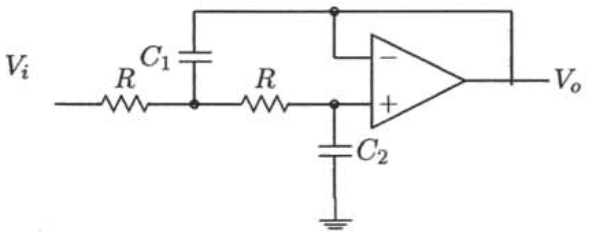


Figura 2.6: Filtro paso bajas

Para el circuito de la figura 2.6 (ref[19]), los capacitores se calculan de la siguiente manera con base en las frecuencias que se deseen suprimir: Las resistencias R se eligen a algun valor deseado (1kΩ, p.e.). El valor de los capacitores se encuentra utilizando la ecuación de escala

$$C_n = \frac{C_i}{2\pi f_p R} \tag{2.3}$$

R es el valor elegido de la resistencia y C_i se elige de una tabla dependiendo de la aproximación, del orden del filtro y el tamaño del rizo (Aproximación Chebyshev con rizo en la banda de paso o la de supresión, la aproximación Butterworth no tiene rizos en ninguna banda), f_p es la frecuencia de paso para la cual el filtro esta calculado.

Linealización

La linealización es muy importante en los sistemas de adquisición de datos. Aún los dispositivos cuya salida es razonablemente lineal pueden necesitar linealización adicional cuando se requieren mediciones precisas.

La salida del transductor puede ser linealizada usando un amplificador cuya ganancia sea función del voltaje de entrada y por lo tanto presentando una salida lineal. Por ejemplo, la señal de salida de un transductor puede variar exponencialmente con respecto a alguna variable, como temperatura, como se muestra en la figura 2.7, El voltaje de salida producido por el transductor puede ser expresado como:

$$V = V_o e^{\theta T} \tag{2.4}$$

Donde:

V	=	voltaje de salida del transductor a la temperatuta T
V_o	=	voltaje de salida del transductor a una temperatura de referencia
θ	=	constante del exponente
T	=	temperatura

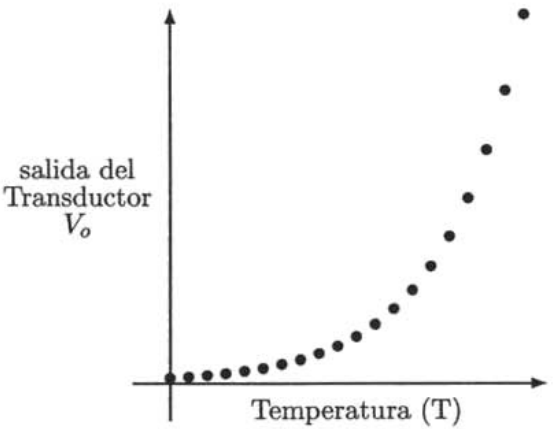


Figura 2.7: Curva de transferencia para un transductor que muestra una relación no lineal entre la temperatura y el voltaje de salida

Este voltaje puede ser linealizado usando un amplificador que su salida varie inversamente al voltaje de entrada y de acuerdo a la función logaritmo natural:

$$V_{out} = K \ln V_{in} \tag{2.5}$$

Donde:

V_{out}	=	voltaje de salida del amplificador
V_{in}	=	voltaje de entrada del amplificador
K	=	constante de calibración

Sustituyendo ec. 2.4 en la ec. 2.5 resulta:

$$\begin{aligned} V_{out} &= K \ln V_o e^{\theta T} \\ &= K(\ln V_o + \ln e^{\theta T}) \end{aligned}$$

Por lo tanto podemos decir que:

$$V_{out} = K \ln V_o + K\theta T \tag{2.6}$$

La ecuación 2.6 es una ecuación lineal. Por lo tanto, podemos ver que la salida del amplificador es lineal con respecto a la temperatura, pero con un factor de escala $K\theta$ y un nivel de DC $K \ln V_o$ como se muestra en la figura 2.8.

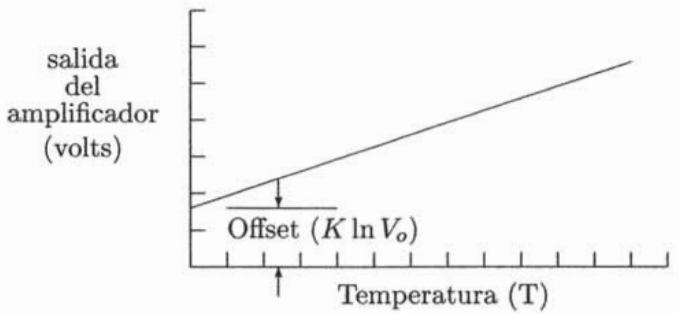


Figura 2.8: Señal del transductor linealizada con nivel de DC.

2.4 Etapa de despliegue de información

Para el despliegue se tienen diferentes formas de llevarlo a cabo aunque existan instrumentos de medición [3] que no dispongan de un elemento para vizualizar el valor. Estos instrumentos se les llaman ciegos ya que no tienen indicación visible de la variable y se utilizan principalmente como alarmas en termostatos (temperatura) y preostatos (presión). A diferencia de estos instrumentos, los instrumentos indicadores poseen una escala graduada concéntrica (medidores de presión, velocidad, temperatura) o exéntrica (medidores de flujo, vóltmetros, ampérmetros) o con dígitos en forma alfanumérica (termómetro, tacómetro digitales) y los instrumentos registradores, que hacen un trazo continuo o a puntos de la variable y pueden ser circulares[4] o de gráfico rectangular o alargado[5] (muestran el comportamiento de la variable en un periódo largo de tiempo).

El propósito de los instrumentos registradores es el de mostrar la relación entre dos o mas variables. Una variable es dependiente mientras la otra es independiente, la variable dependiente siempre debe estar en el eje Y y la variable independiente en el eje X.

Para hacer indicadores numéricos se pueden utilizar LED's en arreglos de 7 segmentos que proporcionan la vizualización de cualquier dígito numérico (0–9). Otra forma de presentar los datos numéricos y también caracteres alfanumérico es mediante un indicador de cristal líquido (LCD), este se usa en la mayoría de los multímetros aunque también los hay de carátula graduada.

Otro sistema para despliegue de datos pero en forma de gráfico es el osciloscopio el cual es un instrumento que permite observar la variación del nivel de voltaje de una señal.

La forma mas versatil de despliegue y que permite tener todos los tipos en un solo dispositivo es mediante un instrumento virtual el cual reside en la computadora como un programa que interactua con la interfase utilizada que en este caso es GPIB.

---

[3] Un instrumento de medición es un dispositivo que convierte una cantidad a ser medida en una forma útil para interpretación. Los instrumentos de medición reciben señales del acondicionador y proporcionan una salida comprensible para el usuario.

[4] Gráficos polares, son utilizados cuando es necesario indicar características direccionales como patrones de antenas y micrófonos, Son hechos con circulos concéntricos igualmente espaciados y líneas radiales desde el origen y marcadas con grados respecto al origen.

[5] De tipo lineal x vs y o logarítmico semilogx, semilogy o logx–logy.

2.5 Interfase GPIB

La interfase GPIB[6], también conocida como el estándar IEEE488 (publicado en marzo de 1975), fue diseñada para transferir datos entre instrumentos y la PC. Fue originalmente desarrollada en 1965 por Hewlett–Packard quienes le llamaron interfase HPIB[7], la norma GPIB fue también adoptada en europa por la IEC[8] bajo la denominación IEC625.1 que modificó el conector de 24 terminales (ver figura 2.9(a)) a 25 (solo en europa, ver figura 2.9(b)). La IEEE continuó sus trabajos para ampliar y mejorar la norma IEEE488-1975 que pasó a denominarse IEEE488.1, posteriormente en 1987 se publicó la norma IEEE488.2 y finalmente un consorcio de empresas fabricantes de instrumentos electrónicos publicaron en 1990 la norma SCPI[9] que define un conjunto de comandos de programación idéntico para todos los instrumentos desarrollados de acuerdo con ella, esto de manera independiente al tipo de interfase utilizada (RS232, GPIB, etc.). Hoy en día muchos sistemas, multímetros digitales, generadores de señales, interruptores y fuentes de poder programables pueden obtenerse con el estándar SCPI el cual da soporte a las normas HPIB, IEEE488.1 y IEEE488.2.

El IEEE488 es una interfase paralela asíncrona de 8 bits con señales de control especiales, el cual esta estandarizado en hardware y software y tiene las siguientes características:

- Ocho líneas para escritura y lectura de datos.
- Tres líneas de control de flujo (señales de sincronización).
- Cinco líneas especiales para el control del bus y las interrupciones.
- Velocidad de transferencia máxima de 1 Mbyte/s, limitada por la velocidad del receptor mas lento.
- Conectores especiales apilables.
- Un máximo de 15 instrumentos conectados a un bus común.
- 2 metros de longitud entre instrumentos (Máxima longitud total de 20 metros).
- El bus puede ser configurado en estrella, serie o una combinación con una sola tarjeta de interfase. Ver figura 2.10.

[6]General Purpuse Interface Bus
[7]Hewlett–Packard Interface Bus
[8]International Electrotechnical Commision
[9]Standard Commands for Programmable Instruments

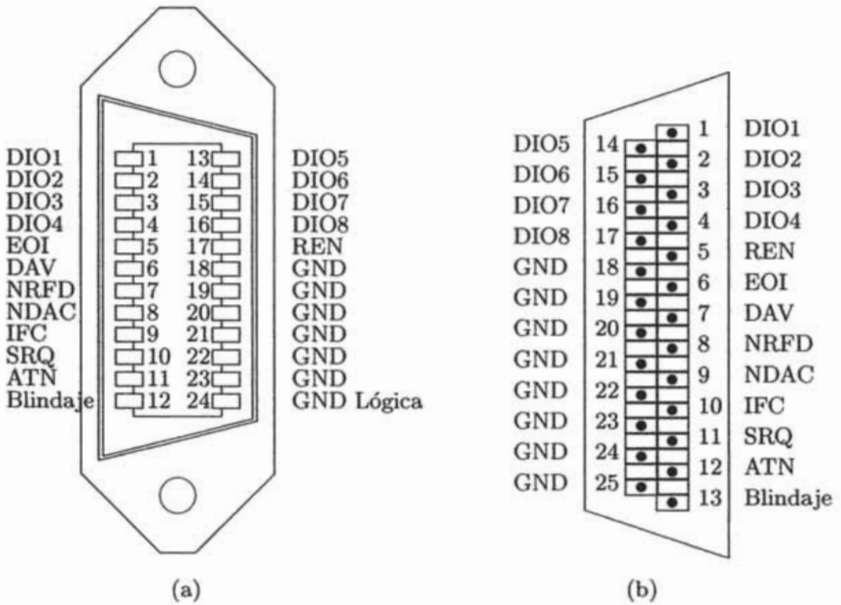


Figura 2.9: Conectores IEEE488, (a) americano, (b) europeo.

2.5.1 Protocolo

La interfase GPIB asigna un dispositivo como su *controlador del sistema*, el cual es un dispositivo que es capaz de especificar quien es emisor y quien es receptor para la transferencia de comandos o datos o si existe otro controlador pasarle el mando, en este caso este controlador recibe el nombre de *controlador activo* pero solo puede haber un solo controlador del sistema. El *controlador del sistema* es generalmente la tarjeta de interfase para la PC que esta siendo usada para gobernar todas las acciones, como en este caso solo se utilizará esta tarjeta como único controlador este será el controlador activo y del sistema y me referiré a el solo como controlador. El controlador maneja todos los accesos al bus y puede usar las líneas de control especiales para este propósito. Los dispositivos pueden ser *receptores*, que son capaces de recibir datos a través de la interfase cuando son direccionados para ese objetivo; tal es el caso de impresoras, fuentes de poder y otros intrumentos programables. Los *emisores* son dispositivos capaces de transmitir los datos en el bus cuando son direccionados para ese fin, ejemplo de estos dispositivos son vóltmetros, sistemas de adquisición de datos o cualquier otro instrumen-

to programable. Solo puede haber un solo emisor direccionado en la interfase a la vez, generalmente el controlador funciona como un receptor mientras que el dispositivo hace el papel de emisor. Todo esto pasa dinámicamente bajo instrucciones de software las cuales pueden ser manejadas por subrutinas y comandos que son proporcionadas por el fabricante de la interfase y del dispositivo.

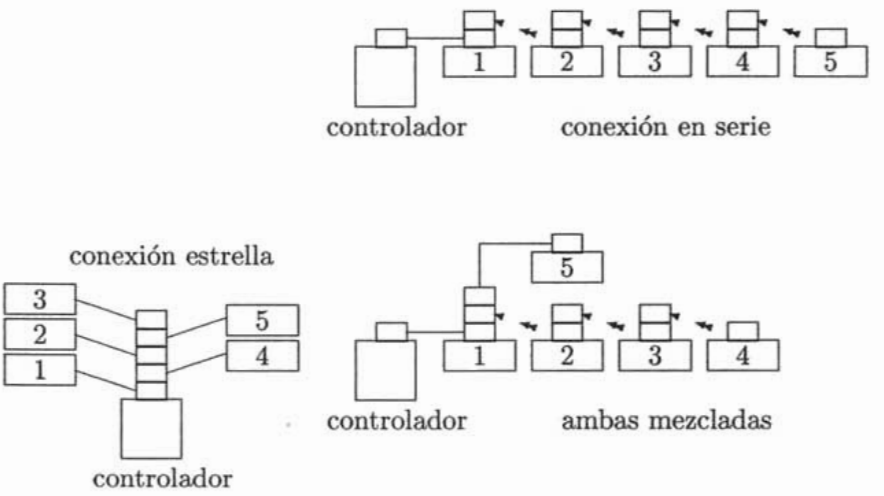


Figura 2.10: Diferentes topologías para GPIB

En los conectores de la figura 2.9, las ocho líneas (DI01–DI08) del bus de datos transportan los mismos en forma de bits en paralelo o un byte en serie a través de la interfase. También son utilizadas por el protocolo para enviar comandos y direcciones como bytes.

Las tres líneas de control (DAV[10] ,NDAC[11] ,NRFD[12]) operan en forma asíncrona (handshake). Dos de estas señales (NRFD y NDAC) se conectan a cada dispositivo por medio de una compuerta "OR" alambrada. Los receptores activos que no están listos para aceptar datos mantienen la línea NRFD en bajo, con este esquema, el emisor activo no puede enviar mas datos hasta que el receptor mas lento este listo, liberando la linea NRFD y mandandola a un nivel alto.

[10] data valid
[11] not data accepted
[12] not ready for data

El emisor controla la línea DAV para indicar cuando el dato esta listo para transmitirse, y los receptores controlan las líneas NRFD y NDAC para indicar su preparación para recibir los datos y la recepción de los datos, respectivamente. Como se muestra en el diagrama de tiempo de la figura 2.11, el emisor activo manda los 8 bits de datos y 2 μs después la línea DAV se manda a un nivel bajo. Este retraso de 2 μs proporciona el tiempo para que los datos lleguen a niveles válidos. Después de que la línea DAV se va a un nivel bajo, los receptores responden mandando la línea NRFD a un nivel bajo para prevenir alguna transferencia adicional de datos. Los receptores aceptan el dato (byte) a sus velocidades individuales. Cuando cada receptor ha aceptado el dato, este libera la línea NDAC. Cuando el último receptor libera la línea NDAC y la manda a un nivel alto. Cuando el emisor activo ve que la línea NDAC se va a un nivel alto, este se detiene de controlar el bus de datos y libera la línea DAV (nivel alto). Los receptores mandan la línea NDAC a un nivel bajo. Una vez que el controlador ha configurado los receptores y emisores, no toma lugar en la transferencia de datos.

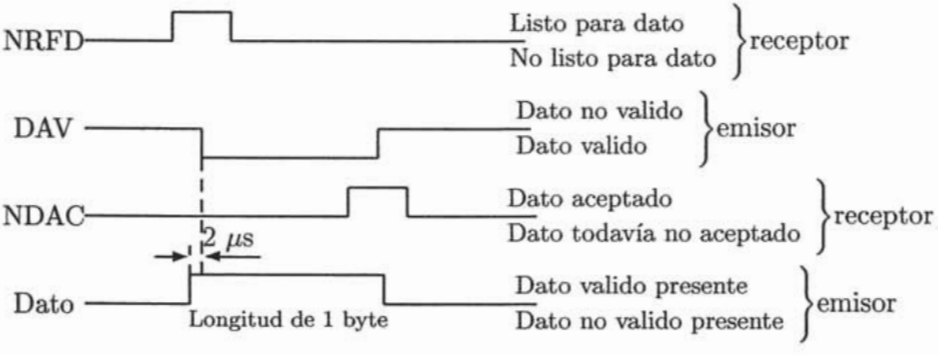


Figura 2.11: Diagrama de tiempo del bus IEEE-488

La operación del bus es generalmente controlada por la tarjeta de interfase en la computadora. Esta transmite comandos a otros dispositivos en el bus para hacerlos funcionar como emisores o receptores. Las dos maneras por medio de las cuales esto puede ser realizado es usando uno de los comandos de las líneas de control o emitiendo (dictando) comandos codificados sobre el bus mientras la línea ATN esta en alto.

2.5.2 Comandos

Comandos unilínea

Los comandos de *uni-línea* ocurren cuando una de las líneas dedicadas a controlar el bus y las interrupciones tiene un pulso enviado sobre ella, y estos afectan a todos los instrumentos que están conectados y son definidos como sigue:

Interface Clear (IFC). Todo el intercambio de datos que existe en el bus se detiene incondicionalmente cuando IFC es confirmado por el o los dispositivos en $100\mu s$. Solo el controlador del sistema puede usar IFC. Todos los emisores y receptores se reestablecen a su estado no direccionado y el monitoreo serial es deshabilitado.

Remote Enable (REN). Es usado por el controlador del sistema para poner a los dispositivos en modo de programación remota. El comando REN permite que los instrumentos en el bus sean programados una vez que han confirmado el comando. Si REN es verdadero, cualquier dispositivo direccionado puede ser programado por el controlador.

Attention (ATN). Si ATN es confirmado, entonces el dato en el bus es un comando y el dispositivo debe de ser capaz de responder al controlador en 200 ns y detener su tarea actual. Es utilizado para direccionar emisores, receptores y obtener estados de los dispositivos. Esto permite a los dispositivos discriminar entre transmisión de datos normales en el bus y comandos especiales multi-línea los cuales pueden necesitar que una acción sea tomada por los dispositivos direccionados.

End or Identify (EOI). Cuando ATN es confirmado, la línea EOI es usada por el controlador activo para realizar el monitorea paralelo, cuando ATN es falso, EOI puede ser usada po el emisor para indicar el último de una ráfaga de bytes.

Comandos multilínea

Los *comandos multi-línea* o *universales* son enviados a los dispositivos en el bus cuando ATN es confirmado (verdadero). Los dispositivos conectados al bus que no se les ha asignado el estado de receptor no reciben datos del bus, pero reaccionan a comandos direccionados. Los comandos universales actuan sobre todos los dispositivos conectados al bus, incluyendo a los no-receptores. La intención de los comandos universales definidos por el

estándar IEEE488 es ser útiles de manera general, pero acciones específicas dependerán sobre la forma en la cual cada comando sea implementado en un instrumento en particular. Los comandos universales y sus acciones son las siguientes:

Device Clear (DC). Causa que los todos los dispositivos regresen a su estado inicial (esto es dependiente del dispositivo).

Local Lockout (LLO). Este previene contra la modificación de parámetros del dispositivo desde su panel frontal por el operador. Si se requiere del control del panel frontal, el controlador puede direccionar el instrumento y puede mandar un comando Go To Local (GTL). El uso de LLO previene interferencia (diferente del apagado del dispositivo) por parte del operador en el panel frontal cuando las mediciones se están llevando a cabo.

Parallel Poll Unconfigure (PPU). PPU reestablece todos los dispositivos de monitoreo paralelo a un estado de descanso (IDLE). No se puede obtener una respuesta de estado de monitoreo en paralelo después de que se ha ejecutado el PPU.

Serial Poll Enable (SPE). SPE causa que los dispositivos confirmen el servicio de petición de línea (SRQ) cada vez que una condición de interrupción ocurra y proporciona uno o dos bytes para reportar el estado cuando son revisados por el controlador.

Serial Poll Disable (SPD). Contrarresta los efectos de SPE y regresa a todos los dispositivo a su configuración de emisor.

Comandos direccionados

El siguiente grupo de comandos multi–línea se les conoce como *Comandos Direccionados* o *Comandos Primarios.* Solo dispositivos a los cuales se les ha direccionado ser receptores por un comando anterior responderán a los comandos direccionados. Los resultados específicos de ejecución de estos comandos deben ser revisados en el manual del instrumento. Los comandos direccionados reconocidos por la especificación IEEE488 son los siguientes:

Group Execute Triger (GET). Este comando dispara todos los dispositivos direccionados como receptores para realizar algún servicio tal como tomar una lectura de datos y regresarlos sobre el bus. También permite la sincronización de mediciones entre múltiples dispositivos. Los detalles de la respuesta son dependientes del dispositivo.

Selected Device Clear (SDC). Cuando un receptor reconoce un SDC, este se inicializa a su estado de encendido. SDC solo reestablece a aquellos dispositivos que son receptores.

Go To Local (GTL). Este es también específico a los dispositivos receptores, pero contrarresta el efecto del comando universal LLO. Causa que un dispositivo permita la modificación de parámetros a traves de su panel frontal.

Parallel Poll Configure (PPC). Es usado para programar la respuesta de monitoreo paralelo de un dispositivo específico. Una respuesta única debe ser configurada para cada dispositivo.

Take Control (TCT). Este comando permite al controlador activo asignar el control del bus a otro controlador. Solo un dispositivo puede ser el controlador activo. Puede no parecer obvio el por qué la necesidad del comando TCT, pero es útil si dos o mas computadoras van a tomar lecturas de un instrumento en común.

Comandos secundarios

Los dos comandos adicionales se les conoce como *comandos secundarios* y se refieren completamente a técnicas de monitoreo de estados usadas para permitir al controlador decidir cuales dispositivos necesitan atención.

Parallel Poll Enable (PPE). Este comando requiere que cada dispositivo en el bus haya sido configurado para tener una respuesta de monitoreo única. PPE es usado después de que el controlador ha avisado a los otros dispositivos que se detengan de recibir, y seleccionado dispositivos para escuchar en secuencia, y ha usado el comando PPC para asignar una respuesta única para cada dispositivo de manera separada.

Parallel Poll Disable (PPD). Contrarresta los efectos de PPE para prevenir escuchar a los dispositivos que proporcionan respuestas de monitoreo paralelo.

Direccionamiento y Comandos con dirección secundaria

Cada instrumento y el controlador tienen un número de dirección física primaria en el intervalo de 0-31. La dirección se determina normalmente fijando unos interruptores en el exterior del instrumento (para instrumentos que tienden a ser obsoletos en los que son mas modernos esto se hace

por software) antes de usarlo, muchos dispositivos despliegan su dirección cuando se encienden o se presiona un botón de su panel frontal.

Como los dispositivos pueden ser configurados como emisores o receptores, el byte de dirección para ser receptor se calcula sumando el número decimal 32 a la dirección base (física), mientras que para ser emisor el byte de dirección se calcula sumando el valor decimal 64 a la dirección base.

Los bytes o comandos de dirección se conocen como TAD (Talk ADress) y LAD (Listen ADress) (ver tabla 2.7) o simplemente TA y LA.

El controlador también tiene una dirección para ser receptor y otra para emisor que le permiten transferir datos. Las direcciones 21 y 30 son direcciones comunes para emisor/receptor en equipo Hewlet–Packard aunque puede ser cualquiera. Si un dispositivo se configura con una dirección ya existente pueden ocurrir problemas sin lógica o incomprensibles. Las direcciones del controlador para emisor y receptor son conocidas como MTA (My Talk Adress) an MLA (My Listen Adress).

Los comandos que permiten que los dispositivos reciban, emitan y reciban direcciones secundarias se les llama *comandos con dirección secundaria*, estos comandos son recibidos en el bus por todos los dispositivos, pero solo aquellos dispositivos que han sido asignados como receptores y emisores actuan de acuerdo al comando que tiene la dirección incluida como parte del mismo. Los comandos con dirección secundaria son interpretados por los dispositivos direccionados como receptor o emisor por el comando anterior (es decir que si después de que al dispositivo se le envió un LAx o TAx, se le envía la dirección secundaria a la cual se le conoce como SC), con esto es posible tener hasta 961 direcciones (31×31). La línea ATN es confirmada con estos comandos como con los comandos de control ya mencionados anteriormente. Cada dirección de comando de receptor (LAx) es un byte ASCII en el intervalo numérico de 32-62, el cual corresponde a las direcciones 0-30. Los comandos de dirección de emisores (TAx) son bytes ASCII en el intervalo de 64-94 para las direcciones de 0-30. Los comandos de dirección secundaria (SCx) son enviados con valores numéricos en el intervalo de 96-127 para direcciones secundarias de 0-31. Los direcciones secundarias son utilizadas en sistemas de instrumentación modulares donde se tienen diferentes estántes o chasises que contienen diferentes interfases y sistemas. Finalmente, dos comandos son proporcionados para que todos los dispositivos dejen de escuchar UNL (unlisten) o hablar UNT (untalk).

DEC	HEX	IEEE488	tipo
1	01	GTL	direccionado
4	04	SDC	direccionado
5	05	PPC	direccionado
8	08	GET	direccionado
9	09	TCT	direccionado
17	11	LLO	universal
20	14	DCL	universal
21	15	PPU	universal
24	18	SPE	universal
25	19	SPD	universal
32-62	20-3E	LAx	dir. receptor
63	3F	UNL	—
64-94	40-5E	TAx	dir. emisor
95	5F	UNT	—
96-127	60-7F	SCx	dir. secundaria

Tabla 2.7: Codigos IEEE488

Capítulo 3

Herramienta de desarrollo

3.1 Ambiente de desarrollo de HP VEE 40
3.1.1 Operaciones en el ambiente de desarrollo 42
3.1.2 Objetos 45
3.1.3 Reglas de propagación en los objetos 47
3.1.4 Depuración de un programa 48
3.2 Creación de programas de prueba 50
3.3 Lectura y escritura de datos 55
3.4 Formas de controlar instrumentos 61
3.4.1 Utilización del INSTRUMENT MANAGER 61
3.4.2 Objeto DIRECT I/O 63
3.4.3 PANEL DRIVERS 66
3.4.4 COMPONENT DRIVER 67
3.5 Análisis y despliegue de datos 69
3.6 Funciones en HP VEE 73
3.6.1 Objetos de usuario 73
3.6.2 Funciones de usuario 74
3.6.3 Librerías 76

HP VEE es una herramienta de programación visual que esta optimizada para aplicaciones científicas e ingenieriles, particularmente de prueba y medición. El usuario escribe programas conectando bloques, muy parecido al crear un diagrama de flujo. Una vez que el programa esta completo, el usuario puede crear una "vista de panel" para ocultar la implementación y generar una interfase al usuario.

HP VEE incluye objetos para:

- Control de flujo (como ciclos y estructuras condicionales).
- Realización de operaciones matemáticas (incluyendo álgebra de matrices y booleana, probabilidad y estadística, cálculo y procesamiento de señales.)
- Procesamiento de cadenas de caracteres.
- Lectura y escritura de archivos (incluyendo la habilidad de crear y leer registros que contienen diferentes tipos de datos e información, y "conjuntos de datos" que consisten en conjuntos de dichos registros.)
- Desplegado y graficación de datos.

El usuario puede construir interfases que desplieguen objetos como termómetros, tanques, y alarmas, objetos para desplegar mapas de bits y objetos para recibir una entrada de datos.

También el usuario puede construir sus propios objetos ("UserObjects") de elementos de HP VEE, convertirlos a funciones ("UserFunctions"), y crear librerías de tales funciones.

HP VEE puede (casi de manera transparente) usar una gran variedad de tipos de datos como coordenadas y números complejos. Se pueden examinar facilmente los contenedores a los que HP VEE transfiere el flujo de datos para examinar su contenido. HP VEE también incluye herramientas de depuración que permiten seguir la ejecución y el flujo de datos dentro del programa, examinar los datos en las líneas y terminales de entradas y fijar puntos de ruptura. Muy importante, HP VEE proporciona tres niveles de control para un instrumento:

1. **INSTRUMENT DRIVERS** que simula el panel frontal de un instrumento el cual permite el acceso al panel frontal (desde la pantalla de la PC) del mismo y permite hacerle configuraciones sencillas, también se le pueden agregar entradas y salidas de acuerdo a las funciones específicas del instrumento.

2. **COMPONENT DRIVERS** Tiene la misma funcionalidad que el INSTRUMENT DRIVER, la diferencia es que en este objeto no aparecen las funciones del instrumento de manera gráfica. Si se necesita ejecutar una función específica es necesario agregar una entrada (o entradas) de datos y de igual forma si se requiere recuperar un dato leido del instrumento se le tiene que agregar una salida (o salidas).

3. **DIRECT I/O** que proporciona control a nivel de comandos sobre un instrumento a través de la interfase GPIB.

HP VEE proporciona la interfase **BUS I/O MONITOR** para permitir a los usuarios seguir y revisar los comandos enviados al instrumento[1].

[1]Los comandos vistos en el capítulo de la Interfase GPIB y en el Apéndice A

3.1 Ambiente de desarrollo de HP VEE

HP VEE es una herramienta que reduce de manera importante el tiempo de desarrollo de aplicaciones de medición y prueba ya que las mismas se construyen conectando íconos de manera similar al ensamblado de un diagrama de bloques (ver figura 3.1).

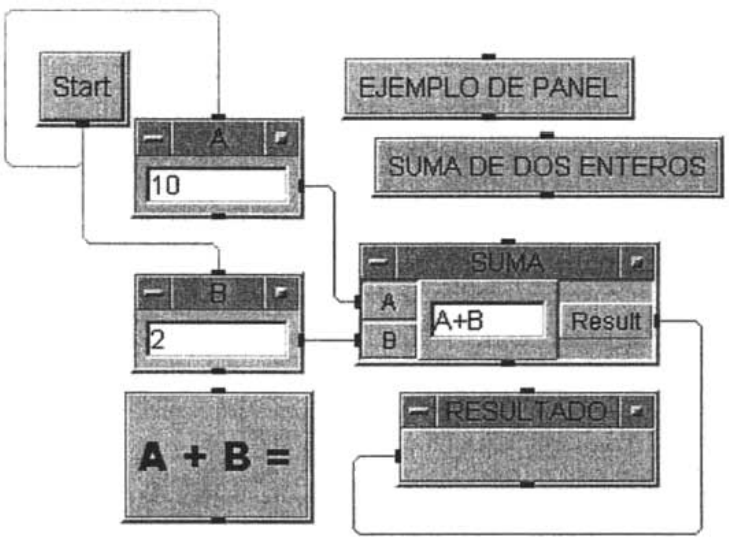


Figura 3.1: Ejemplo de programa.

Existen dos diferentes posibilidades de iniciar una sesión en HP VEE:

1. Presionando el botón de INICIO⇒PROGRAMAS⇒HP VEE⇒HP VEE.
2. En el escritorio se le da doble click al ícono de acceso directo a HP VEE.

Una vez iniciada la sesión se observará el ambiente que se muestra en la figura 3.2, donde se pueden identificar 5 áreas:

1. La **Barra de menús** proporciona los comandos e íconos para la construcción de programas.

    - **File**, permite el guardado y recuperado de programas, así como también hacer la configuración general de HP VEE.

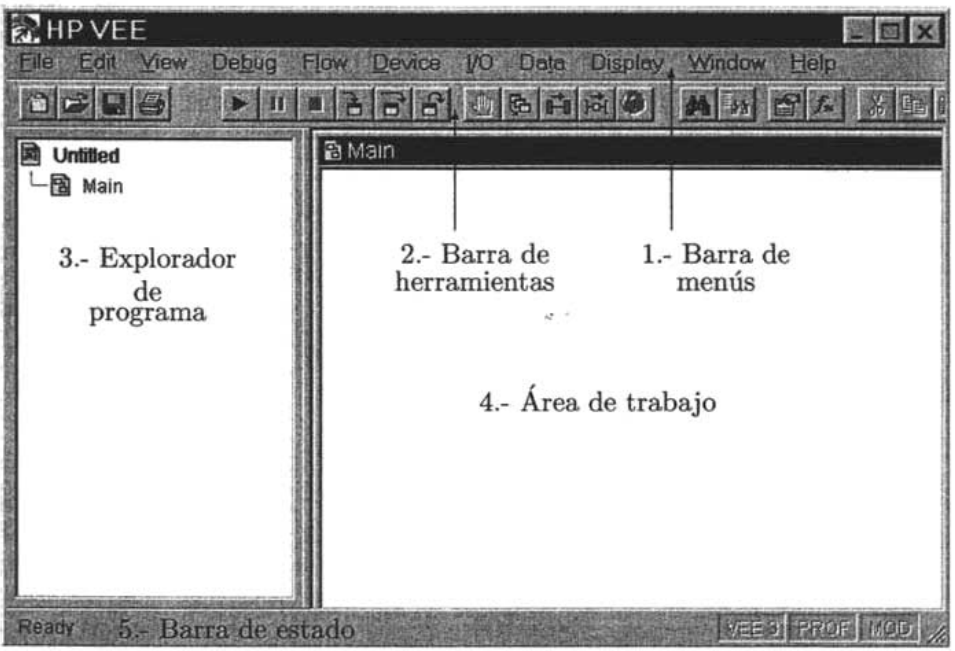


Figura 3.2: Ambiente de desarrollo de HP VEE

- **Edit**, permite la selección, copiado y borrado de objetos o grupos de objetos de HP VEE así como también el acceso a la utilería **Find** para localizar los diferentes elementos del programa.
- **View**, permite inspeccionar las vaiables de HP VEE, selección de diferentes ventanas del espacio de trabajo y acceso al perfilador el cual permite determinar los niveles de carga del programa.
- **Debug**, controla las herramientas de revisión (paso a paso, puntos de revisión, seguimiento) para el ambiente de HP VEE.
- **Flow**, proporciona objetos para operaciones condicionales, ciclos, modificación de flujo de datos y retardos.
- **Device**, proporciona objetos para operaciones matemáticas, manejo de funciones y librerías de usuario, generadores de señales virtuales y varias herramientas de programación útiles (contadores, acumuladores, temporizadores, registros de corrimiento, comparadores).

