



UNIVERSIDAD AUTÓNOMA DEL ESTADO DE HIDALGO

INSTITUTO DE CIENCIAS BÁSICAS

E INGENIERÍA

**"APLICACIONES DIDÁCTICAS DEL GPIB A NIVEL
LICENCIATURA Y POSTGRADO PARA LA UAEH"**

T E S I S

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

P R E S E N T A

MAGALY SIERRA VITE

ASESOR: M.en.C ELÍAS VÁRELA PAZ

PACHUCA DE SOTO HIDALGO, MARZO DE 2007

AGREDECIMIENTO

El presente trabajo está dedicado:

- > A mi hija por ser mi motivación para seguir estudiando.
- > A mis padres que me apoyaron en todos los aspectos para concluir satisfactoriamente mis estudios.
- > A Isaac por el apoyo incondicional que me ha brindado.
- > A mis hermanas por su apoyo moral.
- > Al M en C. Elias Várela Paz por asesorarme en este trabajo.
- > A los catedráticos por su amistad y todas sus enseñanzas.

Atte. Ing. Magaly Sierra Vite

OBJETIVO GENERAL

Conocer las ventajas que tiene el uso del bus GPIB para el Instituto de Ciencias Básicas e Ingeniería, orientado a resolver el problema de conectividad en instrumentación y proponer una serie de prácticas de laboratorio aplicables a algunas carreras de nivel ingeniería y postgrado, especialmente en aquellos casos en que se ocupan instrumentos de medición como voltímetros, amperímetros, osciloscopio, etc.

Las carreras en que se imparten prácticas de laboratorio con estas características son, entre otras:

Nivel licenciatura:

- > Ingeniería en electrónica y telecomunicaciones.
- > Ingeniería industrial.
- > Licenciatura en sistemas computacionales.
- > Física y tecnología avanza.

Nivel postgrado:

- > Ciencias en sistemas computacionales.
- > Ciencias en administración y control.
- > Ciencias en ingeniería de manufactura.
- > Ciencias en ingeniería industrial.

OBJETIVOS ESPECÍFICOS

- > Proporcionar al estudiante la información para el uso del bus GPIB ya existente en el Centro de Investigaciones Avanzadas de Ingeniería Industrial, ubicado en el laboratorio de Hidráulica.
- > Plantear las ventajas relacionadas con el uso de esta herramienta y difundir su conocimiento.
- > Fomentar la inclusión de este equipo en las licenciaturas y postgrados en nuestra casa de estudios.
- > Minimizar, con el empleo de este equipo, el error de lectura humana muy frecuente en las prácticas de laboratorio.
- > Aprovechar la ventaja que nos permite este equipo, de manejar en tiempo real la respuesta de los dispositivos a emplear.

JUSTIFICACIÓN

La realización de este trabajo atiende a la falta y necesidad de acervo que nos permita el conocimiento y aplicación sobre el equipo existente en el Centro de Investigaciones Avanzadas de Ingeniería Industrial, dentro de nuestra máxima casa de estudios; para el mejoramiento y calidad de nuestras prácticas con el fin de una buena proyección profesional y así obtener un nivel y rendimiento más competitivo en el ámbito laboral. Se puede implementar en las siguientes licenciaturas y postgrados:

Nivel licenciatura:

- > Ingeniería en electrónica y telecomunicaciones.
- > Ingeniería industrial.
- > Licenciatura en sistemas computacionales.
- > Física y tecnología avanza.

Nivel postgrado:

- > Ciencias en sistemas computacionales.
- > Ciencias en administración y control.
- > Ciencias en ingeniería de manufactura.
- > Ciencias en ingeniería industrial.

La implementación de GPIB es para mejorar el funcionamiento y organización de las conexiones de nuestros dispositivos del laboratorio de investigaciones.

Índice General

Agradecimiento	III
Objetivo General	V
Objetivos Específicos	VII
Justificación	IX
Índice de Figuras	XV
Índice de Tablas	XVII
Índice de Fórmulas	XVII
Introducción	XIX
Capítulo 1. Antecedentes del bus GPIB	1
1.1 Antecedentes	1
1.2 Características eléctricas del GPIB	3
1.3 Hardware disponible para sistemas GPIB	4
1.3.1 Controladores GPIB (IEEE 488)	6
1.3.2 Tipos de interfaces	10
1.3.3 Controladores vs. Convertidores	18
Capítulo 2. Protocolos de GPIB	19
2.1 Introducción	19
2.2 La norma IEEE 488.1	19
2.2.1 Mensajes remotos. Mensajes dependientes del interfase	21
2.3 La norma IEEE 488.2	22
2.3.1 Registros de estado	23
2.4 Protocolo básico	24
2.5 Protocolos de excepción	26
2.6 Estatus de un equipo protocolo IEEE 488.2	28
2.6.1 Modelo operativo de los equipos (IEEE 488.2) protocolos	31
2.7 Protocolo de transferencia elemental del bus GPIB IEEE-488	33

Capítulo 3. Componentes que se utilizan en el control de la instrumentación por GPIB	35
3.1 Tarjeta Hardware de conexión al bus GPIB	35
3.2 Bus GPIB Nivel Físico(IEEE 488.1) Estructura del bus GPIB	36
3.2.1 Tipos de mensajes que intercambian los equipos	38
3.2.2 Modos de operación de un equipo	38
3.3 Niveles y señales lógicas en el GPIB	41
3.3.1 Significado de las líneas del bus GPIB	43
Capítulo 4. Comandos	47
4.1 Comandos del bus	47
4.2 Comandos Talk/üsten	48
4.2.1 Mecanismos de finalización de mensajes de datos	49
4.3 Comandos universales	49
4.4 Comandos Addressed (ACG)	50
4.5 Comandos comunes	51
4.6 Comandos SCPI	52
4.7 VISA e IVI	54
4.8 Ejemplos de transferencias de datos por el bus GPIB	56
4.9 Drivers	58
Capítulo 5. Interfaces con el bus GPIB	61
5.1 Componentes de la interfase	61
5.2 Fases de una sesión de comunicación de un instrumento	63
5.2.1 Conexión del objeto al instrumento	69
5.2.2 Comunicación con un instrumento VISA o GPIB	70
5.2.3 Desconexión y destrucción de un GPIB object y VISA object	70
5.3 Software existente para GPIB. Programas propietarios	71
5.3.1 LabWindows/CVI y LabView de National Instruments	71
5.3.2 MatLab con Instrument Control Toolbox	72
5.4 El futuro de la conexión de instrumentos de medida	73
5.4.1 USB: Universal Serial Bus	74
5.4.2 Etherneth	75

5.4.3 Firewire (IEEE 1394)	75
Capítulo 6. Instalación y configuración de la tarjeta Aligent 82350B PCI GPIB Interfase y comunicación con instrumentos	77
6.1 Introducción	77
6.2 Pasos para la instalación	77
Capítulo 7. Como realizar prácticas de laboratorio utilizando la comunicación GPIB	85
7.1 Adquisición de voltaje	85
7.1.1 Pasos para adquirir el voltaje	88
7.2 Adquisición de resistencia	95
7.2.1 Pasos para adquirir resistencia	98
7.3 Adquisición de temperatura	103
7.3.1 RTDs y termopares	103
7.3.2 Pasos para adquirir temperatura	112
7.4 Control de voltaje	122
7.4.1 Pasos para controlar voltaje	126
7.5 Control de velocidad de un motor	134
7.5.1 Tacogenerador	134
7.5.2 Convertidores de frecuencia voltaje	135
7.5.3 Frecuencímetro multifunción	136
7.5.4 Tacómetro	136
7.5.5 Procedimiento y metodología	136
7.5.6 Pasos para controlar velocidad	139
Conclusión	157
Bibliografía/Bibliografía electrónica	159
Anexo	161
Glosario	183

Índice de Figuras

Figura 1.1: Configuración de un sistema GPIB.	2
Figura 1.2: Cable de conexión GPIB.	3
Figura 1.3: Tarjeta GPIB	5
Figura 1.4: Tarjeta controladora.	6
Figura 2.1: Estructura genérica de programación de un sistema GPIB	29
Figura 2.2: Registros en la programación.	29
Figura 2.3: Modelo de intercambio de un equipo.	32
Figura 2.4: Transferencia del bus GPIB.	33
Figura 3.1: Componentes necesarios para la instrumentación GPIB.	36
Figura 3.2: Interpretación básica de los comandos en la interfaz.	37
Figura 3.3: Señales lógicas del bus GPIB.	42
Figura 4.1: Ejemplo de la comunicación con comandos Talk/üsten.	48
Figura 4.2: Ejemplo de la comunicación con comandos universales.	50
Figura 4.3: Ejemplo de la comunicación con comandos Addressed.	51
Figura 4.6: Jerarquía.	53
Figura 4.7: Señal de transferencia iniciada en controller.	57
Figura 4.8: Señal de transferencia iniciada en comando MxA.	57
Figura 5.1: Componentes de interfaz.	61
Figura 5.2: Bits en una interfaz.	62
Figura 5.3: Propiedades de un objeto.	68
Figura 5.4: Estructura del driver IVI de un osciloscopio.	74
Figura 6.1: Pasos.	78
Figura 6.2: Conectores.	80
Figura 7.1: Diagrama de Flujo de cómo adquirir voltaje.	87
Figura 7.2: Tarjeta Agilent instalada en CPU.	88
Figura 7.3: Comunicación con instrumentos.	88
Figura 7.4: Multímetro 34401A	89
Figura 7.5: Configurar multímetro.	89
Figura 7.6: Abrir un VI en blanco.	90
Figura 7.7: Ventana de diagrama a bloques y funciones.	90

Figura 7.8: Ventana de panel frontal y funciones.	92
Figura 7.9: Panel frontal del programa LabView para adquirir voltaje.	93
Figura 7.10: Diagrama a bloques para adquirir Voltaje.	94
Figura 7.11: Diagrama de Flujo de cómo adquirir resistencia.	97
Figura 7.12: Panel frontal de LabView para adquirir resistencia.	101
Figura 7.13: Diagrama a bloques de LabView para adquirir resistencia.	101
Figura 7.14: Termopar tipo K.	106
Figura 7.15: Linealización de la temperatura.	114
Figura 7.16: Diagrama de Flujo de cómo adquirir temperatura.	115
Figura 7.17: Panel frontal de LabView para adquirir temperatura.	120
Figura 7.18: Diagrama a bloques de LabView para adquirir temperatura.	120
Figura 7.19: Diagrama de Flujo de cómo controlar el voltaje.	125
Figura 7.20: Panel frontal de LabView para controlar el voltaje.	130
Figura 7.21: Diagrama a bloques de LabView para controlar el voltaje.	131
Figura 7.22. Partes de un tacogenerador.	134
Figura 7.23: Gráfica de linealización.	138
Figura 7.24: Diagrama de Flujo de cómo controlar el voltaje.	138
Figura 7.25: Panel frontal de LabView para el control de velocidad de un motor.	148
Figura 7.26: Diagrama a bloques de LabView para el control de velocidad de un motor.	149
Figura 7.27: Diagrama a bloques para controlar la velocidad de un motor, en la parte false.	150

Índice de Tablas

Tabla 2.1: Tabla de bits 1.	30
Tabla 2.2: Tabla de bits 2.	31
Tabla 4.1: Naturaleza de cada comando.	47
Tabla 4.2: Comando Talk/Listen.	48
Tabla 4.3: Comandos Univerales.	49
Tabla 4.4: Comandos Addressed.	50
Tabla 4.5: Comandos comunes.	52
Tabla 4.6: Comandos de un osciloscopio, según el estándar SCPI.	54
Tabla 6.1: Requerimientos.	79
Tabla 6.2: Parámetro.	81
Tabla 7.1: Coeficientes de Seebeck y tensiones de saluda	110
Tabla 7.2: Ventajas y Desventajas.	111
Tabla 7.3: Tabulación de temperatura.	113
Tabla 7.5: Tabulación de velocidad.	157

Índice de Fórmulas

Resistencia Total	104
Resistencia eléctrica	105
Resistencia de temperatura	105
Diferencia de potencial	123

INTRODUCCIÓN

El bus es el conjunto de líneas (cables) de hardware utilizados para la transmisión de datos entre los componentes de un sistema informático. Un bus es en esencia una ruta compartida que conecta diferentes partes del sistema como el procesador, la controladora de unidad de disco, la memoria y los puertos entrada y salida, permitiéndoles transmitir información.

Por lo general supervisado por el microprocesador, se especializa en el transporte de diferentes tipos de información.

Se caracterizan por el número de bits que pueden transmitir en un determinado momento.

Como este es parte integral de la transmisión interna de datos y como los usuarios suelen tener que añadir componentes adicionales al sistema, la mayoría de los buses de los equipos informáticos pueden ampliarse mediante uno o más zócalos de expansión (conectores para placas de circuito añadidas). Al agregarse estas placas permiten la conexión eléctrica con el bus y se convierte en parte efectiva del sistema.

Se refiere al camino que recorren los datos desde una o varias fuentes hacia uno o varios destinos y es una serie de hilos contiguos. En el sentido estricto de la palabra, esta definición solo se aplica a la interconexión entre el procesador y los periféricos.

Es simplemente un conjunto compartido de pistas trazadas en la placa de circuito principal, al que se conectan todas las partes que controlan y forman el ordenador. Cuando un dato pasa de un componente a otro, viaja a lo largo de este camino común para alcanzar su destino. Cada chip de control y cada byte de memoria del PC están conectados directa o indirectamente al bus. Cuando un nuevo componente se inserta en uno de los conectores de expansión,

queda unido directamente al bus, convirtiéndose en un objeto más de la unidad completa.

Cualquier información que entra o sale de un sistema ordenador se almacena temporalmente en al menos una de las distintas localizaciones que existen a lo largo del bus. La mayor parte de las veces el dato se sitúa en la memoria principal, que en la familia PC está formada por miles de posiciones de memoria de 8 bits. Pero algún dato puede acabar en un puerto, o registro, durante unos instantes, mientras espera que la CPU lo envíe a una posición adecuada.

Generalmente los puertos y registros almacenan sólo uno o dos bytes de información a la vez, y se utiliza normalmente como lugares de parada intermedia para los datos, que se están enviando de un lugar a otro. Siempre que se utiliza una posición de memoria, o un puerto, como lugar de almacenamiento, su localización está marcada por una dirección que la identifica individualmente. Cuando el dato está listo para ser transferido, se transmite primero su dirección de destino por el bus de direcciones, el dato realiza su transferencia por el bus de datos.

El objetivo de conectar una tarjeta a un bus de expansión es que ésta funcione como si estuviera directamente conectada al procesador. Con el fin de hacer factible estas características.

El GPIB permite una diversidad de conexiones con otros dispositivos del equipo de laboratorio a las diferentes carreras que existen dentro del ICBI.

Este equipo ya está vigente dentro de nuestro Centro de Investigaciones Avanzadas de Ingeniería Industrial y no tendremos que aplicar ningún gasto, por falta de personal capacitado y bibliografía existente de este equipo, no está en funcionamiento aún y su aplicación es importante tanto en el ámbito académico como en el laboral. El equipo se encuentra disponible a estudiantes a nivel licenciatura y postgrado que suelen no conocer el óptimo desempeño de este equipo ya es compatible con otras marcas de equipos.

La temática del trabajo está distribuida de la siguiente forma:

Capítulo 1: Se desglosará todo lo referente al bus GPIB así como la importancia y características del mismo.

Capítulo 2: Menciona los protocolos y normas a seguir para los estándares establecidos para el mejor rendimiento del equipo.

Capítulo 3: Modos de operación, control y desempeño del bus.

Capítulo 4: Se manejan los diferentes comandos y drivers a utilizar para la comunicación GPIB.

Capítulo 5: Conexiones y ventajas sobre otros buses

Capítulo 6: Configuración y pasos a realizar para la instalación de la tarjeta ALIGENT.

Capítulo 7: Se desarrolla una propuesta de prácticas de laboratorio, diseñadas de tal forma que los alumnos incursionen desde los niveles inferiores de la comprensión de buses, hasta la aplicación de prácticas.

Capítulo 1

ANTECEDENTES DEL BUS GPIB

1.1 Antecedentes.

Este *bus* empezó con Hewlett-Packard Instrument Bus (HP-IB) que es un estándar bus de datos digital de corto rango desarrollado en los años 70's. El cual fue diseñado para conectar dispositivos de examen y medida (por ejemplo multímetros, osciloscopios, nanovoltmetro, etc) con dispositivos que los controlen como un ordenador. Otros fabricantes copiaron en HP-IB, llamando a su implementación Bus de Instrumentación de Propósito General (GP-IB). En 1978 el bus fue estandarizado por el *Institute of Electrical and Electronics Engineers (IEEE)* como el IEEE-488 (488.1).

El IEEE-488 permite que 15 dispositivos inteligentes compartan un simple bus, con el dispositivo más lento determinando la velocidad de transferencia. La máxima velocidad de transmisión está sobre 1 Mbps, se muestra la conexión en la figura 1.1.[19]

El bus se compone por 16 líneas y 8 tierras físicas, agrupadas en tres grupos de acuerdo con sus funciones: bus de datos, bus de control de transferencia de datos y bus general. Algunas de ellas tienen retornos de corrientes comunes y otras tienen un retorno propio, lo que provoca un aumento del número de líneas totales.

Además del IEEE otros comités han adoptado el HP-IB. El *American National Standards Institute (ANSI)* lo llama ANSI Standard MC 1.1, y para la *International Electrotechnical Commission (IEC)* es el IEC Publication 625-1. En junio de 1987 el IEEE aprobó una revisión del estándar para instrumentos programables llamado

IEEE-488-1987 (488.2), en él se definieron códigos, formatos, protocolos y comandos comunes para todos los instrumentos.

No fue planeado por los diseñadores de HP específicamente para ser un interfaz de propósito general entre periféricos y ordenadores. Tales usos del bus fueron hechos por el *Commodore PET/CBM* para la comunicación con discos, impresoras, módems, etc. donde los dispositivos conectados al ordenador hablaban (*talking*) y escuchaba (*listening*) las líneas para realizar sus tareas. Todas las máquinas de 8 bits de Commodore utilizaron un IEEE-488 propietario con conectores DIN en vez del típico del HP-IB. Los ordenadores de HP también han usado este bus con un protocolo llamado CS-80.

En la actualidad la mayoría de los instrumentos electrónicos de medida pueden ser controlados mediante este bus. Recientemente el IEEE ha adoptado una nueva revisión hecha por National Instruments que permite más velocidad.[20]



Figura 1.1: Conexión de un sistema GPIB.

1.2 Características eléctricas del GPIB.

El bus de transmisión de datos de GPIB es de 8 bits en paralelo, y lógica negativa con niveles *TTL* estándar (cierto si el voltaje es 0.8 V y falso si el voltaje es 2.0 V). Los cables y conectores tienen el aspecto típico mostrado en la figura 1.2:

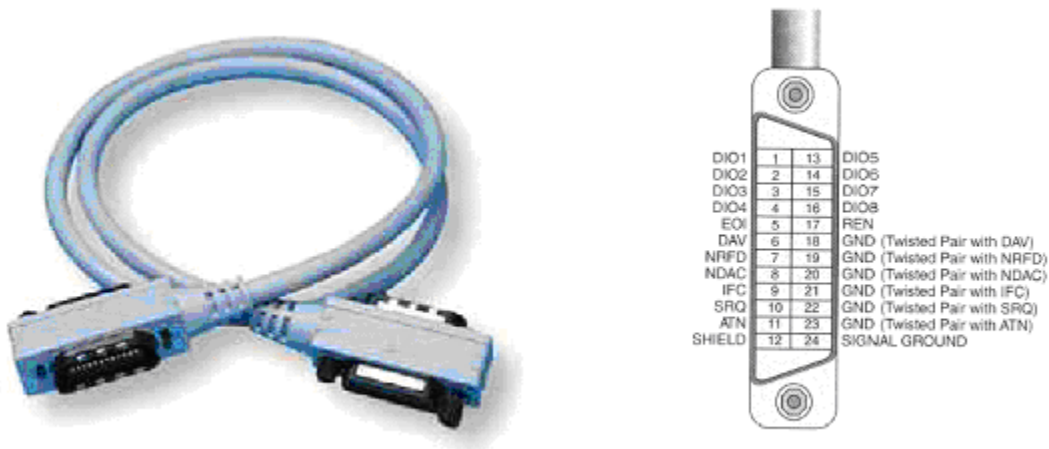


Figura 1.2: Cable de conexión GPIB (aspecto físico y distribución de señales).

El bus consta de 24 pines, repartidos de la siguiente forma:

- 8 líneas de transmisión de datos (DIO1-DIO8).
- 3 líneas para el control asíncrono de la comunicación (NRFD, NDAC y NRDAV). Mediante estas líneas se verifica la correcta transmisión de los datos, que es una de las fortalezas del GPIB.
- 5 líneas que gestionan la transmisión de comandos ATN, IFC, REN, SRQ y EOI que se menciona en el siguiente capítulo.

- Las 8 restantes componen las tierras de las diferentes líneas.

Para que el bus GPIB alcance la velocidad de transmisión para el que fue diseñado (hasta 8 Mb/s), deben cumplirse los siguientes requisitos:

- Puede haber un máximo de 15 dispositivos conectados al bus, y al menos dos tercios de ellos deben estar encendidos.
- La separación máxima entre dos dispositivos es 4 m, y la separación promedio en toda la red debe ser menor de 2 m.
- La longitud total de la red no debe exceder los 20 m.[21]

1.3 Hardware disponible para el sistema GPIB.

Un sistema típico consta de un ordenador con una tarjeta controladora GPIB (figura 1.3), más los instrumentos (compatibles con IEEE 488). Existen tarjetas GPIB para prácticamente todos los ordenadores presentes en el mercado (PC, Macintosh, estaciones Sun, Silicon Graphics, DEC Alpha, HP RS/6000, etc). En el caso concreto del PC, las controladoras GPIB pueden conectarse al bus *ISA*, *PCI*, *PCMCIA (portátiles)*, *USB*, *Ethernet*, *Firewire*, los puertos serie y paralelo. Existen asimismo adaptadores para los estándares de comunicación RS-232(puerto serie) y RS-485(puerto paralelo).[7]

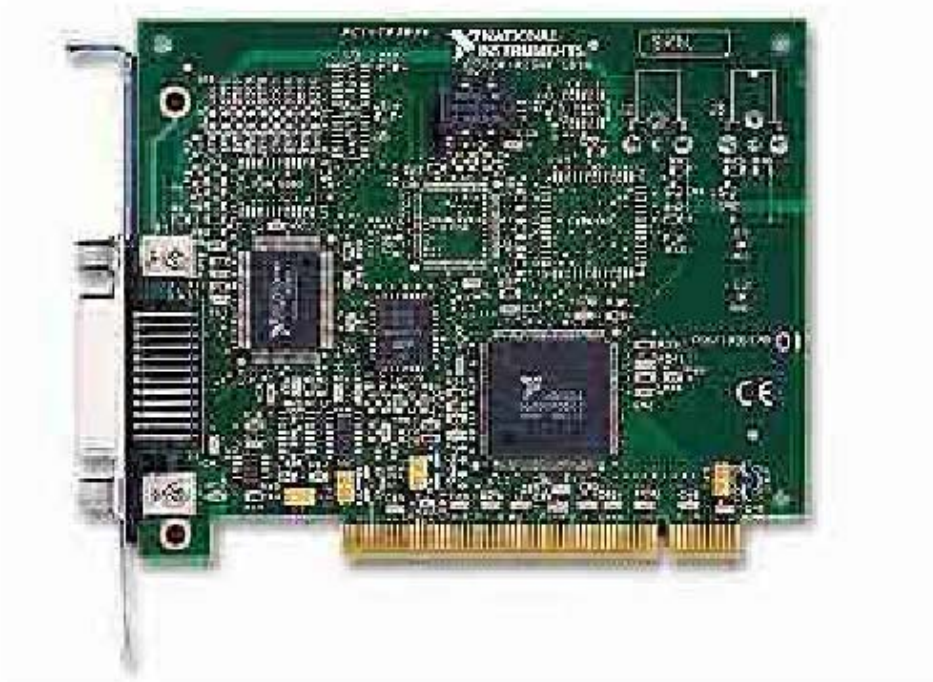


Figura 1.3: Tarjeta GPIB.

En la mayoría de los sistemas industriales, un PC con el sistema operativo DOS es suficiente para controlar el bus GPIB. La tarjeta GPIB se instala en un *slot* ISA o PCI libre. Si tanto la tarjeta como el sistema operativo son *plug and play*, en principio será reconocida y configurada automáticamente. En caso contrario (por ejemplo, bajo S.O.), debemos elegir de forma manual:

- El puerto de entrada/salida, entre los valores 100h y 3F8h.
- La línea de petición de interrupción (IRQ), entre 2 y 7, si queremos utilizarla.

Estos parámetros deben tomarse de forma que no produzcan conflictos con otras tarjetas instaladas en el PC.[9]

1.3.1 Controladores GPIB (IEEE 488).

GPIB (IEEE 488) consiste de una o más tarjetas de *interfaz* y uno o más instrumentos. Las tarjetas de interfaz son controladores GPIB, (figura 1.4). que gobiernan el flujo de información en el bus. Esto se logra mandando comandos a los instrumentos para que se comuniquen, entre sí o respondan a pedidos de servicio de los dispositivos. Puede haber más de un controlador GPIB en un bus, pero solo uno puede tener el control a la vez. Un controlador GPIB puede pasar el control del bus de sí mismo a otro controlador en el bus, pero solo un controlador puede tener el control de bus. Este tipo de controladores es conocido como Controlador-A-Cargo (Controller-In-Charge, CIC o System Controller) y es responsable del manejo total de bus.[29]

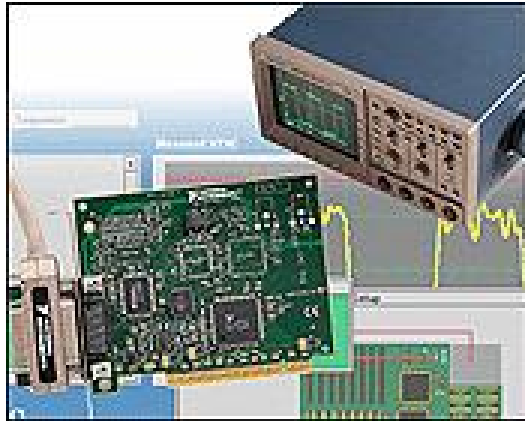


Figura 1.4: Tarjeta controladora.

Ejemplos de Hardware que utilizan el bus GPIB.[30]

Estos son algunas dispositivos que se pueden conectar al bus, e indicamos algunos datos técnicos y básicos para su uso.

Convertidor y Controlador GPIB - RS – 232 NI GPIB-232CT A



- Transforma un puerto RS-232 en uno GPIB Controla o integra un dispositivo RS-232 en un sistema GPIB
- NI-488.2 para Windows/DOS NI GPIB-232CV-A
- Conversiones entre protocolos GPIB y RS-232

Controladores GPIB -- rs-485



- Transforma un Puerto RS-485 en uno GPIB. Controla o integra un dispositivo RS-485 en un sistema GPIB
- 256 KB de búer RAM incluido
- Velocidades de transferencia de datos RS-485 hasta de 38.4 kbytes/s
- Soluciones en extensiones GPIB a bajo costo
 - El software NI-488.2 hace el enlace RS-485 transparente.
 - Extensiones hasta de un 1.2 km (4,000 ft)
- NI-488.2 para Windows/DOS
- Versiones DC disponibles

Puerto Paralelo PC



- ASIC TNT4882
 - Total compatibilidad IEEE 488.2 y máximo desempeño
- 3 modos de puerto paralelo bidireccionales
- Tamaño compacto
- NI-488.2 para Windows/DOS
- Versiones DC disponibles

IEEE 1393 (FireWire)



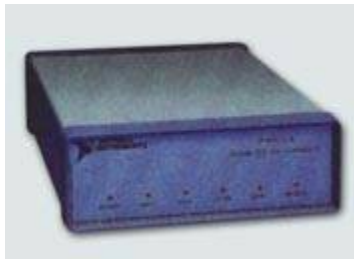
- ASIC TNT4882
- Total compatibilidad IEEE 488.2 y máximo desempeño
 - 1.5 Mb/s empleando IEEE 488.1
 - 4.0 Mb/s empleando IEEE HS488
- Controla hasta 14 instrumentos GPI
- Configuración Plug and play
- Insertable en caliente
- NI-488.2 para Windows Me/98

Extensores



- Extensores en fibra óptica
- 1 km máximo en longitud de cable GPIB
- Transmisión serial de alta velocidad
- Máximas velocidades de transferencia
- - 1.1 Mb/s empleando IEEE 488.1
- - 2.8 Mb/s empleando HS488
- Extensión paralela de 300 m máximo en longitud de cable GPIB
- 28 dispositivos pueden ser conectados
- Velocidades de transmisión hasta de 900 kb/s
- Software transparente

Expansor



- Expande el número máximo de dispositivos GPIB de 15 a 28
- Suministra 1600 V de aislamiento eléctrico entre sistemas GPIB
- Transparente para usuarios de software

1.3.2 Tipos de Interfaces.

National Instruments elabora varias tarjetas de interfaz GPIB (IEEE 488) insertable para los buses de computadoras tradicionales como: ISA, PCI, PCMCIA y más. Normalmente, sólo cuenta con una tarjeta de interfaz GPIB por computadora. Pero se pueden tener varias tarjetas si necesita controlar más instrumentos (el límite es 14 instrumentos por Controlador-A-Cargo).

Adicionalmente a estas interfaces, los controladores GPIB (IEEE 488) pueden escapar los confines de una computadora y pasar por los puertos Ethernet, paralelo, serial, USB y FireWire. Estas alternativas a las interfaces GPIB cuentan con toda la funcionalidad de una tarjeta insertable y además expanden grandemente la flexibilidad de sus aplicaciones.[30]

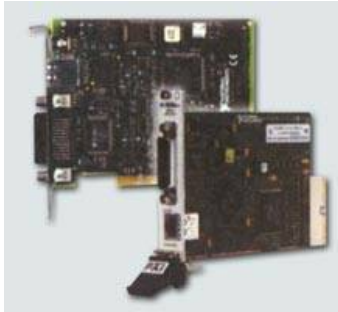
Ejemplos de interfases GPIB Multiplataforma.

PCI y PXI



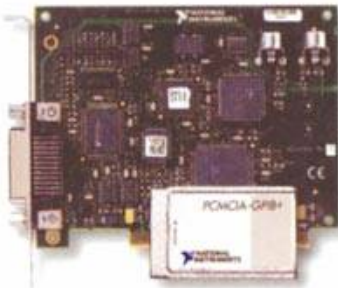
- TNT4882 ASIC
- - Compatibilidad total con IEEE 488.2 y máximo desempeño
- MITE PCI ASIC - Controlador bus-maestro DMA
- Máximas velocidades de transferencia GPIB
- - 1,5 Mb/s empleando IEEE 488.1
- - 7,7 Mb/s empleando HS488

Controlador GPIB y Puerto Ethernet para PCI y PXI



- Búferes FIFO para desacoplar transferencia GPIB de transferencia PCI
- Puerto monitor GPIB para diagnóstico de tarjeta y nivel de bus
- Total prueba funcional en sistema con modo de loop-back
- Máximas velocidades de transferencia GPIB
 - 1,5 Mb/s empleando IEEE 488.1
 - 7,7 Mb/s empleando HS488
- Puerto Ethernet compatible con 100BaseTX
- Duplex total tanto a 10 como 100 Mb/s

Controlador/ Analizador GPIB



- Controlador IEEE 488.2 y analizador GPIB
- Tarjeta simple, requiriendo sólo una ranura
- Solución de problemas de hardware y software IEEE 488
- Software Analizador GPIB para Windows de actividad HS488 en capturas/monitoreo con interfaz de usuario simple
- Compatibilidad Plug and Play, incluye NI-488.2

Ethernet - TCP/IP



- Instrumentación controlada IEEE 488.2 donde sea que exista una red de trabajo TCP/IP basada en Ethernet
- Compatible con par cruzado (1 OBaseT o 1 OOBBaseTX)
- NI-488.2 para Windows 2000/NT/XP/Me/9x, TRU64 UNIX (Digital UNIX), Solaris, HP-UX, Mac OS, Mac OS X
- Velocidad de transferencia sostenida sobre 800 kb/s

Bus Serial Universal



- ASIC TNT4882
 - Total compatibilidad IEEE 488.2 y máximo desempeño
- Controla hasta 14 instrumentos GPIB
- Tamaño compacto y peso liviano
- Sin interruptores DIP-configuración plug and play
- No requiere fuente de poder externa
- No requiere cable GPIB para conectarse a instrumentos
- NI-488.2 para Windows 2000/XP/Me/98

PC Card (PCMCIA)



- ASIC TNT4882
 - Total compatibilidad IEEE 488.2 y máximo desempeño
- Tarjeta PCMCIA Tipo II
- Máximas velocidades de transferencia GPIB
- Compatibilidad Plug and Play
- Bajo consumo de potencia (65 mA)

ISA/EISA



- ASIC TNT4882
- Total compatibilidad IEEE 488.2 y máximo desempeño
- Totalmente sin jumpers Compatibilidad Plug and Play
- Máximas velocidades de transferencia GPIB (Windows)
 - 1,5 Mb/s empleando IEEE 488.1
 - 2,9 Mb/s empleando HS488 (EISA)

Productos GPIB para Linux



National instruments ofrece una completa versión de LabView y GPIB para Linux. El Software controlador GPIB está disponible a través del ambiente de desarrollo de LabView.

PC/XT



- Total compatibilidad con Controlador IEEE 488.2 y mayor desempeño empleando la NAT4882 ASIC
- Velocidades de transferencia de 300 kbytes.