- **I/O**, permite la configuración y selección de objetos de programación de E/S tales como Direct I/O, drivers plug & play, y drivers de HP VEE, etc.
- **Data**, proporciona acceso a herramientas para el manejo de datos como objetos lista, cajas de diálogo, acceso a arreglos y variables.
- **Display**, proporciona acceso a objetos para hacer despliegues de texto, gráficas e indicadores.
- **Help**, despliega ayuda en línea (con acceso a programas de ejemplo).

2. La **Barra de herramientas** contiene botones que funcionan como atajos a las tareas más comunes de HP VEE, si se posiciona el ratón sobre cualquiera de estos botones, aparecerá un mensaje que indica su función.

3. El **Explorador de programa** muestra la jerarquía de funciones del programa.

4. El **Área de trabajo** es donde se construye el programa con íconos, cuando se hace una función de usuario (o un objeto de usuario), esta tendrá su propia ventana y se podrá navegar facilmente entre ellas utilizando el explorador de programa.

5. La **barra de estado** explica el objeto que se selecciona así como también proporciona información acerca del modo de programación actual.

3.1.1 Operaciones en el ambiente de desarrollo

- Guardado, cerrado y comienzo de un nuevo programa (opción por omisión).
  - FILE⇒SAVE AS (SAVE o presionando el botón del disco flexible y completar la ventana de diálogo).
  - FILE⇒EXIT,o *CTRL-E.*
  - FILE⇒NEW,*CTRL-N* o presionando el botón de página en blanco.

- Ejecutar, detener, detener momentaneamente un programa.
  - DEBUG⇒RUN/RESUME, *CTRL-G* o presionando el botón run.
  - DEBUG⇒STOP o presionando el botón stop.
  - DEBUG⇒PAUSE, *CTRL-P* o presionando el botón pause.
- Borrado de un objeto del área de trabajo.

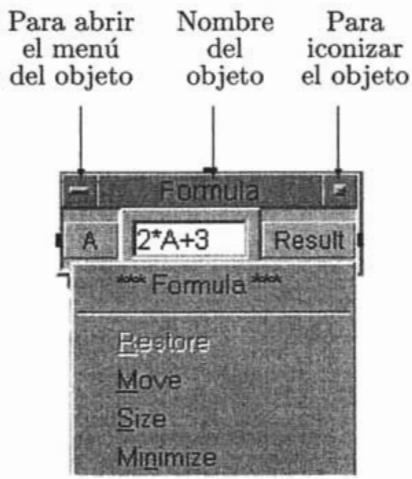


Figura 3.3: Estructura general de la barra del objeto.

  - Se posiciona el ratón sobre el objeto, se abre su menú como en la figura 3.3 y se selecciona *CUT*, en el menú EDIT⇒CUT o se selecciona el objeto y se presiona *CTRL-X* o se presiona el botón cortar de la barra de herramientas.
- Copiado y pegado de un objeto.
  - Para copiar un objeto se selecciona el mismo, se presiona *CTRL-C*; o en el menú EDIT⇒COPY; o en el botón copy.
  - Para pegar un objeto (previamente se ha copiado) se selecciona el menú EDIT⇒PASTE o *CTRL-V* o se presiona el botón paste.

- Duplicado de un objeto.
  - Se abre el menú del objeto y se selecciona *CLONE*, se posiciona en el lugar deseado.
- Modificación del nombre de un objeto.
  - Se abre el menú del objeto y se selecciona *PROPERTIES*, aparece una ventana de diálogo con el nombre actual resaltado. Se escribe el nuevo nombre y se presiona OK.
- Modificación del nombre de la terminal de un objeto.
  - Se hacen dos clicks en el nombre de la terminal, con lo que aparecerá una caja de diálogo en la que está resaltado el nombre de la misma, se modifica y se presiona el botón OK, en esta misma caja de diálogo se puede seleccionar el tipo de dato y forma esperado en la terminal del objeto, pero esto generalmente no se modifica ya que los objetos hacen la conversión al tipo de dato en forma automática.
- Selección de uno o varios objetos.
  - Para un solo objeto se le da un click sobre el mismo; para varios objetos se presiona *CTRL* y se dibuja un rectángulo alrededor de los objetos a seleccionar, o sin soltar la tecla *CTRL* se van seleccionando uno por uno los objetos deseados. Para seleccionarlos todos, se selecciona en el menú EDIT⇒SELECT ALL.
- Creación, borrado y tipos de líneas de flujo de datos entre objetos.
  - Para unir dos objetos con una línea de flujo se hace click sobre la terminal de salida de datos o secuencia de salida del objeto y se mueve el ratón hacia la terminal de entrada o secuencia de entrada de otro objeto (ver figura 3.4).
  - Para borrar una línea de flujo se posiciona el ratón sobre el área de trabajo, se abre su menú y se selecciona *DELETE LINE* (con lo que aparecen unas tijeras y se le da click a la línea que se quiera eliminar), o se selecciona en la barra de menús EDIT⇒DELETE LINE.
  - En el ambiente de desarrollo, HP VEE asigna diversos colores a las líneas de datos que se conectan entre ellos, este color esta basado en el tipo de datos que fluye por dicha línea:

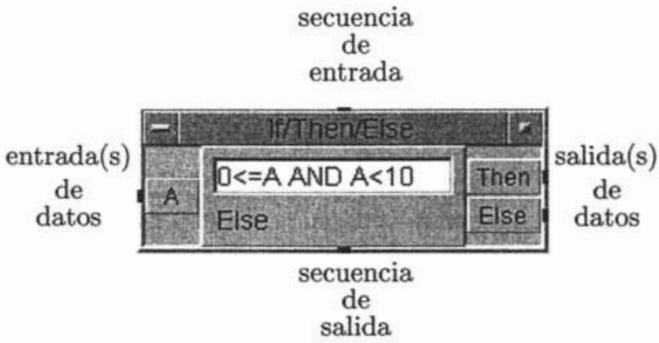


Figura 3.4: Estructura general de objetos y sus terminales.

  * AZUL: numérico de tipo entero o real.
  * AZUL: complejo de los tipos complex y Pcomplex.
  * NARANJA: Cadenas de caracteres.
  * GRIS: secuencia(no tiene valor).
  * NEGRO: tipo de datos desconocido.

- Desplazamiento del área de trabajo.
  - (Hay que asegurarse de que haya al menos un objeto en el área de trabajo). Se necesita poner el ratón en cualquier parte del fondo del área de trabajo. Presionar y sostener el botón izquierdo del ratón y mover el área de trabajo en cualquier dirección.

En HP VEE existen dos vistas, la detallada, la cual muestra las conexiones entre los diferentes objetos y que corresponde al código fuente en un lenguaje textual (como en la figura 3.1); y la vista de panel (como en la figura 3.5) que es una vista que esconde las conexiones entre los objetos y solo presenta los resultados o los indicadores que en ella se hayan dispuesto.

Para crear un panel se seleccionan los objetos que se deseen que aparezcan en el y se selecciona la opción ADD TO PANEL en el menú EDIT o presionando el botón ADD TO PANEL.

3.1.2 Objetos

Todos los objetos tienen una estructura general como se muestra en las figuras 3.3, 3.4, 3.6. Para que puedan funcionar correctamente todas las

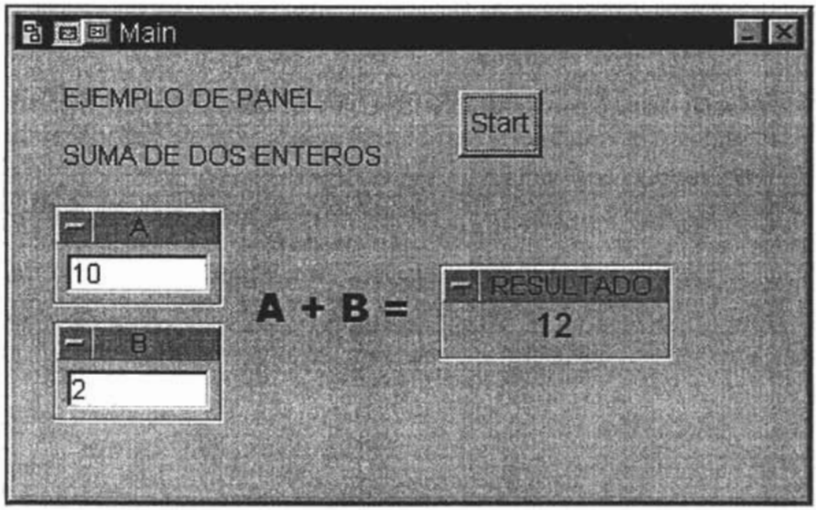


Figura 3.5: Vista de panel.

Figura 3.6: Estructura general del objeto como ícono.

entradas del objeto tienen que estar conectadas de lo contrario se produce un error.

El orden de los eventos que se llevan a cabo dentro de un objeto es el siguiente (ver figura 3.7):

1. Si la terminal de secuencia de entrada esta conectada, el objeto no operará hasta que reciba el mensaje de ejecución (un "ping"). De cualquier manera, la terminal de secuencia de entrada no tiene que estar conectada.

2. Todas las terminales de entradas de datos deben de tener datos antes de que el objeto opere, (se pueden agregar terminales de entrada/salida de datos en la mayoría de los objetos).

3. El objeto lleva a cabo su tarea. En este caso se calcula el seno de A ($\sin(A)$) y el resultado se deposita en la terminal de salida.

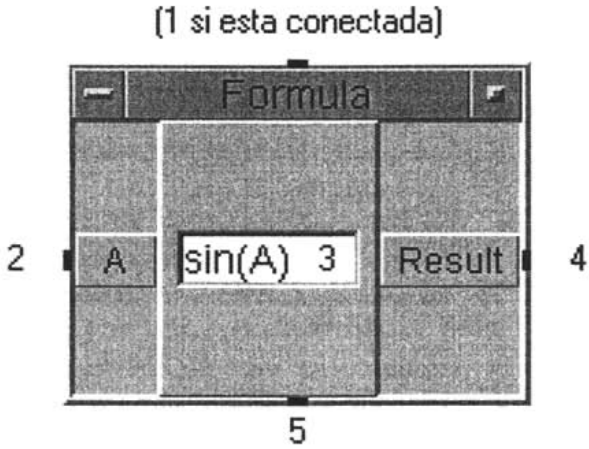


Figura 3.7: Orden de los eventos de un objeto.

4. La terminal de salida de datos se dispara. El objeto espera una señal del siguiente objeto de que el dato fue recibido antes de que su operación sea completada.
   Por lo tanto, un objeto dado no dispara su terminal de secuencia de salida hasta que todos los objetos conectados a su terminal de salida de datos han recibido datos.

5. La terminal de secuencia de salida de datos se dispara.

3.1.3 Reglas de propagación en los objetos

El siguiente es un resumen de las reglas de propagación que sigue HP VEE cuando un programa se ejecuta en modo compilado[2]:

- Los datos fluyen a través de los objetos de izquierda a derecha, las secuencias fluyen de arriba hacia abajo.

- Todas las terminales de datos y la terminal XEQ[3] deben de estar conectada.

[2]En HP VEE hay dos modos disponibles, el Compilado (rápido) y compatible VEE 3 (lento), para mayor información referirse a la ayuda en línea.

[3]Es una terminal que forza la operación de un objeto aun si las terminales de entrada o secuencia de entrada no han sido activadas. Esta terminal solo esta disponible en los objetos COLLECTOR y SAMPLE & HOLD.

- Los objetos que no tienen terminal o secuencia de entrada operan primero.
- Todas las terminales de entrada deben ser activadas antes de que un objeto opere (excepto el objeto JCT[4]).
- Si la terminal de secuencia de entrada de un objeto esta conectada, esta debe ser activada antes de que el objeto pueda operar.
- Los objetos operan una vez a menos que se les conecte un objeto de ciclo (For Count etc.), o a menos que sea forzado a operar por la terminal XEQ.
- Las terminales de control[5] se ejecutan de manera inmediata y no causan que el objeto opere o se propague.
- Cuando se genera un error en un objeto en su terminal de error[6], la terminal de error se propaga en lugar de la salida de datos. De cualquier modo, la terminal de secuencia de salida se activa, (si no hay terminal de error, se despliega un mensaje de error).
- Subtareas en paralelo operan en cualquier orden.
- Tareas múltiples operan en cualquier orden.

3.1.4 Depuración de un programa

Para la eliminación de fallas en un programa, HP VEE cuenta con el menú debug en el cual destacan los siguientes comandos, que también tienen sus correspondientes botones para acceso rápido:

- **Show Data Flow**, Muestra la ruta que toma el dato a través del programa. Un pequeño marcador cuadrado es usado para trazar el flujo del dato a lo largo de las líneas.

[4]Es un objeto que manda el valor mas actual a la salida, para mayor información ver la ayuda en línea.

[5]Es una terminal de entrada opcional asíncrona que transmite datos al objeto sin esperar a que el objeto contenga datos en sus otras terminales, son usados comunmente para limpiar o auto-escalar el display.

[6]Es opcional que atrapa cualquier error que ocurra en un objeto. En lugar de obtener un mensaje de error, el número de error es mandado a esta terminal.

- **Show Execution Flow**, Resalta el objeto que se esta ejecutando actualmente. El borde del objeto se resalta y así permanece hasta que termina su operación. La transacción que se esta ejecutando actualmente se resaltará también en los objetos que incluyan una lista de transacciónes tales como *Direct I/O, To File, From File*, etc.

- **Step Over**, Cada vez que se lleva a cabo, ejecuta el programa operando un objeto a la vez, sin abrir objetos y funciones de usuario. El siguiente objeto a operar tendrá el borde amarillo.

- **Step Into**, Cada vez que se lleva a cabo, se ejecuta el programa objeto por objeto, incluso los objetos dentro de los objetos y funciones de usuario, pero no en objetos que utilicen transacciones y no esta disponible para objetos que han sido asegurados. El siguiente objeto a operar tendrá el borde de color amarillo.

- **Step Out**, Completa la ejecución del objeto o la función de usuario (cuando se ha utilizado Step Into como comando anterior) y se detiene momentaneamente en el siguiente objeto.

- **Line Probe**, Despliega la información del contenedor que se transmite en una línea entre dos objetos. Cuando se selecciona la línea a revisar (dandole un click) aparece una lupa como apuntador del ratón y se abre una caja de diálogo que nos proporciona la información.

- **Object Probe**, Tiene la misma función que Line Probe solo que con este comando no es necesario dar un click sobre la línea ya que el valor aparece de manera automática.

3.2 Creación de programas de prueba

En esta parte se creará un programa que realiza el filtrado de una señal senoidal utilizando un filtro promediador[7]. En la figura 3.8 se muestra el programa completo, la señal puede ser cualquier conjunto de datos combinado a la cual el usuario puede agregar un nivel de ruido y también introducir el orden del filtro. Los resultados se muestran en una gráfica y se creará un panel de usuario como el de la figura 3.9.

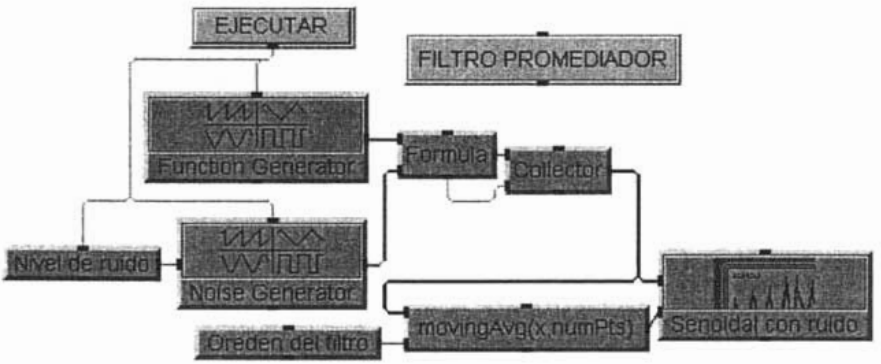


Figura 3.8: Programa para hacer filtrado.

Los objetos utilizados son los siguientes:

- DEVICE⇒VIRTUAL SOURCE⇒FUNCTION GENERATOR
- DEVICE⇒VIRTUAL SOURCE⇒NOISE GENERATOR
- DEVICE⇒FORMULA
- DATA⇒CONTINUOUS⇒REAL KNOB
- DATA⇒COLLECTOR
- DATA⇒CONSTANT⇒INTEGER
- DISPLAY⇒XYTRACE
- DISPLAY⇒LABEL

[7]Un filtro promediador es aquel que toma una determinada cantidad de muestras de una señal, las acumula o suma y las divide por el número de muestras tomadas, al factor por el cual se divide la suma acumulada se le llama orden del filtro o tamaño de la ventana.

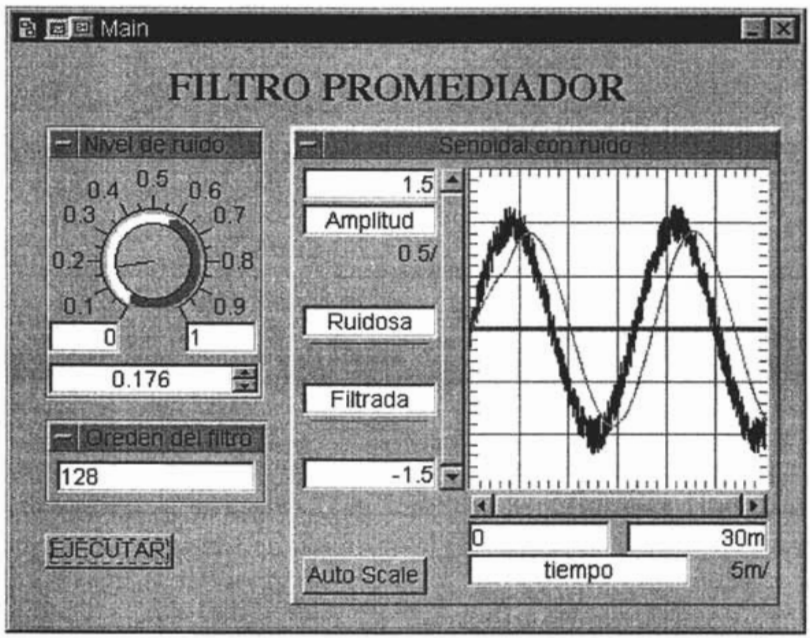


Figura 3.9: Panel de usuario del filtro promediador.

- FLOW⇒START
- DEVICE⇒MATH & USER FUNCTIONS, seleccionar del tipo Built-in functions, categoria, filtrado de datos y nombre **movingAvg**.

Primero vamos a tomar el objeto **FUNCTION GENERATOR** para que genere una función senoidal cuyo número de puntos sea de 1024, una frecuencia de 60 Hz, duración (Time Span) de 30 ms (todos los parámetros anteriores pueden ser modificados para cualquier valor).

Después vamos a tomar el objeto **NOISE GENERATOR**, este debe de tener la misma duración (30 ms) y el mismo de número de puntos (1024) que el objeto **FUNCTION GENERATOR** porque de lo contrario los dos vectores (la señal senoidal y el de ruido) no podrán ser sumados posteriormante. A este objeto se le agregará la terminal (en el menú del objeto seleccionar Add Terminal) de Amplitud con lo que se verá en la parte izquierda del mismo la nueva terminal.

Para el que el nivel de ruido sea variable va a utilizar el objeto **REAL KNOB**, al cual en sus propiedades se le modificará el nombre por omisión con el de "Nivel de ruido", la terminal de salida de este objeto será conectada a la terminal de "Amplitud" del objeto generador de ruido.

Para hecer la suma de la señal senoidal y el ruido se va a utilizar el objeto **FORMULA** al cual se le agregará una terminal de entrada de datos y en la casilla de edición se colocará la fórmula que se desee ejecutar que en este caso es $a + b$.

Para tener los datos que se están generando listos para ser procesados o manipulados es necesario guardarlos o mantenerlos en algún sitio y esto se hace mediante el objeto **COLLECTOR** el cual genera un arreglo de los datos de entrada de manera dinámica y funciona de la siguiente manera, el colector tiene dos terminales de entrada "Data" y "XEQ", en la terminal "Data" van los datos que se están recibiendo y cuando se terminan de generar datos de la fuente, esta manda la señal de secuencia de salida (que se conecta a la terminal "XEQ" del colector) la cual disparará en el colector el evento de que los datos han cesado de generarse. En este momento se tiene un vector a la salida del colector.

Para introducir el orden del filtro se toma el objeto **INTEGER**, la salida de este objeto va conectada a la terminal de entrada "numPts" de la función **movingAvg**, en la entrada "x" de la misma función va conectada la salida del colector (que tiene el vector de datos).

Para hacer el despliegue gráfico de la señal filtrada y la ruidosa se hará uso del objeto **XYTrace** al cual se le agregará una terminal de entrada de datos, en sus propiedades, en la parte de **Traces & Scales ...** se modifican los nombres de los datos de entrada para diferenciar las dos señales. habiendo hecho lo anterior, se conectan la salida del colector y la salida de la función **movingAvg** a las entradas del objeto **XYTrace**.

El objeto **LABEL** es solo para poner un título al programa, se revisan sus propiedades y se cambia su nombre por el deseado.

El objeto **START** se utiliza para que al presionarlo se ejecute el programa, a este también en sus propiedades se le modifica su nombre para que aparezca la leyenda de "EJECUTAR" en el mismo, su terminal de salida se conecta a la terminal de secuencia de entrada de los objetos FUNCTION GENERATOR, NOISE GENERATOR y Nivel de ruido. Una vez hecho lo anterior el programa ya puede ejecutarse.

Ahora para hacer el panel de usuario se hace lo siguiente, se seleccional los objetos que se deseen que aparezcan en el panel, y se presiona el botón en la barra de herramientas ADD TO PANNEL. Los objetos seleccionados se pondrán en el panel tal y como se seleccionaron, se pueden mover y

acomodar dentro del espacio del panel según se requiera. Hay que tener cuidado en la vista de los objetos ya que si algunos o todos los objetos seleccionados estaban en su forma de ícono, así aparecerán en el panel, si un objeto se requiere en su forma abierta y se selcccionó en su forma de ícono, simplemente se selecciona y se borra del panel y se agrega nuevamente en la forma requerida.

La programación visual es en general intuitiva siguiendo las reglas de propagación y las propiedades de los objetos pero dependiendo de la forma de programar se puede mejorar la velocidad de ejecución del programa, las siguientes técnicas de programación ayudan a un mejor desempeño de la aplicación.

- **Realizar cálculos sobre arreglos cuando sea posible**
  El aplicar funciones matemáticas sobre arreglos mejora grandemente el comportamiento del programa ya que es mas facil para HP VEE procesar un arreglo que un dato a la vez.

- **Si es posible, mantener todos o la mayoría de los objetos como íconos**
  Mientras mas información se tenga que mantener y actualizar en la pantalla, mayor será el tiempo que tome el programa para ejecutarse. En el caso de contadores, es mas eficiente tenerlos en su vista de íconos ya que se ejecutan mas rápido.

- **Reducir el número de objetos en el programa**
  Para poder reducir el número de objetos se utiliza el objeto **FORMULA** dentro del cual se introduce la expresión a evaluar, en lugar de tomar objetos por separados, así como también si se necesita un dato constante este se escribe directamente y no a través del objeto **CONSTANT**. Y anidar funciones dentro de la lista de parámetros de otras.

Además de las anteriores, existe otras formas de optimización:

- Ejecutar el programa en la vista de panel en lugar de la vista detallada ya que en la vista de panel se mantienen menos objetos en la pantalla.

- Usar variables globales en lugar de valores propagados (especialmente en arreglos muy largos) dentro y fuera de objetos y funciones de usuario. El declarar variables globales permite el uso de variables locales.

- Graficar un arreglo de datos en lugar de graficar cada punto por separado. Si los valores en X están igualmente espaciados utilizar el objeto **XYTrace** en lugar de **X vs. Y plot**.

- Utilizar un objeto **If/Then/Else** con condiciones múltiples en lugar de múltiples objetos **If/Then/Else**.

- En **Strip Charts** y **Logging Alphanumeric**, fijar el tamaño del buffer en las propiedades al menor número posible para la aplicación.

- Utilizar el operador condicional, **(condición ? exp1 : exp2)**, en lugar del objeto **If/Then/Else** con una **Junction** y una **Gate**.

- Cuando se lean datos de un archivo utilizar la transacción ARRAY 1D TO END:(*) en lugar de realizar la transacción READ en un elemento a la vez y utilizando la terminal EOF.

- Cuando se utilicen mapas de bits fijarlos como centrados o de tamaño actual en lugar de escalados.

- En indicadores como el **Fill Bar** o **Thermometer**, deshabilitar la opción **Show digital Diplay**.

- En alarmas de color, si se esta cambiando entre los colores de manera rápida, deshabilitar la opción **Show 3D Border**.

3.3 Lectura y escritura de datos

Escritura de datos a un archivo

HP VEE utiliza *transacciones* de Entrada/Salida para comunicarse con instrumentos, archivos, cadenas, el sistema operativo, interfases y otros programas. Las transacciones son especificaciones para entrada y salida usadas por cietos objetos, estos incluyen **To File**, **From File**, **Direct I/O**. Las transacciones aparecen como frases listadas en la vista abierta de estos objetos. Por ejemplo, para escribir datos de cualquier tipo a un archivo se utiliza el objeto **To File** en el menu I/O⇒TO FILE y una o varias transacciones.

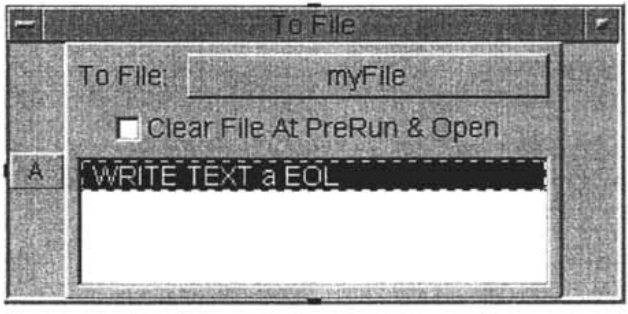


Figura 3.10: Objeto TO FILE.

En la figura 3.10 la entrada al objeto **TO FILE** se está enviando al archivo llamado **myFile** (por omisión, puede modificarse), se pueden agregar mas entradas según se necesite. La parte que esta resaltada se le llama transacción. Cuando se le da un click en la transacción aparece una caja de diálogo para configurar la sentencia de la transacción (figura 3.11).

Hay diferentes formas de la caja de diálogo de una transacción dependiendo del objeto que se este utilizando[8], pero hay ciertos elementos comunes, las *"acciones"*, la *"codificación"*, el *"formato"*, y la secuencia *"final de línea"*(EOL). Una transacción para escribir datos tiene generalmente el siguiente formato:

<acción><codificación><lista de expresiones><formato><EOL>

[8]Generalmente son todos los objetos del menú I/0 excepto EXECUTE PROGRAM(UNIX y PC) y HP BASIC/UX(UNIX)

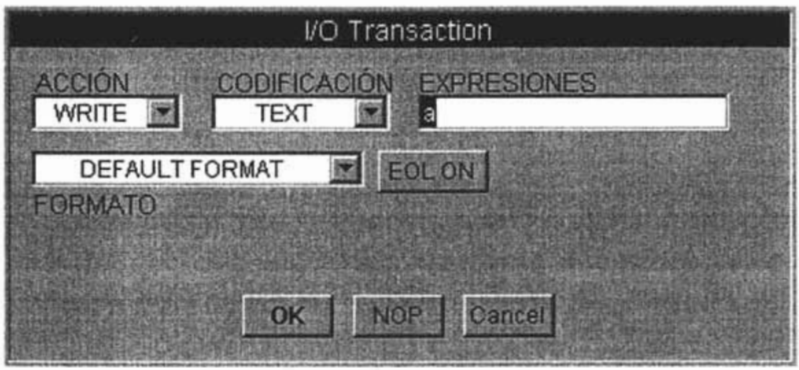


Figura 3.11: Caja de diálogo de una transacción de E/S.

Las acciones mas comunes disponibles son **READ**,**WRITE**, **EXECUTE** y **WAIT**:

ACCIÓN	**EXPLICACIÓN**
READ	Lee datos de la fuente específica usando el formato y la codificación especificados.
WRITE	Escribe datos al destino específico usando la codificación y formato especificados.
EXECUTE	Ejecuta un comando específico. Por ejemplo, EXECUTE REWIND reestablece el apuntador de archivo de lectura o escritura al inicio del mismo sin borrar su contenido. EXECUTE CLOSE cierra un archivo abierto.
WAIT	Espera el número de segundos especificado antes de la transacción siguiente.

La *codificación* y *formato* se refieren a la forma en como el objeto se empaqueta y se envía. Por ejemplo, una codificación TEXT manda los datos como caracteres ASCII.

ACCIÓN	CODIFICACIÓN	EXPRESIÓN	FORMATO
READ	TEXT	ALGEBRAICA	CHAR
WRITE	BYTE	MATEMATICA	TOKEN
EXECUTE	CASE		STRING
WAIT	BINARY		INTEGER
	BINBLOCK		OCTAL
	CONTAINER		HEX
			REAL
			COMPLEX
			PCOMPLEX
			COORD
			TIME STAMP

Figura 3.12: Formato de una transacción

CODIFICACIÓN	EXPLICACIÓN
TEXT	Lee o escribe todos los tipos de datos en formato ASCII, el cual puede ser facilmente editado o portado a otras aplicaciones de software. Los datos numéricos de HP VEE son convertidos a texto automaticamente.
BYTE	Convierte datos numéricos a enteros binarios y envia o recibe el byte menos significativo.
CASE	Mapea un valor enumerado o un entero a una cadena y lee/escribe la cadena. Por ejemplo, se puede usar CASE para aceptar números de error y escribir mensajes de error.
BINARY	Maneja todos los tipos de datos en formato binario específico de la máquina.
BINBLOCK	Usa bloques IEEE488.2 de encabezado de longitud definida con todos los tipos de datos de HP VEE en archivos binarios.
CONTAINER	Usa formato de texto específico de HP VEE con todos los tipos de datos.

En una transacción de **escritura**, una *"lista de expresiones"* es simplemente una lista de expresiones separadas por una coma que necesitan ser evaluadas para producir un dato enviado. La expresión puede ser compuesta de expresiones matemáticas, un nombre de una terminal de entrada, una cadena de caracteres constante, una función de HP VEE, una función de usuario, o una variable global. En una transacción de **lectura**, la lista de expresiones debe consistir de una lista de nombres de terminales de salida separada por comas indicando el dato a leer. EOL (fin de línea) puede ser habilitada o deshabilitada dentro de las propiedades del objeto.

Tabla 3.1: Formatos para la transacción READ TEXT.

FORMATO	**DESCRIPCIÓN**
CHAR	Lee cualquier caracter de 8 bits.
TOKEN	Lee una lista contigua de caracteres como una unidad llamada token. Los tokens son separados por un delimitador que uno especifica tal como una coma o un espacio.
STRING	Lee una lista de caracteres de 8 bits como una unidad. La mayoría de los caracteres de control son leidos o rechazados. Se llega al final de la cadena cuando el número de caracteres especificados se han leido, o cuando se encuentra un caracter de nueva línea.
INTEGER	Lee una lista de caracteres y los interpreta como la representación de un entero decimal o no decimal. Los únicos caracteres que se consideran como parte de un entero decimal son 0123456789-+. HP VEE reconoce el prefijo 0x(HEX) y todos los formatos numéricos no decimales especificados por el IEEE488.2: #H(HEX), #Q(OCTAL),#B(BINARIO).

Tabla 3.1: (continua ...)

OCTAL	Lee una lista de caracteres y los interpreta como la representación octal de un entero. Estos caracteres incluyen 01234567. HP VEE reconoce el prefijo numérico no decimal IEEE488.2 #Q para números octales.
HEX	Lee una lista de caracteres y los interpreta como la representación hexadecimal de un entero. Estos caracteres incluyen 0123456789abcdefABCDEF El caracter 0x es el prefijo por omisión; no es parte del número y es leido e ignorado. HP VEE también reconoce el prefijo númerico no decimal IEEE488.2 #H para números en hexadecimal.
REAL	Lee una lista de caracteres y los interpreta como la representación decimal de un número real (en punto flotante). Todas las notaciones comunes son reconocidas incluyendo signos al inicio del número, exponentes signados y puntos decimales. Los caracteres reconocidos como parte de un real son 0123456789-+.Ee. VEE reconoce ciertos caracteres como sufijos multiplicadores: P para 10^{15}, T para 10^{12}, G para 10^{9}, M para 10^{6}, K o k para 10^{3}, m para 10^{-3}, u para 10^{-6}, n para 10^{-9}, p para 10^{-12}, f para 10^{-15}.
COMPLEX	Lee el equivalante a dos REALes y los interpreta como un número complejo. El primer número es leido como la parte real y el segundo número es la parte imaginaria.
PCOMPLEX	Lee el equivalante a dos REALes y los interpreta como un número complejo en forma polar (notación de fasor). El primer número es la magnitud y el segundo es el ángulo. Se deben de especificar unidades de medición para la fase en una transacción.

Tabla 3.1: (continua ...)

COORD	Lee el equivalente o dos o mas REALes y los interpreta como coordenadas rectangulares.
TIME STAMP	Lee uno de los formatos de tiempo de HP VEE los cuales representan el día del calendario y/o la hora.

3.4 Formas de controlar instrumentos

3.4.1 Utilización del INSTRUMENT MANAGER

En el menú I/O⇒INSTRUMENT MANAGER se despliega la caja de diálogo del manejador de instrumentos (figura 3.13). Esta caja de diálogo se usa para agregar, eliminar, editar y actualizar instrumentos de la lista de instrumentos preestablecida u agregar algún otro. El manejador de instrumentos nos permite seleccionar tres objetos para poder controlar los mismos, **DIRECT I/O**, **PANEL DRIVERS** y **COMPONENT DRIVERS**.