HP-UX



- ASIC TNT4882
 - Compatibilidad Total con IEEE 488.2
- Máximas velocidades de transferencia GPIB
- Total compatibilidad con interfaz PCI 2.1
- Disponibilidad de juego de desarrollo en controlador NI-488 NI GPIB-ENET/100
- Controlador Ethernet-a-GPIB
- Control de instrumentos donde exista una red de trabajo de 10 o 100BaseTX TCP/IP
- Ejecución de programas existentes escritos con NI-488.2 o NI-VISA sin modificaciones

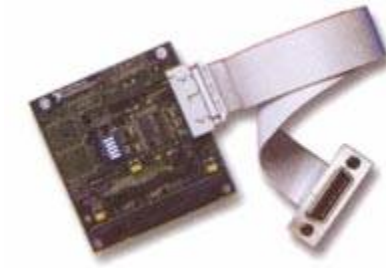
Ejemplo de interfaz GPIB para Computadoras de Escritorio, Laptops y Estaciones de Trabajo - Soluciones OEM.[33]

Solaris de Sun



- Controlador Ethernet-a-GPIB
- ASIC TNT4882
 - Total compatibilidad IEEE 488.2
- Controlador Ethernet-a-GPIB
- Máximas velocidades de transferencia GPIB
 - 1.4 Mb/s empleando IEEE 488.1
 - 3.6 Mb/s empleando HS488 (SBus)
 - 7.7 Mb/s empleando HS488 (PC)
- Controlador Ethernet-a-GPIB
- NI-488.2 para Solaris incluido
 - NI GPIB-ENET/100
- Controlador Ethernet-a-GPIB
- Control de instrumentos donde exista una red de trabajo a 10 o 100BaseTX TCP/IP
- Ejecución de programas existentes escritos con NI-488.2 o NI-VISA sin modificaciones

Sistemas PC/104



- ASIC TNT4882C -Total compatibilidad 1EEE488.2
- Máxima velocidad de transferencia GPIB
 - Sobre 1.5 Mb/s empleando IEEE 488.1
 - Sobre 1.6 Mb/s empleando HS488
- Interfaz de bus ISA a 16-bit con empaquetamiento y desempaquetado byte-a-palabra
- Puerto monitor GPIB para diagnóstico de tarjeta y nivel de bus
- Controlador GPIB de grado industrial

IBM RS/6000 - Micro Channel



- Compatibilidad total de controlador con IEEE 488.2 y máximo desempeño empleando NAT4882 y Turbo488 ASICs
- Circuitería POS para selección de direcciones de E/S, nivel de interrupción y canal DMA (sin jumpers o switches en hardware)
- NI-488 para AIX

➤ **Proveedores de Instrumentos/Soluciones OEM – Software**



- API amigable al usuario para desarrollo de instrumentos compatibles IEEE-488.2 con bus firmware independiente
 - Soporte de bus GPIB
 - Soporte de bus Ethernet (VXI-11)
- API diseñado especialmente para aplicaciones sin controlador
- Juego de código fuente disponible como DDK

Proveedores de Instrumentos/Soluciones OEM Hardware



- TNT4882 - GPIB Comunicador/Receptor ASIC de bajo costo y alto desempeño
 - Paquete QFP de 100-pin
- NAT9914 - Chip reemplazable por Controlador TI TMS9914A
 - Paquete DIP a 40-pin
 - Paquete PLCC a 44-pin
 - Paquete QFP a 44-pin
- NAT7210 - Chip reemplazable por controlador NEC pPD7210
 - Paquete DIP a 40-pin

1.3.3 Controladores vs Convertidores.

Un controlador GPIB (IEEE 488) gobierna las comunicaciones de un bus, en contraste un convertidor GPIB simplemente traduce los datos de un tipo de bus a otro tipo de bus. Por ejemplo, el GPIB-232CV-A (convertidor GPIB a RS-232 serial) traduce comandos seriales a comandos GPIB. La mayoría de las interfaces GPIB son convertidores y controladores. Solo el GPIB-232CV-A es convertidor solamente. La mayoría de los convertidores actúan como calle de dos sentidos: Pueden convertir de GPIB a la interfaz alterna y viceversa. Por ejemplo, con el GPIB-485CT-A (el controlador GPIB a serial RS-485), se puede usar el puerto RS-485 en su computadora para comunicarse con un instrumento GPIB o puede conectar un instrumento RS-485 a un bus GPIB existente. En el primer caso, el GPIB-485CT-A tiene todas las capacidades completas de un controlador GPIB (por ejemplo, puede controlar hasta 14 instrumentos con él), pero en el segundo caso, el GPIB-485CT-A trabaja con un solo instrumento RS-485 para hacerlo parecer como un instrumento GPIB.

Excepciones a este concepto de calle de dos sentidos son los GPIB-USB, el GPIB-1394, y el GPIB-ENET, los cuales solo pueden ser utilizados como instrumentos GPIB. Por ejemplo, el GPIB-ENET le permite controlar hasta 14 instrumentos GPIB a través de una conexión Ethernet, pero no se puede controlar más instrumentos Ethernet a través de la conexión GPIB.[31

Capítulo 2

PROTOCOLOS DE GPIB.

2.1 Introducción.

En Informática y Telecomunicaciones, un protocolo es una convención, o estándar, o acuerdo entre partes que regulan la conexión, la comunicación y la transferencia de datos entre dos sistemas. En su forma más simple, un protocolo se puede definir como las reglas que gobiernan la semántica (significado de lo que se comunica), la sintáxis (forma en que se expresa) y la sincronización (quién y cuándo transmite) de la comunicación.

Los protocolos pueden estar implementados bien en hardware (tarjetas de red), software (drivers), o una combinación de ambos.

2.2 La norma IEEE 488.1.

Pin-out del bus GPIB.

- 8 líneas de datos bidireccionales: DIO7....DIO0.
- 3 líneas de control de transferencia (handshaking) entre talker-listener.
- 5 líneas de administración de la interface del bus.

Por las 8 líneas de datos (bidireccional) se transportan mensajes en código ASCII de 7 bits.

Por las tres líneas de handshaking (comunicación asíncrona) se establece el acuerdo entre un talker y un/varios listener/s:

- DAV (Data Valid) El talker indica a los listeners que hay un dato válido (correcto y estable) en el bus.
- NRFD (Not Ready for Data) Los listeners indican si están o no preparados para recibir los datos del talker. Sólo puede pasar a nivel alto cuando todos están listos para recibir datos.
- NDAC (Not Data Accepted) Los listeners indican al talker si han aceptado o no el dato. Sólo puede pasar a nivel alto cuando todos los han aceptado.

Las cinco líneas de administración del interface del bus son:

- IFC (InterFace Clear) El controlador reinicializa todos los instrumentos a un estado inactivo predeterminado. Para ello la pone a cero durante un intervalo de tiempo. Cada instrumento vigila el estado de la línea continuamente y debe responder a su cambio en 100 microsegundos.
- ATN (AtTeNtion) El controlador la activa cuando quiere indicar que por el bus circulan mensajes de interface (entre controlador e instrumentos). La desactiva cuando circulan mensajes de dispositivo (entre listener y talker). Cada instrumento vigila el estado de esta línea continuamente.
- REN (Remote Enable) El controlador habilita a un instrumento a ser controlado local (desde su panel frontal) o remotamente.

- EOI (End-OR-Identify) El talker la activa para indicar al listener y al controlador que ha terminado de transmitir todos los datos. Los listeners monitorizan esta línea y al detectar que está activa completan la transmisión. También tiene otra función. Cuando es accionada por el controller, con la línea ATN puesta a cero, es para hacer una llamada en paralelo.
- SRQ (Service ReQuest) Cualquier instrumento solicita al controlador que necesita de sus servicios. Es una especie de interrupción. Acto seguido, el controlador hace un polling (sondeo) para detectar qué instrumento le ha solicitado.[20]

2.2.1 Mensajes remotos. Mensajes dependientes del interface. Funciones de la interface.

Con todos estos nombres se nombra a aquellos mensajes que sirven para preparar una comunicación (handshaking tal como se ha visto antes) entre talker y listener. Para ello se precisa que el controlador mantenga el protocolo adecuado. Estos mensajes circularán por las 8 líneas DIO7....DIO0 acompañados por alguna señal de administración del bus (IFC, ATN, REN, EOI, SRQ). Los mensajes dependientes de la interface por mencionar algunos son:

- GTL (Hex01) Go To Local. Cambia el modo de funcionamiento del instrumento: local/remoto.
- GET (Hex08) Group Execute Trigger. Para poner en marcha las funciones del instrumento.

- LLO (Hex11) Local Lockout. Inhibe la acción de los controles locales de un instrumento con capacidad de programación remota.
- DCL (Hex14) Device Clear. Todos los instrumentos capaces de responder vuelven a un estado inicial predefinido.
- PPU (Hex15) Parallel Poll Unconfigure. Deja a todos los dispositivos que responden a la llamada en paralelo, en estado pasivo (incapaces de responder a llamadas en paralelo).
- SPD (Hex19) Serial Poll Disable. Termina la llamada serie.
- MLA0-30 (Hex20-3E) My Listener Address Pone a un instrumento como listener.
- UNL (Hex 3F) UNListen Desdireziona a los listeners activos en este momento.
- MTA0-30 (Hex40-5E) My Talker Address Pone a un instrumento como talker.
- UNT (Hex5F) UNTalk Desdireziona a los talkers activos en este momento.

2.3 La norma IEEE 488.2

La norma IEEE 488.2 fue creada en 1987 para completar algunos aspectos entre la compatibilidad entre instrumentos que no quedaban especificados en la norma IEEE 488.1

Básicamente esta norma define una serie de comandos comunes a todos los instrumentos. También define una estructura de bytes donde el instrumento guarda y actualiza toda la información referente al estado del instrumento.[35]

2.3.1 Registros de estado.

La norma contempla tres registros de estado. Además cada uno de estos registros tiene un registro de habilitación/deshabilitación (Enable Register) que permite enmascarar ciertos bits. Los tres registros son:

1. Standard Event Status Register (8 bits), contiene las siguientes informaciones.

- 1.1. Operation Complete (operación completa).
- 1.2. Request Complete (petición completa).
- 1.3. Query Error (error de pregunta).
- 1.4. Device Error (error de dispositivo).
- 1.5. Execution Error (ejecución de error).
- 1.6. Command Error (comando de error).
- 1.7. Power On (alimentación encendida o apagada).

Para cambiar y conocer las máscaras disponemos de los comandos *ESE y *ESE?. Para conocer el contenido de todo el Standard Event Status Register, disponemos del comando *ESR?.

2. Questionable Data (16 bits), contiene las siguientes informaciones:

- 2.1. Voltage Overload(caso de sobretensión).
- 2.2. Current Overload(en caso de sobrecorriente).
- 2.3. Ohms Overload(sobrerresistencia).
- 2.4. Limit Test Fail LO(limite el suspenso de prueba inferior).
- 2.5. Limit Test Fail HI(limite el suspenso de prueba superior).

Para leer el contenido de todo el registro, podemos utilizar este comando combinado: STAT:QUES:EVEN? y para habilitar/deshabilitar así como conocer el contenido de la máscara STAT:QUES:ENAB? y STAT:QUES:ENAB, respectivamente.

3. Status Byte (8 bits), registro de estado ya existente en la norma IEEE 488.1 que contiene los siguientes bits:

3.1 Questionable Data, si alguno de los bits del registro Questionable Data está activo y habilitado.

3.2 Message Available, si hay un mensaje que el instrumento quiere enviar. Tiene una cola de saluda (output buffer) donde están los posibles mensajes a enviar.

3.3 Standard Event, si alguno de los bits del registro Standard Event Status Register está activo y habilitado (ver gráfico más adelante)

3.4 Request Service (este bit no tiene máscara) el instrumento ha pedido o no una petición al controlador.

Para leer el contenido de este registro, utilizamos el comando: *STB?. Para conocer el registro de máscara se usa *SRE? y para habilitarlo *SRE.

En el diagrama de abajo, se ven todos estos contenidos. Fijarse que la suma (OR lógico) de los bits (sólo aquellos habilitados) de los registros Standard Event Status Register y Questionable Data activa los dos bits del Status Register.

2.4 Protocolo básico.

GPIB es un bus y un protocolo estándar utilizado para el control y comunicación a través de instrumentos de medida, tales como polímetros digitales, osciloscopios, etc, este bus permite configurar sistemas automáticos en el laboratorio y en la industria con gran flexibilidad y potencia.

Se describe el estándar GPIB, incluyendo sus características eléctricas, lenguaje de comunicación y las herramientas existentes para la programación.[22]

- El equipo y el controlador se comunican intercambiando mensajes de ordenes y mensajes de respuesta.
- Los mensajes de ordenes son enviados por el controlador, y pueden ser de dos tipos:
 - ❖ Ordenes de control: Que requieren un cambio de estado del equipo, pero que no requieren ninguna respuesta.
 - ❖ Ordenes de requerimiento: Que solicitan información sobre el estado del equipo o sobre información que posee.
- El equipo se comunica (envía un mensaje de salida) como respuesta de una orden de requerimiento.
- El controlador sólo admite un mensaje de salida (respuesta de una orden de requerimiento), y lo requiere antes de enviar un nuevo mensaje de ordenes. En caso contrario se genera una situación de bloqueo.
- La regla básica del protocolo es: "El equipo solo habla cuando está dispuesto a ello, y en ese caso, tiene que hablar antes de que se le ordene hacer una cosa nueva".
- Cuando el equipo es encendido, o cuando recibe una orden de inicialización "*CLS" el buffer de entrada y de salida son inicializada, y el parser es inicializado a la raíz de su árbol de ordenes.

- El equipo y el controlador se comunican intercambiando mensajes de ordenes y de respuestas completos. Esto significa que el controlador siempre debe terminar un mensaje de ordenes, antes de intentar leer una respuesta.
- Si el equipo envía un mensaje de respuesta, el controlador debe siempre leer de forma completa el mensaje, antes de que envíe un nuevo mensaje de ordenes al equipo.
- El controlador puede enviar un mensaje conteniendo múltiples ordenes de requerimientos. A esto se le denomina un "requerimiento compuesto". Los diferentes requerimientos dentro del mensaje deben estar separados por el delimitador ";". Los mensajes de respuesta son encolados en la cola de salida, separados entre sí también, por el delimitador ";"
- Los comandos son ejecutados en el orden en que han sido recibidos.

2.5 Protocolos de excepción.

Cuando un error ocurre en el intercambio de la información, este no termina en la forma normal, sino que sigue un protocolo de excepción:

- a) Equipo direccionado para hablar sin nada en la cola:
 - Si es consecuencia de que el equipo no haya recibido una orden de requerimiento, el equipo indicará un error de encolamiento, y no enviará ningún byte por el bus.

- Si es como consecuencia de que la orden de requerimiento previa no se ejecutó como consecuencia de un error, el equipo no indica ningún error de encolamiento, sino que el equipo espera a recibir el siguiente mensaje del controlador.

b) Equipo direccionado para hablar sin que ningún equipo escuche:

En este caso, el equipo esperará o bien a que algún equipo escuche, o a que el controlador tome el control.

c) Error de orden:

Se genera cuando se detecta un fallo de sintaxis o una orden no reconocible.

d) Error de ejecución:

Se genera si un parámetro está fuera de rango, o si el equipo se encuentra en un estado que no permita la ejecución del comando requerido.

e) Error específico del equipo:

Se produce cuando el equipo es incapaz de ejecutar una orden, como consecuencia de una razón estrictamente dependiente de él, y no del protocolo seguido por el bus.

f) Error de encolamiento:

Se genera si no se sigue el protocolo de lectura de los datos de la cola de salida.

g) Condición inconclusa:

Si el controlador intenta leer un mensaje de respuesta antes de que el programa haya concluido de ejecutar la orden que los genera. En este caso el Parser, se inicializa a si mismo, la respuesta ya elaborada es limpiada de la cola de salida, y ningún dato es transferido por el bus.

h) Condición interrumpida:

Si el controlador no lee completamente el mensaje generado por un mensaje de requerimiento, y envía otro mensaje de orden, el equipo genera un error de encolamiento, y el segmento de mensaje de salida no leído es eliminado. La orden que interrumpe es inafectada.

i) Bloqueo de buffer:

El equipo alcanza un estado de bloqueo si el buffer de entrada, está lleno y también resulta llena la cola e salida. Esta situación ocurre si un mensaje de orden muy largo que contiene órdenes de requerimiento ha sido enviado, y genera un mensaje de salida superior al que puede contener la cola. El controlador no puede terminar de enviar el mensaje de entrada porque no cabe, y el buffer de entrada no se vacía porque espera que su cola de salida sea vaciada para concluir la orden en ejecución. En este caso el equipo rompe el bloqueo, limpiando la cola de salida y generando un error de encolamiento.[23]

2.6 Estatus de un equipo por el protocolo IEEE 488.2.

El estándar IEEE 488.2 ofrece un mecanismo estandarizado de presentar y mostrar el estado interno del equipo. A través de este mecanismo, se puede tener información de si el equipo tiene dato dispuesto para transferir, así como si algún tipo de error ha ocurrido. Figura 2.1).

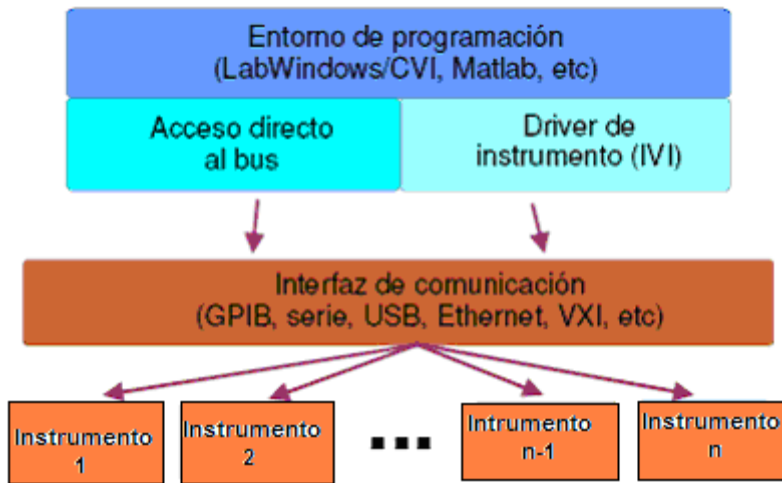


Figura 2.1: Estructura genérica de programación de un sistema GPIB.

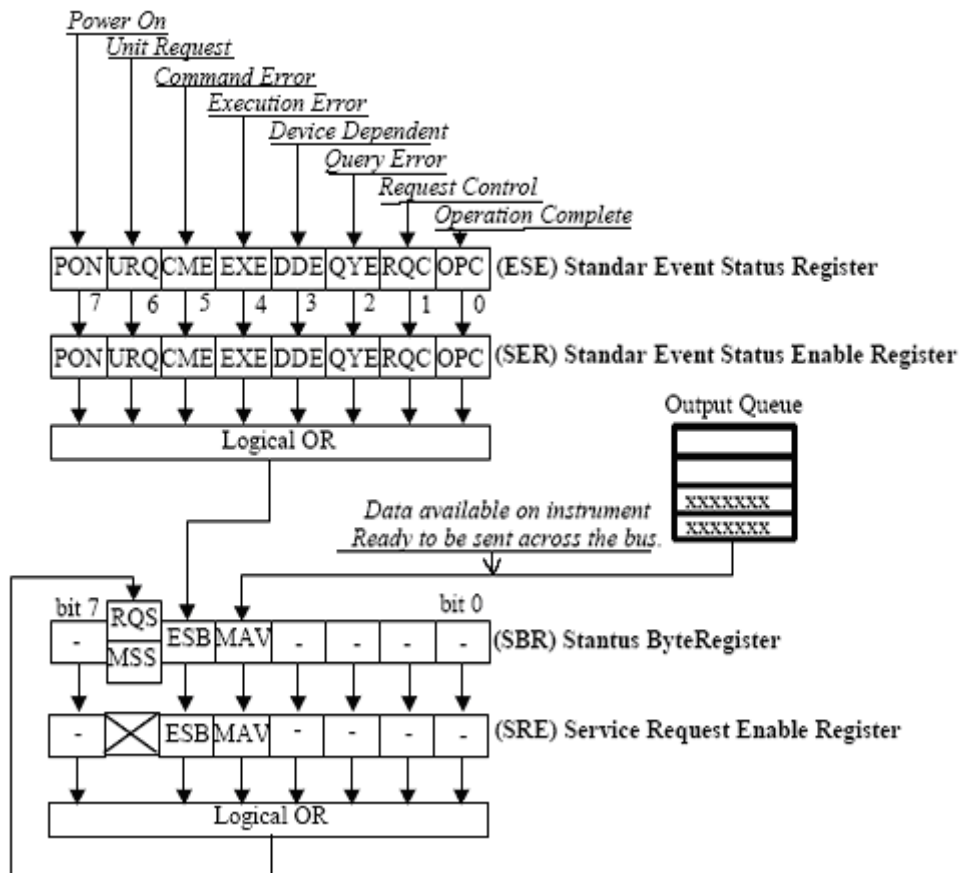


Figura 2.2: Registros en la programación.

El mecanismo se basa en cuatro registros(Figura 2.2)

- Status Byte Register (SBR): Cada bit está asociado con un tipo específico de evento. Cuando un evento ocurre, el instrumento establece el correspondiente bit a 1. Se puede habilitar e inhibir el efecto de cada bit del SBR a efectos de requerir atención (bit RQS), con el correspondiente bit del registro SER. Se puede determinar que eventos han ocurrido por leer los establecidos en el registro SBR.(Tabla 2.1)

Bit	Etiqueta	Descripción
0-3	-	Instrumento específico mensajes resumidos.
4	MAV	El Mensaje que el bit Disponible indica si los datos están disponibles en la Cola de Salida. MAV es 1 si la Cola de Salida contiene datos. MAV es 0 si la Cola de Salida es vacía.
5	ESB	El Bit de Estado de Acontecimiento indica si uno o varios acontecimientos permitidos Han ocurrido. ESB es 1 si un acontecimiento permitido ocurre. ESB y 0 si ningún acontecimiento permitido ocurre. Usted permite acontecimientos con el Estado de Acontecimiento Standar de Registro Permitido.
6	MSS	El Estado de Amo(Maestro) resume el ESB y el MAV. MSS tienen 1 si MAV O ESB es 1. MSS tienen 0 si tanto MAV como ESB son 0. ¿Estos son obtenidos de comando *STB?.
	RQS	El bit de Servicio de Petición indica que los instrumentos solicitan el servicio del regulador GPIB. Estos son obtenidos por una encuesta sucesiva.
7	-	Instrumento específico mensajes resumidos.

Tabla 2.1: Tabla de bits 1.

- Event Status Register (ESR-registro de estado de acontecimiento): Byte de máscara para determinar el si se establece el Request Service.

Por ejemplo, si se desea conocer si se ha producido un error en el instrumento, el bit 5 del registro SER debe ser establecido.

- Event Status Enable Register (ESE-registro de estado acontecimiento permite registro): Cada bit de este registro está asociado con un estado específico del instrumento. Cuando el estado cambia, el instrumento establece el correspondiente bit a 1. Se puede habilitar o inhibir los eventos

que van a ser proyectados sobre el bit ESB, a través de los correspondientes bits de máscara de registro SER. Se puede leer el evento que ha ocurrido leyendo el contenido del registro ESR.

- Service Request Enable Register (SER-petición de servicio permite registro): Mascara de los bits del registro ESE a efecto de requerir el servicio a través del registro SRB.(Tabla 2.2)

Bit	Etiqueta	Descripción
0	OPC	La Operación Completa el bit indica que todas las órdenes han completado.
1	RQC	El bit de Control de Petición no es usado según la mayor parte de instrumentos.
2	QTE	El bit de Error de Pregunta indica que el instrumento intentó leer y Vaciar el buffer de salida, o que los datos en el buffer de salida fueron perdidos.
3	DDE	El bit de Error Dependiente de Dispositivo indica que un error de dispositivo ocurrió (error de prueba).
4	EXE	El bit de error de ejecución indica que un error ocurrió cuando los dispositivos ejecutaban un una orden o un Query.
5	CME	El bit de error de orden indica que el error de sintaxis de orden ocurrió.
6	URQ	El bit de Petición de Usuario no es usado según la mayor parte de instrumentos.
7	PON	El Power sobre el bit indica que está encendido.

Tabla 2.2: Tabla de bits 2.

Por ejemplo, si deseamos conocer cuando ocurre un error de ejecución, se deben establecer los bits 4 del registro ESE, y además se debería establecer el bit 5 del registro SER a fin de que el evento de error de interés sea reportado por el bit ESB del registro SBR.[28]

2.6.1 MODELO OPERATIVO DE LOS EQUIPOS (IEEE-448.2) PROTOCOLOS.

El estándar IEEE-488.2 define los modos de operación básicos de los equipos que lo satisfacen. Define los protocolos de intercambio de mensajes con

el que el equipo y el controlador se comunican, así como, facilidades básicas de control del instrumento.

Define:

- a) Cuando los equipos están dispuestos a escuchar o a hablar a lo otros equipos.
- b) Que ocurre cuando no se cumplen las normas establecidas en el protocolo.

Los equipos que satisfacen los modos operativos definidos en el estándar IEEE-488.2., necesariamente satisfacen los niveles de comunicación físicos definidos en el estándar IEEE-488.1. La implicación inversa no es requerida. El modelo de intercambio de un equipo que satisface el estándar IEEE-488.2, se visualiza en la figura 2.3:

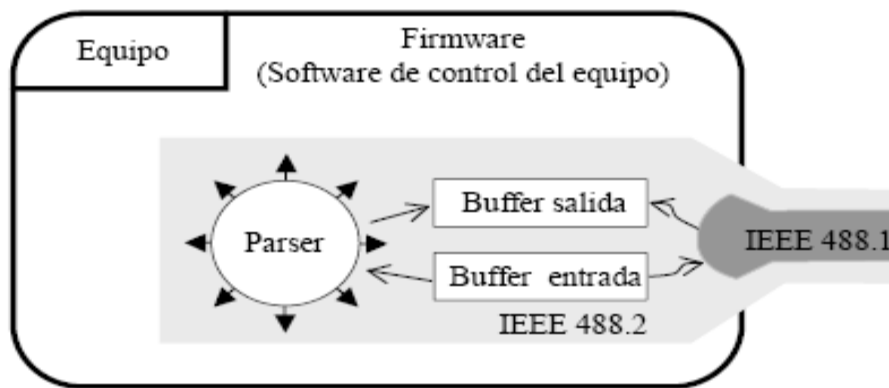


Figura 2.3: Modelo de intercambio de un equipo.

Buffer de entrada: Es el área de memoria en la que las ordenes y requerimientos de entrada son almacenadas, antes de que sean interpretadas y ejecutadas por el *Parser*. El buffer de entrada, permite que el equipo controlador del instrumento, almacene en el buffer un string conteniendo una o varias ordenes que llevan un

cierto tiempo ser ejecutada, y mientras esto ocurre pueda realizar otras operaciones con otros equipo.

Buffer de salida: Es el área de memoria en el que los datos de salida (mensajes de salida) son almacenadas en espera de que el controlador decida leerlas.

Parser: Es el controlador interno del equipo que interpreta los mensajes completos almacenados en el buffer de entrada, los ejecuta, y en caso de que sean de requerimiento, deposita en la cola de salida los datos que resulten.[35]

2.7 Protocolo de transferencia elemental del bus GPIB IEEE-488.

El bus GPIB IEEE-488 es un bus *asíncrono* que implementa un protocolo completamente bloqueante (*fully interlocked*). En las transferencias de información puede haber varios destinos o receptores, esto hace necesario líneas independientes para indicar aceptación de datos **DA** y finalización del ciclo actual **RDY**. Como consecuencia de esta separación en dos líneas, además de los 4 *flancos* necesarios para la implementación del protocolo completamente bloqueante, se producen 2 flancos que carecen de relevancia. El protocolo está esquematizado en la figura 2.4:

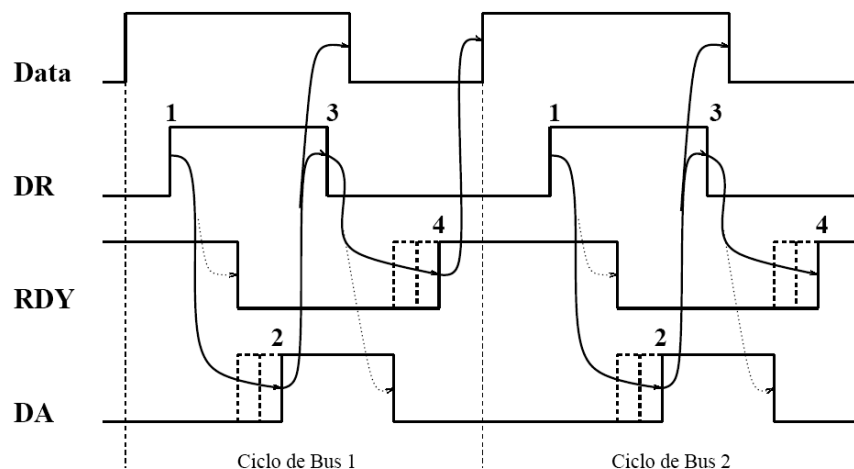


Figura 2.4: Transferencia del bus GPIB.

1. El maestro comprueba si todos los esclavos están listos (**RDY**=alto) y, si es así, coloca los datos y activa **DR** (flanco 1).
2. Los esclavos responden colocando a nivel bajo sus señales **RDY** y, conforme cogen los datos, ponen sus señales **DA** a nivel alto. La línea **RDY** señala un nivel bajo en el momento que un esclavo baje su señal **RDY**, ya que es un **AND**—cableado. Como **DA** es también un **AND**—cableado, la línea **DA** sólo alcanza el nivel alto cuando el más lento de todos haya cogido los datos (flanco 2).
3. Como respuesta al flanco de subida de **DA**, el maestro indica que ha recibido que todos han cogido los datos bajando **DR** (flanco 3) y retira los datos.
4. Los esclavos observan que los datos han sido retirados e indican que están listos para una nueva transferencia poniendo sus señales **DA** a nivel bajo y sus señales **RDY** a nivel alto. Como **RDY** es un **AND**—cableado la línea alcanza el nivel alto cuando el más lento de todos los esclavos indique que está listo para una nueva transferencia (flanco 4).[32]

Capítulo 3

COMPONENTES QUE SE UTILIZAN EN EL CONTROL DE LA INSTRUMENTACIÓN POR GPIB.

3.1 Tarjeta hardware de conexión al bus GPIB.

Para su control dispone de software interno de control que interpreta los mensajes que recibe por el bus GPIB e interacciona con el *firewire* propio del equipo. El “Parser” es el thread de gestión del intercambio de mensajes por el bus GPIB.(Figura 3.1)[4]

- Nivel Físico (IEEE 488.1): La comunicación entre los equipo se basan en un bus físico, compuesto por un conjunto de líneas con niveles lógicos bien definidos y con protocolos de comunicación basados en los estados lógicos de las líneas.

- Nivel Operativo (IEEE 488.2): El protocolo operativo básico dentro del que se encuadra el intercambio de información, datos e instrucciones básicas de control.

- Driver GPIB (SCPI): El computador interacciona con el bus GPIB a través de una tarjeta de control hardware que resuelve y atiende los dos protocolos anteriores. El propio fabricante ofrece una interfaz software implementada por un conjunto de funciones que permiten el acceso de los

programas a la funcionalidad del bus. Puede ofertar una interfaz constituida por una librería de funciones que corresponde al lenguaje SCPI.

- Driver VISA: Estándar de driver que ofrece un conjunto de librerías estandarizadas que permiten integrar equipos conectados por diferentes medios de comunicación.
- Instrumentation Control ToolBox: Conjunto de funciones Matlab que permite gestionar los driver GPIB o Visa, y establecer comunicación con los equipos.
- Matlab (Versión 6.5): Programa interactivo que ofrece un lenguaje de programación de alto nivel y unos conjuntos de librerías matemáticas y gráficas muy extensas.

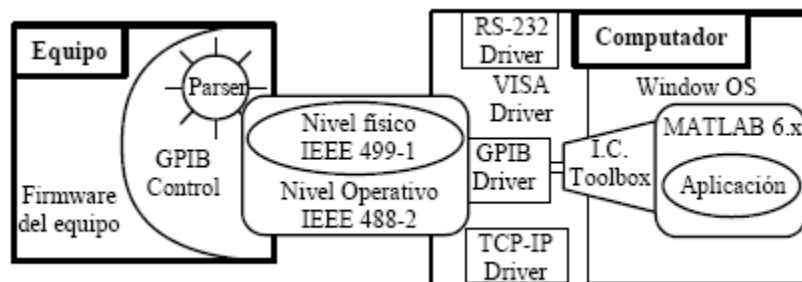


Figura 3.1: Componentes necesarios para la instrumentación GPIB.

3.2 Bus GPIB Nivel Físico (IEEE-488.1). Estructura del bus GPIB.

El bus GPIB está basado en 16 líneas activas, además de la tierra. Estas 16 líneas se organizan en tres buses:

- Bus de Datos (DIO1-DIO8) (Data input/output): Es un bus bidireccional de 8 líneas orientado a la transferencia de byte o de caracteres ASCII.