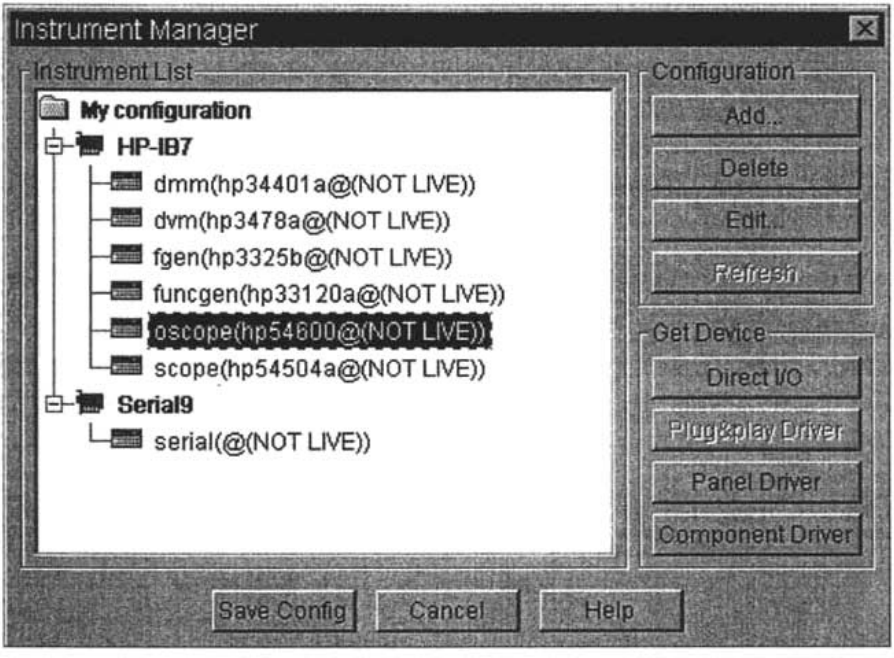


Figura 3.13: Manejador de instrumentos.

La caja de diálogo del **INSTRUMENT MANAGER** tiene tres botones en la parte de abajo: Save Config, Cancel y Help y esta dividida en tres áreas:

- Instrument List. Para seleccionar un instrumento en la lista solo se da un click. Para agregar un objeto I/O de instrumento en el área de trabajo se presiona el botón *Get device*. Para editar un instrumento de la lista se le da doble click.

- Get Device. Esta área determina el tipo de objeto I/O de instrumento que se agraga al área de trabajo. Hay botones de selección para Direct I/O, Plug & Play driver, Panel Diver y Component Driver. Los botones esta deshabilitados cuando no están disponibles para el instrumento actual.

- Configuration. Esta área permite cambiar la configuración del instrumento. Hay cuatro opciones:

  - Add... - Permite agregar un instrumento a la lista. Cuando se presiona Add... la caja de diálogo de configuración del dispositivo aparece. Se puede configurar el instrumento deseado que se quiera agregar llenando los campos de esta caja de diálogo.
  - Delete - Borra el instrumento seleccionado de la lista. Si se comete un error, se presiona el botón Cancel para recuperar el instrumento por medio de la operación salir sin guardar los cambios.
  - Edit... - Permite editar la configuración de un objeto existente. Cuando se presiona Edit... aparece la caja de diálogo de configuración del instrumento en la cual se le hacen los cambios. También se puede hacer doble click en el instrumento que se quiera editar.
  - Refresh - Busca interfases y dispositivos que estén disponibles (conectados). Si el nodo raíz de la lista se selecciona, *Refresh* buscará los dispositivos y las interfases disponibles. Si una interfase específica es seleccionada, Refresh buscará a todos los dispositivos conectados a la interfase seleccionada. Si un dispositivo es seleccionado, Refresh enviará el comando *IDN? y tratará de identificar el dispositivo y lo reiniciará.

Un objeto **DIRECT I/O** puede ser operado con o sin el instrumento conectado a la computadora. Si se desea manejar un instrumento conectado se debe configurar una dirección correcta y diferente de cero y habilitar *Live Mode*. La dirección y Live Mode se controlan por medio de la caja de diálogo de configuración del objeto, si la direccción es cero o *Live Mode* esta apagado, el objeto **DIRECT I/O** del instrumento opera pero no intenta comunicarse con el instrumento.

Los **PANEL DRIVER** pueden usarse interactivamente dentro de un programa. Para modificar el valor de un componente individual se presiona el campo que contiene el valor y se completa la caja de diálogo resultante.

Para hacer una medición y desplegar el resultado se presiona la correspondiente lectura de salida o el display XY dentro de la vista abierta del Panel Driver.

Es posible tener mas de un objeto manejando un instrumento. También es posible tener multiples copias del mismo driver u objeto DIRECT I/O, cada una manejando un instrumento diferente o mandando secuencias de comandos distintas a un mismo instrumento. En cada caso, es el nombre de la configuración el que determina cual objeto maneja cual instrumento.

Antes de que HP VEE pueda comunicarse con un instrumento la interfase del mismo y del hardware deben de ser configuradas de manera adecuada.

3.4.2 Objeto DIRECT I/O

Con este objeto se puede realizar la comunicación con cualquier instrumento que tenga una interfase GPIB. Aunque esto implica un esfuerzo adicional que el de utilizar el PANEL DIVER, es útil para hacer que el instrumento ejecute una secuencia de comandos específica. También, el objeto *DIRECT I/O* produce en general velocidades de ejecución mas rápidas. El escoger el mejor control sobre un instrumento depende de la disponibilidad del driver, la necesidad para el desarrollo rápido de pruebas y los requerimientos de rendimiento. En la figura 3.14 se muestran las vistas iconizadas de los objetos **DIRECT I/O** para el multímetro digital (HP34401A), el generador de señales (HP33120A) y el osciloscopio (HP54603B) los cuales se utilizarán de manera extensiva en el capítulo de Aplicaciones.

Figura 3.14: Objetos direct I/O del multímetro, generador de señales y osciloscopio respectivamente.

Menú del objeto (figura 3.15)

- Show Config - Despliega la configuración del instrumento.
- Add Trans - Agrega una transacción al final de la lista.
- Insert Trans - Inserta una transacción antes de la transacción actualmente seleccionada.
- Cut Trans - Elimina la transacción actual seleccionada, pero la guarda en la transación buffer "cut-and-paste".
- Copy Trans - Copia la transacción actual seleccionada a la transacción buffer "cut-and-paste".
- Paste Trans - Pega la transacción que ha sido previamente "cortada" o "copiada".
- Print Trans - Imprime la caja de diálogo detallada que describe la transacción actual seleccionada.

Figura 3.15: Menú parcial del objeto DIRECT I/O.

Parámetros de la vista abierta

Las acciones del objeto **DIRECT I/O** se fijan mediante el uso de transacciones (Ver figura 3.11 en la página 56), si un objeto de este tipo no se le ha agregado alguna transacción todavia tiene el aspecto de la figura 3.16.

- READ - Lee datos de un instrumento utilizando la codificación y el formato especificados.

- WRITE - Escribe datos a un instrumento utilizando la codificación y el formato especificados.
- EXECUTE - Para un instrumento con GPIB, envia cualquiera de los comandos Selected Device Clear o Go To Local, Remote o Group Execute Trigger. Para un instrumento serial envia Reset o Break.
- WAIT - Espera el número de segundos antes de ejecutar la siguiente transacción.

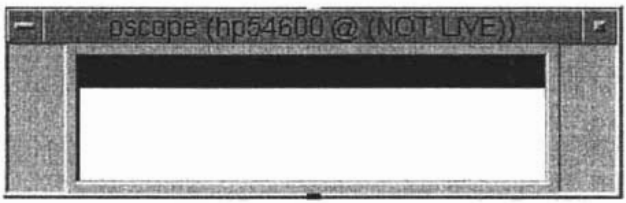


Figura 3.16: Objeto DIRECT I/O en vista abierta.

Codificaciones de DIRECT I/O

Referirse a la figura 3.11 en la página 56.

- TEXT - Escribe todos los datos en una forma legible. Los datos de HP VEE son automaticamente convertidos a texto conforme son escritos.
- BINARY - Escribe todos los tipo de datos en formato binario específico de la computadora.
- BINBLOCK - Escribe todos los tipos de datos de HP VEE en archivos binarios con encabezados de longitud determinada por el IEEE488.2. Cuando se usa BINBLOCK con un instrumento que manda el carácter alimentación de línea (line-feed), se selecciona END Byte (LF) para que HP VEE consuma el caracter line-feed. Si el instrumento no manda line-feed, seleccionar No END byte.
- CONTAINER - Escribe todos los tipos de datos en formato de texto independiente de la plataforma.

3.4.3 PANEL DRIVERS

El Panel Driver es un objeto para el control de instrumentos que forza a igualar todas las funciones de configuración en el panel del mismo correspondiente con el panel de control desplegado en la vista abierta del objeto.

Se utiliza para controlar el estado de un instrumento por medio del panel de control desplegado en la vista abierta. Antes de que el Panel Driver pueda ser usado, el driver del instrumento deseado debe estar instalado (ver figura 3.17).

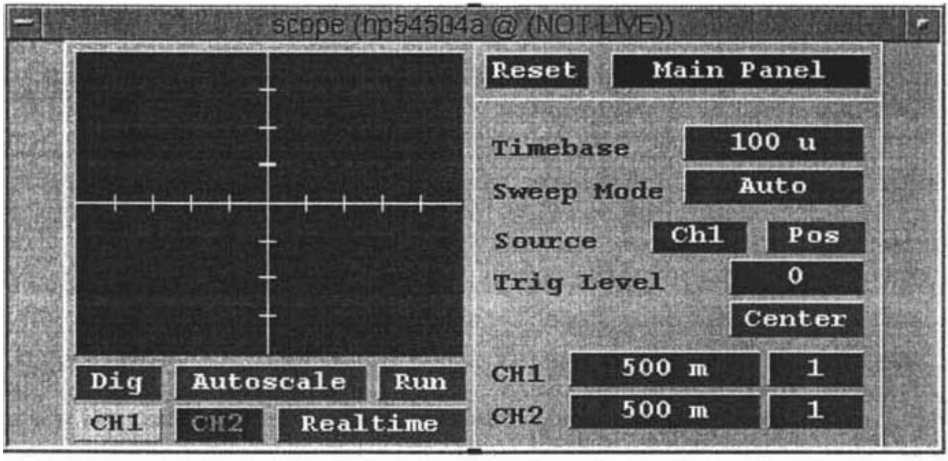


Figura 3.17: Panel driver para un osciloscopio.

Para agregar un objeto Panel Driver al área de trabajo de HP VEE, hay que presionar en el menú I/O⇒INSTRUMENT MANAGER para desplegar la caja de diálogo del INSTRUMENT MANAGER. Revisar la lista de instrumentos configurados para ver si hay alguno para el instrumento que se desea usar. Si el instrumento que se requiere esta en la lista y esta configurado apropiadamente:

1. Presionar una vez en el instrumento para selecionarlo.
2. Presionar en *Panel Driver* para agregar el panel del instrumento para ese instrumento.

Si el instrumento deseado esta en la lista, pero es necesario cambiar su configuración, se presiona *Edit...* para cambiar la configuración utilizando las cajas de diálogo, se guardar la configuración (*Save Config*), y después se siguen los dos pasos anteriores.

Si el instrumento deseado no esta en la lista, se presiona *Add...*, se configura usando las cajas de diálogo, se guarda la configuración y se siguen los pasos 1 y 2 anteriores.

Menú del objeto.

- Add Terminal by Component ⇒ Select input Component - Se usa para agregar campos visibles en el panel como entradas al *Panel Driver.*
- Add Terminal by Component ⇒ Select output Component - Se usa para agregar campos visibles en el panel como salidas a un *Panel Driver.*
- Show Config - Despliega la configuración de un instrumento (Se puede editar esta configuración utilizando el menú I/O ⇒ Instrument Manager).
- Sync - Sincroniza el estado de un driver de instrumento al estado del instrumento sobre las siguientes condiciones:
  - El código de identificación contiene un componente sync.
  - El código de identificación tiene *Live Mode* habilitado.

Los *Panel Drivers* pueden ser operados con *Live Mode* deshabilitado. Esto permite ejecutar el programa sin comunicación con el instrumento, aún si el instrumento no esta conectado a la computadora.

Es posible tener mas de un objeto controlando un instrumento. Esto es, dos objetos *Panel Driver* pueden controlar *fgen* en el mismo programa. Estos objetos realizan llamadas (recalls) de estados cuando operan.

Se pueden agragar entradas y salidas para los componentes de un *Panel Driver.* Las entradas realizan las acciones grupo, de panel y acciones acopladas para el componente. Las terminales de control están disponibles como entradas para inicializar el instrumento, cambiar de manera programada la dirección del instrumento y el periodo de tiempo de espera (timeout), y para una señal sync. Las salidas realizan get actions y panel get actions.

3.4.4 COMPONENT DRIVER

Es un control de instrumento que lee y escribe valores a componentes que se seleccionan específicamente. No se despliega panel para el instrumento, lo cual difiere del objeto *Instrument Panel.*

Se usa para controlar un instrumento configurando los valores de solo unos pocos componentes al mismo tiempo. Antes de usar el objeto *Component Driver* el driver del instrumento requerido debe de estar instalado (ver figura 3.18).

Figura 3.18: Component driver para un osciloscopio.

Para agregar un objeto *Component Driver* al área de trabajo de HP VEE se presiona I/O ⇒ INSTRUMENT MANAGER para desplegar la caja de diálogo del Instrument Manager. Se revisa en la lista de instrumentos configurados para ver si hay el instrumento que se quiere utilizar, si el instrumento que se requiere esta en la lista y esta configurado de manera apropiada:

1. Se presiona una vez el instrumento para seleccionarlo.
2. Se presiona el botón *Componet Driver*

Para que sea útil, al menos una entrada o una salida debe agregarse al objeto *Component Driver*. Más de una entrada o salida pueden agregarse y ambas pueden aparecer en el mismo objeto. Una entrada realiza las acciones del componente. Una salida realiza la obtención de acciones (get actions). Las terminales de control están disponibles como entradas para inicializar, cambiar la dirección del dispositivo y fijar el tiempo de espera (timeout).

3.5 Análisis y despliegue de datos

HP VEE empaqueta los datos en contenedores, los cuales acarrean los mismos entre los objetos, cada contenedor tiene un dato de tipo y forma específica:

Tabla 3.2: Tipos de datos

Tipo de dato	Descripción
Int32	Entero de 32 bits en complemento a dos (-2147483648 a 2147483647)
Real(o REAL64)	Un real de 64 bits en conformidad con el estándar IEEE754(+/-1.797693138623157E308)
PComplex	Un componente en magnitud y fase en le forma (mag,@ fase). La fase esta por omisión en grados, pero puede cambiarse a radianes o gradientes con FILE⇒DEFAULT PREFERENCES⇒TRIG MODE.
Complex	Un complejo en forma rectangular teniendo partes real e imaginaria en la forma (real, imag). Cada componente es real.
Waveform	Un tipo de dato compuesto de valores en el dominio del tiempo que contiene valores reales espaciados en forma par, lineales y el total de duración de la forma de onda. La forma de un **Waveform** debe ser un arreglo unidimensional.
Spectrum	Un dato compuesto de valores en el dominio de la frecuencia que contiene valores PComplex de puntos y el valor mínimo y máximo de frecuencia. Los datos del dominio puede mapearse como logarítmicos o lineales. La forma de un **Spectrum** debe ser un arreglo unidimensional.

Tabla 3.2: (continua ...)

Coord	Un dato compuesto que contiene al menos dos componentes en la forma (x,y,...), cada componente es real. La forma de un **Coord** debe ser un escalar o arreglo unidimensional.
Enum	Una cadena de texto que tiene asociada un entero. Se puede accesar al escalar utilizando la función ordinal(x).
Text	Una cadena de caracteres alfanuméricos.
Record	Un tipo de dato compuesto con un campo para cada tipo de dato. Cada campo tiene un nombre y un contenedor, el cual puede ser de cualquier tipo y forma (incluyendo **Record**).

Adicionalmente a los diez tipos anteriores, HP VEE tiene tres tipos mas, usados solo para instrumentos I/O. Todos los enteros son almacenados y manipulados internamente por HP VEE como **Int32**, y todos los números reales son almacenados y manipulados como **Real64**. De cualquier manera, los instrumentos manejan generalmente enteros de 16 bits y algunos manejan números reales de 32 bits.

Tipo Instrument I/O	Descripción
Byte	Byte en complemento a dos(-128 a 127). (Usado en READ BINARY, WRITE BINARY, y WRITE BYTE, para la comunicación entre instrumentos I/O.)
Int16	Entero de 16 bits en complemento a dos (-32768 a 32767).
Real32	Un real de 32 bits en conformidad con el estándar IEEE754 (+/-3.40282347E+/-38).

HP VEE convertirá automáticamante los datos anteriores a un tipo de dato interno apropiado. Para hacer el despliegue de algún dato o grafico se tienen los objetos de la tabla 3.3.

Tabla 3.3: Tipos de desplegadores

Desplegador	Descripción
AlphaNumeric	Despliega valores como texto o números, Requiere SCALAR, ARRAY 1D, o ARRAY 2D.
Logging AlphaNumeric	Despliega valores como texto o números cuando son introducidos de manera continua. Requiere SCALAR o ARRAY 1D.
Indicador⇒Meter, Thermometer, Fill Bar, tank, Color Alarm	Todos estos desplegadores muestran números en la forma que sugieren sus nombres. Todos tienen intervalos en colores diferentes (usualmente tres), pero meter tiene cinco. El **Color Alarm** puede simular un LED con un mensaje de texto parpadeando en cada intervalo de la alarma.
XY Trace	Despliega graficamente arreglos mapeados o un conjunto de valores cuando los datos en y están igualmente espaciados con los valores de x. El valor de x que es generado automaticamente depende del tipo de dato a trazar. Por ejemplo, un trazo con reales generará valores de x reales igualmente espaciados; mientras que un trazo de Waveform generará valores de x de tiempo.
Strip Chart	Despliega graficamente los datos recientes que son generados mientras el programa corre. Para cada valor de entrada de y, el valor de x es incrementado por un tamño de paso específico. Cuendo el nuevo dato sobrepasa el límite al lado derecho del desplegador, este automaticamante se recorre para mostrar el dato mas actual.
Complex Plane	Despliega complejos, complejos polares (Pcomplex), o valores Coord en un eje real contra imaginarios.

Tabla 3.3: (continua ...)

X vx Y plot	Despliega graficamente valores cuando esta disponible la información para x y y de manera separada.
Polar Plot	Despliega graficamente datos en una escala polar cuando estos están disponibles de manera separada.
Waveform (Time)	Despliega graficamante formas de onda o espectros en el dominio del tiempo. Los espectros son convertidos automaticamente a el dominio del tiempo con una IFFT. El eje x son las unidades de muestreo de la forma de onda de entrada.
Spectrum (Freq)	Un menú que contiene desplegadores en el dominio de la frecuencia: espectro de magnitud, Espectro de fase, Magnitud contra fase (Polar), y Magnitud contra fase (Smith). Las entradas deben ser del tipo Waveform, Spectrum o arreglo de Coord. Las entradas Waveform son cambiadas automaticamente al dominio de la frecuencia con una FFT.
Picture	Un objeto usado para poner una imagen en la vista de panel. Los formatos soportados son: BMP, ICO (mapa de bits de X11), GIF (GIF87a).
Label	Un objeto usado para poner una etiqueta de texto en la vista de panel. El color y el tipo de letra pueden cambiarse facilmente a traves de las propiedades del objeto estando en la vista de panel.
Beep	Proporciona un tono audible para resaltar una parte del programa.
Note pad	Usa una nota de texto para hacer mas claro el programa.

3.6 Funciones en HP VEE

3.6.1 Objetos de usuario

Un objeto de usuario (*UserObject*) proporciona los medios para encapsular un grupo de objetos que realizan una tarea en un solo objeto. Esta encapsulación permite:

- Usar técnicas de diseño modular en la construcción de un programa en HP VEE. Esto permite resolver problemas complejos a través de un enfoque organizado. Los objetos de usuario (*UserObjects*) permiten usar la técnica de diseño top-down para crear un programa más flexible y versatil.
- Construir objetos definidos por el usuario que se pueden guardar en una librería para su posterior reutilización. Una vez que un objeto de usuario es creado y guardado, puede ser pegado en otros programas.

Características de los objetos de usuario

Cuando se agrega un objeto de usuario a la ventana principal este aparece en la vista de ícono y permanece en esta forma en el programa. Cuando se pasa a la vista abierta aparece el área de trabajo del objeto dentro de la cual se puede construir un segmento de programa agregando objetos y conectandolos. Las áreas de terminales acomodan las terminales de datos y control de tal manera que el objeto de usuario pueda comunicarse con el resto del programa (ver figura 3.19).

Contextos en objetos de usuarios

La ventana principal y los objetos de usuario representan contextos[9] separados dentro de un programa de HP VEE, tal como los subprogramas representan contextos separados dentro de un programa en C o en algún otro lenguaje. Como se muestra en la figura 3.19, se pueden anidar objetos de usuario lo cual genera contextos adicionales, en esta misma figura hay tres contextos y se pueden agregar mas objetos a cada contexto.

1. La ventana principal es un contexto que contiene *Objeto1*.
2. *Objeto1* es un contexto y contiene a *Objeto2*.
3. *Objeto2* es un contexto que puede contener otros objetos.

[9]Un contexto es al área específica del nuevo objeto.

Figura 3.19: Contextos entre objetos de usuario.

Propagación en objetos de usuario.

La propagación en un programa que contiene objetos de usuario es afectada por el hecho de que estos tienen un contexto separado, y siguen las siguientes reglas de propagación:

- Todas la terminales de entrada (y la terminal de secuencia de entrada si es que esta conectada) deben de ser activadas antes de que cualquier objeto dentro del objeto de usuario opere.
- Una vez que las terminales de entrada (y la terminal de secuencia de entrada si es que esta conectada) han sido activadas, el objeto de usuario opera y los objetos dentro del objeto de usuario operan. Los objetos dentro del objeto de usuario operan siguiendo las reglas de propagación, estos objetos comparten tiempo en operación con objetos externos en diferentes tareas. El objeto de usuario no bloquea la operación de objetos fuera del objeto de usuario.
- Las terminales de salida de datos del objeto de usuario no se propagan hasta que que todos los objetos dentro del mismo terminan de operar (a menos que se de la orden de salida prematura del objeto de usuario o por un error). Solo aquellas terminales de salida activadas desde adentro pasan los datos a la parte exterior. Cuando se activa cada terminal de salida de datos solo se propaga un contenedor.

3.6.2 Funciones de usuario

Se puede crear una función de usuario (*UserFunction*) a partir de un objeto de usuario ejecutando *Make UserFunction* dentro del menú del objeto de

usuario. También se puede crear seleccionando un grupo de objetos utilizando el menú *Edit* ⇒ *Create UserFunction.*

Se puede utilizar *Edit* ⇒ *UserFunction* para editar la función de usuario una vez que ha sido creada. La funciones de usuario existen en el ambiente de HP VEE y pueden ser llamadas con *Call Function* o desde ciertas expresiones.

Menú del objeto

- Edit ⇒ - Un menú padre que encabeza el menú de edición de contexto de la función de usuario. Este menú contiene las mismas alternativas como el menú principal de edición. La siguientes alternativas son solo válidas dentro del contexto de la función de usuario:
  - Clean Up Lines - Rutea las líneas en la función de usuario alrededor de los objetos.
  - Add To Panel - Crea una vista de panel para la función de usuario (si la vista de panel no existe actualmente) conteniendo los objetos seleccionados.
  - Create UserObject - Crea un objeto de usuario de los objetos seleccionados actualmente.
  - Create UserFunction - Crea una función de usuario de los objetos seleccionados actualmente.
- Make UserObject - Lo opuesto a la operación *Make UserFunction.* Cambia la función de regreso a un objeto de usuario.
- Save Secured Version - Guarda una versión del objeto de usuario la cual no puede ser editada. Se preguntará el nombre bajo el cual se guardara el objeto con extensión .vee.
- Locate - Resalta el lugar de la función de usuario en el explorador de programa.
- Generate Call - Crea una llamada a la función de usuario la cual puede ser colocada en otra parte del programa
- Cut - Borra la función de usuario de HP VEE y de la memoria.
- Close - Esconde la ventana de edición de la función de usuario. La función de usuario reside en memoria y no se necesita cerrar la ventana de edición de la función de usuario antes de ejecutar el programa.

No solo se puede llamar una función de usuarion con *Call Function*, se puede llamar desde cualquier expresión cuya evaluación sea retardada hasta el tiempo de ejecución. Estas incluyen expresiones en los objetos *Formula, If/Then/Else, Get values, Get Field, Set Field, From DataSet* o en *Transacciones I/O*. La sintaxis de una llamada a función de usuario en una expresión permite solo un valor de regreso. Así, una llamada a una función de usuario en una expresión solo regresara el valor de su primer salida, si hay salidas adicionales (excluyendo la secuencia de salida), sus valores serán descartados, si no hay salidas el valor de regreso será indefinido.

3.6.3 Librerías

Para crear una librería de objetos de usuario primero se crean los objetos en el área de trabajo, después se seleccionan y se guardan con *File* ⇒ *Save Objects* y se completa la caja de diálogo. Para utilizar una librería de objetos se selecciona *File* ⇒ *Merge*, en la caja de diálogo se preguntará por el nombre del archivo, se selecciona el archivo y la librería se agrega al área de trabajo y se posiciona donde se desee.

Se puede crear una librería de varias funciones de usuario y después guardarlas a un archivo. Esta librería puede ser importada a un programa usando el objeto *Device* ⇒ *Import Library* y borrada con el objeto *Device* ⇒ *Delete Libray.* o con *File* ⇒ *Merge Library.*

La diferencia entre *Delete* e *Import Library* con *Merge Library* es que con estos se carga y se elimina la(s) librería(s) al momento de ejecución del programa, mientras que con el último la(s) librería(s) siempre están en memoria y se tienen que cargar manualmente antes de ejecutar el programa.

Capítulo 4

Aplicaciones

4.1 Descripción general 78
4.2 Toma de muestras de una población 80
4.3 Medición de temperatura 89
4.4 Adquisición de datos de transductores 99
4.5 Ajuste de curvas 113
4.6 Conversión de variables eléctricas 117
4.7 Osciloscopio 124
4.8 Integración con el lenguaje C 128
4.9 Generación de funciones 135
4.10 Operaciones entre funciones 140
4.11 Puerto serie 149

4.1 Descripción general

Las aplicaciones, que se presentan en este capítulo muestran una gran parte del espectro de posibilidades de la herramienta *HP VEE.*

La aplicación MUESTREADOR.VEE consiste en la implementación de un adquisidor de datos para voltaje, corriente y resistencia la cual es útil para tomar mediciones de manera controlada presionando un botón una vez que se tenga la lectura de alguna variable de interés.

La aplicación TEMPERATURA.VEE hace la adquisición de las lecturas del multímetro para tres transductores de temperatura (uno a la vez), dos son de voltaje (termopar y transistor) y uno de resistencia (termistor) de manera continua (uno a la vez) hasta que el usuario decida que es un número suficiente de datos o ya no se note variación significativa en la gráfica mostrada.

En la sección *Adquisición de Datos de Transductores* se hace uso del programa MUESTREADOR.VEE el cual se modificó y se le nombró MUESTREADOR_MODULOS.VEE para la captura de pares coordenados de datos (x,y) generados por los transductores, también se presenta como se pueden manejar dos instrumentos de manera simultánea.

La aplicación AJUSTE.VEE complementa a las tres anteriores ya que se utiliza para obtener el módelo matemático del conjunto de datos generado por los transductores.

La aplicación VISUALIZACION_VARIABLES.VEE hace uso del programa AJUSTE.VEE para obtener los modelos matemáticos de los transductores en los que se adquirieron sus datos. Una vez obtenido el modelo matemático el valor tomado por el multímetro se escala de acuerdo al modelo de tal manera que en el display de la aplicación aparece la lectura del valor de la variable física y no eléctrica.

Algo importante y poderoso que tiene HP VEE es el poder integrar dentro de su programación funciones escritas en lenguaje "C" en forma de DLLs lo que amplia el campo de sus aplicaciónes y alcance, así como los métodos para resolver problemas o algoritmos de procesamiento específicos. El subcapítulo *Integración con el lenguaje C* muestra esta potencialidad.

La aplicación GENERADOR.VEE presenta la capacidad de generación de todas las formas de onda del instrumento HP33120A y la aplicación OPERACIONES.VEE ejemplifica como el resultado de la operación matemática entre dos generadores de funciones virtuales es una forma de onda arbitraria que puede descargarse al HP33120A como un vector de datos y convertirse en una función definida por el usuario para su uso posterior.

La aplicación OSCILOSCOPIO.VEE muestra como este puede aprovecharse como capturador de la forma de onda desplegada en su display y la aplicación PUERTO_SERIE.VEE permite el uso del puerto serie de la PC para el manejo y control de un instrumento programable que cuente con un puerto serie RS232.

4.2 Toma de muestras de una población

Esta aplicación se llama MUESTREADOR.VEE y su objetivo es adquirir las lecturas del multímetro digital que pueden ser de voltaje, corriente y resistencia, entre otros [1] y guardarlas a un archivo . El esquema general de conexión PC–Instrumento se muestra en la siguiente figura.

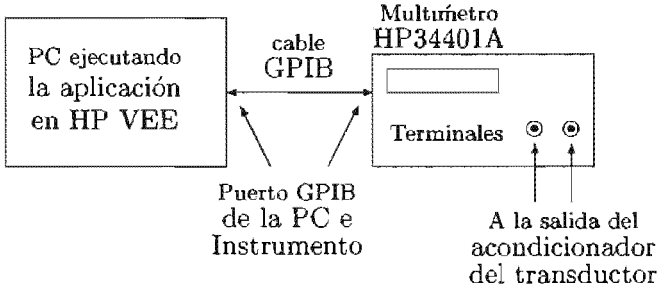


Figura 4.1: Esquema de conexión PC–Instrumento (HP34401A).

Al ejecutar esta aplicación aparecerá el panel de usuario como el de la figura 4.2 y una caja de diálogo preguntando por el número máximo de muestras y el tipo de medición a tomar, (ver figura 4.3 y su implementación en la figura 4.4), por omisión el número máximo de muestras es 10 sin embargo el usuario puede adecuar el número de muestras de acuerdo a la población o al experimento que este realizando. La implementación de la aplicación completa se muestra en la figura 4.5. En la figura 4.3 aparecen las opciones de tipos de medición que es posible realizar con la aplicación, a continuación se muestra la descripción de cada una de estas opciones, sin embargo, si se requiere información adicional acerca de las mismas, esta puede encontarse en el manual de usuario del multímetro digital.

DCV Medición de voltaje de corriente directa.

ACV Medición de voltaje de corriente alterna.

RAT RATIO, calcula un intervalo, el multímetro mide el voltaje de referencia de CA aplicado a las terminales Sense y el voltaje aplicado en las terminales Input del multímetro.

$$intervalo(RATIO) = \frac{\text{Voltaje señal CA}}{\text{Voltaje referencia}CA}$$

[1] Incluyendose valores de frecuencia, periódo y los que se especifiquen en el manual de usuario para el instrumento utilizado, que en este caso es un HP34401A.

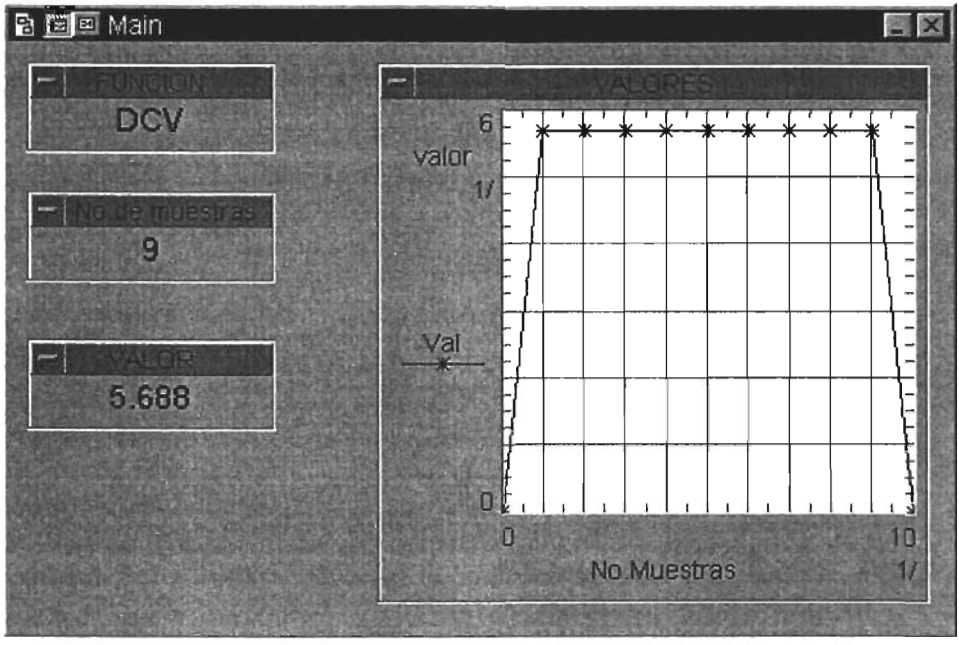


Figura 4.2: Panel de usuario para el muestreador.

DCI Medición de corriente de voltaje directo.

ACI Medición de corriente de voltaje alterno.

FREQ Medición de frecuencia.

PER Medición de periódo.

FRES (four wire resistance o 4 líneas) es la forma mas exacta de medir resistencias pequeñas, este método se utiliza cuando hay conexiones numerosas o longitudes de cable largas entre el multímetro y el dispositivo de en prueba.

RES Medición de resistencia con dos líneas.