- Bus de sincronización de la transferencia de datos: Es un conjunto de tres líneas (DAV: Data valid, NRFD: Not Ready For Data y NDAC: Not Data ACcepted) que se utilizan de forma coordinada para asegurar la transferencia de datos entre los equipos.
- Bus de control: Está constituido por 5 líneas (ATN: ATteNtion, IFC: InterFace Clear, SRQ: Service ReQuest, REN: Remote ENable, y EOI: End Or Identify) que se utilizan para transferir comandos entre los equipos relativos al modo de interpretar los datos que se transfieren o comandos básicos de gobierno de la interfaz del bus(figura 3.2).[7]

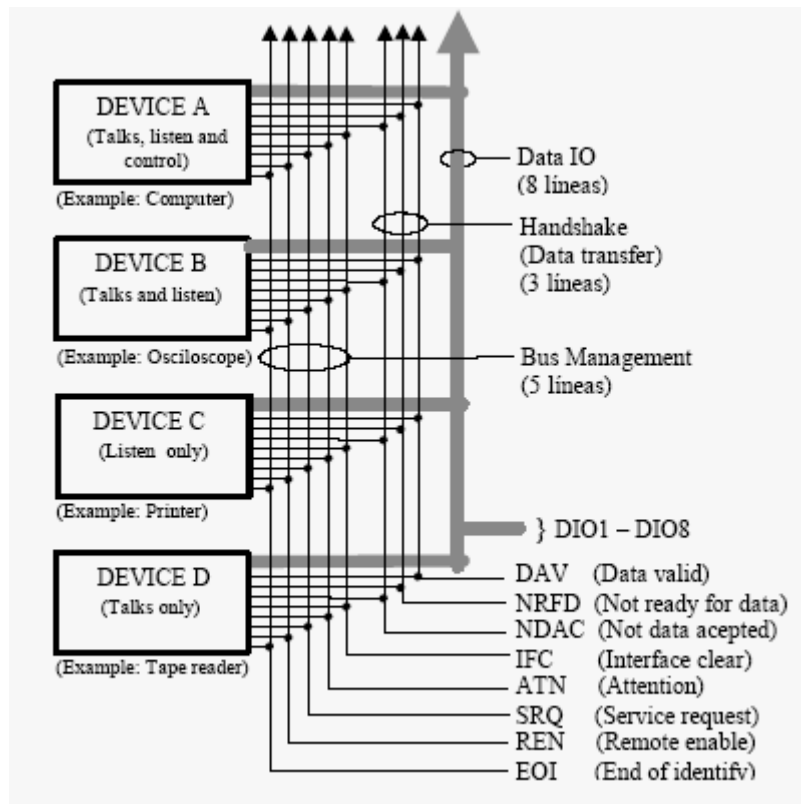


Figura 3.2: Interpretación básica de los comandos en la interfaz.

3.2.1 Tipos de mensajes que intercambian los equipos.

Entre los equipos conectados al bus GPIB se transfieren mensajes constituidos por secuencias de byte por transacción. De acuerdo con el estado de la señal de control "ATN", existen dos tipos de mensajes:

- "Data": Mensaje que contiene información relativa a la funcionalidad de un equipo. Ejemplos son: instrucción de programación, resultado de medida, estatus de un equipo, etc.
- "Commnad": Mensaje que tiene como función controlar el modo de operación del bus. Ejemplos son: Inicialización del bus, cambio del modo de operación de un equipo, transferencia del control, etc.

3.2.2 Modos de operación de un equipo.

En cada momento, unos equipos conectados al bus GPIB pueden estar operando como uno o varios de los siguientes modos de comportamiento:

- "Controller": Con capacidad de establecer quien envía o recibe datos y el modo de operación del bus (solo un equipo puede ser "controller").
- "Talker": Con capacidad de enviar datos a otros equipos.
- "Listener": Con capacidad de recibir datos de otros equipos.
- "Idler": Sin ninguna capacidad respecto del bus.

"CONTROLLER": Equipo con capacidad de transferir mensajes de tipo Command a los otros equipos. Existen dos tipos de controller:

"System Controller": Tiene capacidad hardware de tomar el control del bus en todo momento, a través de las líneas "IFC" y "REN".

En un bus sólo puede existir un único System Controller y está caracterizado por tener las capacidades hardware especial de poder establecer el estado de las líneas "IFC" y "REN".

"Active controller": Tiene la capacidad de transferir mensajes de tipo Command para:

- Establecer los modos de operación "Listener" y "Talker" en los restantes equipos.
- Enviar los comandos de inicialización y sincronización del bus.
- Supervisar mediante encuesta el status de los equipos.

Cada bus puede tener conectado uno o mas dispositivos capaces de asumir la función de "active controller", aunque en cada momento, solo uno de esos equipos puede operar como tal.[26]

En una situación estándar, hay un computador conectado al bus que actúa a la vez como "system controller" y como único "active controller". En algún momento un equipo puede requerir del controlador del sistema su interés en convertirse en controlador activo, a fin de llevar a cabo una operación compleja, tal como transferir unos datos al "plotter", o almacenar un fichero en un disco, etc.

Como respuesta a este requerimiento el controlador de sistema, transferirá el control al equipo que lo ha solicitado, el cual pasa a constituirse en controlador activo del bus. Cuando concluye su operación, retorna de nuevo el control del bus, al controlador de sistema.

Obsérvese que las capacidades propias de controlador de sistema no pueden ser transferidas. Un sistema construido sobre el bus GPIB puede ser configurado en uno de los siguientes tres modos:

- Sin "controller": En esta configuración uno de los equipos debe solo tener capacidad para actuar solo como "talker", y los restantes solo como "listener". La transferencia de datos posibles es desde el "talker" a todos los "listener" simultáneamente.
- Con "controller" único: En esta configuración las transferencias de datos posibles son: Desde el "controller" a los equipos en modo comando y datos, de un equipo al "controller" solo en modo datos, y de un equipo a otro equipo solo en modo datos.
- Con múltiples "controller": En este caso tiene las mismas capacidades que la configuración anterior, solo que en esta también es posible la transferencia entre, equipos de la capacidad de operar como "controller activo".

“TALKER”: Equipo con capacidad de enviar datos. En cada bus puede existir uno o varios equipos con capacidad de enviar datos a otros equipos por el bus, pero en cada instante sólo uno de ellos puede ser establecido por el controller para que opere como Talker. Es el único equipo (además del Active Controller) con capacidad de establecer el estado de las líneas DAV. El equipo Talker sólo puede

enviar un dato si todos los equipos que se encuentran en modo Listener esté en disposición de leerlo (Linea NRFD a valor lógico FALSE).

“LISTENER”: Equipo que recibe y lee todos los datos que se transfieren por el bus. En cada bus pueden existir uno o varios equipos con capacidad de recibir datos desde el bus, y uno o varios de ellos se puede encontrar simultáneamente en modo “Listen”. El Active Controller es el que establece a través de un comando que un equipo pasa o deja de estar en modo Listen. Todos los equipos que se encuentran en estado Listen reciben simultáneamente todos los datos que son transferidos por el bus.

“IDLER”: Estado base sin ninguna respuesta respecto del bus.[11]

3.3 Niveles y señales lógicas en el bus GPIB.

Todas las líneas del GPIB operan con niveles de tensión *TTL*, y utilizando una lógica negativa. Esto significa que un nivel de tensión inferior que 0.8 V corresponde a un estado lógico TRUE, y un nivel de tensión superior a 2.5 voltios corresponde a un estado lógico FALSE.

Las puertas de salida en cada equipo sobre una línea del bus utiliza la tecnología opencollector.

Esto hace que una línea del bus estará en estado "TRUE" lógico (tensión baja) si ese es el valor de salida que corresponde para esa línea en algún equipo. Por el contrario, una línea del bus estará en estado "FALSE" lógico solo si en todos los equipos conectados corresponde a esa línea ese valor.

Cada línea implementa una función lógica OR (Wire-OR)(figura 3.3).

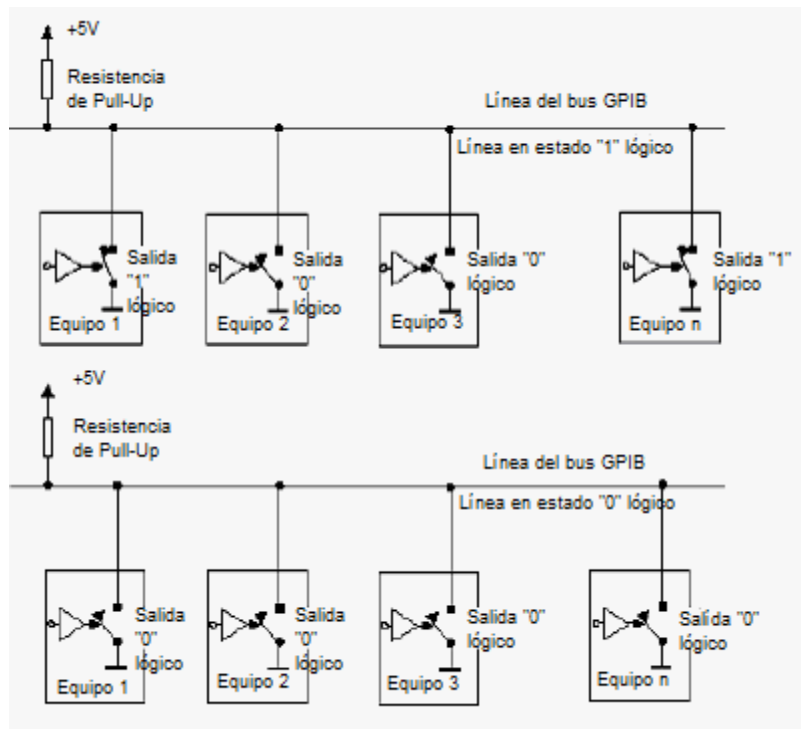


Figura 3.3: Señales lógicas del bus GPIB.

Algunas interfases GPIB, utilizan una tecnología *tri-state* para el control de las líneas del bus, pero siguiendo la misma funcionalidad que con la tecnología "open-collector". Utilizando la tecnología *tri-state* se puede conseguir mayores velocidades de transferencia de datos.

Dada la tecnología de control de líneas que se utiliza, aunque funcionalmente podrían conectarse a un bus, más de 15 equipos (siempre que algunos de ellos sean solo "Listener"), las prestaciones dinámicas del bus pueden deteriorarse con ello.[13]

3.3.1 Significado de las líneas del bus GPIB.

- DATA LINE (DIO1 - DIO8): Las 8 líneas de datos permiten que el único equipo establecido como "talker" (o en otros casos el "controller") envíe un byte en paralelo hacia todos aquellos equipos que en ese instante estén definidos como "listener".

- DATA VALID (DAV): Es una de las líneas de sincronización que permite la transferencia de datos por al bus de datos. Un TRUE lógico en esta línea significa que el equipo establecido como "talker" activo ha establecido unos datos válidos sobre el bus de datos que deben ser leídos por todos los equipos establecidos como "listener".

- NOT READY FOR DATA (NRFD): Es una de las líneas de sincronización, que es gobernada por los equipos establecidos como "listener". Cuando esta línea está en estado lógico TRUE, significa que algún equipo de entre los "listener" no está aún dispuesto para aceptar nuevos datos. El que esta línea se encuentre en estado lógico TRUE, inhibe al equipo "talker" a que inicie el envío de un nuevo dato. El que esta línea esté en estado lógico FALSE, significa que todos los equipos "listener" se encuentran a la espera de un nuevo dato.

- NO DATA ACCEPTED (NDAC): Es la tercera línea de sincronización de datos, y es gobernada por los equipos que están establecidos como "listener". Cuando se encuentra en estado lógico TRUE, significa que alguno de los equipos establecidos como "listener" aún está pendiente de leer un dato, y en consecuencia, el talker debe esperar aún para retirar los datos. Cuando esta línea se encuentra en estado lógico FALSE significa que ya todos los equipos establecidos como "listener" han leído el dato transferido.

- ATTENTION (ATN): Es una señal que establece el "controller" para establecer con un estado lógico TRUE en ella que el dato que se envía por el bus de datos es un comando enviado por el "controller". Cuando esta línea toma el estado lógico FALSE indica que el byte del bus de datos debe ser considerado como un dato.
- INTERFACE CLEAR (IFC): Está bajo el exclusivo control del "system controller". Cuando es establecido en esta línea un estado lógico TRUE, todos los equipos conectados al bus deben ser reseteados, y todos ellos deben pasar a su estado base.
- SERVICE REQUEST (SRQ): Es utilizado por los equipos conectados al bus para comunicar al "controller" que requieren ser atendidos por alguna causa (ha concluido una actividad, se ha producido un error, existe algún dato para transferir, etc.). Cuando el "controller" detecta un estado lógico TRUE en esta línea, debe iniciar una encuesta (polling) para determinar que equipo causó el requerimiento, y en el caso de que proceda, satisfacer su demanda.
- REMOTE ENABLE (REN): Es una línea con la que el "controller" al establecerla a un estado lógico TRUE, habilita a todos los equipos conectados al bus para que reciban datos, o comandos.
- END OR IDENTIFY (EOI): Esta línea tiene dos funciones:
 - a) En primer lugar, el "talker" puede comunicar, poniendo a estado lógico TRUE esta línea, que concluye su envío de datos.

b) En segundo lugar, esta línea es utilizada por el "controller" para iniciar una encuesta paralela. En este caso el "controller" debe poner simultáneamente a estado lógico TRUE las señales ATN y EOI, y como respuesta a ello, los equipos que previamente hayan sido configurados para participar en la encuesta paralela transfieren sus bit de status sobre el bus.[23]

Capítulo 4

COMANDOS.

4.1 Comandos del bus.

Los comandos de bus (mensajes Command) son siempre enviados desde el controller a los otros equipos para sincronizar su estado de operación o para establecer su estado de operación.

En los comandos de bus se envían datos por el bus de datos (si la operación lo requiere) de igual modo en la transferencia de datos, solo que en estos casos la señal ATN es establecida a estado lógico TRUE por el "controller", para indicar que es un comando, y en este caso, todos los equipos con independencia de que sean "talker" o "listener" reciben el comando manteniendo el protocolo con el "controller".

El "controller" puede enviar cinco tipos de comandos de bus a los otros equipos:

"addressed", "listen", "talk", "universal" y "secondary". Solo los 7 bits menos significativos del bus son utilizados en los comandos de bus. Los tres bit b7, b6 y b5. Definen la naturaleza de cada comando (Tabla 4.1):

b7	b6	b5	Tipo comando
0	0	0	Addressed
0	0	1	Universal
0	1	x	Listen
1	0	x	Talk
1	1	x	Secondary

Tabla 4.1: Naturaleza de cada comando.

Los equipos conectados al bus GPIB tienen asignado un código o dirección de bus comprendido entre 0-30. Este código debe ser establecido en cada equipo, bien estableciendo un conmutador hardware presentes en su panel trasero, o bien en algunos casos programando el equipo mediante su software interno. Normalmente, los cinco bits menos significativos de la línea de datos b4, b3, b2, b1, b0 se utilizan en un comando para establece a que equipos hace referencia un comando.

El código 31 ("11111") no es el código de un equipo, y suele estar reservado para hacer referencia a todos los equipos del bus en conjunto.[35]

4.2 Comandos TALK/LISTEN

A continuación se presentan los comandos Talk/Listen (tabla 4.2).

MTA	010dddd	My Talk Address: Establece el modo Talker en el equipo "dddd".
UNT	01011111	Untalk: El equipo en el modo Talker pasa a modo Idler.
MLA	001dddd	My Listen Ardes: Establece el modo Listener en el equipo "dddd".
UNL	00111111	Enlisten: Todos los equipos en modo Listener pasan a modo Idler.

Tabla 4.2: Comando Talk/Listen.

Transacción: Equipo A envía el dato 8.2 a los equipos B y C (figura 4.1).

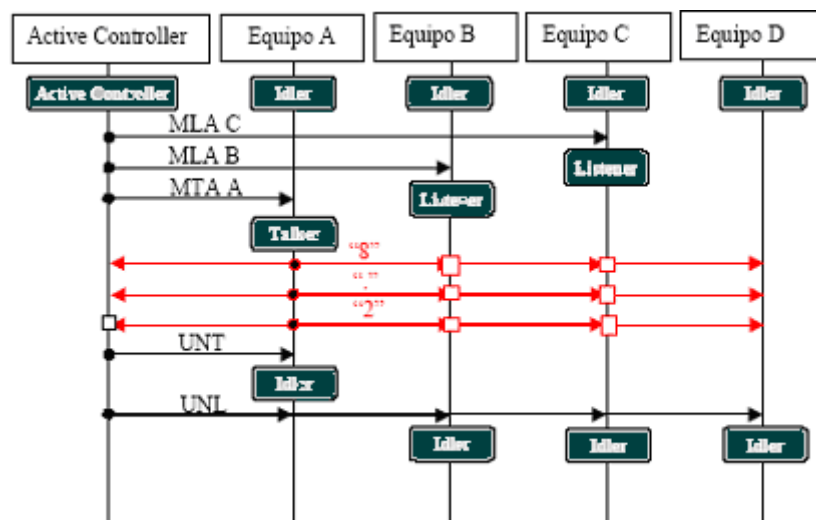


Figura 4.1: Ejemplo de la comunicación con comandos Talk/Listen.

4.2.1 Mecanismos de finalización de mensajes de datos.

Hay tres mecanismos con el que el Talker puede comunicar que el dato que transmite es el último a enviar:

- Método EOI: Cuando envía el último dato el Talker establece la línea de control del bus EOI se establece al estado lógico TRUE.
- Método EOS: El último carácter enviado por el Talker es el que previamente se ha establecido como End of Send. Habitualmente es CR (0x0D) o LF (0x0A). En este caso no es detectado por el hardware del Bus sino por el software del driver.
- Método de cuenta: El controlador para al Talker cuando el número de byte del mensaje convenido (habitualmente establecido en la cabecera del mensaje) se han enviado. El Controlador bloquea al Talker estableciendo las líneas del control del bus NRFD y NDAC al estado lógico TRUE. La cuenta de bytes se realiza por el software del driver.[29]

4.3 Comandos UNIVERSALES (UGC):

Mensajes enviados por el Controller a todos los equipos (tabla 4.3).

LLO	00010001	Local Lockout: Se deshabilitan los paneles de control de todos los equipos conectados al Bus.
DCL	00010100	Device Clear: Inicializa las interfaces hardware/software de los equipos.
PPU	00010101	Parallel Poll Unconfigure: Se cancela la programación previs de los equipos a fin de responder en la encuesta paralela.
SPE	00011000	Serial Poll Enable: Habilita a todos los equipos a fin de que respondan a la encuesta serie.
SPD	00011001	Serial Poll Disable: Deshabilita el modo de encuesta serie.

Tabla 4.3: Comandos Univerales.

Transacción: Controller realiza "Serial Poll" sobre los equipos A,B,C (figura 4.2).

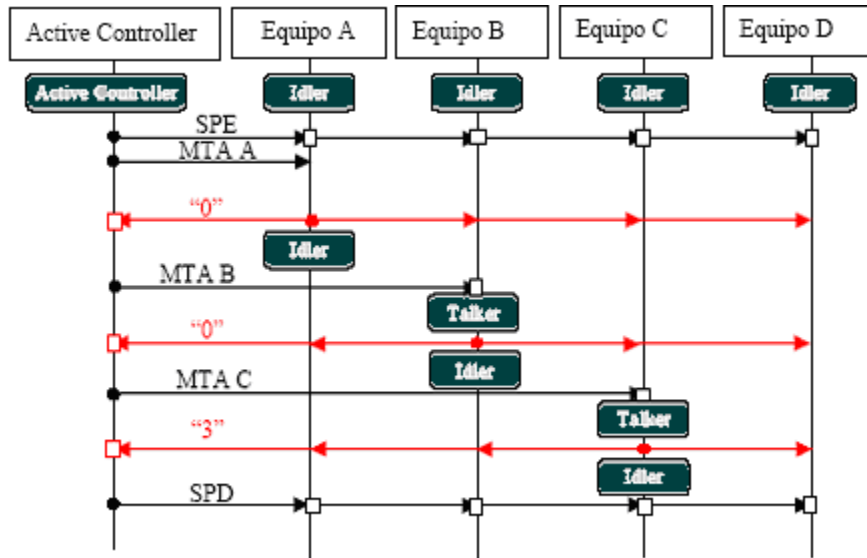


Figura 4.2: Ejemplo de la comunicación con comandos universales.

4.4 Comandos Addressed (ACG).

Van destinados y afectan únicamente a aquellos equipos que previamente han sido establecidos en "Listener" (tabla 4.4).

GTL	00000001	Go To Local: Retorna el control de los paneles a todos los equipos en estado Listener.
SDC	00000100	Select Device Clear: Inicializa las interfaces hardware/software en los equipos en estado Listener
PPC	00000101	Parallel Poll Configure: Configura la respuesta a una encuesta paralela de los equipos en estado Listener. Los equipos quedan a la espera de un comando MSA.
MSA	011xcbbb	My Secondary Address: Establece la línea (bbb) y el estado (c=0⇒requet=FALSE) con la que responden en una encuesta paralela de los equipos Listener.
GET	00001000	Group Trigger: Dispara el Trigger de los equipos en estado Listener.
TCT	00001001	Take Control: Establece como Active Controller al equipo que esta establecido como Listener.

Tabla 4.4: Comandos Addressed.

Transacción: Controller programa y realiza “Pararell Poll” sobre los equipos A,B,C (figura 4.3).

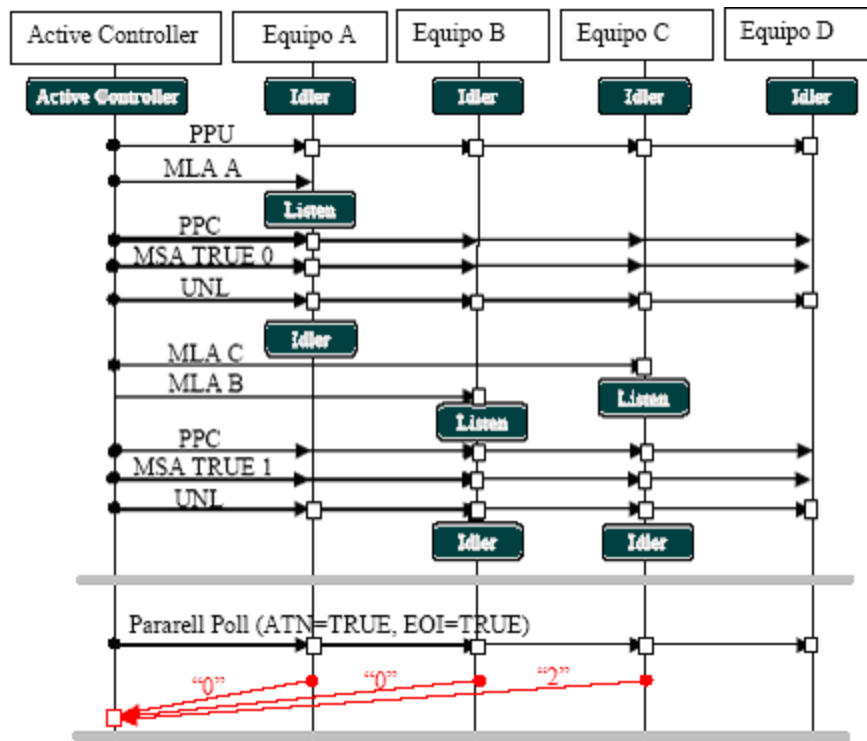


Figura 4.3: Ejemplo de la comunicación con comandos Addressed.[29]

4.5 Comandos comunes.

Los comunes son todos aquellos que son iguales para todos los dispositivos y los especiales son diferentes para cada dispositivo.

Los comandos comunes se encuentran en la tabla 4.5:

Orden	Nombre de la orden	Función
*CLS	Clear Status Command	-Despeja el registro de estado y los registros de incidencia.
*ESE	Event Status Enable Command	-Habilita bits del registro de habilitación de incidencias
*ESE?	Event Status Enable Query	-Interroga el registro de habilitación de Incidencias estándar.
*ESR?	Event Status Register Query	-Interroga el registro de Incidencias estándar.
*IDN	Identification Query	-Identifica tipo de instrumento y versión software.
*LRN?	Learn Device Setup Query	-Requiere el estado actual del equipo. -Fija el bit de "Operación completa" del registro estándar.
*OPC	Operation Complete Command	-Responde con "1" si se han ejecutado ordenes previas.
*OPC?	Operation Complete Query	-Requiere la opción instalada en el equipo.
*OPT?	Option Identification Query	-Restaura el estado del equipo del registro save/recall.
*RCL	Recall Command	-Sitúa al equipo en el estado básico de referencia.
*RST	Reset Command	-Almacena el estado actual en un registro save/recall.
*SAV	Save Command	-Habilita los bits del registro de habilitación de Byte de estado.
*SRE	Service Request Enable Command	- Requiere el contenido del registro SER de habilitación del byte de estado
*SRE?	Service Request Enable Query	- Requiere el estado del registro resumido del Byte de estado.
*STB?	Read Status Byte Query	- Arranca o dispara la operación del equipo de forma remota.
*TRG	Trigger Command	-Requiere el resultado del autotest del equipo.
*TST?	Self-Test Query	- Espera a que se realicen todas las operaciones pendientes.
*WAI	Wait-to-Continue Command	

Tabla 4.5: Comandos comunes.

4.6 Comandos SCPI.

A pesar de los estándares IEEE 488.1 y 488.2, existía libertad para que cada fabricante eligiera los comandos de sus instrumentos. En 1990 un grupo de empresas fabricantes de instrumentos acordaron crear un conjunto de órdenes con una sintaxis común, que fue llamada SCPI (Comandos Estándar para

Instrumentos Programables). Lógicamente, SCPI se construyó respetando los principios del anterior 488.2.

Si dos instrumentos (por ejemplo, dos osciloscopios), de fabricantes distintos, se adhieren al estándar SCPI, es teóricamente posible intercambiarlos con mínimas modificaciones en el programa de control.

Los comandos SCPI se escriben como texto ASCII, y tienen una estructura jerárquica por niveles, separados por dos puntos(figura 4.6):

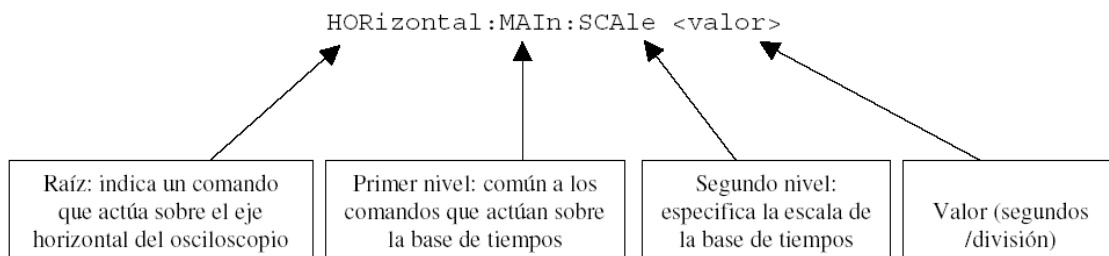


Figura 4.6: Jerarquía.

Los caracteres en mayúsculas son necesarios para especificar la orden, mientras que los que están en minúsculas pueden suprimirse, sirviendo sólo para facilitar la lectura de programas por usuario. Los comandos en sí pueden ser escritos indistintamente en mayúsculas o minúsculas. Así, SCALE, sca y scale representan todos al mismo comando.

Por ejemplo: HOR:MAIN:SCA 5.0E-4b establece la escala de la base de tiempos en 500 μ s/división. El signo de dos puntos (:) separa los niveles de la jerarquía. Si quisiéramos preguntar al osciloscopio por la escala actual de la base de tiempos habría que escribir: HOR:MAIN:SCA?

Los comandos se pueden concatenar utilizando un punto y coma (;). Por ejemplo, se puede escribir la orden: ACQuire:MODE AVE; NUMAVg 16 que indica al osciloscopio que la adquisición va a ser en modo promedio (average) y que va a constar de 16 muestras. Nótese que los niveles jerárquicos ACQuire:MODE, al ser comunes, sólo necesitan ser indicados una vez; es decir, la orden anterior equivale a: ACQuire:MODE AVE; ACQuire:MODE NUMAVg. [10]

Una ventaja del estándar SCPI es la definición homogénea de comandos para todos los aparatos de una misma clase; por ejemplo, para un osciloscopio, las principales categorías de raíz se muestran en la Tabla 4.6:

ACQuire	Adquisición de señales (mustra, promedio).
CALibrate	Calibración y diagnóstico del aparato.
CURSor	Control del cursor.
DISplay	Control de la pantalla (formato YT oXT, contraste).
Horizontal	Control de la base de tiempos (posición, escala).
MATH	Operaciones matemáticas sobre las ondas (suma, resta, FFT).
MEASUrement	Medidas sobre las formas de onda (amplitud, frecuencia).
TRIGger	Control del disparo (fuente, nivel, pendiente).
CH	Control vertical de las señales (posición, amplitud).
CURV,DAT yWFMP	Captura de ondas del osciloscopio.

Tabla 4.6: Comandos de un osciloscopio, según el estándar SCPI.

4.7 VISA e IVI.

Como se ha podido ver, la programación de un sistema GPIB a base de comandos SCPI es bastante laboriosa. Aunque hace años era la única opción disponible; hoy en día disponemos de drivers para los principales entornos de programación que permiten el acceso a los instrumentos a más alto nivel.

Históricamente, el primer paso hacia la estandarización fue VISA (Virtual Instruments Software Architecture), un convenio de Agilent y NI para acceder a los instrumentos de la misma forma independientemente de la interfaz física (GPIB, puerto serie, etc).

En 1998 surgió el consorcio IVI (Interchangeable Virtual Instruments) entre una treintena compañías, incluyendo usuarios de sistemas como Boeing, y fabricantes de hardware como Agilent, Tektronix, NI, etc., con el objetivo de alcanzar una estandarización de los drivers de los instrumentos.

En concreto, IVI aporta las siguientes novedades:

- Adopción de VISA (independizando la programación respecto al bus de interfaz).
- Posibilidad del intercambio de instrumentos, incluso de distintos fabricantes.
- Posibilidad de trabajar con instrumentos simulados durante el desarrollo de aplicaciones, cuando la disponibilidad de los equipos está restringida.
- Acceso a los instrumentos mediante una caché de estado, que optimiza el tráfico del bus cambiando el estado del instrumento sólo de forma incremental.
- Posibilidad de programación multihilo.

Los drivers IVI están clasificados en ocho clases o plantillas, correspondientes a los siguientes instrumentos de medida:

- Fuentes DC.
- Multímetros digitales.
- Generadores de funciones.
- Osciloscopios.

- Medidores de potencia.
- Generadores de RF.
- Analizadores de espectros.
- Conmutadores de señales (switches)

Con una librería IVI, el programador de C puede emplear rutinas de alto nivel sin necesidad de conocer el conjunto de comandos SCPI que el instrumento entiende.

Por ejemplo, la rutina C usada para configurar la salida de un generador de funciones de Agilent mediante el driver IVI es: `hp33120a_ConfigureStandardWaveform` (Generador, "1", `HP33120A_VAL_WFM_SINE`, amplitud, 0.00, frecuencia*1e3, 0.00); que es mucho más compacta y comprensible que la secuencia de comandos GPIB necesarios para obtener el mismo efecto. Como consecuencia, el desarrollo de aplicaciones de esta forma se ha agilizado considerablemente respecto al uso de comandos SCPI.[22]

4.8 Ejemplos de transferencias de datos por el GPIB.

Ejemplo de una transferencia de dato iniciado por el "controller" con una instrucción MTA4, seguida de la transferencia de dos datos, por parte del equipo #4, y por último de desactivación del equipo "talker mediante un comando UNT (figura 4.7).

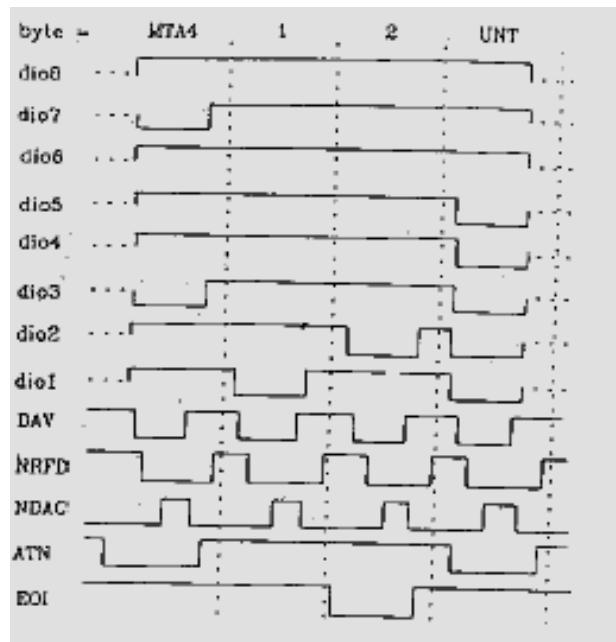


Figura 4.7: Señal de transferencia iniciada en controller.

Ejemplo de la transferencia de datos. En cada caso la transferencia se inicia con un comando MxA, y termina con un comando UNx (figura 4.8).

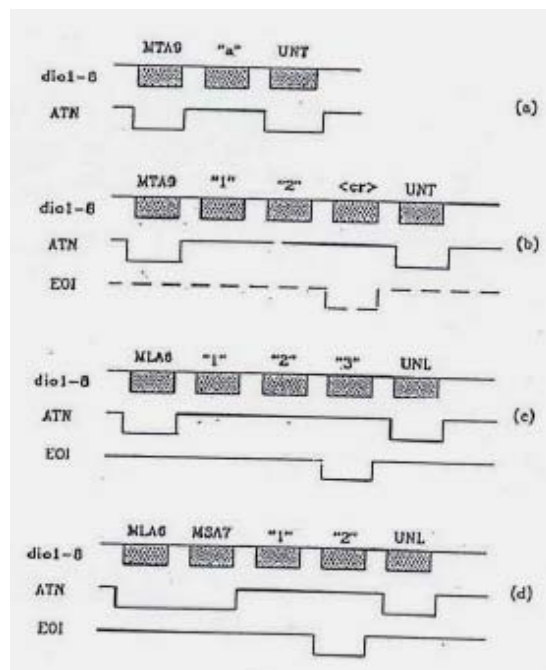


Figura 4.8: Señal de transferencia iniciada en comando MxA

4.9 Drivers.

GPIB Driver: Para poder acceder al hardware debe instalarse el driver correspondiente en el sistema operativo. Este puede ofrecer o no la interfaz VISA.