Al presionar el botón CONTINUAR de la caja de diálogo donde se pregunta por el número de muestras y tipo de medición (figura 4.3) se presentará otra caja de diálogo como la de la figura 4.6 que sirve para confirmar el inicio de la toma de muestras, una ves confirmada esta acción (es decir, después de haber presionado el botón OK) se mostrará una nueva caja de diálogo (figura 4.7) la cual preguntará de manera continua si se desea seguir

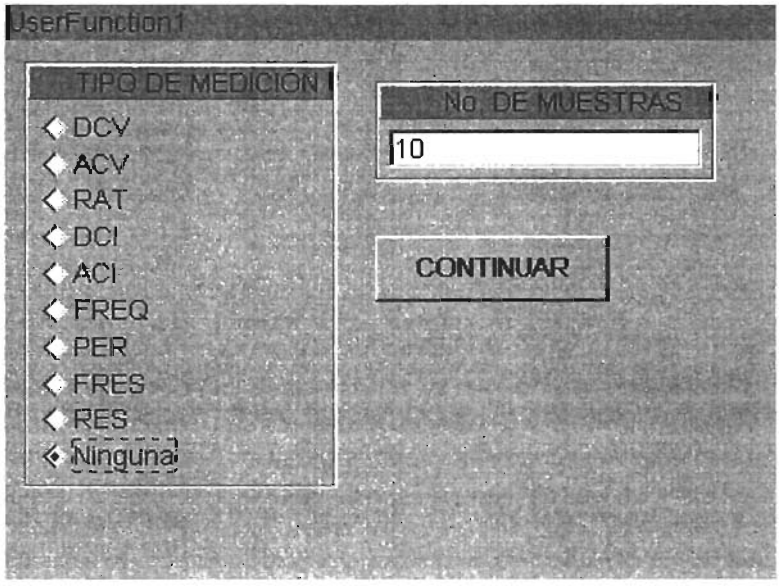


Figura 4.3: Panel para tipo de medición y num. de muestras.

tomando muestras (esto mientras no se llegue al límite máximo de muestras o se presione el botón SALIR o GUARDAR, si se presiona SALIR las muestras tomadas se perderan y si se presiona GUARDAR las muestras se podrán guardar a un archivo).

Para las cajas de diálogo que tienen el botón CANCEL (figs. 4.2 y 4.6), el presionarlo llevará a finalizar la aplicación. El botón salir (fig. 4.7) conducirá a la terminación del programa pero como en este caso se supone que ya se empezaron a tomar muestras el programa guardará automaticamente (antes de salir) las muestras tomadas hasta el momento en un archivo de nombre "resistencias.dat" en el directorio actual de trabajo, el botón "guardar" mostrará la caja de diálogo para guardar las muestras (figura 4.11) bajo el nombre de un archivo que se elija. Para la implementación en HP VEE se utilizaron los siguientes objetos:

- DATA⇒DIALOG BOX⇒MESSAGE BOX, hacer lectura, nueva lectura y error
- DATA⇒DIALOG BOX⇒FILE NAME SELECTION
- DEVICE⇒USERFUNCTION para panel de selección del tipo de medición y num. de muestras

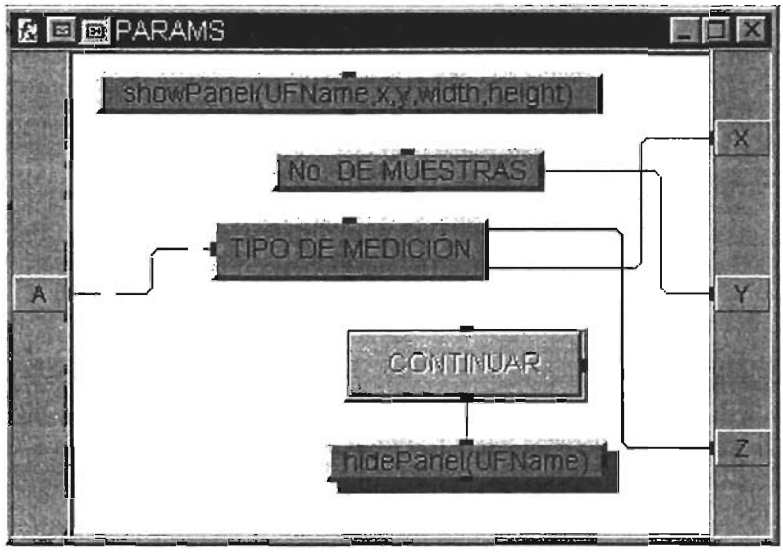


Figura 4.4: Implementación panel para tipo de medición y num. de muestras.

- DATA⇒CONSTANT⇒TEXT (opción de tipo de lectura por omisión, ninguna)
- DATA⇒ALLOCATE ARRAY⇒ALLOC REAL
- DATA⇒ACCESS ARRAY⇒SET VALUES a[2]=b, a[b] = c
- DISPLAY⇒XY TRACE (visualización gráfica)
- DISPLAY⇒ALPHANUMERIC ,no. de muestras, VALOR, FUNCION
- DISPLAY⇒NOTEPAD
- FLOW⇒REPEAT⇒BREAK
- FLOW⇒JUNCTION
- FLOW⇒REPEAT⇒UNTIL BREAK
- FLOW⇒IF/THEN/ELSE
- DEVICE⇒COUNTER
- DEVICE⇒FORMULA (A+1)

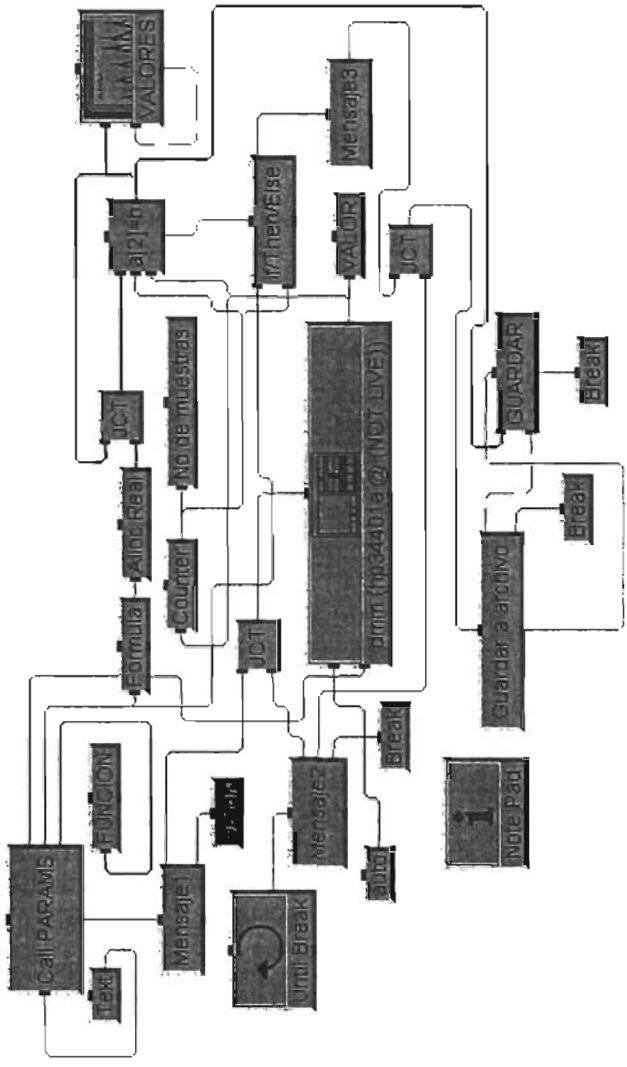


Figura 4.5: Implementación del muestreador.

Figura 4.6: Para iniciar el muestreo.

Figura 4.7: Para tomar muestras.

- I/O⇒TO⇒FILE (save_exit) y (guardar)
- I/O⇒INSTRUMENT MANAGER⇒DMM(hp34401A @ NOT LIVE)), presionar panel driver

El objeto **ALLOC REAL** (figura 4.9) crea un arreglo unidimensional o multidimensional de números reales, para este caso se creará un arreglo unidimensional o vector de tamaño *D*im Size 1. En la vista abierta del objeto se le da el tamaño del arreglo o se le agrega como terminal de entrada y se fija de manera manual el número de dimensiones que en esta caso es uno, también se tiene la opción de inicializar el arreglo a un valor, pero para mayor información en la utilización de este objeto referirse a la ayuda del mismo.

Figura 4.8: Sobrepaso de muestras.

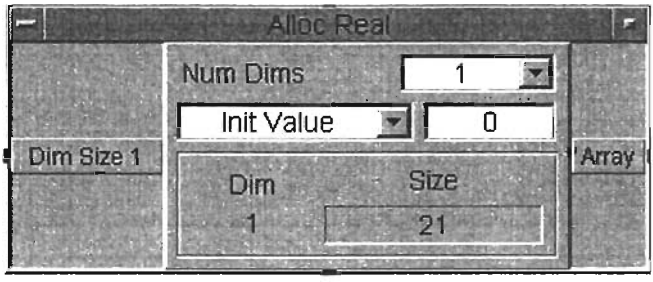


Figura 4.9: Objeto alloc real en vista abierta.

El **PANEL DRIVER** del multímetro digital se muestra en la figura 4.10, se le agregaron las entradas de **RES_AUTO** y **FUNCTION** para tomar resolución automática y seleccionar el tipo de medición, tiene autorango por omisión y también se le agregó la salida **READING** donde se tiene la medición tomada. El objeto **TO FILE** se explicó en la página 55, el objeto **JCT** entrega a su salida el primer valor o valor mas actual de cualquiera de sus entradas e **INTEGER INPUT** es un objeto que genera una caja de diálogo en la que se pregunta por un entero.

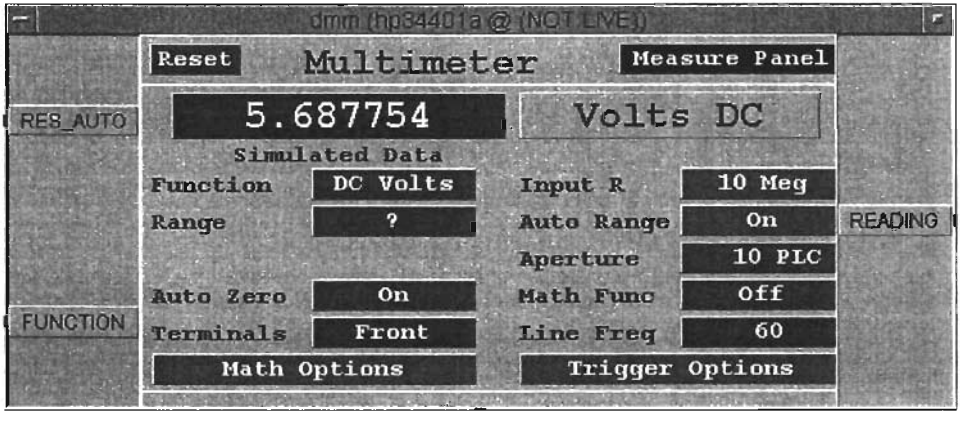


Figura 4.10: Panel Driver del multímetro digital.

Hay diferentes cajas de diálogo las cuales permiten ejecutar distintas acciones dependiendo de su tipo, son bastante flexibles en cuanto a su configuración, permiten agregarles botones, cambiarles su nombre, cambiarles el mapa de bits de presentación y como cualquier objeto de HP VEE permiten que se les agreguen terminales de entrada y salida, todo esto a través del

menú del objeto en la parte de propiedadedes y en la vista abierta del objeto, para una mejor descripción del funcionamiento de estos objetos referirse a la ayuda de HP VEE. Para la caja de diálogo **INTEGER INPUT** que pregunta por el número de muestras se le cambian los textos que van a aparecer al momento de la ejecución, en este caso se le cambia el valor por omisión y los mensajes en caso de que se cometa alguna condición erronea. En el caso de la caja de diálogo de mensaje **HACER LECTURA** solo se modificó la cadena del mensaje y los textos de los botones. Para la caja de diálogo **NUEVA LECTURA** también se cambiaron los textos de mensaje, se agregaron botones y se cambiaron los textos de los botones. La última caja de diálogo es la de error la cual se le suprimió un botón y se le cambio el texto de mensaje.

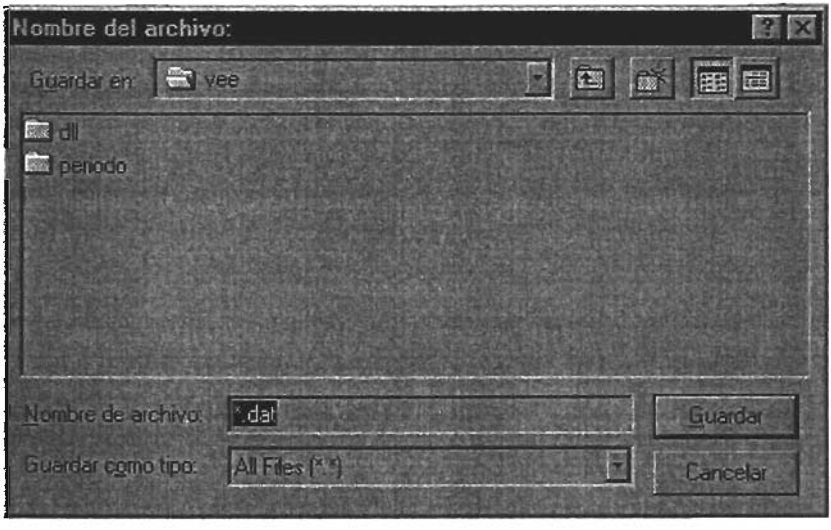


Figura 4.11: Caja de diálogo para guardar las lecturas a un archivo.

Para la implementación en HP VEE del panel donde se pregunta el número de muestras y el tipo de medición se utilizaron los siguientes objetos:

- DATA⇒CONSTANT⇒INTEGER (No. de muestras)
- FLOW⇒CONFIRM(OK) (CONTINUAR)
- DATA⇒SELECTION CONTROL⇒RADIO BUTTONS

- función SHOWPANEL(UFNAME,X,Y,WIDTH,HEIGHT)
- función HIDEPANEL(UFNAME)

Las funciones **SHOWPANEL(UFNAME,X,Y,WIDTH,HEIGHT)** y **HIDEPANEL(UFNAME)** muestran el panel y lo ocultan respectivamente (para mayor información consultar la ayuda de estas funciones), en el objeto **No. DE MUESTRAS** se especifica el número de muestras a tomar y con el objeto **TIPO DE MEDICION** se selecciona la medición ya sea de voltaje, corriente u otra, en este objeto se toma la salida ordinal la cual se conecta a la entrada **FUNCTION** del **PANEL DRIVER** del multímetro en donde se le manda el comando apropiado de tomar la medición seleccionada. Con el botón **CONTINUAR** se confirman las entradas de tipo de medición y número de muestras y se pasa al panel principal (figura 4.2).

4.3 Medición de temperatura

El nombre de esta aplicación es TEMPERATURA.VEE, su objetivo es adquirir los datos generados por tres transductores de temperatura, Termopar tipo J el cual se comporta linealmente dentro de un intervalo, un termistor NTC[2] y un transistor[3] el cual tiene una respuesta lineal según la hoja de especificación de $10\frac{mV}{°C}$ (ver apéndice B). En las figuras 4.12, 4.13 y 4.14 se muestran las conexiones del PC—multímetro—transductor para hacer la obtención de los datos de cada uno de los transductores, las terminales de estos deben estar aisladas del medio donde se esta midiendo de lo contrario se pueden dañar.

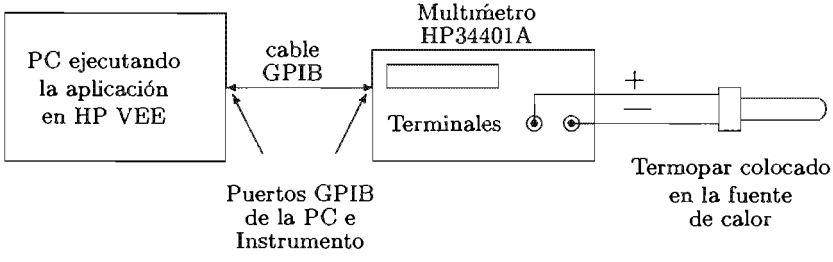


Figura 4.12: Conexión del Termopar tipo J

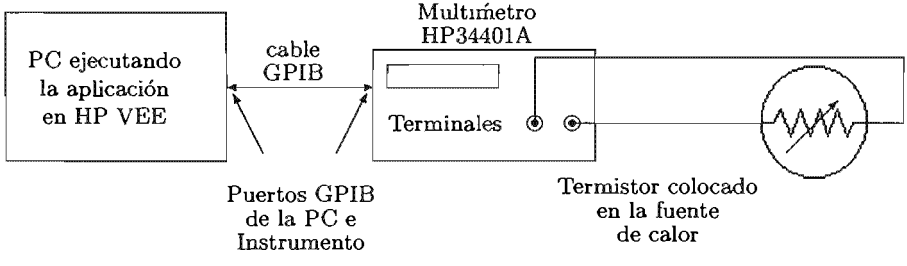


Figura 4.13: Conexión del Termistor NTC

En la figura 4.15 se muestra el panel de usuario de la aplicación y en la figura 4.16 aparece el conjunto de objetos que la integran.

[2]Coeficiente de Temperatura Negativo, la resistencia del decrece con el aumento de la temperatura de manera inversamente proporcional

[3]LM35DZ

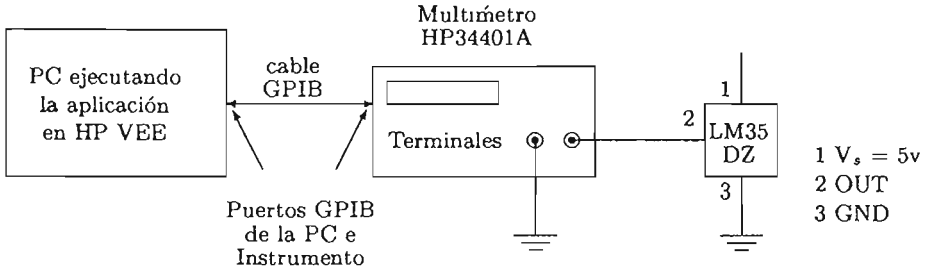


Figura 4.14: Conexión del LM35DZ

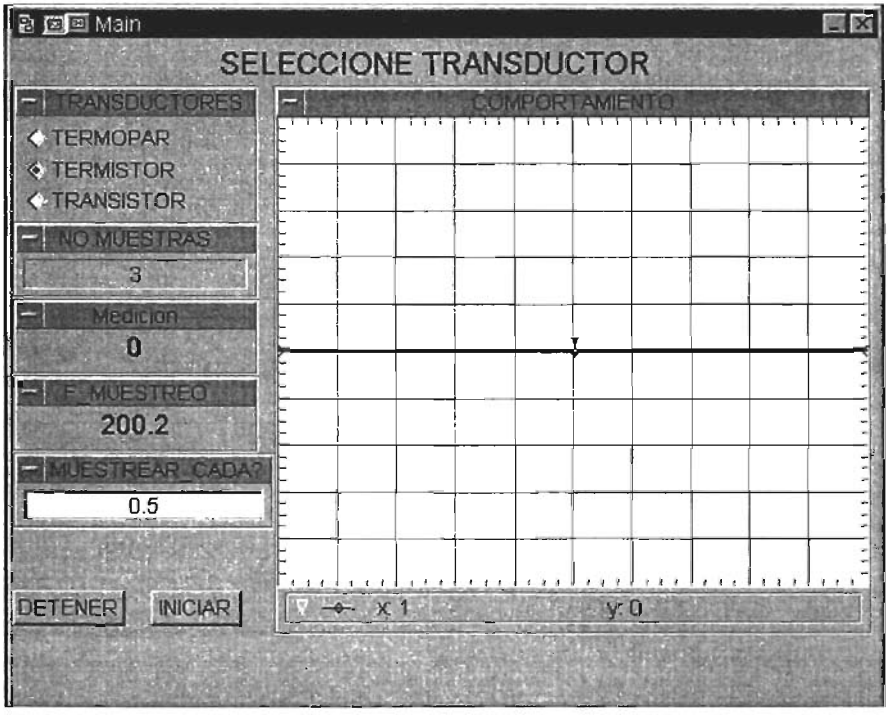


Figura 4.15: Panel de usuario de selección de transductores de temperatura.

La aplicación funciona de la siguiente manera. Hay un objeto que se ejecuta inmediatamente después de que se presiona el botón **INICIAR** el cual es nombrado **TRANSDUCTORES** (figura 4.17) que es de tipo **RADIO BUTTONS**, que dependiendo de la selección que se decida manda a la salida un número ordinal (0, 1, 2), este número se conecta a la entrada de un objeto **IF/THEN/ELSE** (nombrado Sel_sensor como en la figura 4.18) que llama a una función que inicializa las sentencias SCPI para autorango y la resolución máxima (ver figura 4.19), en la figura 4.16 se notan 2 rutinas de inicialización (Call ini_termopar y Call ini_termistor), una para el transductor de voltaje (termopar y transistor) y otra para el transductor de resistencia (termistor), difieren en las sentencias SCPI (ver figura 4.19) los objetos que integran esta función son de tipo texto constante en el menu DATA⇒CONSTANT⇒TEXT, (RANGO, RESOLUCION, VOLTAJE), en la siguiente tabla aparecen el nombre de la función de usuario de inicialización junto con las sentencias SCPI correspondientes a los objetos de texto.

Función de usuario	sentencia SCPI	Objeto
Call ini_termopar	SENS:FUNC "VOLT:DC" Selección de voltaje de DC	VOLTAJE
	SENS:VOLT:DC:RANG:AUTO ON Autorango habilitado	RANGO
	SENS:VOLT:DC:RES MAX Resolución Máxima	RESOLUCION
Call ini_termistor	SENS:FUNC "RES" Selección de resistencia	RESISTENCIA
	SENS:RES:RANG:AUTO ON Autorango habilitado	RANGO
	SENS:RES:RES MAX Resolución Máxima	RESOLUCION

Tabla 4.1: Sentencias SCPI, aplicación TEMPERATURA.VEE.

También dependiendo del transductor seleccionado se llama a una función que escoge el archivo en el cual se guardaran los datos, en este caso el usuario no tiene la libertad de especificar el nombre del archivo a donde guardar los datos, sino que estos se guardan bajo los siguientes nombres:

Transductor	Archivo
Termopar	termopar1.dat
Termistor	termistor1.dat
Transistor	transistor1.dat

De cualquier manera cuando se detiene el programa una vez finalizada la toma de mediciones, aparece una caja de diálgo con el mensaje del archivo bajo el cual se guardaron los datos (ver figura 4.21), esta caja de diálogo se le agrega inicialmente la entrada de mensaje en donde se conecta la ruta del archivo donde se van a guardar los datos.

Figura 4.16: Programa para los tres transductores.

Figura 4.17: Objeto tipo Radio Buttons para selección de transductor.

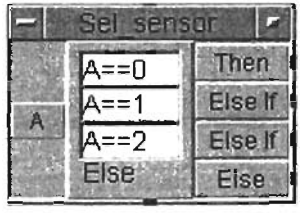


Figura 4.18: Selector del tipo de transductor.

Los objetos utilizados para la aplicación son los siguientes:

- DISPLAY⇒ALPHANUMERIC, 2 objetos (cod_err, medicion, F._muestreo)
- DISPLAY⇒NOTE PAD, INFO
- DISPLAY⇒STRIP CHART, COMPORTAMIENTO
- DISPLAY⇒LABEL, SELECCIONE TRANSDUCTOR
- DEVICE⇒USER FUNCTION, 3 objetos (ini_termopar, ini_termistor, ini_archivo)
- DEVICE⇒TIMER, para frecuencia de muestreo
- DEVICE⇒FORMULA, 2 objetos (FORMULA1, FORMULA2)
- DEVICE⇒COUNTER, NO.MUESTRAS
- FLOW⇒START, botón INICIAR
- FLOW⇒REPEAT⇒NEXT
- FLOW⇒REPEAT⇒BREAK
- FLOW⇒UNTIL BREAK

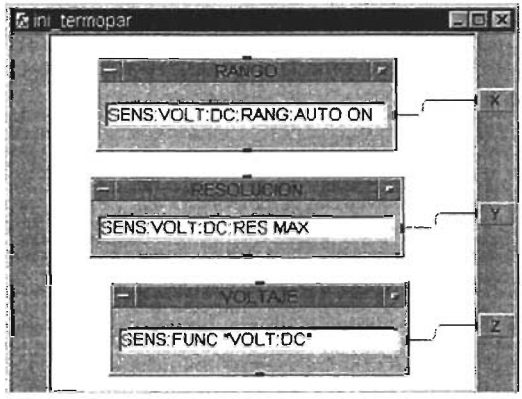


Figura 4.19: Objetos para inicializar la medición.

- FLOW⇒JUNCTION, 4 objetos (JCT1, JCT2, JCT3, JCT4, JCT)
- FLOW⇒IF/THEN/ELSE, 2 objetos (sel_sensor, if/Then/Else)
- I/O⇒TO⇒FILE
- I/O⇒INSTRUMENT DRIVER, DIRECT I/O del HP34401A
- DATA⇒CONSTANT⇒REAL, MUESTREAR_CADA?
- DATA⇒SELECTION CONTROL⇒RADIO BUTTONS, transductores
- DATA⇒DIALOG BOX⇒MESSAGE BOX, mensaje
- DATA⇒TOGGLE CONTROL⇒BUTTON, detener

El objeto que se encarga de hacer todo el proceso de medición es el **DIRECT I/O** del **HP34401A** el cual se muestra en su vista abierta en la figura 4.20, a este objeto se le agregaron 3 entradas para seleccionar el tipo de medición (entrada C para seleccionar voltaje o resistencia), resolución (entrada B) e intervalo (entrada A). También se le agregó una salida para el dato adquirido (salida Y). Las entradas A, B y C corresponden a los comandos SCPI de la tabla 4.1. Las lineas que siguen después de estas entradas hacen lo siguiente, TRIG:SOUR IMM genera un disparo para toma de la lectura de manera inmediata, INIT y FETCH?, transfieren la lactura de la memoria del instrumento a la PC. La transacción READ pone la lectura actual del instrumento en la variable Y.

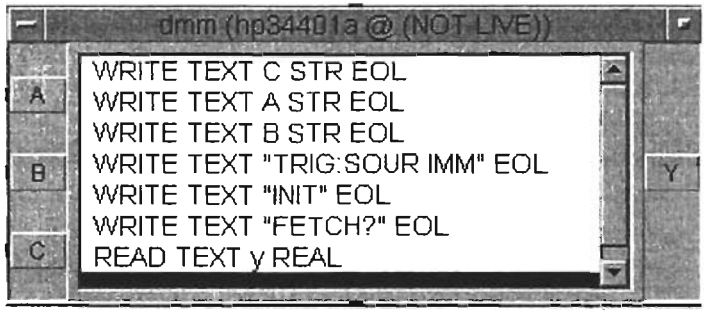


Figura 4.20: Proceso de medición con el objeto Direct I/O.

EL panel de usuario es muy fácil e intuitivo, en la parte superior izquierda se muestra el tipo de transductor a seleccionar, una vez hecho esto ya podemos comenzar a tomar mediciones presionando el botón **INICIAR** y para terminar se presiona el botón **DETENER**. En el panel se van a ir mostrando la cuenta de muestras y el valor actual además de irse graficando los valores conforme se van adquiriendo. Después de presionar el botón **DETENER** aparece la caja de diálogo de la figura 4.21 a la cual se presiona OK para confirmar. Adicionalmente se conecta un temporizador (timer en la figura 4.16) para saber la frecuencia de muestreo.

Figura 4.21: Caja de diálogo indicando el guardado del archivo.

La función que selecciona el archivo en el cual se guardan los datos (figura 4.22) tiene como entrada el número de salida del objeto **TRANSDUCTORES** (figura 4.17), dentro de la función hay otro objeto **IF/THEN-/ELSE** que selecciona el nombre del archivo (0 para termopar, 1 para termistor y 2 para transistor), hay otra cadena de texto que se puede modificar para indicar una ruta donde guardar el archivo, esta ruta junto con el nombre del archivo se concatenan (la sentencia **A+B** del objeto **FORMULA** de la figura 4.22 concatena ambas cadenas de texto) para crear una ruta con

el nombre del archivo el cual se manda a la salida de la función de selección de archivo y se conecta al objeto **TO FILE** cuyo funcionamiento ya fue descrito en la sección de transacciones (página 55), esta función la integran los siguientes objetos:

- FLOW⇒IF/THEN/ELSE
- FLOW⇒JUNCTION
- DATA⇒CONSTANT⇒TEXT, Termistor, transistor, termopar y Text
- DEVICE⇒FORMULA

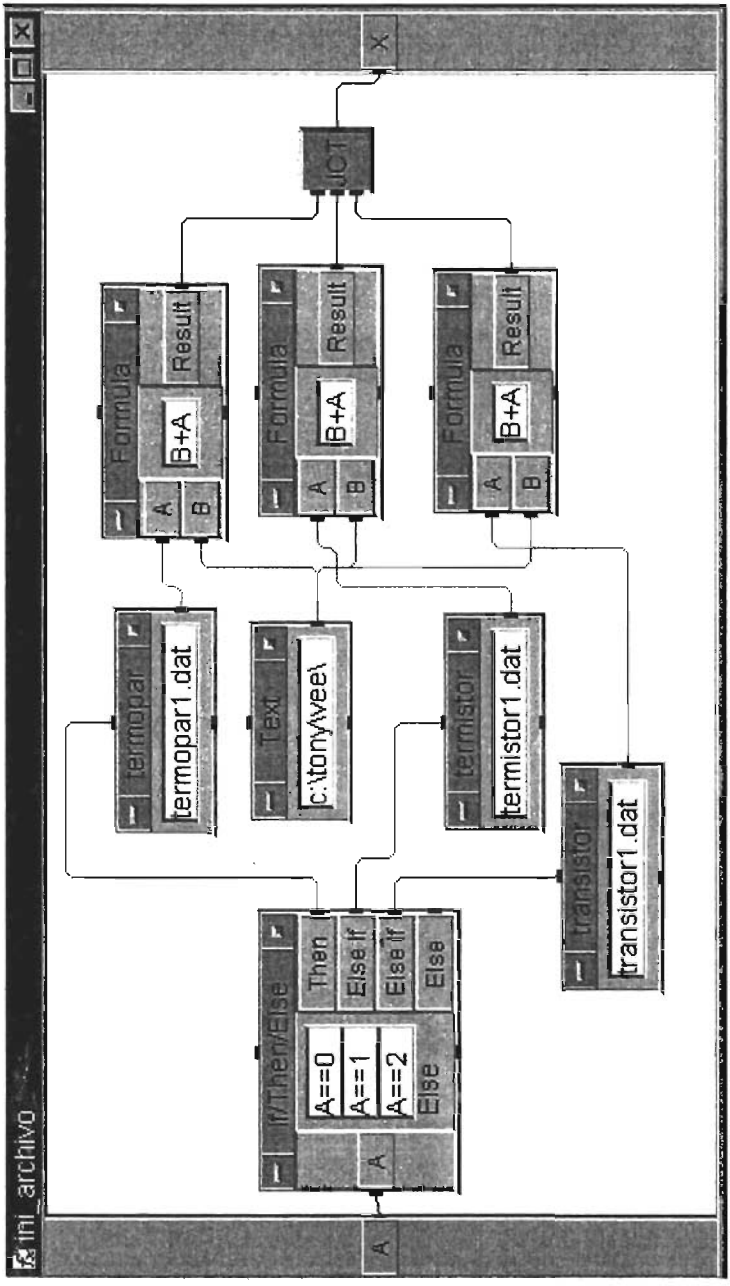


Figura 4.22: Objetos para inicializar archivo de guardado de datos.

4.4 Adquisición de datos de transductores

El laboratorio de Medición e Instrumentación cuenta con módulos didácticos que consisten en el transductor y su acondicionador y que proporcionan en su salida una magnitud de voltaje, corriente o resistencia equivalente a la magnitud física a la cual responden (kilogramos, desplazamiento, presión, luminosidad, etc.). La aplicación MUESTREADOR.VEE fue modificada de tal manera que con cada medición de salida de los transductores se pudiera introducir el valor de la variable independiente al cual corresponde dicha medición (ver figura 4.23), La aplicación MUESTREADOR.VEE ya modificada se llama MUESTREADOR_MODULOS.VEE. La forma de conexión de la PC–Instrumento se muestra en la figura 4.24.

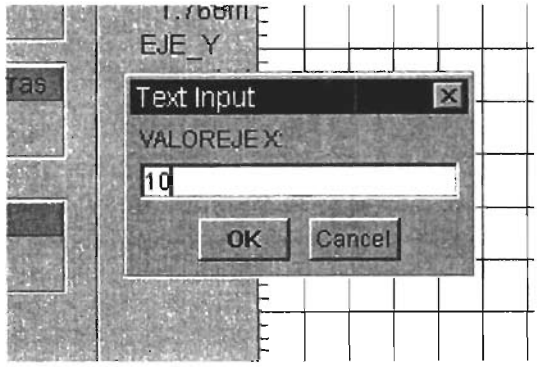


Figura 4.23: Introducción del valor de la variable independiente

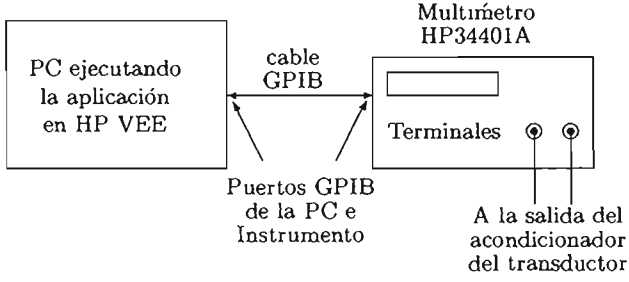


Figura 4.24: Esquema de conexión PC–HP34401A.

Los objetos que se agregaron a la aplicación MUESTREADOR.VEE y modificarla a MUESTREADOR_MODULOS.VEE son los siguientes:

- FLOW⇒JUNCTION (JCT_X)
- DATA⇒DIALOG BOX⇒TEXT INPUT, VALOR_EJE_X, entrada del valor en el eje x
- DATA⇒ALLOCATE ARRAY⇒REAL (ARREGLO_X)
- DATA⇒BUILD DATA⇒COORD (arreglo de coordenadas x,y)
- DEVICE⇒MATH&FUNCTIONS⇒OPERATORS⇒ASSIGNMENT

El objeto de asignación funciona del la siguiente manera, tiene tres entradas, una donde se le pasa el arreglo para almacenar los datos, otra para el índice del arreglo y otra para el valor a asignar de acuerdo al índice, en la figura 4.25 aparecen estos objetos para el valor del eje x y del eje y (EJE_X y EJE_Y respectivamente) la entrada uno es donde se pasa el arreglo (objeto ARREGLO_X y ARREGLO_Y respectivamente), la entrada dos es la que lleva el índice del arreglo en el que se van a almacenar los valores y la entrada tres es la entrada del valor, a la salida de este objeto se tiene un arreglo con los valores almacenados y que posteriormente se empaquetan en un arreglo de coordenadas mediante el objeto BUILD COORD. En la figura 4.26 se muestra la implementación de la aplicación MUESTREADOR_MODULOS.VEE.