Para comprobar desde Matlab el hardware y los drivers disponibles se puede utilizar la función “instrhwinfo”. Esta orden nos devuelve la lista de las interfaces soportadas por el entorno aunque no estén instaladas en el ordenador. Para obtener la información acerca de una determinada interfaz, debemos incluir como parámetro de la orden el identificador de dicha interfaz. La información que visualiza incluye los adaptadores instalados y la sintaxis para crear los objetos Matlab de control de la instrumentación.

En el laboratorio el driver con interfaz VISA es “GPIB0”. Interface Driver Adapter: Es un objeto Matlab que permite formular la comunicación con un instrumento, y por tanto permite su manejo, a través del entorno de programación Matlab. Representa el enlace a través del que se pasa la información entre Matlab y el driver correspondiente.

- de driver DEC DC21x40 and compatible PCI controllers (including 21140 100bT cards).
- ed driver NE2000 and 1000 WD/SMC 8003, 8013 and Elite Ultra (8216) 3Com 3c503 HP 27247B and 27252A And clones of the above.
- le driver DEC EtherWORKS II and EtherWORKS III controllers.
- ie driver AT&T EN100/StarLAN 10 3COM 3c507 Etherlink 16/TP NI5210 Intel EtherExpress.

- is driver Isolan AT 4141-0 Isolink 4110.
- el driver 3com 3c501 (does not support Multicast or DMA).
- eg driver 3com 3c505 Etherlink/+.
- ze driver IBM PCMCIA credit card adapter.
- Inc drive Lance/PCnet cards (Isolan, Novell NE2100, NE32-VL)(*).
- ep driver 3com 3c509 (Must disable PNP support on card).
- cx driver Cronyx/Sigma multiport Sync/Async (Cisco and PPP framing).
- zp driver 3Com PCMCIA Etherlink III (aka 3c589)(A-C only).
- fea driver DEC DEFEA EISA FDDI controller.
- fpa driver DEC DEFPA PCI FDDI controller.
- fe driver Fujitsu MB86960A/MB86965A Ethernet cards.

NOTA: Los drivers marcados con (*) pueden tener problemas.

Aquí hay una lista de drivers que no se han podido catalogar en las categorías anteriores.

- b004.c Driver for B004 compatible Transputer boards.
- ``ctx" driver Driver for CORTEX-I Frame grabber.
- ``gp" driver Driver for National Instruments AT-GPIB and AT-GPIB/TNT boards.
- ``pca" driver Driver for PC speakers to allow the playing of audio files.
- ``spigot" driver Driver for the Creative Labs Video Spigot.
- gsc driver Driver for the Genuis GS-4500 Hand scanner.
- joy driver Driver for a joystick.
- labpc driver Driver for National Instrument's Lab-PC and Lab-PC+.
- ``uart" driver Stand-alone 6850 UART for MIDI.
- psm driver PS/2 mouse port .
- tw.c Driver for the X-10 POWERHOUSE

Capítulo 5

INTERFACES CON EL BUS GPIB

5.1 Componentes de una interfaz.

Para el control desde un computador de un sistema constituido por un conjunto de equipos todos ellos interconectados a través de un bus GPIB, se necesita incorporar al computador varios componentes hardware y software (figura 5.1):

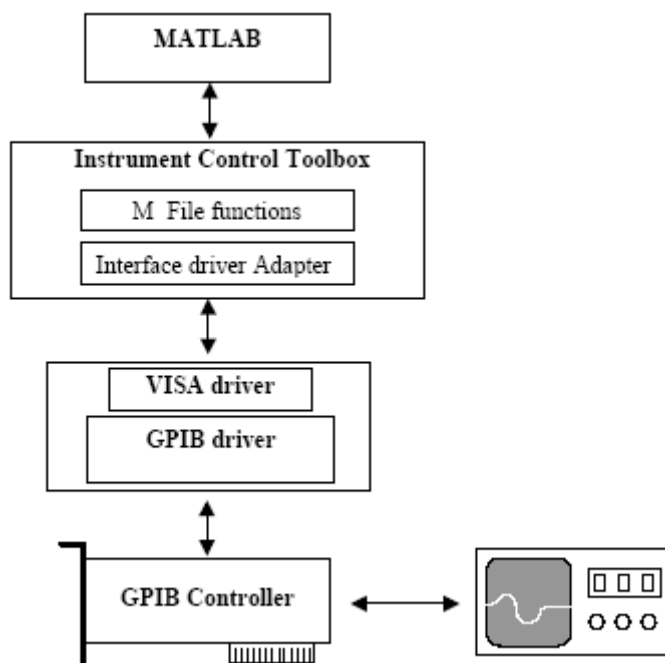


Figura 5.1: Componentes de interfaz.

GPIB Controller: Es la tarjeta hardware incorporada al computador, con capacidad de controlar, leer y escribir sobre las líneas físicas que constituyen el bus GPIB.

Se utilizan dos tipos de tarjetas:

- Tarjeta 82350 B PCI (Agilent): Es un controlador para PCI que soporta Plug&play.

Requerimientos mínimos del sistema: Windows 95/98 /Me/2000/XP, Pentium -200, 32 MB RAM, 50 MB de espacio libre de disco, PCI port . Estandars que soporta: IEEE-488.1 y IEEE-488.2 compatible, SICL yVISA 2.2.

- Tarjeta HP-82335 (Hewlett-Packard): Interface para bus ISA que no soporta Plug&Play ni la interfaz VISA. Configuración: Antes de ser instaladas, esta tarjeta debe ser configuradas, a fin de que ocupen zonas de memoria independientes y de que no se solapen con las zonas de memoria utilizadas por otros periféricos. Esta configuración se realiza mediante una combinación adecuada en un conmutador tipo DIP interno que posee la tarjeta. Como se indica en la figura 5.2, a cada combinación válida se le asocia un código que identifica a la tarjeta y que se denomina SELECT_CODE (Sel_code). Este código es la clave a través del que el driver identifica la interfaz sobre la que debe ejecutar las funciones que se programen. En nuestro caso, en cada computador existe una sola, interfaz, que siempre se configura con "SELECT_CODE = 7".[31]

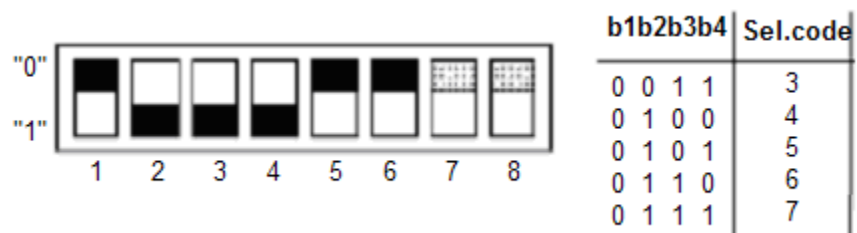


Figura 5.2: Bits en una interfaz.

GPIB Driver: Para poder acceder al hardware debe instalarse el driver correspondiente en el sistema operativo. Este puede ofrecer o no la interfaz VISA.

Para comprobar desde Matlab el hardware y los drivers disponibles se puede utilizar la función “instrhwinfo”. Esta orden nos devuelve la lista de las interfaces soportadas por el entorno aunque no estén instaladas en el ordenador. Para obtener la información acerca de una determinada interfaz, debemos incluir como parámetro de la orden el identificador de dicha interfaz. La información que visualiza incluye los adaptadores instalados y la sintaxis para crear los objetos Matlab de control de la instrumentación.

En el laboratorio el driver con interfaz VISA es “GPIB0”.

Interface Driver Adapter: Es un objeto Matlab que permite formular la comunicación con un instrumento, y por tanto permite su manejo, a través del entorno de programación Matlab. Representa el enlace a través del que se pasa la información entre Matlab y el driver correspondiente.

Cada objeto está asociado a un controlador y a un solo instrumento, por lo que será necesario crear uno por cada instrumento que forme parte del sistema.

5.2 Fases de una sesión de comunicación con un instrumento.

Las fases que se siguen durante toda sesión de comunicación con un instrumento son las siguientes:

1. Creación de los objetos GPIB o VISA

Ejemplo: Creación de un objeto GPIB de un objeto Matlab para controlar un equipo con el address 15.

```
generador= gpib('agilent',7,15);
```

Creación de GPIB object:

```
obj= gpib('vendor', boardindex, primaryaddress)
```

siendo: obj: Objeto Matlab GPIB.

'vendor': Nombre del vendedor del driver {'agilent', 'cec', 'iotech',
'keithley', 'mcc', 'ni'}

boardindex: El índice de la tarjeta GPIB

primaryaddress: El GPIB address primario {1 – 30}

Ejemplo: Creación de un objeto VISA para controlar por GPIB un generador que tiene asignado el address 15 en el bus.

```
generador= visa('agilent', GPIB0::15::INSTR)
```

Creación de un VISA object:

```
obj= visa('vendor', rsrcname)
```

siendo: obj: Objeto Matlab VISA.

'vendor': Nombre del vendedor del driver {'agilent', 'ni', 'tec'}

rsrcname: El nombre del recurso VISA que controla el instrumento:

GPIB[board::primary_address::[secondary_address]::INSTR

2. Configuración de las propiedades del GPIB Object y VISA Object

Propiedades de GPIB Object y VISA Object:

Todo objeto GPIB o VISA tiene un conjunto de propiedades que pueden ser utilizadas, bien para leer su estado o bien para configurar el modo de comunicación con los instrumentos.

- BytesToOutput: Indica el número de bits salen del buffer.
- OutputBufferSize: Tamaño específico del buffer de salida en bytes.
- Timeout: El tiempo de espera para la completa lectura o escritura de la operación.
- TransferStatus: Indica si la operación asincrónica de lectura o escritura esta en proceso.
- ValuesSent: Indica el total número de valores escritos en el instrumento.
- BytesAvailable: Indique el número de bytes disponibles en el buffer de entrada.
- InputBufferSize: Tamaño específico de entradas al buffer en bytes.
- ValuesReceived: Indica el número total de valores leídos en cada instrumento.
- BytesAvailableFcn: Especifique la función de callback (retrollamada) de M-file (archivo M) para ejecutar cuando un número especificado de bytes esta disponible en el buffer de entrada.
- BytesAvailableFcnCount: Especifique el número de bytes que deben estar disponibles en el buffer de entrada para generar un evento disponible de bytes.
- BytesAvailableFcnMode: Especifique si el evento disponible de bytes es generado después de que un número especificado de bytes está disponible en el buffer de entrada, o después de que un terminador es leído.

- ErrorFcn: Especifique la función de callback (retrollamada) de M-file (archivo de M) para ejecutar cuando un evento de error ocurre.
- OutputEmptyFcn Especifique la función de callback (retrollamada) de M-file (archivo de M) para ejecutar cuando el buffer de salida esta vacío.
- TimerFcn: Especifique la función de callback (retrollamada) de M-file (archivo de M) para ejecutar cuando un período predefinido de tiempo pasa.
- TimerPeriod: Periodo específico de tiempo entre los eventos.
- RecordDetail: Especifique la cantidad de información guardada a un archivo de registro.
- RecordMode: Especifique si los datos y la información del evento son guardados a un archivo de registro o a múltiples archivos de registro.
- RecordName: Nombre específico del archivo grabado.
- RecordStatus: Indica si el dato y la información del evento han sido guardados correctamente.

Propiedades con un Proposito General

- ByteOrder: Especifique la orden en la cual el instrumento almacena bytes.
- Name: Especifique un nombre descriptivo para el objeto del instrumento.
- Status: Indica si el objeto del instrumento esta conectado a otro instrumento.

- Tag: Indique si la etiqueta esta asociada al objeto del instrumento.
- Type: Indique el tipo de objeto.
- UserData: Especifique datos que usted quiere asociar con un objeto de instrumento. Propiedades específicas.
- BoardIndex (GPIB y VISA): Especifique el número de índice de la tabla de GPIB.
- BusManagementStatus (GPIB): Indique el estado de las líneas de dirección del bus GPIB.
- CompareBits (GPIB): Especifique el número de bits que deben igualar al carácter EOS para completar una operación leída, o afirmar la línea EOI.
- EOIMode (GPIB y VISA): Especifique si la línea EOI es afirmada al final de una operación escrita.
- EOSCharCode (GPIB y VISA): Especifique el carácter EOS.
- EOSMode (GPIB y VISA): Especifique cuando el carácter EOS es escrito o leído.
- HandshakeStatus (GPIB y VISA): Indique el estado de las líneas handshake de GPIB.
- PrimaryAddress (GPIB y VISA): Especifique la dirección primaria del instrumento GPIB.

- SecondaryAddress (GPIB y VISA): Especifique la dirección secundaria del instrumento GPIB. Las propiedades de un objeto pueden leerse desde un programa Matlab mediante las sentencias:

- ❖ `Get(generator, 'BoardIndex')` .

- ❖ `Generator.BoardIndex`.

Así mismo, las propiedades de un objeto pueden establecerse mediante las sentencias:

- `Set(generator, 'Timeout', 5.0)` .

- `Generator.Timeout= 5.0`.

Matlab ofrece una aplicación para visualizar/cambiar las propiedades del un objeto GPIB o VISA de forma interactiva. Para ello seleccionar el objeto en el Workspace Browser e invocar la herramienta Explore-> Call Property Inspector (figura 5.3):

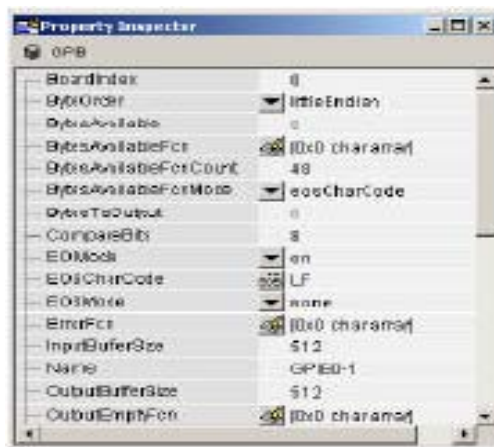


Figura 5.3: Propiedades de un objeto.

Valores por defecto Todas la variables tienen definido un valor por defecto, por lo que si no son establecidas en el proceso de configuración, son estos valores los que permanecen.

Los valores por defecto de algunas propiedades importantes son:

- ByteOrder: [littleEndian | bigEndian].
- EOIMode: [on | off].
- EOSMode: [none | read | write | read&write].

5.2.1 Conexión del objeto al instrumento.

Una vez que se ha creado el objeto, debemos realizar la conexión del mismo con el instrumento al que corresponde. Este paso se realiza mediante la sentencia:

- `fopen(generator)`

Esta sentencia cambia el estado de la propiedad `generator.Status` de 'closed' a 'open'.

Hay que tener cuidado con el momento en que realizamos la conexión, pues existen propiedades del objeto, tales como `InputBufferSize` or `OutputBufferSize`, que solo se pueden establecer antes de la llamada a `fopen`.

5.2.2 Comunicación con un instrumento VISA o GPIB.

Comunicar con un instrumento implica transferir o recibir mensajes ASCII o bloques de datos binarios con el instrumento. Esto se realiza a través de un conjunto de funciones que ofrece Matlab y que se aplican al objeto que constituye el adaptador del instrumento.[18]

La comunicación puede realizarse:

- Sincronamente: La función que invoca la comunicación se suspende hasta que esta concluye y luego retorna el control.
- Asíncronamente: La función retorna inmediatamente el control al programa sin que su efecto haya concluido. Posteriormente se accede de nuevo al objeto para verificar el estado de aquella, o se responde mediante funciones “callback” a los eventos que proceden de los instrumentos. Esta forma de comunicación solo tiene sentido si se realiza un estrategia de programación concurrente. No la utilizaremos en este curso.

5.2.3 Desconexión y destrucción de un GPIB object y VISA object.

Cuando se haya concluido la comunicación con un instrumento dentro de un programa, el adaptador debe ser cerrado:

- `fclose(generator).`

Esta sentencia cambia el estado de la propiedad `generator.Status` de ‘open’ a ‘closed’

El objeto se destruye y la variable se elimina de memoria con las sentencias:

- delete(generator).

- clear generador.