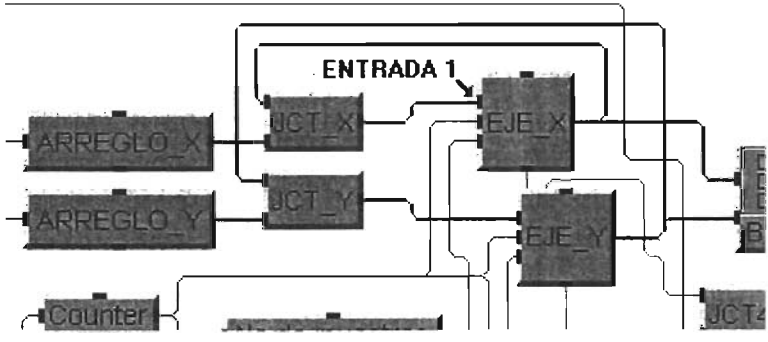


Figura 4.25: Objeto de asignación para almacenado de valores.

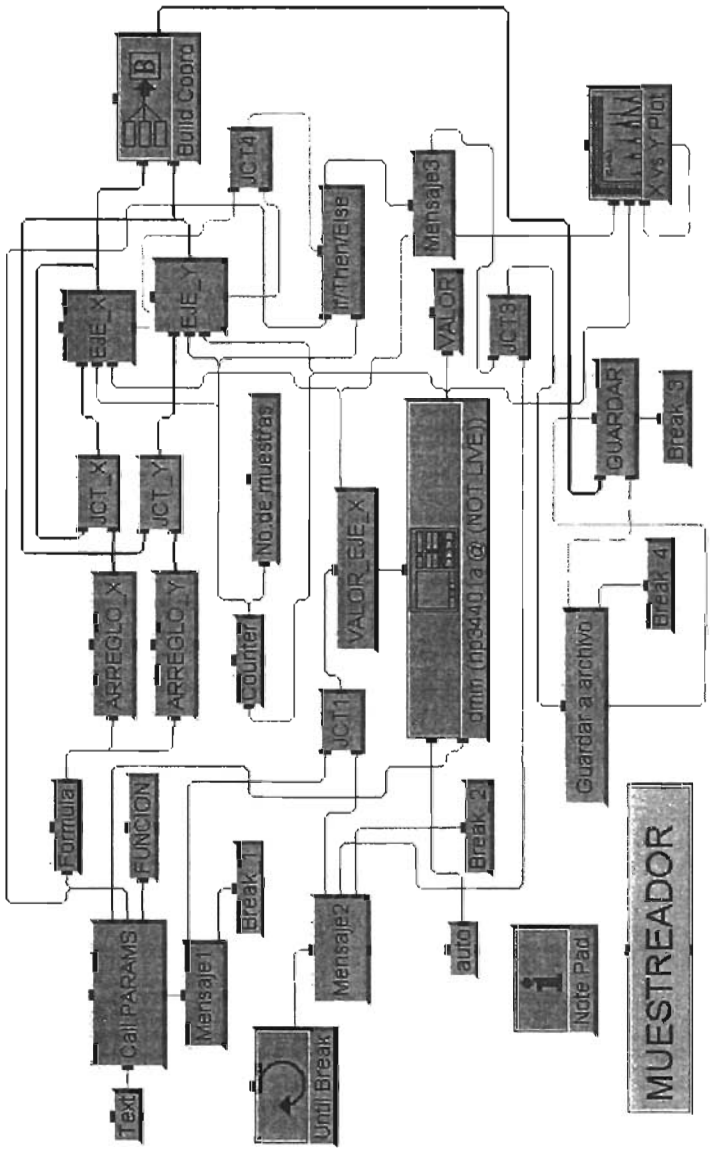


Figura 4.26: Programa MUESTREADOR_MODULOS.VEE.

El panel de usuario de la figura anterior se muestra a continuación junto con la caja de dialógo donde se pregunta por el valor correspondiente en el eje x de la medición tomada.

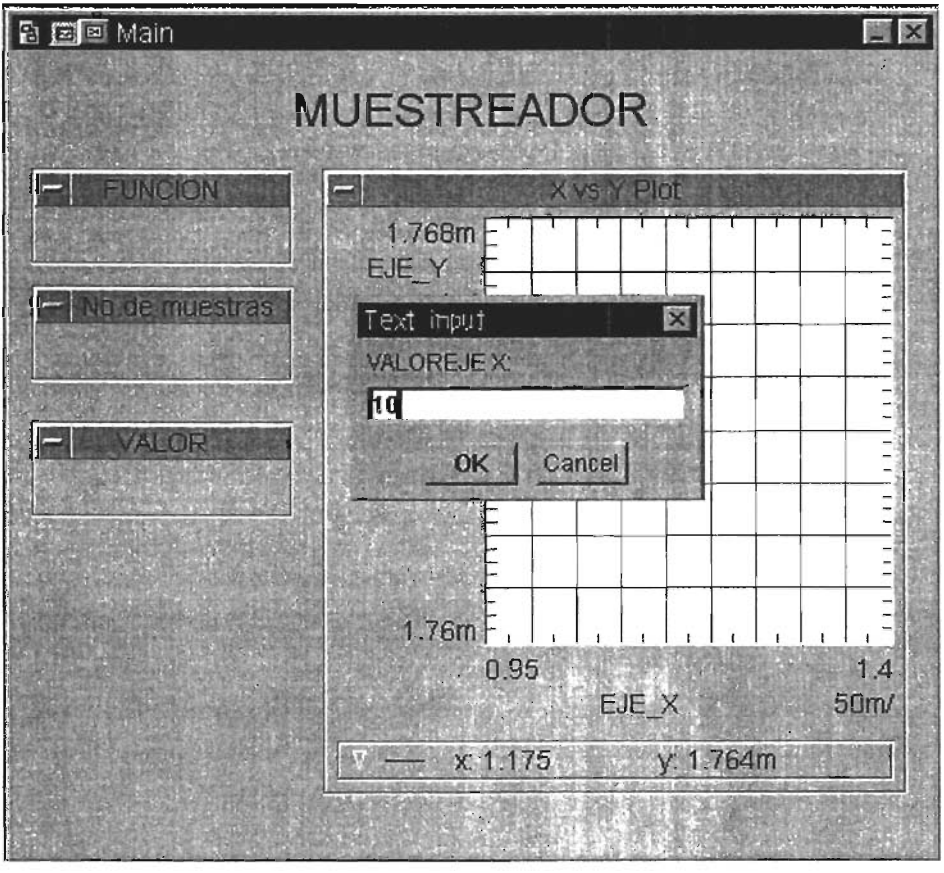


Figura 4.27: Panel de la aplicación MUESTREADOR_MODULOS.VEE.

El objetivo de esta aplicación es el de adquirir y guardar las lecturas del multímetro en forma de pares coordenadas (x,y) generadas por los transductores del Laboratorio de Medición e Instrumentación, en específico de cinco módulos , el G22, G25 y G24–MIL27, G27 y G11 (desplazamiento, fuerza, presión, LVDT[4], luminosidad[5]), haciendo uso de la aplicación MUESTREADOR_MODULOS.VEE. Para el módulo de luminosidad, específicamente con el fototransistor, fue necesario hacer otra aplicación que pudiera manejar dos multímetros con el esquema de la figura 4.28. Como la aplicación MUESTREADOR_MODULOS.VEE guarda los datos en un archivo, este puede manipularse después con cualquier otra herramienta de graficación y análisis que no necesariamente tiene que ser HP VEE.

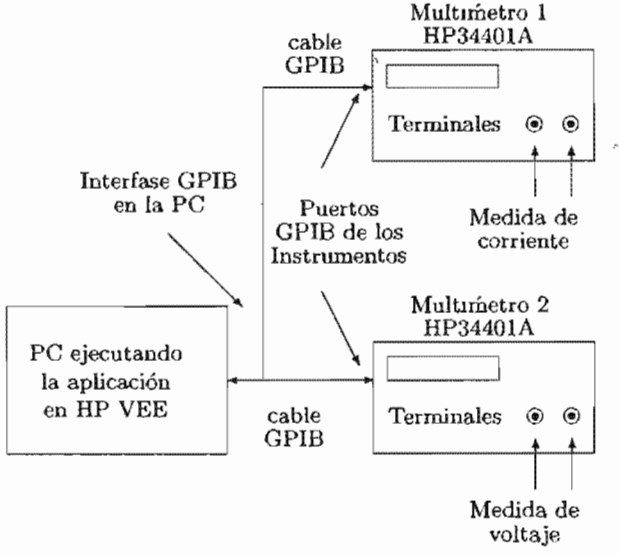


Figura 4.28: Esquema de conexión PC–dos multímetros.

[4]Transformador Diferencial de Variación Lineal.
[5]Fotorresistencia, fotodiodo y fototransistor

El módulo de desplazamiento es de tipo potenciométrico lineal, el cual mediante el acondicionador de señal convierte el valor de la resistencia en un valor de voltaje [6]. Para poder iniciar con la caracterización de este transductor primero se debe de calibrar el módulo acondicionador bajo el siguiente parámetro, a la distancia de cero milímetros el voltaje a la salida del acondicionador es de 8 volts. Una vez habiendo hecho la calibración se toman mediciones sucesivas del voltaje de salida a intervalos de distancia uniformes de 3 mm hasta llegar a una distancia de 30 mm que es el máximo desplazamiento para este transductor. La gráfica de los datos obtenidos se muestra en la figura 4.29.

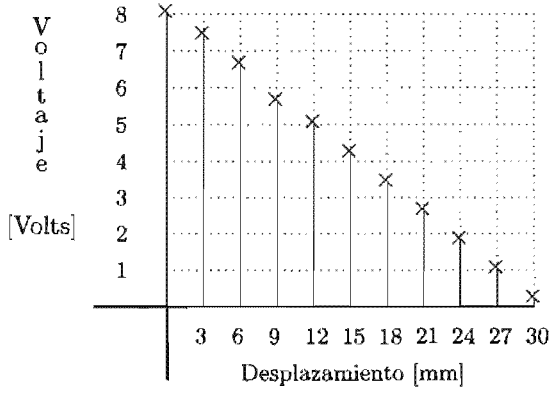


Figura 4.29: Gráfica Módulo G22 (desplazamiento).

El módulo de fuerza utiliza una celda de carga la cual tiene extensómetros resistivos en configuración de puente (similar al de la figura 2.5). La deformación que sufren los extensómetros en el puente produce una salida de voltaje muy pequeña la cual debe de amplificarse y filtrarse mediante el acondicionador de señal integrado en el módulo. Para este módulo es importante no exceder el límite de peso establecido ya que el equipo puede dañarse. Este módulo también debe de calibrase a un valor de peso con cierta salida de voltaje que en este caso es $0.1\frac{V}{kg}$. una vez hecha la calibración se procede a medir las salidas de voltaje con pesos de 250 gramos hasta llegar hasta 5.75kg, la figura 4.30 muestra la gráfica de los datos obtenidos del transductor de fuerza.

[6] Para mas información acerca del equipo referirse al manual de Prácticas de Transductores y Convertidores Eléctricos.

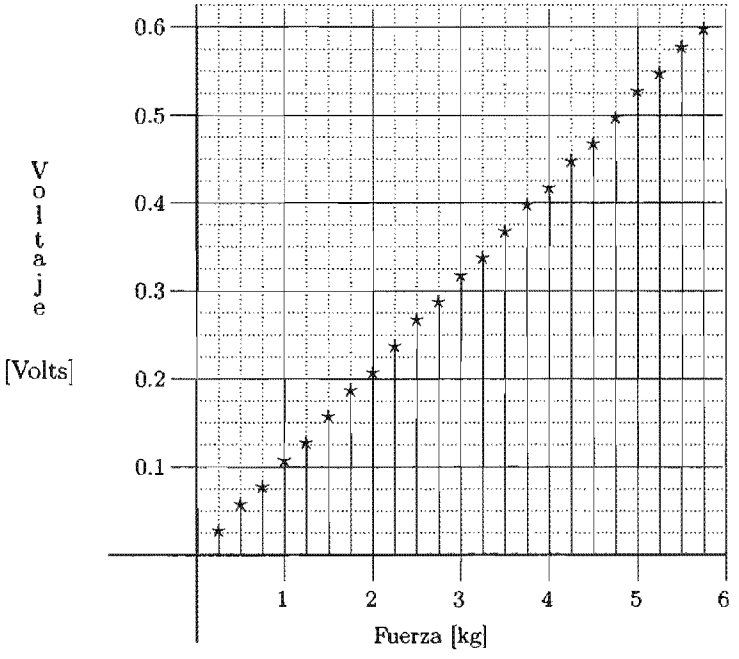


Figura 4.30: Gráfica Módulo G25 (fuerza).

El módulo de presión consta de la parte acondicionadora y de un émbolo con un manómetro que va conectado al elemento semiconductor piezorresistivo[7]. Este módulo se calibra con los siguientes valores, a presión de 0.14 bar el voltaje de salida es de 200mV y los intervalos de medición serán de 0.2 bar hasta el máximo valor de presión que se consiga. La figura 4.31 muestra la gráfica de los datos obtenidos del transductor de presión.

El LVDT es otro transductor de distancia en el que también se hace una calibración previa[8]. Este, dependiendo de la dirección de desplazamiento produce un voltaje negativo o positivo de ±volts, en la figura 4.32 se muestra la gráfica de los datos.

El módulo de transductores de luminosidad consta de una fotorresistencia, un fotodiodo y un fototransistor. Para la obtención de las lecturas del fototransistor fue necesario un programa mediante el cual se manejan dos

[7]Su entrada es una presión diferencial respecto al ambiente.

[8]Consultar el manual de prácticas de Transductores y Convertidores Eléctricos

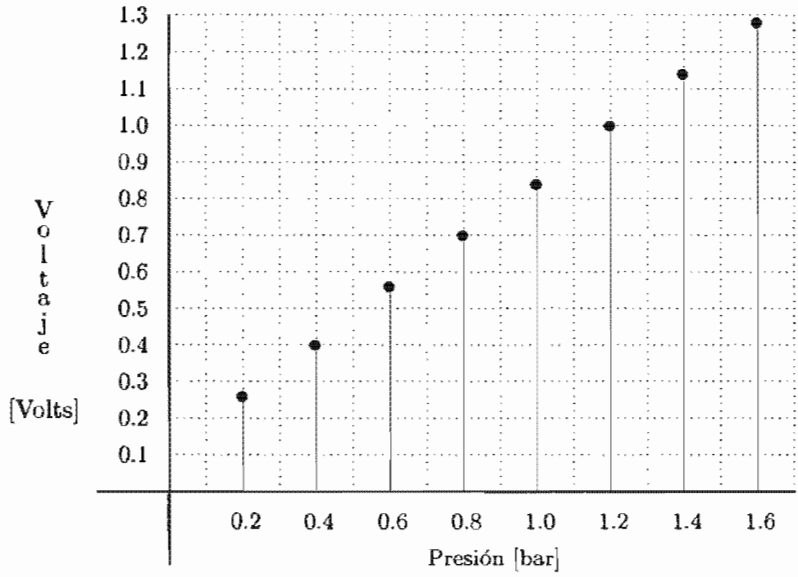


Figura 4.31: Gráfica Módulo G24 y MIL27(presión).

multímetros, uno para el voltaje de colector a emisor y otro para la corriente de colector, el programa se muestra en la figura 4.36 y para construirlo se utilizaron los siguientes objetos:

- DIRECT I/O (HP34401A_VOLT_1,2,3 y HP34401A_CURR_1,2,3)
- I/O⇒TO⇒FILE
- DISPLAY⇒X VS Y PLOT
- FLOW⇒REPEAT⇒UNTIL BREAK
- FLOW⇒CONFIRM (OK)
- FLOW⇒START
- FLOW⇒IF/THEN/ELSE
- FLOW⇒REPEAT⇒NEXT
- FLOW⇒REPEAT⇒BREAK

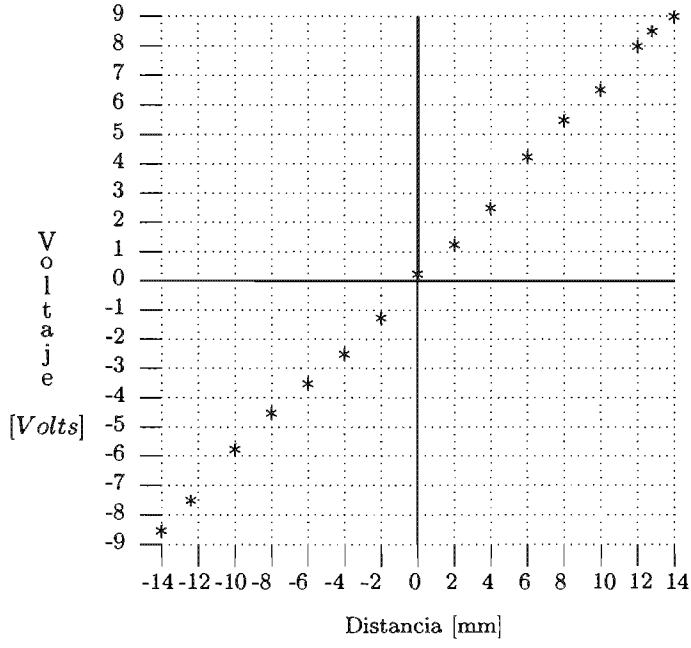


Figura 4.32: Gráfica Módulo G27 (LVDT).

- DATA⇒CONSTANT⇒INTEGER (integer y muestras)
- DATA⇒CONSTANT⇒TEXT (VOLT y CURR)
- DATA⇒BUILD DATA⇒COORD
- DATA⇒COLLECTOR (VOLT y CURR)
- DEVICE⇒FORMULA (A+1)
- DEVICE⇒ACUMULATOR

El objeto **START** se encarga de que las inicializaciones de ambos multímetros se lleven al mismo tiempo, en estas inicializaciones se introducen los siguientes comandos para cada multímetro (**HP34401A_VOLT_1** y **HP34401A_CURR_1**), ***RST** le hace un reset y ***CLS** limpia cualquier mensaje de error que se haya producido en el multímetro.

Sentencia SCPI	Descripción
"*RST"	Reestablecimiento del instrumento a un estado inicial
"*CLS"	Limpieza del cualquier mensaje, incluidos los de error

Lo siguiente es indicar la medición que va a hacer cada multímetro, en el objeto **HP34401A_CURR_2** se introduce una lista de comandos para medir corriente DC:

Sentencia SCPI	Descripción
A	Variable con una cadena de texto del comando SCPI
"SENS:CURR:RANG:AUTO ON"	Medición de corriente, con autorango habilitado
"SENS:CURR:DC:RES MAX"	Corriente de directa con la resolución máxima habilitada
"TRIG:SOUR:IMM"	Las mediciones se disparan de manera inmediata

La variable "A" se inicializa con la cadena **SENS:FUNC "CURRENT:DC"** mediante el objeto **CURR**. Para medir voltaje DC en el objeto **HP34401A_VOLT_2** se introducen los comandos:

Sentencia SCPI	Descripción
A	Variable con una cadena de texto del comando SCPI
"SENS:VOLT:RANG:AUTO ON"	Medición de voltaje, con autorango habilitado
"SENS:VOLT:DC:RES MAX"	Voltaje de directa con la resolución máxima habilitada
"TRIG:SOUR:IMM"	Las mediciones se disparan de manera inmediata

Y la variable "A" se inicializa con el texto **SENS:FUNC "VOLT:DC"** en el objeto **VOLT**.

Ahora que ya se ha especificado cual instrumento mide voltaje y cual mide corriente se entra en un ciclo infinito con el objeto **UNTIL BREAK**, el cual dura el número de mediciones a tomar. La salida del objeto **UNTIL BREAK** se conecta a la secuencia de entrada del objeto **HP34401A_VOLT_3** y la secuencia de salida de este se conecta a la secuencia de

entrada del objeto **HP34401A_CURR_3**, ambos tienen los siguientes comandos para recuperar el valor de la lectura y a ambos se les agrega una salida:

Sentencia SCPI	Descripción
"INIT"	Las lecturas están en la memoria interna del multímetro
"FETCH?"	Transferencia de las lecturas de la memoria interna al buffer de salida
X	Transacción de lectura, Las mediciones se guardan en la variable de salida **X**

La variable "X" es una variable donde se guarda la lectura actual del instrumento y que se propaga a la salida de los objetos anteriores.

Nótese que cuando se introduce la sentencia SCPI en la transacción aparece el texto **WRITE TEXT "<sentencia>" EOL** o **WRITE TEXT <variable> EOL** este texto es agregado automáticamente de acuerdo al tipo de transacción seleccionada en el objeto **DIRECT I/O** (en este caso de lectura o escritura), lo mismo sucede con el texto **READ TEXT <variable> REAL**. Por omisión se entiende que todas las sentencias SCPI se van a escribir al instrumento y que después de una sentencia SCPI de recuperación de lecturas sigue la transacción *READ*.

La secuencia de salida del objeto **HP34401A_CURR_3** se conecta a la secuencia de entrada del botón **OK**, con esto se logra detener la lectura hasta que se presione este botón, cuando este es presionado se activa el objeto **INTEGER** el cual manda su valor (en este caso es uno) al objeto **ACUMULATOR** el cual se encarga de llevar la cuenta de las lecturas junto con el objeto **FORMULA** cuyo contenido es **A+1**.

Cada que se presiona el botón **OK** los valores de voltaje y corriente se introducen en los objetos colectores de voltaje y corriente (**collector_VOLT** y **collector_CURR**) y después estos valores forman una coordenada con el objeto **BUILD COORD** la cual se escribe a un archivo con el objeto **TO FILE**.

El objeto **IF/THEN/ELSE** sirve para detener las mediciones (con el objeto **BREAK**) una vez llegado el contador (objeto **ACCUMULATOR** y **FORMULA** al número máximo de mediciones o continuar con ellas (con el objeto **NEXT**).

El objeto **X VS Y PLOT** despliega el comportamiento de las mediciones conforme se van tomando. Este programa no tiene un panel de usuario.

Para el comportamiento del fototransistor se tomaron los voltajes de colector–emisor y la corriente de colector a 4 diferentes valores de distancia de la fuente de luminosidad.

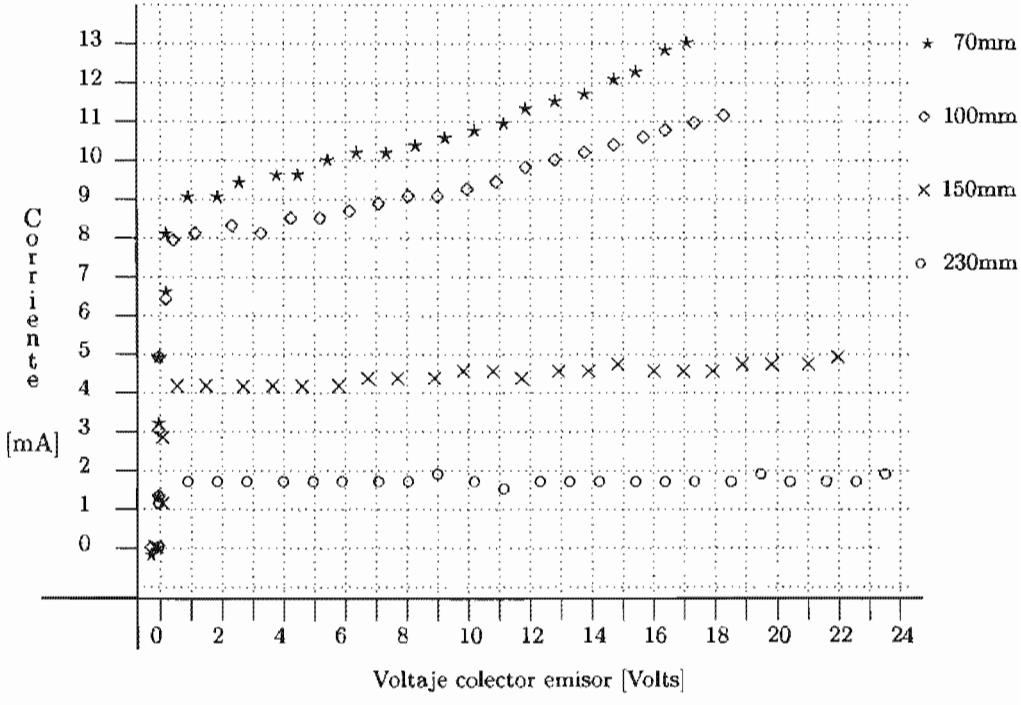


Figura 4.33: comportamiento del fototransistor.

El comportamiento del fotodiodo se hizo tomando el valor del voltaje en una resistencia de carga a diferentes valores de distancia, la característica de corriente es similar a la de voltaje dividida por el valor de la resistencia de carga que es de 100kΩ[9] y con la fotorresistencia se mide su resistencia a intervalos constantes de distancia de la fuente luminosa.

[9]Para mayor información revisar el manual de Prácticas de Transductores y Convertidores Eléctricos

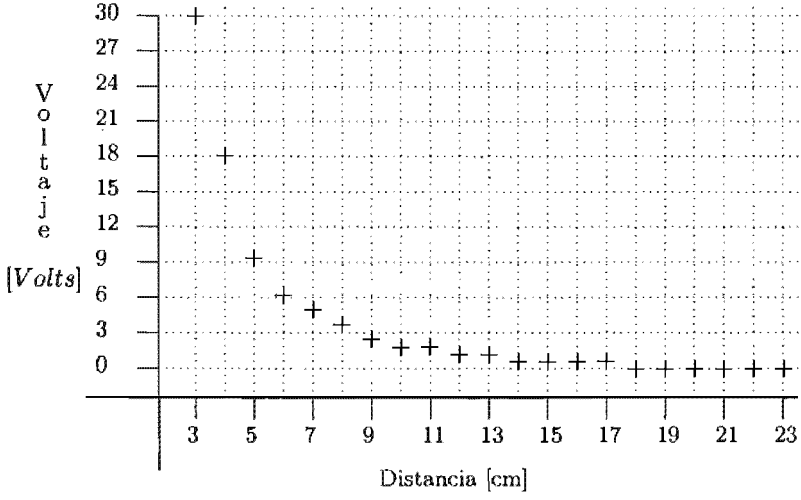


Figura 4.34: comportamiento del fotodiodo.

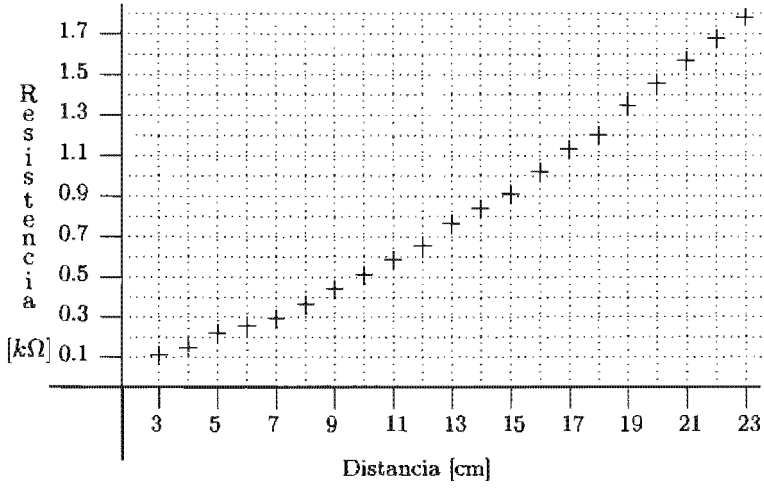


Figura 4.35: comportamiento de fotoresistencia.

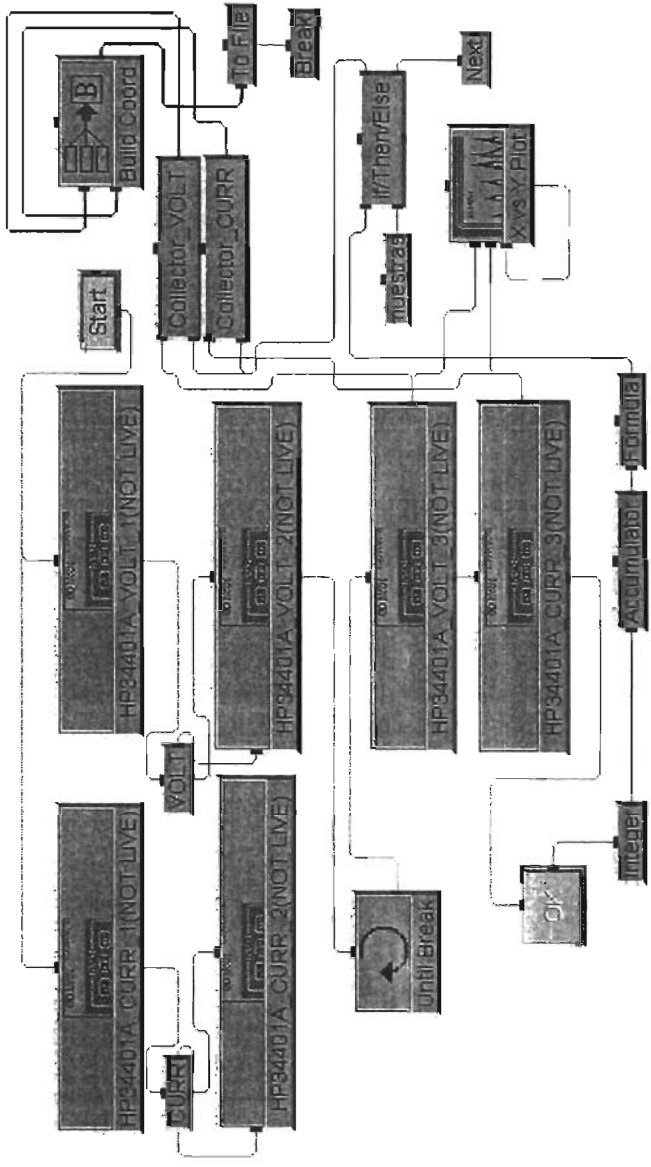


Figura 4.36: Programa para controlar dos multímetros.

4.5 Ajuste de curvas

El objetivo de esta aplicación AJUSTE.VEE es obtener el modelo matemático de un conjunto de datos que fueron previamente adquiridos y guardados a un archivo mediante el programa MUESTREADOR_MODULOS.VEE (datos de voltaje, corriente y resistencia con su respectiva variable independiente). En la siguiente figura se muestra la implementación de esta aplicación.

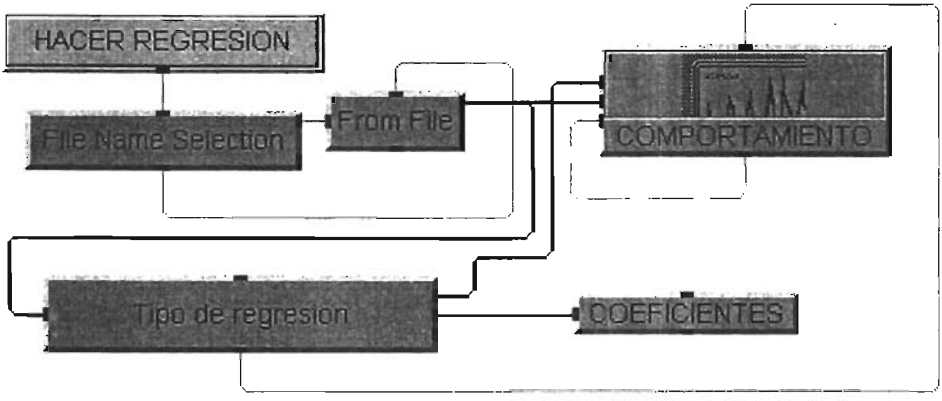


Figura 4.37: Implementación del programa de ajuste.

Cuando se ejecuta esta aplicación inmediatamente aparece una caja de diálogo como la de la figura 4.38 en la que se pregunta por el archivo de datos, esta caja de diálogo es generada por el objeto **File Name Selection** que se muestra en la figura 4.39. Previamente hay que seleccionar el tipo de regresión mediante el objeto **Tipo de regresion** (figura 4.40), por omisión esta seleccionado el tipo de regresión lineal. Si el tipo de regresión no es la adecuada será necesario seleccionar una que se ajuste a la forma de la gráfica de datos.

El panel de usuario de esta aplicación se muestra en la figura 4.41, en el hay un objeto nombrado **COEFICIENTES** de tipo **ALPHANUMERIC** en donde se muestran los mismos como resultado de la regresión seleccionada y en la gráfica se muestran las curvas con los datos ajustados y no ajustados.

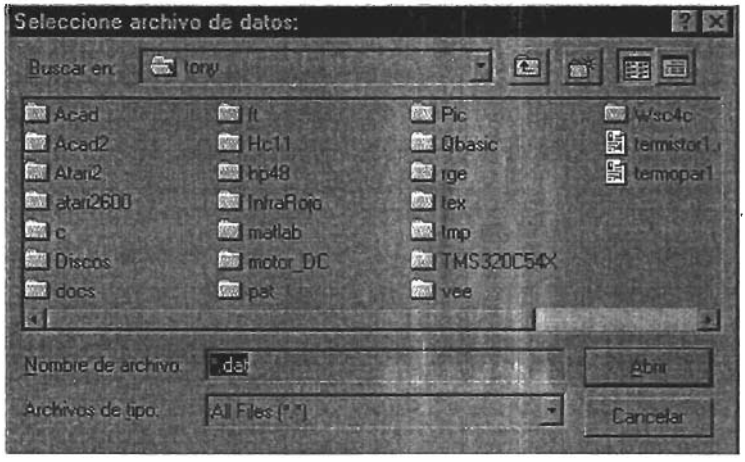


Figura 4.38: Caja de diálogo para selección de un archivo.

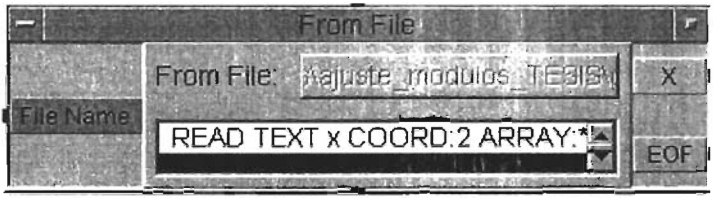


Figura 4.39: Vista abierta objeto File Name Selection.

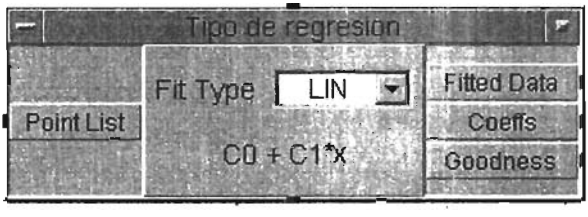


Figura 4.40: Objeto para hacer la regresión de los datos.

Los objetos utilizados en este programa son los siguientes:

- I/O⇒FROM⇒FILE
- DATA⇒DIALOG BOX⇒FILE NAME SELECTION
- DISPLAY⇒XY TRACE
- DISPLAY⇒ALPHANUMERIC
- FLOW⇒START (HACER REGRESION)
- DEVICE⇒REGRESION

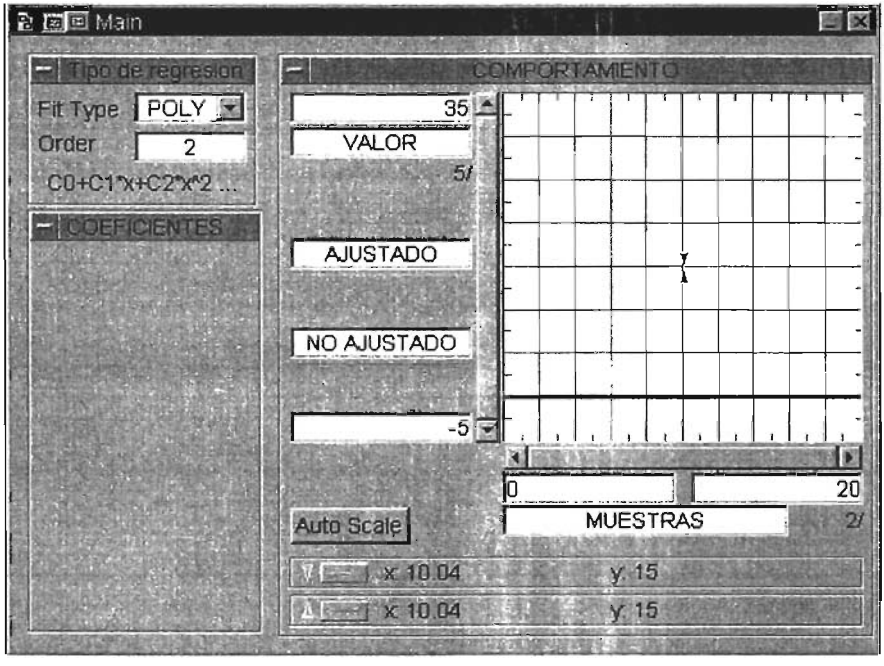


Figura 4.41: Panel del programa de ajuste.

El objeto **File name selection** (figura 4.39) tiene una ruta de búsqueda del archivo de datos por omisión, sin embargo, el usuario tiene la libertad de seleccionar el archivo de datos en una ruta distinta.

Al objeto **From file** (figura 4.42) se le agregó una salida para los datos leidos y una entrada para el nombre del archivo, este objeto funciona de manera inversa al objeto **To File** (página 55). Tiene una transacción **READ**

TEXT x REAL ARRAY:* la cual lee todo el archivo y lo guarda en la variable "x" hasta que se encuentre con un fin del mismo (EOF).

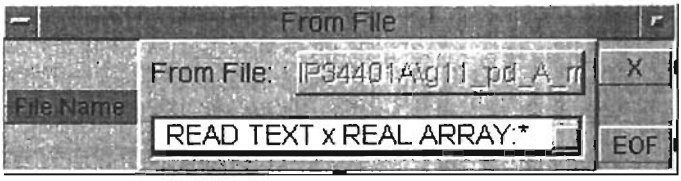


Figura 4.42: Vista abierta objeto From File.

4.6 Conversión de variables eléctricas

El nombre de esta aplicación es VISUALIZACION_VARIABLES.VEE, su objetivo es mostrar el valor de la variable física a la cual responde el transductor seleccionado a partir del modelo matemático generado por el conjunto de datos obtenidos de las lecturas de la variable eléctrica de salida del acondicionador de ese transductor. El esquema de conexión se muestra en la siguiente figura.

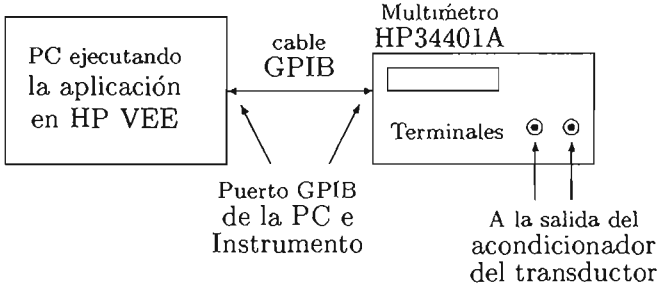


Figura 4.43: Esquema de conexión para visualizador.

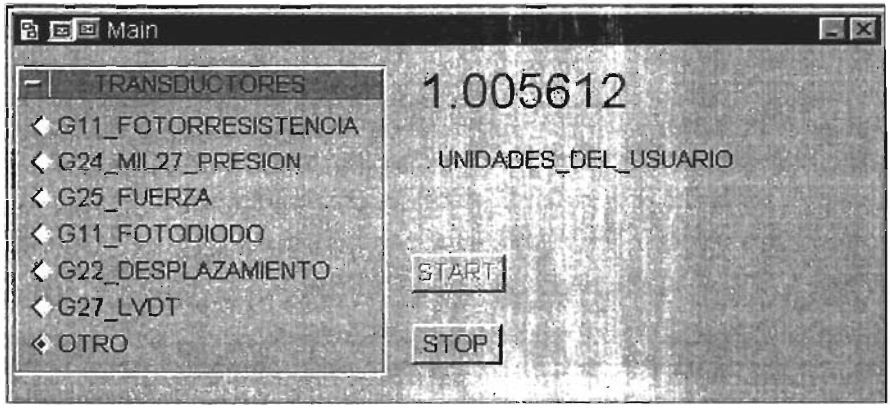


Figura 4.44: Panel del vizualizador de cantidades eléctricas a físicas

Para utilizar esta aplicación primero es necesario seleccionar el transductor con el cual se va a trabajar (ver figura 4.44), y a continuación presionar el botón **START** con lo que se mostrará el valor con las unidades correspondientes al transductor seleccionado.

Para la construcción de la aplicación se utilizaron los siguientes objetos:

- I/O⇒INSTRUMENT MANAGER⇒DIRECT I/O DEL HP34401A, objetos numerados del 1 al 5
- DATA⇒SELECTION CONTROLS⇒RADIO BUTTONS, TRANSDUCTORES
- FLOW⇒START
- FLOW⇒DELAY
- FLOW⇒REPEAT⇒NEXT
- FLOW⇒REPEAT⇒BREAK
- FLOW⇒IF/THEN/ELSE 2 objetos
- FLOW⇒UNTIL BREAK, 2 objetos
- FLOW⇒JUNCTION, 5 objetos JCT1, JCT2, JCT3, JCT4, JCT5
- FLOW⇒DELAY
- DATA⇒TOGGLE CONTROL⇒BUTTON, STOP
- DATA⇒CONSTANT⇒INTEGER
- DATA⇒CONSTANT⇒TEXT, 6 objetos, P_F_FD_D_LVDT, FOTORESISTENCIA, MM, KG, BAR, OHM, DEFINIR
- DEVICE⇒FORMULA, 7 objetos, RESISTENCIA, PRESION, FUERZA, FOTODIODO, DESPLAZAMIENTO, LVDT, OTRO
- DISPLAY⇒LABEL, 2 objetos, unidades y su valor

La implementacion de esta aplicación se muestra en la figura 4.45. Esta toma el valor de la salida del acondicionador del transductor ya calibrado ya que de no ser así se desplegarán datos de conversión erroneos. La selección del módulo se lleva a cabo con el objeto **IF/THEN/ELSE_1**, y también este mismo objeto activa las unidades de medición respectivas que se van a desplegar (objetos tipo Text **KG**, **MM**, **BAR**, **OHM** y **DEFINIR**, esta última opción es cuando el usuario necesite definir otra unidad diferente a las que ya están preestablecidas en la aplicación, ver figura 4.45), asi como el modelo matemático a utilizar para hacer la conversión de la cantidad

eléctrica obtenida del la salida del transductor del módulo seleccionado. Los modelos matemáticos están en los objetos **FORMULA** de la figura 4.45 (**RESISTENCIA**, **PRESION**, **FUERZA**, **FOTODIODO**, **DESPLAZAMIENTO**, **LVDT**), asi como también **OTRO** para el caso de que el usuario defina un modelo matemático distinto a los que ya están preestablecidos en la aplicación y contienen lo siguiente:

Módulo	Modelo
RESISTENCIA	$A * 0.1174 + 32.58$
PRESION	$A * 1.37 - 0.1199$
FUERZA	$A * 9.921 + 5.438^{-3}$
FOTODIODO	$A^{-0.5644} + 191.5$
DESPLAZAMIENTO	$A * (-3.824) + 30.4$
LVDT	$A * 1.616 - 0.2679$

Los modelos matemáticos anteriores fueron calculados con la aplicación AJUSTE.VEE a partir de los datos obtenidos de los módulos didácticos del Laboratorio de Medición e Instrumentación mediante la aplicación MUESTREADOR_MODULOS.VEE, el archivo de datos generado por esta última aplicación esta en la forma de pares coordenados (x,y), es decir $y = f(x)$, sin embargo, para que se pueda vizualizar en el panel de usuario la cantidad física a partir de la lectura de la variable eléctrica es necesario despejar la variable dependiente (y) del modelo matématico generado por la aplicación AJUSTE.VEE, es decir, hay que poner x en función de y ($x = f(y)$), o, de manera alternativa a hacer el despeje de la variable dependiente y aislar la independiente, se puede intercambiar x con y en el archivo de pares coordenados de cada módulo y generar el modelo a partir del intercambio de coordenadas.

El módulo de **RESISTENCIA** tiene en su salida unidades de resistencia y se convierten a milímetros mediante su modelo matemático, el módulo de **PRESIÓN** tiene en su salida volts y se convierten en bar, el de **FUERZA** tiene en su salida volts y se convierten en kg y los módulos **FOTODIODO**, **DESPLAZAMIENTO**, **LVDT** tienen en su salida volts y se convierten a milímetros (mm).

La inicialización del instrumento se hace con el objeto **DIRECT I/O 1** (en la figura 4.45) el objeto tiene el siguiente contenido:

Sentencia SCPI	Descripción
"*RST"	Reestablecimiento del instrumento a un estado inicial
"*CLS"	Limpieza del cualquier mensaje, incluidos los de error

Una vez inicializado el instrumento y verificado que tipo de módulo se va a utilizar se selecciona la sentencia SCPI correcta, esto se hace con los objetos de tipo **TEXT** que contienen lo siguiente:

Objeto	Sentencia SCPI
FOTORESISTENCIA	SENS:FUNC "RES" sentencia SCPI para medición de resistencia
P_F_FD_D_LVDT	SENS:FUNC "VOLT:DC" sentencia SCPI para medición de voltaje DC

Y que seleccionan el tipo de variable a medir, en este caso una es para medir voltaje (**DIRECT I/O 2**), previamente se le ha agregado una entrada y contiene lo siguiente:

Sentencia SCPI	descripción
A	Variable con una cadena de texto con la sentencia SCPI para medición de voltaje
"SENS:VOLT:DC:RANG AUTO ON"	medición de voltaje DC con autorango
"SENS:VOLT:DC:RES MAX"	resolución máxima
"TRIG:SOUR IMM"	se disparan las mediciones al ejecutar esta línea

La variable "A" recibe el comando SCPI de uno de los objetos tipo **TEXT** (FOTORESISTENCIA o P_F_FD_D_LVDT) para la medición de voltaje o resistencia. El objeto **DIRECT I/O 4** que aparece en la figura 4.45 tiene los comandos que se muestran a continuación para medir resistencia, la variable "A" tiene la misma función que en el objeto **DIRECT I/O 2**.

Sentencia SCPI	descripción
A	variable con una cadena de texto con la sentecia SCPI para medición de resistencia
"SENS:RES:RANG AUTO ON"	medición de resistencia con autorango
"SENS:RES:RES MAX"	resolución máxima
"TRIG:SOUR IMM"	se disparan las mediciones al ejecutar esta línea

Para ambos casos la primera línea indica el tipo de cantidad a medir (en este caso resistencia y voltaje), la segunda línea pone al multímetro en autorango, la tercer línea selecciona la resolución máxima y la última línea hace que la lectura se dispare de manera inmediata. Una vez fijados estos parámetros se inicia un ciclo infinito en el que se va a estar vizualizando en el panel de usuario la variable que se esta midiendo pero ya convertida a valor de variable física, las lecturas se obtienen a cada instante con los objetos **DIRECT I/O 3** para voltaje y **DIRECT I/O 5** para resistencia, a ambos se les agrega una salida para la recuperación de la lectura.

Sentencia SCPI	descripción
"INIT"	la lectura se almacena en la memoria del instrumento
"FETCH?"	transfiere la lectura de la memoria al buffer de salida
X	transacción de lectura, las lecturas se guardan en la variable de salida **X**

Las unidades de medición que se despliegan se seleccionan con el objeto **IF/THEN/ELSE_1** y activa también los modelos matemáticos y las unidades a desplegar. El objeto **JCT1** selecciona que módulos son de voltaje (LVDT, FUERZA, PRESION, FOTODIODO, DESPLAZAMIENTO), el objeto **JCT2** se encarga de seleccionar las unidades de los módulos que sean de desplazamiento (LVDT, FOTODIODO, DESPLAZAMIENTO). El objeto **JCT3** se encarga de tomar las unidades a desplegar para cada módulo. Por último el objeto **JCT4** se encarga de propagar la medición ya convertida del módulo seleccionado en el objeto **LABEL** al cual se le ha agregado una entrada para recibir la medición resultante.

Los objetos de texto **MM**, **BAR**, **OHM** y **KG** contienen el texto de las unidades correspondientes para ser despliegadas en otro objeto **LABEL** que también se le ha agregado una entrada para este propósito.

Cuando el valor se ha desplegado en el objeto **LABEL**, la secuencia de salida de este objeto se conecta al objeto **STOP** que es de tipo **TOGGLE BUTTON** (se le ha agregado la entrada de valor por omisión) sirve para detener la aplicación dependiendo de su estado (uno o cero). La salida de este botón se conecta en la entrada del objeto **IF/THEN/ELSE_2** si el valor de entrada es uno (botón **STOP** presionado) se activa el objeto **INTEGER** que contiene un cero y cuya salida se conecta a la entrada del objeto **STOP** para dejarlo en su estado original, cuando esto sucede se detiene la aplicación. En caso contrario la aplicación continua ejecutándose.

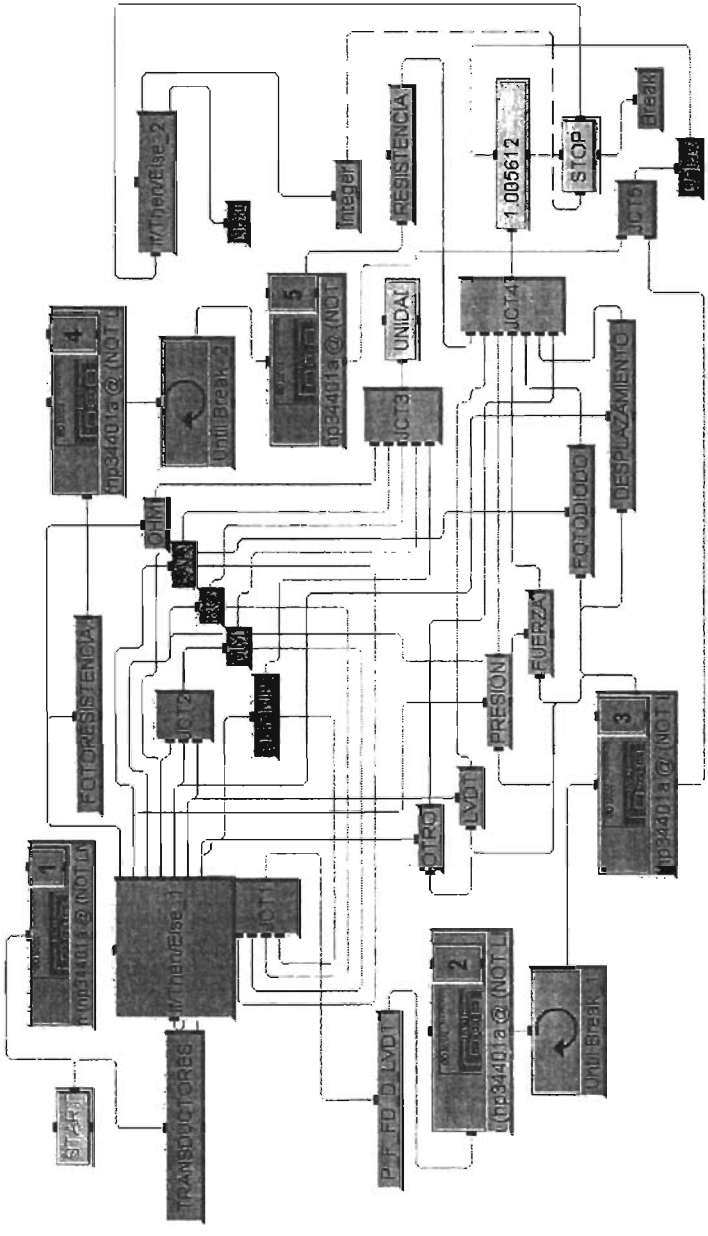


Figura 4.45: Programa VISUALIZACION_VARIABLES.VEE

4.7 Osciloscopio

El nombre de esta aplicación es OSCILOSCOPIO.VEE y su objetivo es adquirir los datos del canal uno del osciloscopio para posteriormente guardarlos a un archivo y ser analizados con cualquier otra herramienta como una hoja de cálculo. El esquema de conexión PC–Instrumento se muestra en la siguiente figura.

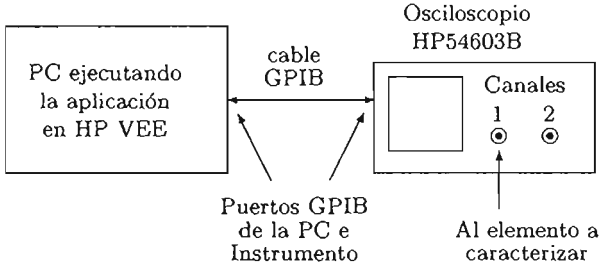


Figura 4.46: Esquema de conexión PC–HP54603B.

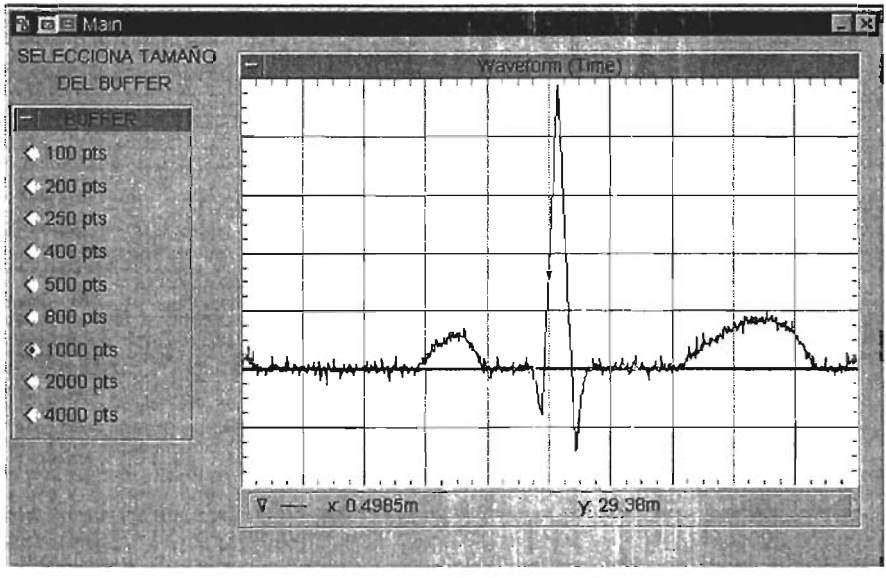


Figura 4.47: Panel del programa para manejar el HP54603B.

El instrumento utilizado es un HP54603B que tiene un ancho de banda de 60 MHz. Para la construcción de esta aplicación se utilizaron los siguientes objetos:

- DISPLAY⇒LABEL, SELECCIONA TAMAÑO y DEL BUFFER
- DISPLAY⇒NOTE PAD
- DISPLAY⇒ALPHANUMERIC
- DISPLAY⇒WAVEFORM
- FLOW⇒REPEAT⇒UNTIL BREAK
- FLOW⇒REPEAT⇒BREAK
- DATA⇒SELECTION CONTROL⇒RADIO BUTTONS, BUFFER
- DATA⇒DIALOG BOX⇒FILE NAME SELECTION, GUARDAR ARCHIVO
- I/O⇒INSTRUMENT MANAGER⇒COMPONENT DRIVER DEL HP54600
- I/O⇒TO⇒FILE

El panel de usuario de la aplicación se muestra en la figura 4.47 y muestra los tamaños de buffer disponibles y la gráfica[10] del mismo actualizandose. La implementación del programa se muestra en la figura 4.48 y funciona de la siguiente manera: el objeto **UNTIL BREAK** hace un ciclo infinito por medio del cual los datos obtenidos del osciloscopio irán actualizando al objeto **WAVEFORM (Time)** que se encarga de mostrar la gráfica de los valores. El ciclo infinito también constantemente activa al objeto **RADIO BUTTONS** (BUFFER) en el cual se selecciona el tamaño del buffer de datos a guardar, al objeto **RADIO BUTTONS** se le agregan opciones para manejar todos los tamaños de buffer que se pueden seleccionar; 100, 200, 250, 400, 500, 800, 1000, 2000, 4000 puntos, entre mas puntos haya en el buffer mas lenta va a ser la actualización del mismo en el objeto **WAVEFORM (time)**. La salida del objeto **RADIO BUTTONS** (donde entrega un número ordinal) se conecta a la entrada **NUM_POINTS** (previamente agregada) del **COMPONET DRIVER** del HP54600 y la salida **WF_CH1**

[10]Señal simulando un latido cardiaco originado por el generador arbitrario de funciones HP33210A con las características de 63.46mV_{pp} de 1kHz.

(que previamente fue agregada también) del **COMPONET DRIVER** del HP54600 se conecta al objeto **ALPHANUMERIC** para ver los datos que se van actualizando. El objeto **NOTE PAD** y los objetos **LABEL** SELECCIONA TAMAÑO y DEL BUFFER funcionan simplemente para hacer la descripción del programa y ponerla como una nota y como etiquetas que se despliegan en el panel de usuario.

El objeto **FILE NAME SELECTION**(GUARDAR ARCHIVO) abre una caja de diálogo en la cual se introduce la ubicación y nombre del archivo bajo el cual se van a guardar los datos. La salida de este objeto que tiene el nombre del archivo se conecta al objeto **TO FILE** al cual previamente también se le ha agregado la entrada **File Name** y que es el que se encarga de guardar el buffer de datos.

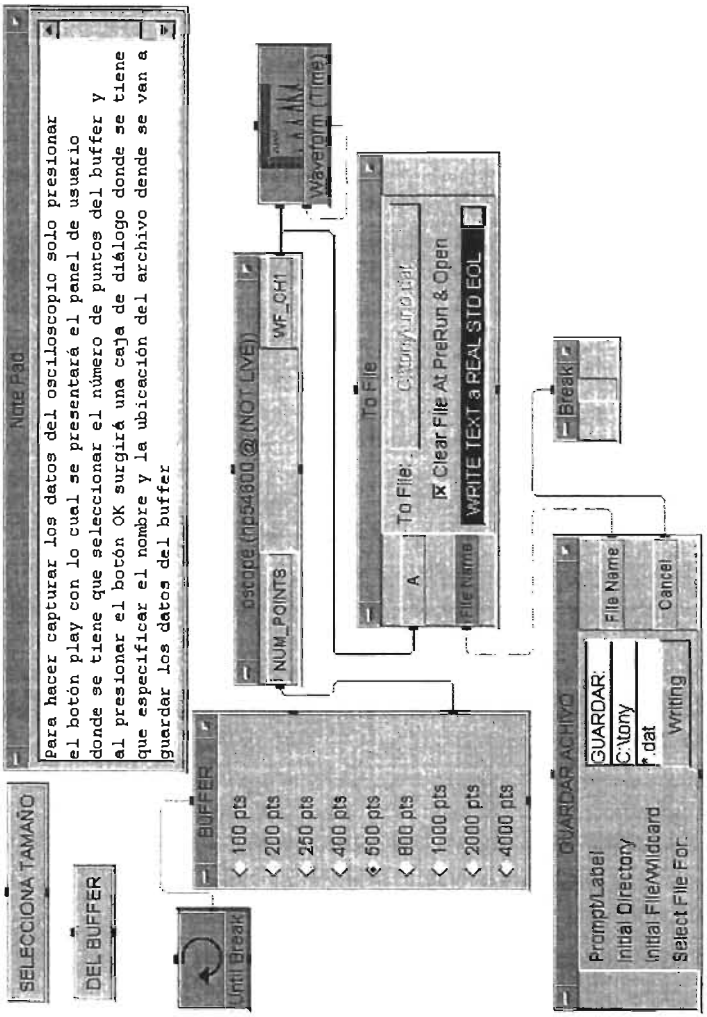


Figura 4.48: Implementación del programa para manejar el HP54603B.

4.8 Integración con el lenguaje C

En esta seccción se proporcionan los lineamientos para que el usuario haga la integración de llamadas a una DLL[11] dentro del ambiente de HP VEE. El uso de estas librerías se vuelve útil y ventajoso para propósitos específicos que no están disponibles en HP VEE, o para funciones que sean mas facilmente programables en otro lenguaje, en este caso "C". Hay que notar que aunque el programar una DLL agrega complejidad al proceso de desarrollo de aplicaciones, se gana velocidad en la ejecución de las rutinas. En la sección "Operaciones entre funciones" se utiliza ampliamente la DLL.

Las DLL's (Dynamic Link Library) son módulos compilados que contienen una o varias funciones y datos que se cargan en tiempo de ejecución dentro del espacio de memoria del programa que las llama, en este caso, HP VEE es el programa que mediante el objeto *Import Library* carga la DLL. Dentro de las DLL se definen dos tipos de funciones: *exportables* e *internas*. Las funciones *exportables* pueden ser llamadas desde cualquier objeto de HP VEE mientras que las funciones *internas* solo pueden ser llamadas por funciones que están dentro de la DLL sean o no exportables.

Otra ventaja del uso de funciones en una DLL es el proporcionar seguridad al código fuente de las rutinas propietarias. Una desventaja en el uso de las DLL's es que tienen que programarse tomando en cuenta las posibles condiciones de error de tal manera que no permitan su propagación una vez que se ha llegado a esa condición y manejar la cantidad de memoria interna que demanden ya que si no se calculan y planean correctamente estos elementos, normalmente el programa en HP VEE se detiene y se tiene que reestablecer la computadora.

La construcción de una DLL involucra el uso de un compilador[12] que genere una DLL de 32 bits ya que las DLL de 16 bits solo pueden utilizarse en versiones de HP VEE 3.2 hacia abajo. No voy a describir el desarrollo de la DLL en un ambiente de programación específico ya que cada usuario puede usar el compilador de su agrado.

La escritura del código de una función dentro de la DLL es idéntico a cualquier otra función, la diferencia son los modificadores en la declaración

[11] Dentro de HP VEE estas DLL son llamadas librerías una vez que han sido importadas.
[12] En este caso Borland C++ Builder 3.

de las mismas. En "C" un modificador es una palabra reservada que en una declaración de función o variable altera la forma en como son llamadas, compiladas y guardadas. Una palabra reservada es un identificador utilizado por el compilador para propósitos y funciones especiales y no deben ser usadas como identificadores de variables y/o funciones, para mas información sobre esto referise al manual de programación del compilador utilizado.

Como se describió al inicio de esta sección, una DLL puede contener una o varias funciones y si se desea que todas puedan ser utilizadas dentro del ambiente de HP VEE tienen que declararse exportables, esto se logra mediante el modificador `_declspec{dllexport}`, adicionalmente a este modificador se utiliza `extern "C"`.

HP VEE tiene la misma convención del llamado a funciones del lenguaje "C" estándar, pero como el compilador utilizado generalmente es de C++, este tiene una convención de llamado a funciones diferente debido a que C++ permite el sobrecargado de funciones (lo que no existe en "C"), es decir, pueden existir varias funciones que tengan exactamente el mismo nombre pero el compilador las distingue por el número, tipo de parámetros y tipo de dato que regresan. Para que el compilador anule la llamada a la función como si fuera sobrecargada se utiliza el modificador `extern "C"` como parte de su declaración. A continuación se muestra un ejemplo del formato general de una función exportable,

```
#define DLLEXPORT _declspec{dllexport}

extern "C" DLLEXPORT tipo_dato nombre_funcion(parametros)
{
//cuerpo_de_la_funcion
.
.
.
.
.
}
```

Y el formato general de una función interna o simplemente una función es el siguiente.

```
tipo_dato nombre_funcion(parametros)
{
//cuerpo_de_la_funcion
.
.
.
.
}
```

Para la función exportable, la directiva del preprocesador `#define DLLEXPORT _declspec{dllexport}` permite abreviar `_declspec{dllexport}` en la cadena `DLLEXPORT` para no tener que escribir `_declspec{dllexport}` cada vez que se incluya una nueva función que sea exportable dentro del módulo de funciones de la DLL. Lo siguiente en el listado es la declaración de la función con los modificadores; `tipo_dato` es el tipo de dato válido para "C" compatible con HP VEE que va a regresar la función (ver tabla 4.2). Tiene lugar mencionar que en el módulo de funciones no todas tienen que ser exportables por las causas ya descritas acerca de proporcionar seguridad al código propietario.

Para que en HP VEE sea posible importar las función que están compiladas dentro de la DLL se tiene que generar un archivo de encabezado donde aparezcan los nombres de las funciones asi como sus argumentos. El siguiente listado muestra el formato de este archivo el cual puede ser nombrado de cualquier manera aunque se recomienda que se llame de la misma manera que la DLL pero con extensión **.h**.

```
(tipo_dato) funcion1(parametros)
(tipo_dato) funcion2(parametros)
.
.
.
(tipo_dato) funcionn(parametros)
```

Los tipos de datos equivalentes de C++ en HP VEE se muestran en la siguiente tabla:

HP VEE	C++
Int32	long int
Real o Real64	double
Text	char *

Tabla 4.2: Equivalencias entre tipos de datos

Para el manejo de arreglos en "C" (ya sean enteros, reales o caracteres) se utilizan apuntadores a enteros (int *), apuntadores a double (double *) y a char (char *). Aunque HP VEE dispone de tipos de datos compuestos de pares de números reales agrupados en arreglos tales como coordenadas, complejos, forma de onda, espectro, éstos primero tienen que ser divididos en sus componentes reales para que puedan ser procesados por la función en la DLL y luego de esto se puedan volver a construir los datos originales si es que asi se desea. De igual manera, "C" permite la composición de datos mediante el uso de estructuras, estas no son compatibles con los tipos de datos compuestos que tiene HP VEE por lo que no pueden ser usadas para recibir los datos compuestos de HP VEE en los o el argumento de funciones.

El objeto que permite integrar la DLL esta en el menu DEVICE⇒IMPORT LIBRARY (ver figura 4.49).

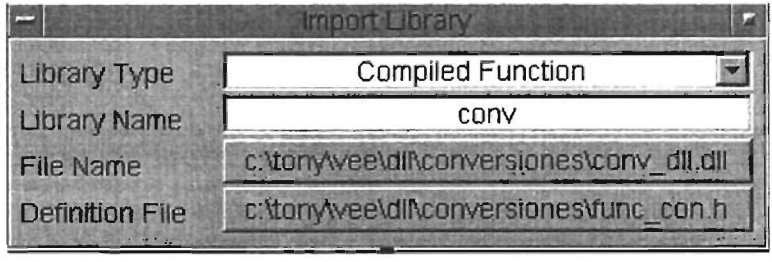


Figura 4.49: Objeto para importar una librería.

En el campo de **Library Type** hay tres opciones de las cuales utilizaremos el de tipo **Compiled Function** ya que se trata de una DLL que en efecto es una función compilada, el siguiente campo es **Library Name** donde vamos a asignar un nombre *lógico* a la DLL dentro del ambiente interno de HP VEE, este nombre es cualquiera de nuestra elección teniendo

cuidado de que identifique el tipo de funciones que representa y que no este duplicado en caso de que no sea la única DLL importada [13], el campo **File Name** es para seleccionar la ruta donde se encuentra la DLL y el último campo **Definition File** es el archivo de encabezado con extensión ***.h**.

Una vez importada la DLL, las funciones que agrupa se llaman mediante el objeto *Call* mostrado en la figura 4.50. Para que una de las funciones de la DLL pueda ser utilizada e identificada se debe de poner una cadena de texto en el campo **Function Name** del objeto *call* con el siguiente contenido: el nombre *lógico* que se le dio a la DLL en el objeto **Import Library**, como separador un punto, seguido del nombre de la función a utilizar que pertenezca a esa DLL y se maneja tal y como si fuera una función interna de HP VEE; adicionalmente se le deben agregar las entradas y salidas, las entradas son los parámetros de la función en el orden en que se declararon en el archivo de encabezado, y la o las salidas dependen de si la función regresa algún dato o no; si se tiene como parámetro de entrada uno o varios apuntadores a cualquier tipo válido este apuntador se tendrá como salida y si la función va a devolver algún resultado ésta salida aparece como *Ret Value*.

Cuando se le agregan las entradas y las salidas al objeto *Call* estas tienen un nombramiento automático, esto no nos debe preocupar ya que cuando la DLL se carga, ésta reasigna los nombres que se encuentran en el archivo de encabezado de la DLL. Lo que si es muy importante es escribir correctamente el nombre de la función y la librería a la cual pertenece.

Otro método para llamar a una función ya compilada es primero importar la DLL como se describió y ejecutar el programa con solo este objeto, una vez hecho esto aparece la librería en el explorador de funciones de HP VEE como se muestra en la figura 4.52.

En el explorador de funciones podemos generar las llamadas a las funciones automaticamente posicionándonos sobre ellas y presionando el botón derecho del mouse con lo que aparecerá un menú que se muestra en la figura 4.52 y después seleccionando **Generate call**.

En la figura 4.50 se muestra un ejemplo de un objeto **Call** que llama

[13] en la figura 4.52 se tienen dos DLL importadas, una llamada *conv* y otra *IIR* con sus respectivas funciones

a la función **angulo** de la librería **conv**, en el campo **Function Name** se identifican estos dos nombres y aparecen las entradas (**x**,**y**) y salidas (**Ret Value**) que pertenecen a dicha función.

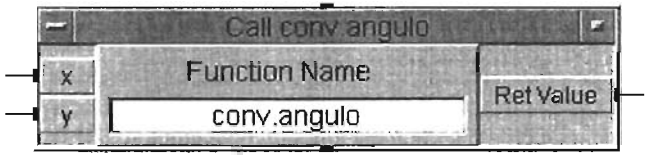


Figura 4.50: Objeto Call para llamar a la función dentro de la DLL.

Una ventaja adicional del uso de DLL's es que las funciones compiladas puede ser llamadas desde el objeto **Formula** (en este caso se deben de incluir sus argumentos) o desde otra expresión dentro de cualquier objeto que se evalue en tiempo de ejecución (previamente hay que cargar la DLL), el nombre *lógico* de la librería a la cual pertenece puede ir o no en el nombre de la función, (este se muestra en la figura 4.51).

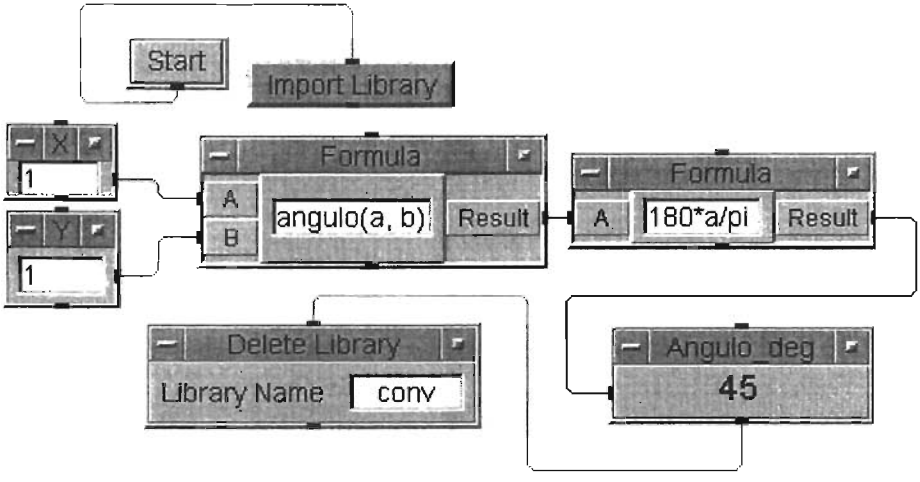


Figura 4.51: Uso de una función compilada desde el objeto Formula.

Una vez terminado el programa, la DLL permanece en memoria a menos que se cierre el ambiente de HP VEE. Si se desea borrar explicitamente alguna DLL que no se ocupe después de cierta tarea aún si el programa en HP VEE sigue utilizando otras librearías se agrega el objeto *Delete Library*

en la aplicación con el nombre *lógico* de la DLL como se muestra en el programa de la figura 4.51.

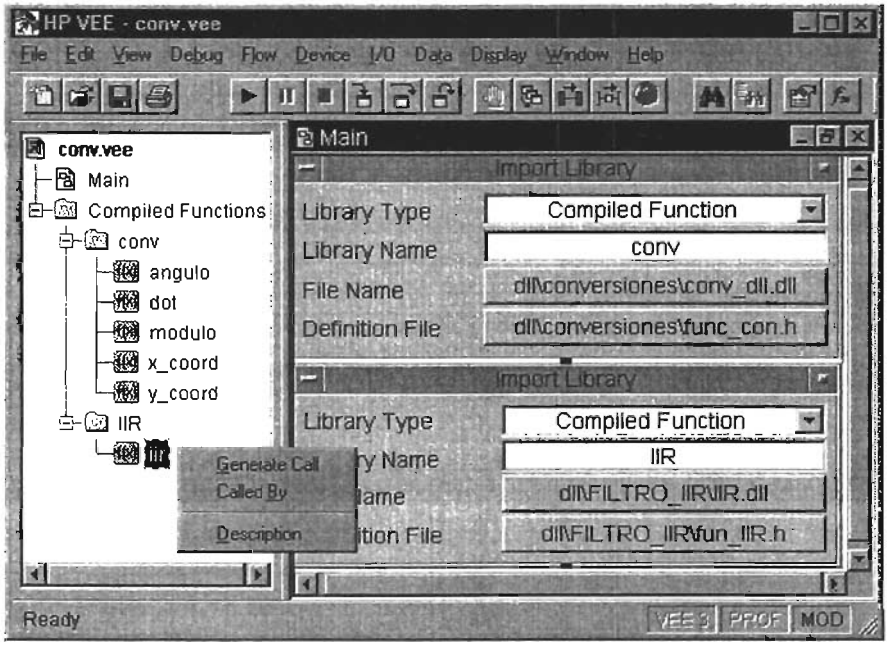


Figura 4.52: Dos librerías con sus funciones respectivas.

4.9 Generación de funciones

El nombre de esta aplicación es GENERADOR.VEE, y su objetivo es demostrar las formas de onda que es posible generar con el HP33120A. El panel de usuario se muestra en la figura 4.54 y el programa completo se muestra en la figura 4.55. El esquema de conexion de la PC—Instrumento se muestra en la siguiente figura.

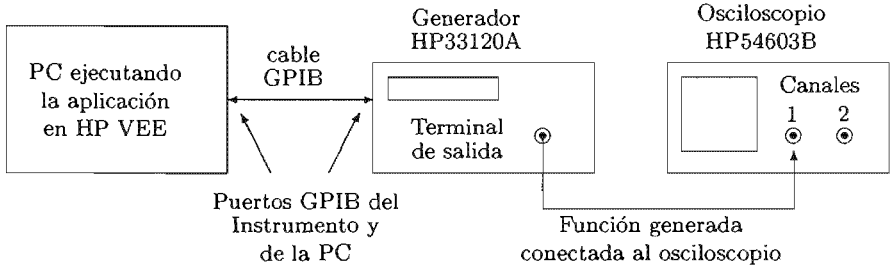


Figura 4.53: Esquema de conexión PC–Instrumento (HP33120A).

La aplicación funciona de la siguiente manera, primero aparece el panel de usuario como el de la figura 4.54, en el se escoge la función que se decida generar asi como sus parámetros (frecuencia, amplitud, offset, etc.). Para que los parámetros y tipo de función tengan efecto en el instrumento es necerario presionar el botón **OK** y automaticamente se generará la funcion seleccionada.

Para la construcción del programa se necesitan los siguientes objetos:

- FLOW⇒REPEAT⇒UNTIL BREAK
- FLOW⇒CONFIRM (OK)
- DATA⇒SELECTION CONTROL⇒RADIO BUTTONS
- DATA⇒SELECTION CONTROL⇒DROP-DOWN LIST
- FLOW⇒IF/THEN/ELSE
- DEVICE⇒FORMULA
- I/O⇒INSTRUMENT MANAGER⇒ PANEL DRIVER del HP33120A
- I/O⇒INSTRUMENT MANAGER⇒DIRECT I/O del HP33120A (2 objetos)

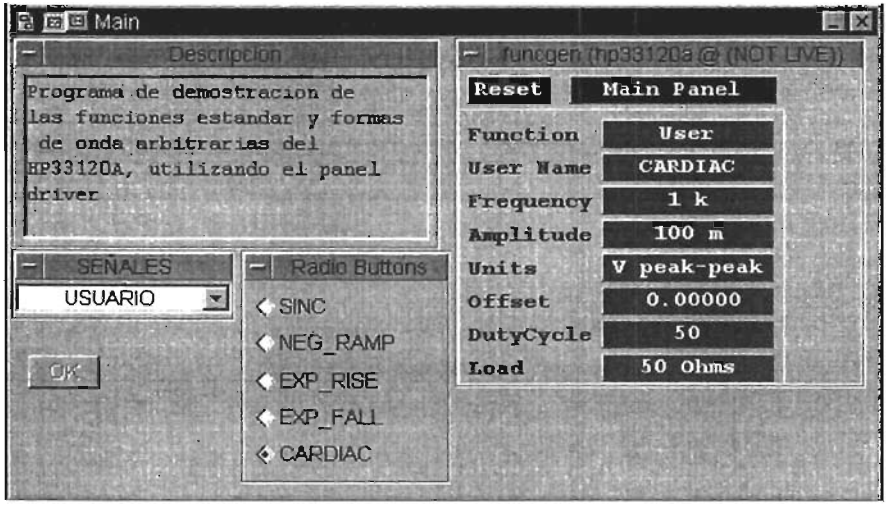


Figura 4.54: Panel de usuario del programa de demostración del HP33120A

- DISPLAY⇒ALPHANUMERIC (dos objetos)
- DISPLAY⇒NOTEPAD

El programa completo de esta aplicación se muestra en la figura 4.55, al ejecutarlo, se genera el ciclo de repetición con el objeto **UNTIL BREAK** el cual activa el botón de confirmación **OK**, este se presiona solo hasta que se ha seleccionado la función que se desea generar, de cualquier manera se tiene una opción por omisión.

Para seleccionar las funciones que aparecen en el objeto **RADIO BUTTONS** es necesario que se seleccione primero la opción de función de USUARIO en el objeto denominado **SEÑALES** ya que el manual de usuario del HP33120A hace la indicación de que para seleccionar las funciones **SINC**, **NEG_RAMP**, **EXP_RISE**, **EXP_FALL**, **CARDIAC**, que son las funciones de usuario, es necesario primero seleccionar o fijar la opción de función de usuario (**USER FUNCTION**), la cual esta dentro de las funciones principales[14].

[14]Las funciones principales son aquellas que se pueden seleccionar de manera directa en el panel frontal del instrumento sin tener que accesar a ningun menu

En el panel de usuario del HP33120A es posible modificar las opciones de amplitud, frecuencia, offset, etc. simplemente se presiona el campo de estos parámetros con lo que aparecerá una caja de diálogo en donde se puede modificar el valor actual.
Una vez que se ha seleccionado el tipo de función y como el objeto **DROP-DOWN LIST** donde se eligen las opciones (objeto nombrado **SEÑALES**) en su salida proporciona un número ordinal[15], este se conecta a la entrada del objeto **IF/THEN/ELSE** en el cual se toma la decisión de las funciones (si son de usuario o las principales), si son las funciones principales la salida ordinal del objeto **DROP-DOWN LIST** previamente se ha conectado al objeto **GATE** el cual deja pasar la información de su entrada a su salida solo cuando su terminal de secuencia de entrada ha sido activada, en este caso se activa cuando el objeto **IF/THEN/ELSE** toma la decisión de que se trata de una función principales.
Cuando se selecciona una función de usuario el flujo de información activa al objeto **RADIO BUTTONS** donde se encuentran las opciones de funciones de usuario, cuando estas son activadas, también se activa el objeto de **DIRECT I/O** para el generador de funciones con la sentencia SCPI **FUNC:SHAP USER** la cual le indica al HP33120A que la funciónde salida es del tipo usuario.

La salida del objeto **DROP-DOWN LIST** que no es la ordinal sino la de la opción seleccionada se conecta a la entrada del objeto **FORMULA** donde se construye otra sentecia SCPI en la cual se indica cual de las funciones de usuario se va agenerar, en la casilla del objeto **FORMULA** aparece el texto **FUNC:USER A**, donde A es el nombre de la función de usuario, la salida de este objeto FORMULA se conecta a otro objeto **DIRECT I/O** donde se le indica al instrumento el tipo de función de usuario a generar.

En el panel de usuario el funcionamiento es mas transparente ya que solo se necesita seleccionar la función deseada (sea o no de usuario) y presionar el botón **OK**. El objeto **NOTEPAD** que aparece en el panel de usuario es solo de manera informativa acerca del programa, en la figura 4.56 aparece la función de usuario **SINC** la cual fue capturada con el programa OSCILOSCOPIO.VEE.

[15]El número ordinal es un entero que va desde cero hasta el número máximo de opciones menos uno y este tipo de objetos proporciona ese número en la terminal ordinal de acuerdo a la opción seleccionada

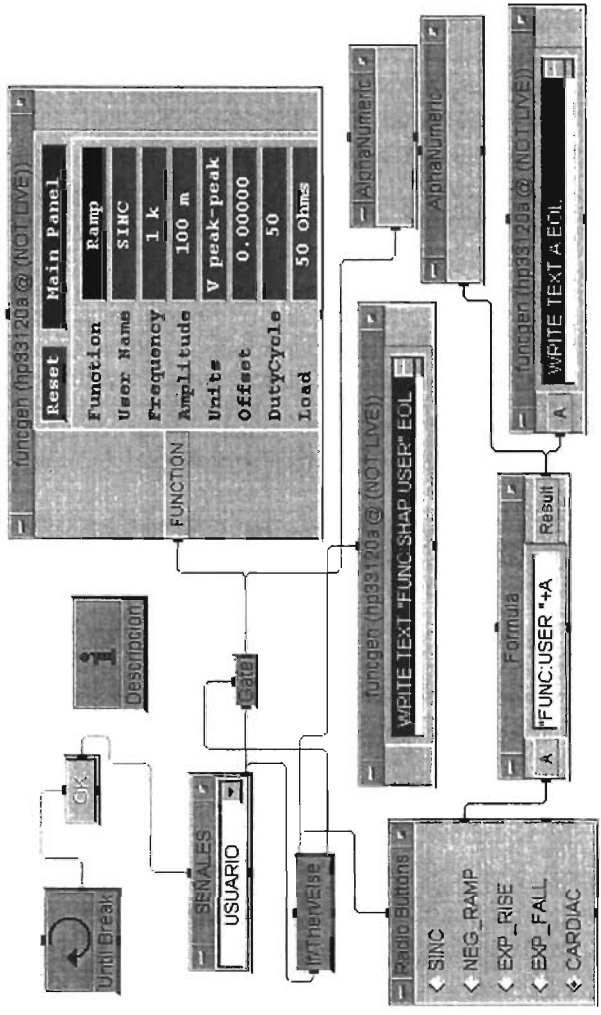


Figura 4.55: Programa de demostración del HP33120A

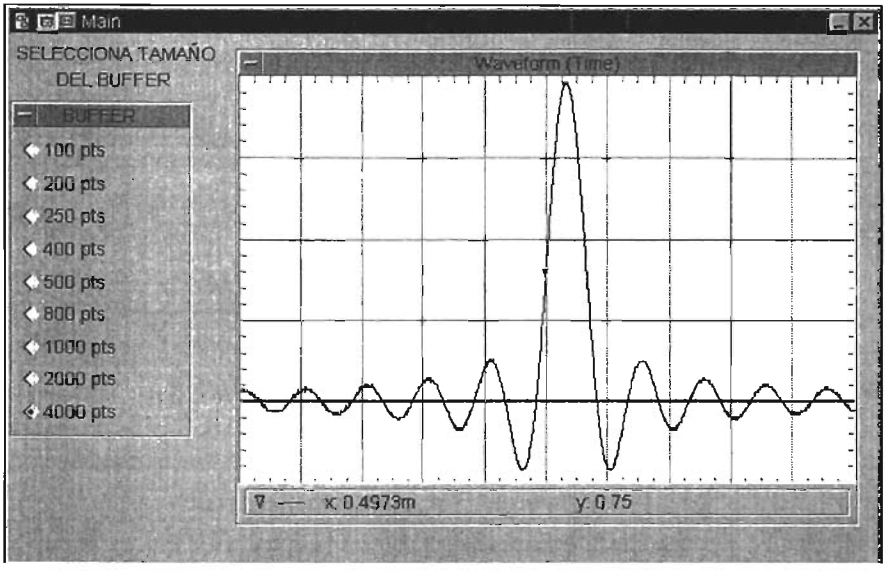


Figura 4.56: Captura de señal SINC con la aplicación OSCILOSCOPIO.VEE.

4.10 Operaciones entre funciones

El HP33120A, como se vió en la aplicación anterior, es un instrumento capaz de generar funciones arbitrarias además de las principales[16], y las denominadas funciones de usuario[17]. Tiene el esquema de conexión de la siguiente figura.

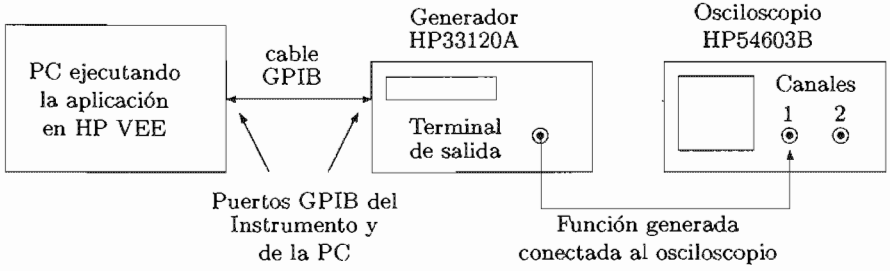


Figura 4.57: Esquema de conexión PC—Generador—Osciloscopio.

El objetivo de esta aplicación (OPERACIONES.VEE) es mostrar la capacidad de este instrumento para generar cualquier forma de onda, en específico, el programa hace las operaciones de suma, resta y multiplicación entre dos señales de tipo principal. Para llevar a cabo tales operaciones el programa carga en memoria una librería de ligado dinámico que fue escrita en lenguaje "C" que construye una cadena de valores (el vector resultado de la operación entre las dos señales) la cual se transfiere al instrumento para poder generar la forma de onda que resulta de hacer la operación.

El panel de usuario se muestra en la figura 4.58 y la implementación en la figura 4.59. Primero hay que seleccionar el tipo de operación a realizar, a continuación se fijan los parámetros de las funciones **A** y **B** (frecuencia, tipo de función, fase y nivel de DC), también se fija el número de puntos de los buffers de las funciones **A** y **B**. Habiendo hecho lo anterior se presiona el botón **OK** y en el desplegador de forma de ondas se visualizará el resultado de la operación seleccionada. En la casilla larga del objeto **ALPHANUMERIC** se muestra la cadena de valores que se escribe al instrumento.

[16] senoidal, triangular, cuadrada, ruido, DC, rampa

[17] sinc, rampa negativa, exponencial creciente, exponencial decreciente, cardiaca

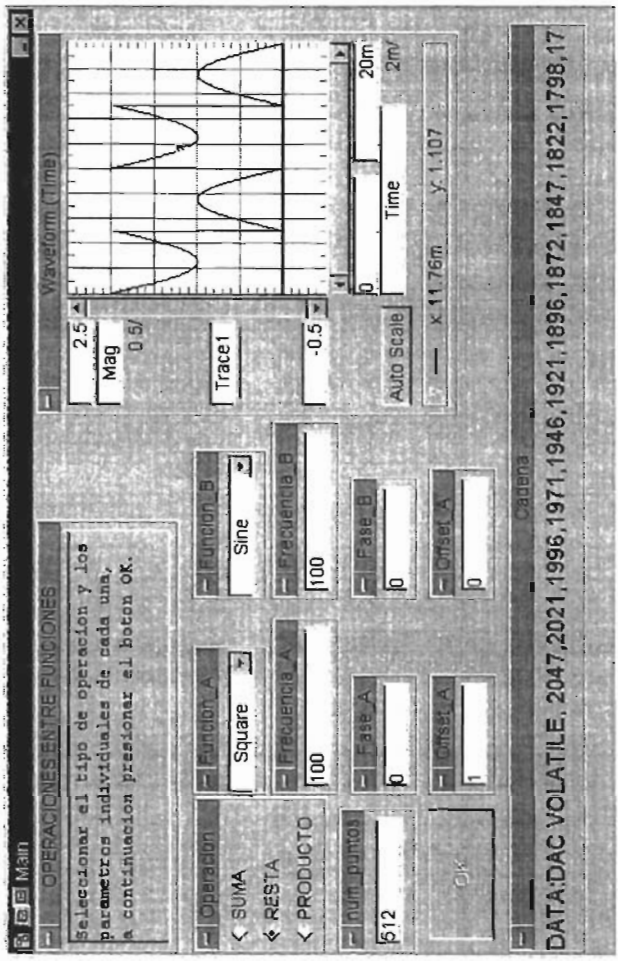


Figura 4.58: Panel de usuario del generador arbitrario de funciones

El programa consta de los siguientes objetos:

- DISPLAY⇒WAVEFORM(Time)
- DEVICE⇒IMPORT LIBRARY
- DEVICE⇒DELETE LIBRARY
- DEVICE⇒CALL,(string.cad_puntos)

- DEVICE⇒DIRECT I/O del HP33120A
- DEVICE⇒VIRTUAL SOURCE⇒FUNCTION GENERATOR, (FUNCION_A, FUNCION_B)
- DEVICE⇒FORMULA (A+B,A-B,AxB)
- FLOW⇒CONFIRM(OK)
- FLOW⇒IF/THEN/ELSE
- FLOW⇒JUNCTION (JCT)
- FLOW⇒REPEAT⇒UNTIL BREAK
- DATA⇒SELECTION CONTROL⇒RADIO BUTTONS (Operacion)
- DATA⇒UNBUILD DATA⇒WAVEFORM
- DATA⇒CONSTANT⇒INTEGER, (num_puntos)
- DATA⇒DIALOG BOX⇒MESSAGE BOX (ERROR_EN_FUNCION_A, ERROR_EN_FUNCION_B, Overflow_Underflow)
- DATA⇒CONSTANT⇒REAL
  (Fase, Frecuencia, Offset para las funciones A y B)
- DATA⇒SELECTION CONTROL⇒DROP-DOWN LIST (Funcion_A, Funcion_B)
- DISPLAY⇒ALPHANUMERIC (cadena)
- DISPLAY⇒NOTE PAD, (OPERACIONES ENTRE FUNCIONES)

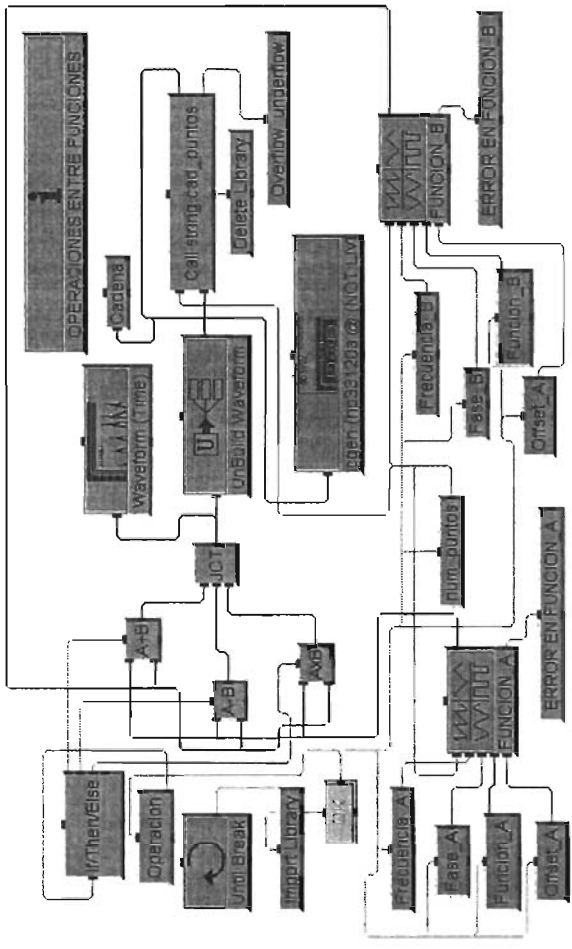


Figura 4.59: Implementación del generador arbitrario de funciones

La librería de ligado dinámico (*build_str_dll.dll*) para la construcción de la cadena de puntos resultado de la operación entre funciones hace lo siguiente:
La función dentro de la librería se llama "cad_puntos" la cual tiene dos argumentos, "num_puntos" y "senal", "senal" es el buffer de puntos resultado de la operación seleccionada y es de tamaño "num_puntos". La función primero reserva suficiente memoria para un arreglo de enteros ("senal_norm") de tamaño "num_puntos" el cual va a contener los valores normalizados y esca-

lados [18] del buffer de entrada "senal", también se reserva suficiente memoria para un arreglo de caracteres ("codigos") el cual va a contener la cadena con los puntos que se va a transferir al instrumento. Hecho lo anterior se busca el elemento mayor, positivo o negativo, del buffer de entreda ("senal"), con este valor se normaliza y se escala seleccionando un valor de 2047, esto se puede hacer con otro valor pero se utiliza este debido a que los códigos del DAC de salida del instrumento van desde -2047 a 2047 para cubrir todo el intervalo de combinaciones del DAC. Después el arreglo de enteros resultante se escribe a un arreglo de caracteres ("codigos") con el comando **DATA:DAC VOLATILE,** para hacer la indicación de que los valores a escribirse son enteros.

A continuación se muestra el contenido del archivo *build_string.cpp* el cual tiene el código fuente de la rutina *cad_puntos* para hacer las operaciones entre las dos funciones **A** y **B** y el archivo de encabezado para HP VEE.

```
#include <stdio.h>
#include <math.h>
#define DLLEXPORT __declspec(dllexport)

extern "C" DLLEXPORT
char *cad_puntos(int num_puntos,double *senal){
char *codigos;
codigos = new char[5*num_puntos+20];
if(!codigos){
    codigos = "err1";
    return(codigos);
}

int *senal_norm;
senal_norm = new int[num_puntos];
if(!senal_norm){
    codigos="err2";
    return(codigos);
}

//encuentra el maximo y el minimo del arreglo
double max = senal[0]; //compara con el primer elemento
```

[18] Con esto me refiero a que el buffer de valores se va a dividir por el elemento mayor y una vez haciendo esto se van a multiplicar por un valor entero el cual se explicara mas adelente.

```
double min = senal[0]; //compara con el primer elemento
int i;
for(i=0;i<num_puntos;i++){
    if(senal[i]>=max){
        max=senal[i];
    }
    if(senal[i]<=min){
        min=senal[i];
    }
}

//normalizacion y escalamiento
int factor;
if(max>0&&min<0) factor = max-min;
if(max>0&&(min<max&&min>0)) factor = max-2*min;
if(max<0&&(min<max&&min<0)) factor = fabs(max-2*min);
if(max==min) factor = max;

for(i=0;i<num_puntos;i++){
senal_norm[i]=(senal[i]/fabs(factor)) * 2047;
if(senal_norm[i]>=2047) senal_norm[i]=2047;
if(senal_norm[i]<=-2047) senal_norm[i]=-2047;
}

//construccion de la cadena
int a,b=0;
char *temp;
a=sprintf(codigos,"DATA:DAC VOLATILE, ");
temp=&codigos[a];
//a temp se le asigna la direccion donde
//acaba la primer cadena
b=b+a; //se inicializa un contador
for(i=0;i<num_puntos-1;i++){
    temp=&codigos[b];
//temp se inicializa con un nuevo valor
    a=sprintf(temp,"\% i,",senal_norm[i]);
//se detecta cuantos caracteres se escribieron
    b=b+a;//se desplaza el contador la cantidad anterior
}
    temp=&codigos[b]; //ultimo numero
```

```
    sprintf(temp,"\% i",senal_norm[num_puntos-1]);
// fin de construccion de la cadena

    delete senal_norm;
    return(codigos);

}//fin funcion
```

Archivo de encabezado *build_string.h* es el siguiente:

```
char *cad_puntos(int num_puntos,double *senal)
```

Para la construcción del programa primero se toma el objeto **RADIO BUTTONS**, se le editan las opciones a elegir, en este caso las operaciones SUMA, RESTA, PRODUCTO. La salida ordinal de este objeto se conecta al objeto **IF/THEN/ELSE** para efectuar la operación adecuada de acuerdo a la selección hecha, en este caso habilitará los objetos fórmula para la operación suma (A+B), resta (A-B), producto (AxB) cuando sean seleccionadas, la salida de los objetos formula de suma, resta, producto se aplican a las 3 entradas del objeto **JCT** el cual ya se ha visto su funcionamiento. Como el resultado de las operación es un tipo de dato denominado forma de onda (Waveform), este se tiene que descomponer en arreglo de tiempo y arreglo de amplitud, el arreglo de amplitud es un arreglo de números en punto flotante el cual se dirige hacia la entrada de la función de "cad_puntos" de la librería que ya fue descrita.

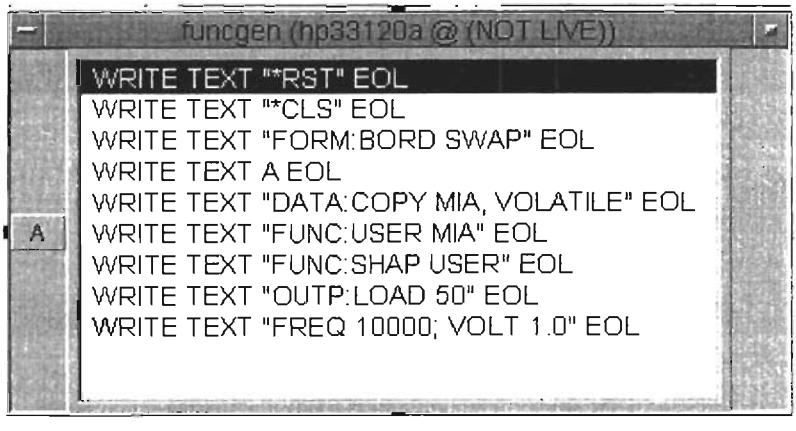


Figura 4.60: Parámetros de inicialización

El objeto **DIRECT I/O** es el encargado de configurar los parámetros de inicio del instrumento, en la figura 4.60 se muestra el contenido de este objeto. En la línea 4 la variable **A** (**DATA:DAC VOLATIL pt1,pt2...**) es la que se le asigna la cadena con los puntos de la forma de onda a transferirse hacia el instrumento, y es la que regresa la función "cad_puntos" de la librería. La línea 1 y 2 hacen un reestablecimiento y limpieza de cualquier error que se haya producido, la línea siguiente intercambia los bytes de los códigos del DAC de salida (el MSB por el LSB). La instrucción **DATA:COPY MIA, VOLATILE**, copia la información transferida por medio de la cadena almacenada en la variable **A** de la línea 4 a la memoria volatil del instrumento y le nombra **MIA** (puede ser cualquier nombre de 8 caracteres alfanuméricos). Después se selecciona la función **MIA** con la sentencia **FUNC:USER MIA** y se hace la indicación de que es una función de usuario (**FUNC:SHAPE USER**). Las últimas dos líneas seleccionan la impedancia de salida (**OUTPUT:LOAD 50**), la frecuencia y la amplitud (**FREQ 10000;VOLT 1.0**).

Para los objetos **FUNCTION GENERATOR**, se les agregan entradas de frecuencia, fase, offset (de tipo **REAL**) y funcion (de tipo **DROP–DOWN LIST**), que son las que se muestran en la figura 4.59, para seleccionar el tipo de función el objeto es del tipo **DROP–DOWN LIST** editado para las formas de onda del instrumento (SINE, COSINE, SQUARE, TRI, +RAMP, -RAMO, DCONLY).

Finalmente para que el programa se vuelva cíclico se agrega el objeto **UNTIL BREAK** junto con el botón de confirmación **OK**, el botón de confirmación forza una pausa del programa hasta que se presiona, en este caso la salida del botón de confirmación se conecta a las secuencias de entrada de los parámetros de frecuencia, num_puntos, fase, función y nivel de DC que a su vez van conectados a los generadores de funciones virtuales. Una vez que se termina la ejecución de la aplicación se activa el objeto **DELETE LIBRARY** para descargar de la memoria la librería de ligado dinámico *build_str_dll.dll*.

Para cierta combinación de los parámetros de frecuencia/núm_de_puntos se puede generar un error a lo que surgirán cajas de diálogo (ERROR_EN-_FUNCION_A, ERROR_EN_FUNCION_B) para corregir la situación, también debido a esto se puede generar un error en la librería con lo que también aparecerá una caja de diálogo (Overflow_Underflow) alertando el error.

El objeto **NOTE PAD** (OPERACIONES ENTRE FUNCIONES) es de tipo informativo para hacer la descripción de la aplicación. En la figura 4.61 aparece la señal resultante de hacer la operación sustracción entre una señal cuadrada de 1 V_{PP} y una senoidal de 0.5 V_{PP}, ambas de 512 puntos, esta figura fue capturada con la aplicación OSCILOSCOPIO.VEE.

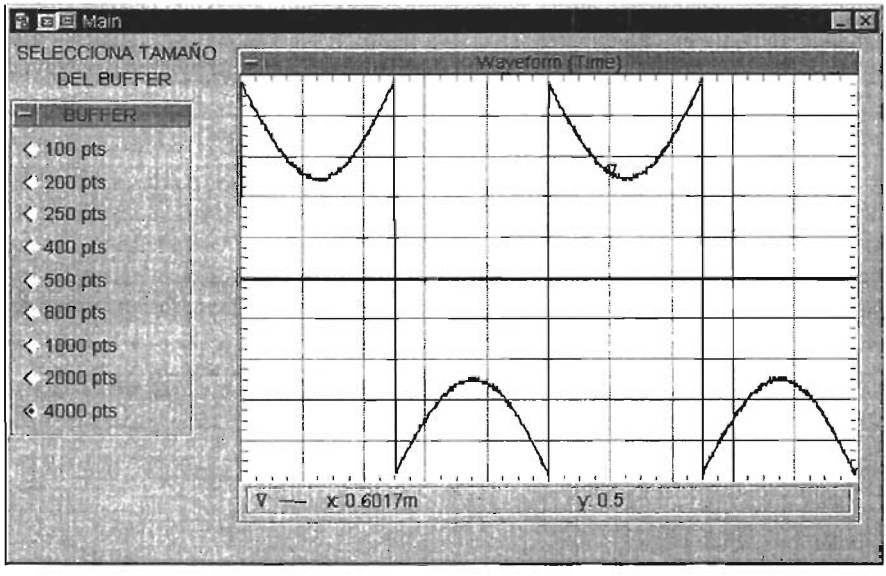


Figura 4.61: Resta de dos señales cuadrada menos senoidal.

4.11 Puerto serie

El nombre de esta aplicación es PUERTO_SERIE.VEE y su objetivo es utilizar el puerto serie de la PC para manejar un instrumento programable que disponga de un puerto serie y que en su electrónica tenga el soporte para comandos SCPI, en este caso el instrumento es el generador de señales HP33120A. El manejo del instrumento se lleva a cabo mediante el uso del objeto **Direct I/O** para el puerto serie y que utiliza transacciones para enviar comandos y datos. La interfase serie es una opción alterna a la interfase GPIB para el manejo del instrumento[19], a continuación se muestra el esquema de conexión PC–Instrumento.

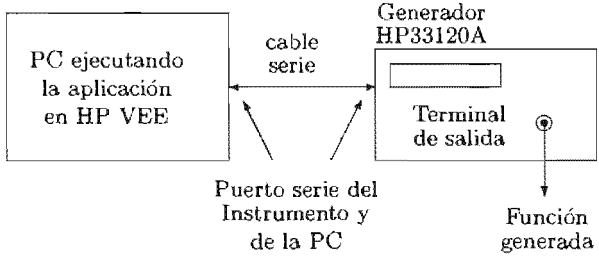


Figura 4.62: Esquema de conexión PC–Instrumento (HP33120A).

Para utilizar el puerto serie de la computadora primero es necesario ejecutar la utilería **I_O Config** (figura 4.63) de la tarjeta GPIB para que agregue la interfase serie a la lista de hardware disponible para comunicación. En esta utilería se selecciona la interfase RS-232 en la parte de "tipo de interfases disponibles" y después se fijan los parámetros del puerto serie como en la figura 4.64 que deben de ser los mismos que tiene el instrumento. En el instrumento se debe de seleccionar el tipo de interfase a utilizar y esta debe de ser RS-232 (por omisión tiene seleccionada la interfase GPIB), a esta configuración se accesa por medio del panel frontal del instrumento con el procedimiento que se enumera en la siguiente página:

[19]Generalmente el objeto **Direct I/O** y el panel driver de la interfase GPIB ofrecen la flexibilidad suficiente, pero el puerto serie se puede usar en el caso de que no se cuente con la tarjeta GPIB y su cable.

1. Presionar el botón SHIFT (de color azul) y presionar el botón ENTER (para seleccionar el menú del instrumento)

2. Presionar el botón > 4 veces para estar en el menu de entrada/salida (INPUT/OUTPUT MENU).

3. Una vez sobre en el menú de entrada/salida presionar el botón ∨ una vez con lo que pasaremos al parámetro de la dirección HPIB (HPIB ADDR).

4. Presionar el botón > una vez con lo cual nos posicionaremos en el menú de la interfase a seleccionar (INTERFACE).

5. Presionar el botón ∨ dos veces con lo cual se seleccionará la interfase RS-232.

6. Presionar el botón ENTER para que los cambios queden guardados en el instrumento.

El instrumento utiliza el puerto serie con control por hardware por lo que el cable de comunicación debe ser armado como se indica en la tabla 4.3. En las figuras 4.63 y 4.66 se muestra la utilería **I/O config** y la ruta de acceso en el ambiente de Windows.

Una vez que los parámetros de comunicación han sido fijados tanto en el instrumento como en la PC y el cable de comunicación está armado correctamente se presiona el botón START y de manera automática se envian los comandos SCPI que están en el objeto **Direct I/O** de la interfase serie, los comandos SCPI se envían de igual manera que con el objeto **Direct I/O** de la interfase GPIB.

Para el programa de la figura 4.65 se utilizaron los siguientes objetos:

- FLOW⇒START
- I/O⇒INTRUMENT MANAGER⇒DIRECT I/O de la interfase serie

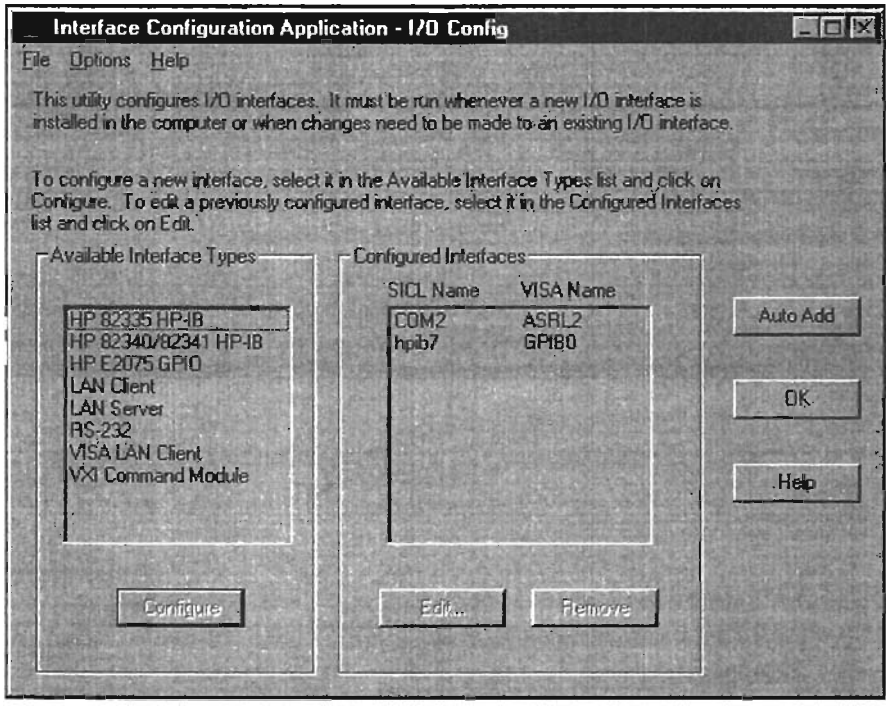


Figura 4.63: Utilería **IO Config** para agregar la interfase serie.

Los comandos SCPI que se envian al instrumento son los siguientes. En este caso se utilizó el HP33120A (generador de funciones arbitrarias).

Sentencia SCPI	descripción
"*RST"	Reestablecimiento del instrumento a un estado inicial
"*CLS"	Limpieza del cualquier mensaje, incluidos los de error
"SYSTEM:REMOTE"	Pone al instrumento en modo remoto, los botones de su panel frontal se deshabilitan
"APPL:SQU 1000,1,0"	Genera una señal cuadrada de 1kHz, 1 V_{pp} y cero offset
"SYSTEM:LOCAL"	Pone al instrumento en modo local, los botones de su panel frontal se habilitan

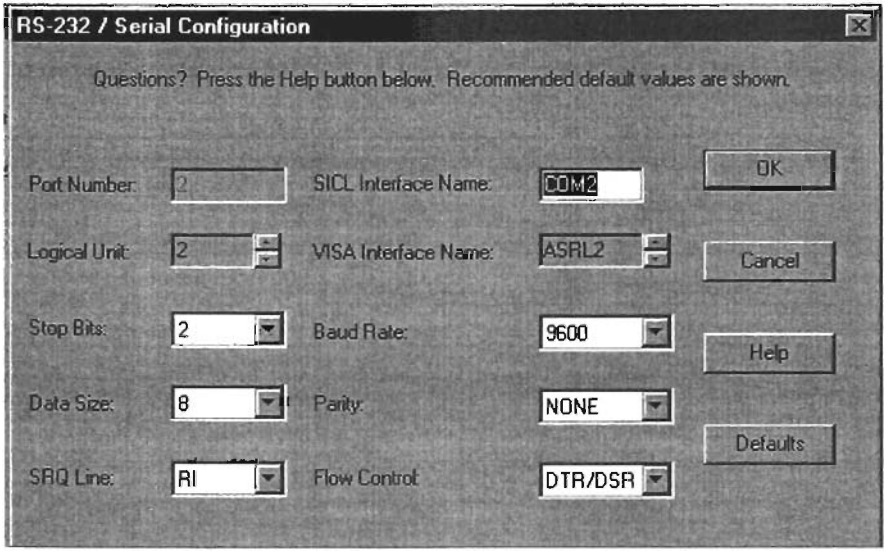


Figura 4.64: Parámetros de comunicación del puerto serie.

Las dos primeras líneas hacen un reestablecimiento del instrumento a un estado inicial, la tercer línea pone el instrumento en modo remoto para que pueda recibir otros comandos, la cuarta línea indica la generación de una señal cuadrada de 1 kHz, 1 volt de amplitud pico a pico y un offset de cero volts. La última línea pone al instrumento nuevamente en modo local, en este modo el usuario puede modificar las señales a generar mediante el panel frontal del instrumento lo que no es posible cuando el instrumento se encuentra en modo remoto.

	DB9 macho (PC)		DB25 hembra (PC)	DB9 (Inst. macho)	
1	DCD	8	DCD	DCD	1
3	TX	2	TX	RX	2
2	RX	3	RX	TX	3
6	DSR	6	DSR	DTR	4
5	GND	7	GND	GND	5
4	DTR	20	DTR	DSR	6
8	CTS	5	CTS	RTS	7
7	RTS	4	RTS	CTS	8
9	RI			RI	9

Tabla 4.3: Correspondencia entre pines del puerto serie de la PC y el Instrumento

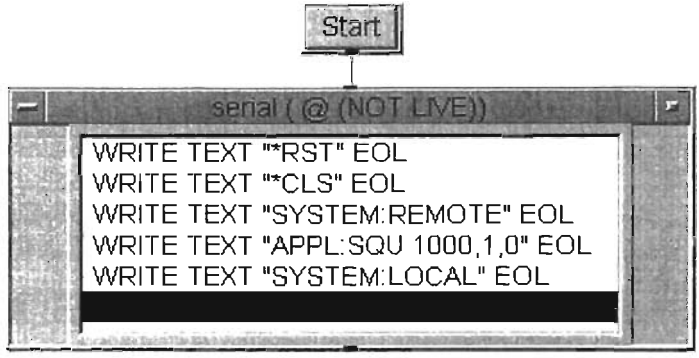


Figura 4.65: Programa para manejar el instrumento mediante el puerto serie.

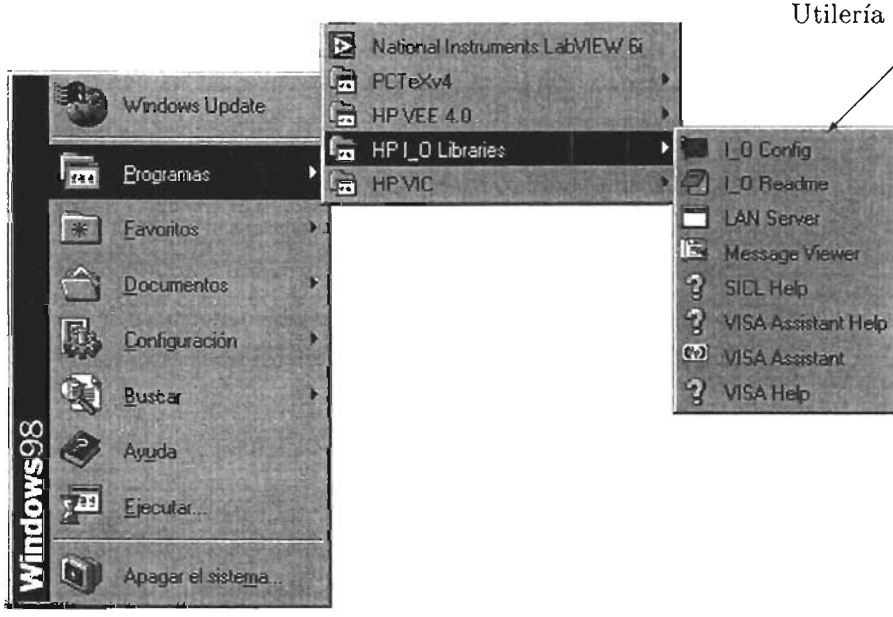


Figura 4.66: Ruta de la utilería **I_O config** para agregar la interfase serie.

Conclusiones

La importancia de la instrumentación virtual ha llevado a mejorar los sistemas de adquisición de datos. El software y hardware son partes importantes de estos sistemas ya que con ambos es posible hacer automáticos los procesos de medición repetitivos que requieren poca o nula supervisión.

Como ya se ha tratado, la herramienta de trabajo para hacer el proceso de adquisición de datos es HP VEE la cual esta enfocada al manejo de equipo que en su hardware incluya soporte para la interfase GPIB y los comandos SCPI. Además, es una herramienta de programación gráfica bastante intuitiva y fácil de utilizar que permite el desarrollo de las aplicaciones con rapidez.

Con la herramienta HP VEE se generaron diversas aplicaciones que muestran un panorama amplio de posibilidades en las que se puede utilizar la herramienta de programación, ya sea para hacer el manejo de instrumentos con la interfase GPIB, adquisición de datos o simulación en procesamiento digital de señales. Quiero mencionar que estas aplicaciones se presentan como una plantilla para que el usuario desarrolle las propias y las adapte a sus necesidades y su propio estilo de programación y el mismo se dará cuenta que esta herramienta le simplifica varias tareas de programación en otro tipo de lenguaje sea o no gráfico.

HP VEE en este trabajo esta orientado a manejar instrumentos principalmente con la interfase GPIB, también se realizó el manejo de un instrumento que cuenta con puerto serie y tiene en su eléctronica el soporte para los comandos SCPI. Aún cuando la interfase GPIB es ampliamente difundida en gran cantidad de equipos de distintas marcas, su uso esta enfocado principalmente a equipo de laboratorio o bancos de pruebas debido a las tareas que se realizan con ella.

Como se ha visto, el sistema de instrumentación basado en PC resulta útil, sobre todo cuando se requiere hacer pruebas en laboratorio, especificamente el caso de estar tomando mediciones (adquisición de datos en un experimento o banco de pruebas de un laboratorio), esta característica será aprovechada para mejorar el desarrollo de las prácticas del Laboratorio de Medición e Instrumentación ya que se cuenta con la tecnología que permite mejorar el proceso de toma de mediciones mediante los equipos HP34401A (multímetro digital), HP33120A (generador de funciones) y el HP54603B (osciloscopio digital), y también presentar esta herramienta como alternativa para aplicaciones a nivel profesional. Por lo anterior el objetivo planteado al inicio de este trabajo se ha cumplido de manera satisfactoria.

GPIB se sigue mejorando continuamente e incorporando a todo tipo de buses e interfases (USB, PCMCIA, PCI, SERIE, EISA), se pueden encontrar adaptadores y conversores para cualquier tipo de requerimiento asícomo tarjetas adquisidoras de datos, de control, interruptores. También se ha integrado cada vez más con los programas de análisis numérico mas populares (por medio de librerías). Adicionalmente se han desarrollado dispositivos que permiten conectar los instrumentos o dispositivos con interfase GPIB a una LAN[20] y hacerlos remotos, de esta manera el campo de aplicaciones con GPIB se ha ampliado aún mas.

[20]Local Area Network, red de área local.

Apéndice A

Comandos SCPI

El estándar SCPI es un lenguaje de programación orientado al manejo de instrumentos tales como osciloscipios, multímetros digitales, generadores de funciones, fuentes de poder, analizadore de espectro, los cuales lo implementan en su CPU interno. El SCPI esta basado en los estándares IEEE488 y IEEE488.2 solo que a diferencia de estos, SCPI proporciona los comandos necesarios para la operación de un instrumento específico. El consorcio inicial de SCPI incluyó a HP, Tektronix, Fluke, Philips, Waveteck, National Instruments, Racal-Dana, Keithley, Bruel & Kajaer.

Utiliza un modelo general de instrumento programable para mantener compatibilidad y representar su funcionalidad desde el punto de vista del SCPI.

El SCPI define tres tipos de compatibilidad:

- **Vertical**, es cuando dos instrumentos del mismo tipo tienen controles idénticos (intervalos de voltaje, tipos de disparos)
- **Horizontal**, es cuando dos instrumentos pueden hacer la misma medición sin importar la técnica utilizada, ambos instrumentos deben utilizar los mismos comandos para hacer la misma medición (un osciloscopio y un multímetro deben medir la frecuencia de una señal de entrada con el mismo comando)
- **Funcional**, es cuando dos instrumentos realizan la misma función con los mismos comandos (un disparo interno).

Los instrumentos específicos para una medición implementan solo algunos de los bloques de la figura A.1, como en el caso de un multímetro HP34401A este solo implementa los bloques 1,2,3,4,5 ya que no genera ninguna señal, el generador de señales implementa los bloques 4,5,6,7,8.

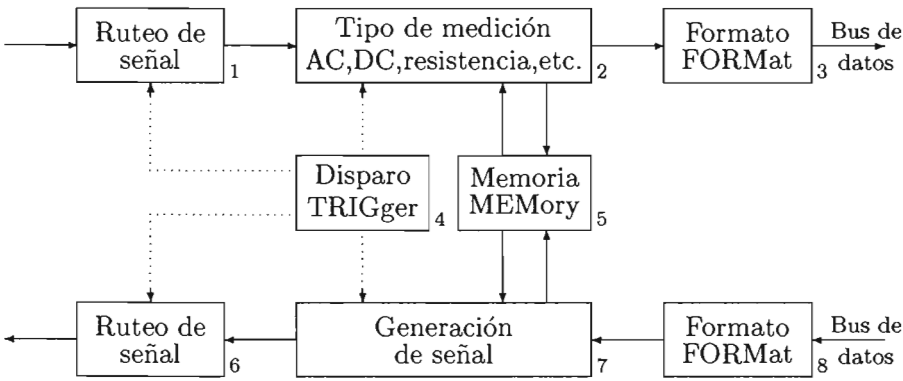


Figura A.1: Modelo de un instrumento programable

El propósito del bloque de ruteo de señal es controlar la ruta de las señales entre un puerto de señal del instrumento y asociarla al tipo de función específica a realizar tal como hacer la medición de corriente, voltaje, resistencia, frecuencia, etc.

El SCPI organiza los comandos en distintos conjuntos llamados *subsistemas*, los comandos para cada subsistema están definidos en una estructura jerárquica similar a la de un sistema de archivos y se le llama *arbol de comandos*, la figura A.2 muestra el arbol parcial del comando `MEASure` para el multímetro digital HP34401A.

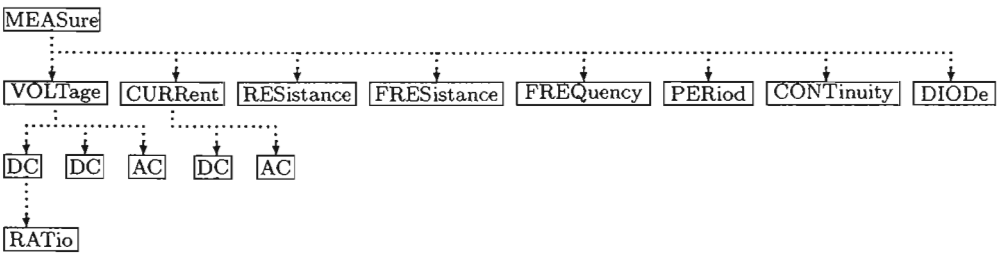


Figura A.2: Arbol parcial del comando MEASure

Para todas las terminales del arbol `MEASure` se tienen parámetros adicionales los cuales son intervalo[1] y resolución dentro de los cuales se puede seleccionar mínimo (MIN), máximo (MAX), por omisión (DEF) o numérico (`MEASure:VOLTaje:DC? 10,0.01`, medir voltaje de DC en el intervalo de 10 volts y con una resolución de 0.01 volts).

Para el caso de la medición del diodo y la prueba de continuidad el intervalo de disparo se configura solamente desde el panel frontal del instrumento, para el diodo se inyecta una corriente de prueba con lo que se emitirá un zumbido (si esta habilitado) si el diodo esta dentro del umbral de 0.3 volts a 0.8 volts con un intervalo de 1 volt y resolución de 100 μ v) y para la continuidad se configura un valor de resistencia para la cual también se emitirá un zumbido (si esta habilitado), el umbral de sonido esta desde 1 Ω hasta 1000 Ω y tiene un intervalo de 1 $k\Omega$ y una resolución de 0.1 Ω. Para ambos casos la corriente de prueba es de 1 mA.

En la figura A.2 las cadenas de caracteres están en mayúsculas y minúsculas, para el SCPI la parte que esta en mayúsculas es una abreviación del comando completo por lo que es indistinto la manera en como se escriba.

Como se ha podido notar, el SCPI es un lenguaje de programación basado en cadenas de caracteres las cuales son decodificadas e interpretadas para que realicen la función específica en el instrumento. En la sección 3.4.1 en la página 61 se mencionan las formas de manejar instrumentos, las mas sencilla es utilizar el panel driver.

La siguientes convenciones son utilizadas en este apéndice:

- Paréntesis cuadrados ([ ]) indican parámetro opcional.
- Corchetes ({ }) agrupan parámetros dentro de un comando.
- Pico–paréntesis (< >) indican que se debe de incluir el valor numérico del parámetro.
- La barra vertical | indica que se tienen múltiples opciones de parámetro.
- Los comandos cuyo parámetro finalice con un signo de interrogación "?" permiten que se recupere el valor de ese parámetro.
- En los comandos SCPI las letras minúsculas son opcionales dentro de la definición del comando.

[1] también conocido como rango

Referencia a comandos SCPI para el HP34401A (multímetro digital)

El comando MEASure proporciona la ruta mas facil de disparar una medición, por ejemplo, ''MEAS:VOLT:DC? 1, 0.001'', mide el voltaje de DC de las terminales de entrada frontales del multímetro en un intervalo de 1 volt y una resolución de 0.001.

```
MEASure
 :VOLTage:DC? {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
 :VOLTage:DC:RATio?  {<rango>|MIN|MAX|DEF},
                                    {<resolución>|MIN|MAX|DEF}
 :VOLTage:AC? {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
 :CURRent:DC? {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
 :CURRent:AC? {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
 :RESsistance?  {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
 :FRESsistance?  {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
 :FREQuency?  {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
 :PERiod?  {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
 :CONTinuity?
 :DIODe?
```

El comando CONFigure proporciona mas flexibilidad, sin embargo no dispara la medición de manera automática.
''CONF:VOLT:DC 1, 0.001''
''TRIG:SOUR IMM'', la medición se dispara de manera inmediata
''READ?'' , la lectura se transfiere al buffer de salida
Otro método de recuperar la lectura es con los siguientes comandos:
''CONF:VOLT:DC 1, 0.001''
''TRIG:SOUR IMM'', la medición se dispara de manera inmediata
''INIT'' ,la lectura se transfiere a la memoria interna del multímetro
''FETC?'',transfiere la lectura de la memoria interna del multímetro al buffer de salida

CONFigure
:VOLTage:DC {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
:VOLTage:DC:RATio {<rango>|MIN|MAX|DEF},
{<resolución>|MIN|MAX|DEF}
:VOLTage:AC {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
:CURRent:DC {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
:CURRent:AC {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
:RESsistance {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
:FRESsistance {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
:FREQuency {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
:PERiod {<rango>|MIN|MAX|DEF},{<resolución>|MIN|MAX|DEF}
:CONTinuity
:DIODe

CONFigure?

El comando SENSe funciona de manera similar al comando MEASure.

```
[SENSe:]
   FUNCTion ''VOLTage:DC''
   FUNCTion ''VOLTage:DC:RATio''
   FUNCTion ''VOLTage:AC''
   FUNCTion ''CURRent:DC''
   FUNCTion ''CURRent:AC''
   FUNCTion ''RESsistance''
   FUNCTion ''FRESsistance''
   FUNCTion ''FREQuency''
   FUNCTion ''PERiod''
   FUNCTion ''CONTinuity''
   FUNCTion ''DIODe''
   FUNCTion?

[SENSe:]
   VOLTage:DC:RANGe {<rango>|MIN|MAX}
   VOLTage:DC:RANGe? [MIN|MAX]
   VOLTage:AC:RANGe {<rango>|MIN|MAX}
   VOLTage:AC:RANGe? [MIN|MAX]
   CURRent:DC:RANGe {<rango>|MIN|MAX}
   CURRent:DC:RANGe? [MIN|MAX]
   CURRent:AC:RANGe {<rango>|MIN|MAX}
   CURRent:AC:RANGe? [MIN|MAX]
   RESistance:RANGe {<rango>|MIN|MAX}
   RESistance:RANGe? [MIN|MAX]
   FRESistance:RANGe {<rango>|MIN|MAX}
   FRESistance:RANGe? [MIN|MAX]
   FREQuency:VOLTaje:RANGe {<rango>|MIN|MAX}
   FREQuency:VOLTaje:RANGe? [MIN|MAX]
   PERiod:VOLTaje:RANGe {<rango>|MIN|MAX}
   PERiod:VOLTaje:RANGe? [MIN|MAX]
```

```
[SENSe:]
  VOLTage:DC:RANGe:AUTO {OFF|ON}
  VOLTage:DC:RANGe:AUTO?
  VOLTage:AC:RANGe:AUTO {OFF|ON}
  VOLTage:AC:RANGe:AUTO?
  CURRent:DC:RANGe:AUTO {OFF|ON}
  CURRent:DC:RANGe:AUTO?
  CURRent:AC:RANGe:AUTO {OFF|ON}
  CURRent:AC:RANGe:AUTO?
  RESistance:RANGe:AUTO {OFF|ON}
  RESistance:RANGe:AUTO?
  FRESistance:RANGe:AUTO {OFF|ON}
  FRESistance:RANGe:AUTO?
  FREQuency:VOLTaje:RANGe:AUTO {OFF|ON}
  FREQuency:VOLTaje:RANGe:AUTO?
  PERiod:VOLTaje:RANGe:AUTO {OFF|ON}
  PERiod:VOLTaje:RANGe:AUTO?

[SENSe:]
  VOLTage:DC:RESolution {<resolución>|MIN|ON}
  VOLTage:DC:RESolution?  [MIN|ON]
  VOLTage:AC:RESolution {<resolución>|MIN|MAX}
  VOLTage:AC:RESolution?  [MIN|ON]
  CURRent:DC:RESolution {<resolución>|MIN|MAX}
  CURRent:DC:RESolution?  [MIN|ON]
  CURRent:AC:RESolution {<resolución>|MIN|MAX}
  CURRent:AC:RESolution?  [MIN|ON]
  RESistance:RESolution {<resolución>|MIN|MAX}
  RESistance:RESolution?  [MIN|ON]
  FRESistance:RESolution {<resolución>|MIN|MAX}
  FRESistance:RESolution?  [MIN|ON]

  FETCh?
  READ?
  INITiate
  TRIGer:SOURce {BUS|IMMediate|EXTernal}
  TRIGer:SOURce?
```

Referencia a comandos SCPI para el HP33120A (generador de señales)

El comando APPLy proporciona el nivel mas facil de programar el generador de funciones a traves de la interfase remota, por ejemplo ''APPLy:SQU 100 KHZ, 1 VPP, 0 V'', genera una señal cuadrada con 1 volt de amplitud pico a pico, 1 kilohertz de frecuencia y cero volts de offset.

```
APPLy:SINusoid [<frecuencia> [,<amplitud>[,<offset>] ]]
APPLy:SQUare [<frecuencia> [,<amplitud>[,<offset>] ]]
APPLy:TRIangle [<frecuencia> [,<amplitud>[,<offset>] ]]
APPLy:RAMP [<frecuencia> [,<amplitud>[,<offset>] ]]
APPLy:NOISe [<frecuencia>|DEFalut [,<amplitud>[,<offset>] ]]
APPLy:DC [<frecuencia>|DEFalut
                        [,<amplitud>|DEFalut[,<offset>] ]]
APPLy:USER [<frecuencia> [,<amplitud>[,<offset>] ]]
APPLy?
```

Los siguientes comandos tienen un nivel de manejo a mas bajo nivel del instrumento ya que proporcionan mas flexibilidad para definir los parámetros de la forma de onda a generar.

```
''FUNC:SHAP SQU''
''FREQ 100 KHZ''
''VOLT 1 VPP''
''VOLT:OFFS 10 V''
```

y genera la misma forma de onda que en ejemplo visto en el comando APPLy.

```
[SOURce:]
  FUNCtion:SHAPe {SINusoid|SQUare|TRIangle|RAMP|NOISe|DC|USER}
  FUNCtion:SHAPe?

[SOURce:]
  FREQuency {<frecuencia>|MINimum|MAXimum}
  FREQuency?  [MINimum|MAXimum]
```

```
[SOURce:]
  VOLTage {<amplitud>|MINimum|MAXimum}
  VOLTage?  [MINimum|MAXimum]
  VOLTage:OFFSet {<offset>|MINimum|MAXimum}
  VOLTage:OFFSet?  [MINimum|MAXimum]
  VOLTage:UNIT {VPP|VRMS|DBM|DEFault}
  VOLTage:UNIT?
```

Comandos para formas de onda arbitrarias

```
[SOURce:]
  FUNCtion {<arb_name>|VOLATILE}
  FUNCtion?
  FUNCtion:SHAPe USER
  FUNCtion:SHAPe?
```

arb_name especifica a una de la 5 formas de onda que trae el instrumento en su firmware ó el nombre de una forma de onda definida por el usuario.

`DATA:DAC VOLATILE {<bloque_binario>|<valor1>,...,<valorn>}`

Se especifican los valores enteros a escribirse en la memoria volatil del instrumento.

`DATA:COPY <nombre_usuario> [,VOLATILE]`

Copia los datos enteros del comando **DATA:DAC VOLATILE** a la memoria volatil del instrumento con el nombre **nombre_usuario**.

`DATA:DELETE <arb_name>`

Borra la forma de onda **arb_name** definida por el usuario.

Comandos para la interfase RS232

```
SYSTem:LOCal
SYSTem:REMote
```

Pone al instrumento en forma remota (se bloquea el panel frontal del instrumento) y en forma local (se habilita nuevamente el panel frontal del instrumento).

Apéndice B

Hoja Técnica

November 2000

National Semiconductor

LM35
Precision Centigrade Temperature Sensors

General Description

The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55 to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM35's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 µA from its supply, it has very low self-heating, less than 0.1°C in still air. The LM35 is rated to operate over a −55° to +150°C temperature range, while the LM35C is rated for a −40° to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO-46 transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35D is also available in an 8-lead surface mount small outline package and a plastic TO-220 package.

Features

- Calibrated directly in ° Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5°C accuracy guaranteeable (at +25°C)
- Rated for full −55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 µA current drain
- Low self-heating, 0.08°C in still air
- Nonlinearity only ±¼°C typical
- Low impedance output, 0.1 Ω for 1 mA load

Typical Applications

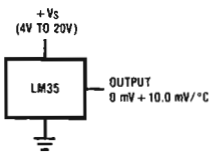
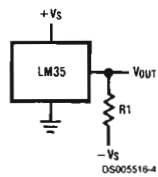


FIGURE 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)

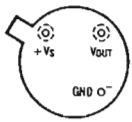


Choose $R_1 = -V_S/50\ \mu A$
V_{OUT} = +1,500 mV at +150°C
= +250 mV at +25°C
= −550 mV at −55°C

FIGURE 2. Full-Range Centigrade Temperature Sensor

Connection Diagrams

TO-46
Metal Can Package*

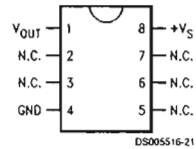


BOTTOM VIEW
DS005516-1

*Case is connected to negative pin (GND)

Order Number LM35H, LM35AH, LM35CH, LM35CAH or LM35DH
See NS Package Number H03H

SO-8
Small Outline Molded Package

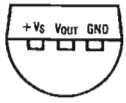


DS005516-21

N.C. = No Connection

Top View
Order Number LM35DM
See NS Package Number M08A

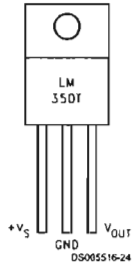
TO-92
Plastic Package



BOTTOM VIEW
DS005516-2

Order Number LM35CZ, LM35CAZ or LM35DZ
See NS Package Number Z03A

TO-220
Plastic Package*



DS005516-24

*Tab is connected to the negative pin (GND).
Note: The LM35DT pinout is different than the discontinued LM35DP.

Order Number LM35DT
See NS Package Number TA03F

Bibliografía

[1] http://ftp.agilent.com/pub/mpusup/pc/vee/vwv_toc.html.

[2] http://www.educatorscorner.com/research/index.shtml.

[3] http://ftp.agilent.com/pub/mpusup/pc/iop/hpibtut/ib0_toc.html.

[4] http://www.scpiconsortium.org.

[5] http://www.quartzdyne.com/software/FreqManual.pdf.

[6] *Control de procesos, curso de instrumentación.* Servicios Industriales Peñoles S.A. de C. V., Minera Rey de Plata, Julio 2000.

[7] Ramón Pallás Areny. *Sensores y Acondicionadores de Señal.* Marocombo, 1810. TA165 P35.

[8] Mariño Espiñeira y Alfonso Lago Ferreiro Enrique Mandano Pérez. *Instrumentación Electrónica.* Alfaomega Grupo editor S.A. de C.V., 1810.

[9] Luces M. Faulkenberry. *Introducción a los Amplificadores Operacionales "Con aplicaciones a CI Lineales".* LIMUSA, 1992. TK7871.5806 F3818.

[10] Ricardo Garibay Jiménez y J. Manuel Gomez Gonzalez Gloria Mata Hernández. *Manual de Prácticas de Transductores y Convertidores Eléctricos.* Departamento de Publicaciones de la Facultad de Ingenieria, Ciudad Universitaria, Mexico D.F., 1993.

[11] Robert Helsel. *Visual Programming with HP VEE.* Prentice Hall PTR, Upper Saddle River, New Jersey 07458, 1997.

[12] Texas Instruments. *Intelligent Opto Sensor.* Texas Instruments, 1996.

[13] T. P. Morrison. *The Art Of Computarized Measurement.* Oxford Science Publications, 1997. TK7878.4 M67.

[14] Harry N. Norton. *Sensores y Analizadores.* Coleción Electrónica Informática. Editorial Gustavo Gili S.A. de C.V., 1984. TA165 N6618.

[15] Hewlett Packard. *Multímetro HP34401A, Guía del Usuario.* Hewlett Packard, 1992.

[16] Hewlett Packard. *HP33120A Function Generator/Arbitrary Waveform Generator, User's Guide.* Hewlett Packard, 1994.

[17] Larry D. Jones y A. Foster Chin. *Electronic Instruments and Measurements.* Prentice Hall, 1991.

[18] Howard M. Berlin y Frank C. Getz, Jr. *Principles of electronic Instrumentation and Measurement.* Macmillan Publishing company, 1988.

[19] David Buchla y Wayne Mclachlan. *Aplied Electronic Instrumentation and Measurement.* Macmillan Publishing Company, 1992. TK7878.4 B79.