5.3 Software existente para GPIB. Programas propietarios.

La primera alternativa (y, en ocasiones, la más práctica), es el uso de software propietario desarrollado por los mismos fabricantes del instrumento. Por ejemplo, el programa gratuito Intuilink de Agilent Technologies, o el programa de pago Wavestar para los osciloscopios de Tektronix Inc. La ventaja evidente de estos programas es que pueden ser empleados nada más conectar los instrumentos, y proporcionan ya hechas las funciones más comunes que uno puede desear realizar, sin necesidad de programar.

Las desventajas son también claras: por tratarse de software cerrado, sólo puede ser usado para la tarea para la que fue diseñado, y además son imposibles de integrar con otros programas.[19]

5.3.1 Labwindows/CVI y LabVIEW de National Instruments.

LabWindows/CVI es un entorno de desarrollo completo basado en ANSI C (el nombre significa C para Instrumentación Virtual). Los aspectos más destacables de este producto son:

- Completas librerías para la comunicación entre dispositivos (puerto serie, paralelo, GPIB, TCP/IP, etc).

- Especial facilidad para el desarrollo de interfaces gráficas adaptadas a los instrumentos de medida (dispone de elementos para mostrar formas de onda, conmutadores, potenciómetros, etc).
- Soporte para los drivers IVI de numerosos instrumentos (extensión .fp).
- Posibilidad de ofrecer el programa final desarrollado mediante una aplicación de instalación.

National Instruments (<http://www.ni.com>), creadores de LabWindows/CVI, ofrecen también LabVIEW, que, con la misma funcionalidad, está orientado a programación gráfica en vez de al desarrollo de código C. NI mantiene la compatibilidad con otros compiladores de propósito general, como las distintas suites de Microsoft Visual Studio (Visual Basic, Visual C++, .NET, etc), a través de su producto Measurement Studio.

5.3.2 Matlab con Instrument Control Toolbox.

Matlab, que fue en origen un conjunto de rutinas para manipulación de matrices, ha evolucionado con el tiempo para convertirse en un entorno de programación de propósito general con gran potencia matemática y aplicabilidad en muchos ámbitos de la ciencia y la ingeniería, gracias a sus módulos de extensión (toolboxes) de procesamiento de señales, control, ecuaciones diferenciales, y un largo etcétera. En muchos casos las toolboxes representan el estado del arte en programación numérica en sus respectivos campos.

Recientemente, los creadores de Matlab, Mathworks (<http://www.mathworks.com>) emprendieron una línea clara de ampliación de su mercado hacia la conexión de

hardware con el PC, distribuyendo toolboxes para el control de tarjetas de adquisición, generación de código y emulación de DSPs, xPC (control remoto de PCs para operación en tiempo real), adquisición de imágenes, etc. La primera versión de la toolbox específica de control de instrumentos de medida apareció en 2001, y la versión 2.1 disponible en la actualidad (2004) está suficientemente depurada para ser competitiva con productos más maduros como CVI.

La toolbox ha sido diseñada de forma que el acceso a los instrumentos de medida se realiza mediante objetos, que pueden ser de dos clases: interfaz o de dispositivo. Mientras que el primer método equivale a la programación con comandos SCPI, el segundo enfoque permite explotar toda la potencia de IVI. Para ello es preciso disponer del driver de Matlab correspondiente (extensión .mdd), obtenible en la página de Mathworks o bien usar una utilidad llamada makemid para convertir (o “envolver”) el driver IVI ya instalado en el sistema. La utilidad midedit permite ver y modificar todos los objetos definidos en los aparatos, sus propiedades, y las funciones ejecutables.

La figura 5.4 muestra el árbol con los objetos predefinidos por el driver IVI sobre un osciloscopio Tektronix y un ejemplo de código para la captura de una forma de onda desde el mismo.

5.4 El futuro de la conexión de instrumentos de medida.

Las dos tecnologías de comunicación con instrumentos más comunes en la industria, GPIB y el puerto serie, provienen de los años 70; es claro que desde entonces la industria informática ha experimentado cambios radicales. ¿Existe alguna tecnología que pueda desbancar al GPIB en el futuro?

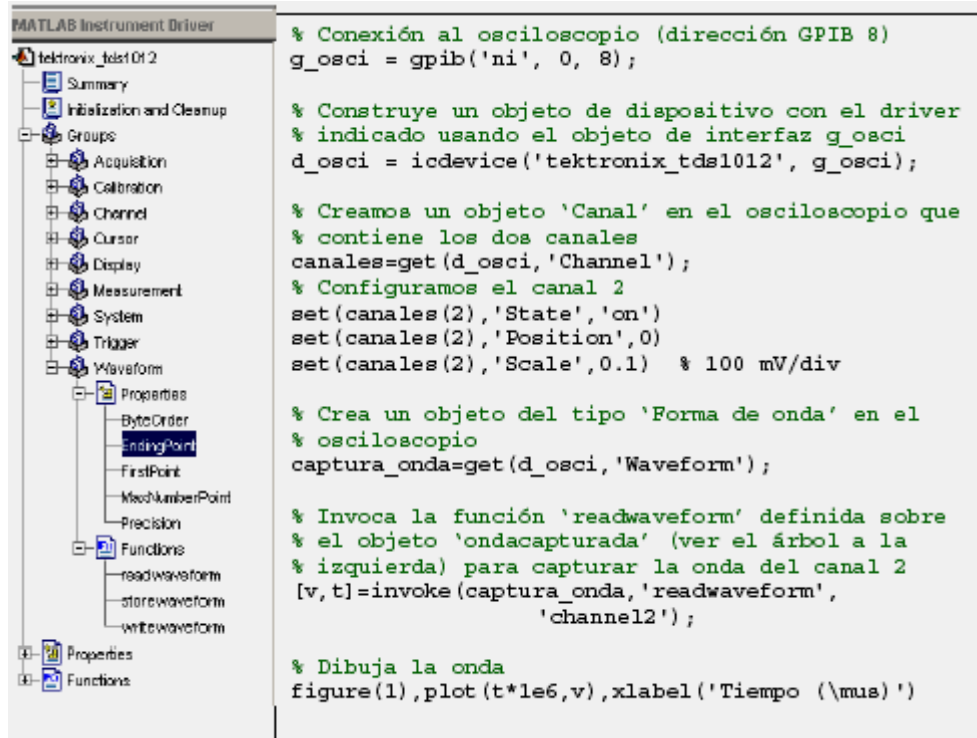


Figura 5.4: Estructura del driver IVI de un osciloscopio.

5.4.1 USB: Universal Serial Bus.

El USB fue creado como puerto para la conexión de periféricos (impresoras, cámaras digitales, unidades de disco, escáners, etc) a los PCs. En el diseño de USB se optó por plug and play, de forma que el PC reconozca y configure los dispositivos en el momento de su conexión. Permite la conexión simultánea de hasta 127 dispositivos en un puerto, con una velocidad de transferencia de datos de hasta 60 Mbytes/s (estándar USB 2.0). USB es un sistema de comunicación barato y está implementado en cualquier PC moderno.

Sin embargo, los cables USB no están preparados para entornos industriales (con posible pérdida de datos ante el ruido electromagnético), carecen de un mecanismo de enganche al PC, y la distancia de conexión está limitada a 30 m.

Además, no existe un protocolo estándar sobre USB: cada fabricante debería desarrollar el suyo propio.

5.4.2 Ethernet.

Ethernet es una tecnología de conexión de ordenadores madura y de amplia implantación. Las ventajas son su ubicuidad, la posibilidad de control remoto, y la facilidad para compartir los instrumentos entre varios usuarios. La velocidad de transmisión de datos tiene un máximo teórico entre 10, 100 o 1000 Mbits/s (estándar Ethernet Gigabit), aunque hay que considerar que raramente es alcanzable en condiciones normales, siendo muy dependiente del tráfico de la red. Una desventaja importante inherente al diseño de Ethernet es que el orden de llegada de los datos no es determinista. También se puede plantear la seguridad global del sistema, y, como factor psicológico en contra el que los ingenieros tuvieran que recurrir a los administradores de red para montar su sistema. Otra alternativa interesante es la Ethernet inalámbrica (IEEE 802.11).

5.4.3 Firewire (IEEE 1394).

Diseñado por Apple, y estandarizado por el IEEE 1394 en 1995, Firewire fue concebido para la transmisión masiva de datos en entornos multimedia (por ejemplo, cámaras de vídeo digitales), alcanzando velocidades de 400 Mbits/s (el siguiente estándar establecerá velocidades de hasta 3.2 Gbits/s). Firewire soporta hasta 16 dispositivos en cadena, pudiendo estar separados mutuamente un máximo de 4.5 m de distancia.

Al contrario que USB, existe un protocolo definido para el control de instrumentos sobre IEEE 1394; sin embargo, pocos instrumentos de medida lo reconocen. Dos inconvenientes para la implantación de este estándar son la ausencia de los

chipsets correspondientes en las placas base de los PCs, por lo que el usuario debe adquirir una tarjeta PCI por separado (Macintosh la incorporan de fábrica) y que los cables Firewire no están inmunizados al ruido, siendo por tanto no aptos para entornos industriales.

Capítulo 6

INSTALACIÓN Y CONFIGURACIÓN DE LA TARJETA ALIGENT 82350B PCI GPIB INTERFACE Y COMUNICACIÓN CON INSTRUMENTOS.

6.1 Introducción.

Es muy importante configurar e instalar adecuadamente una interfase ya que si no se hace correctamente se pueden tener muchos errores a la hora de querer utilizar esa tarjeta o simplemente no la reconoce; esto pasa con cualquier tarjeta.

La forma de configurar es un poco diferente de acuerdo a cada fabricante, cualquier tarjeta se instala igual ya que el CPU cuenta con diferentes ranuras para varios tipos de tarjeta, solo se debe conocer que tipo es para conectarla correctamente.

6.2 Pasos para la instalación.

Se instala un poco diferente cada tarjeta de acuerdo al programa que vayamos a utilizar ya sea especiales para tarjetas de National Instruments o de HP, aquí utilizaremos la tarjeta Agilent 82350B PCI GPIB Interface.

Se van a realizar cinco pasos para esta práctica (figura 6.1):

Paso 1: Se revisan las especificaciones de la PC.

Paso 2: Se instalan librerías de la tarjeta.

Paso 3: Se conectan los instrumentos.

Paso 4: Se configura la tarjeta.

Paso 5: Se comunica con los instrumentos.

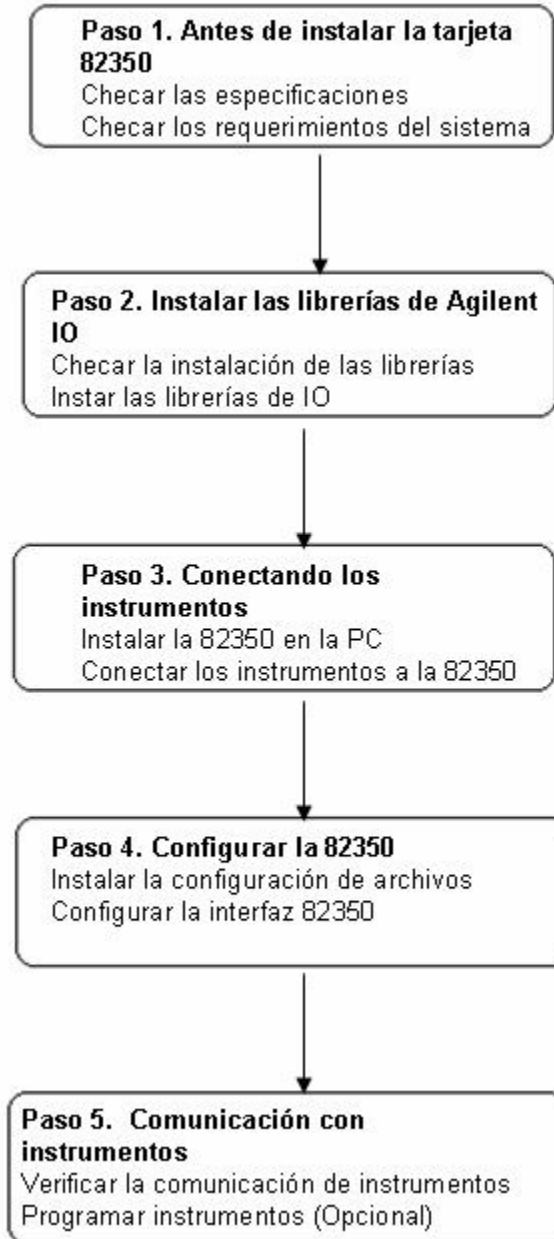


Figura 6.1: Pasos.

Paso 1. Se debe revisar la PC antes de empezar, ya que esta necesita tener ciertas especificaciones: (tabla 6.1)

Requerimientos mínimos en Hardware	
Memoria	100MHz Pentium II (Recomendación 800 MHz) y 64MBytes
Disco Duro	215MB para instalación: 160MB Microsoft.NET, 55MB para las librerías Agilent IO 125MB para la operación: 70MB para Microsoft.Net, 55MB para las librerías de Agilent IO
PCI Bus	De 5V mínimo ya que la tarjeta 83350B es de 5V
Requerimientos mínimos en Software	
Sistema Operativo	Windows 98/Me/2000/XP Nota: La tarjeta de interfaz 82350B se puede instalar en Windows NT 4.0, pero las librerías no. Si utiliza este sistema puede bajar las librerías M.01.01 de Agilent de la siguiente página: http://www.agilent.com/find/iolib .

Tabla 6.1: Requerimientos.

Paso 2. Para instalar las librerías Suite de Agilent:

1. Se desconecta cualquier hardware que utilice el puerto USB y FireWire-VXI.
2. Se inserta el CD-ROM. Se esperan unos segundos y se abrirá el asistente de instalación "Agilent IO Libraries Suite 14.0".
3. Si Windows no lo abre automáticamente
 - Se oprime **start | run**.
 - Escribir <drive>:autorun\auto.exe, donde <drive> es la unidad del lector de CD.

- Después de la instalación aparecerá un icono en la barra de tareas (IO).

Al oprimir ese icono aparecerá todo sobre el control de Agilent IO.

El siguiente paso es insertar la tarjeta en el CPU.

1. Se abre este y se procede a buscar una entrada PCI y se quita la placa para la entrada del bus.
2. Debemos asegurarnos que cerremos apropiadamente el CPU.

Paso 3. Después se conectan los instrumentos que se deseen como en la figura 6.2:

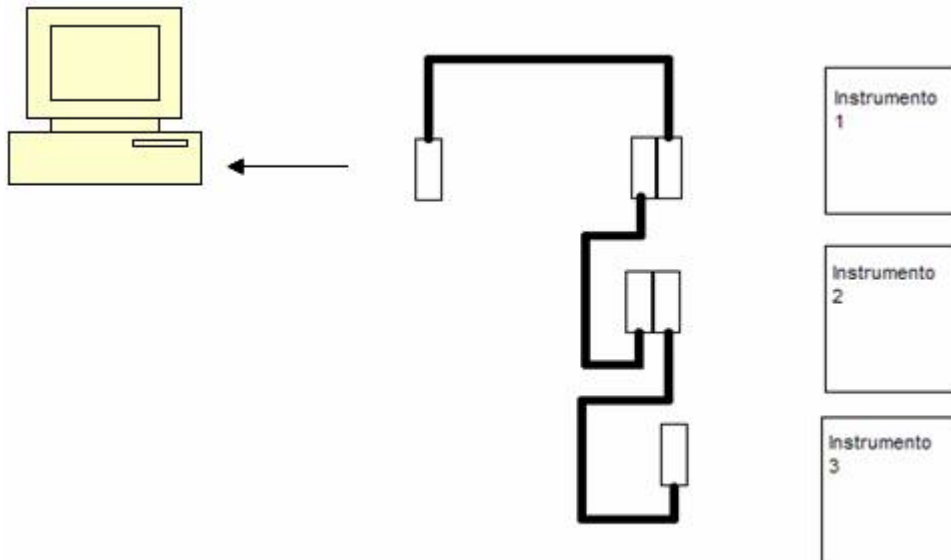


Figura 6.2: Conectores.

1. Se procede a encender la PC y aparecerá un cuadro de dialogo que dice que hay un nuevo hardware encontrado.
2. Se instala.

Paso 4. Se configura la interfaz.

1. Para configurar la interfaz, se oprime el icono que apareció en la barra de tareas “IO” y aparecerá otro cuadro de dialogo “Agilent Connection Expert” (tabla 6.2).
2. Vamos a seleccionar cambiar propiedades y debe de aparecer la Información de la tarjeta que acabamos de instalar.
3. Después se configura la tarjeta GPIB.

Parámetros	
VISA Interface ID	Aquí depende de las tarjetas que se instalen de GPIB ya que si es la primera aparecerá GPIB0, la segunda GPIB1 y así sucesivamente.
GPIB Ardes	Es la dirección del control de la interfaz. Usualmente es 21. Pero puede variar el rango entre 0 – 30.
System Controller	Esta determina si la interfaz puede enviar y recibir datos.
SICL Interface ID	Aquí debe ser muy parecido que en el primer parámetro ya que también es la identificación de la tarjeta, pero esta vez con minúsculas, por ejemplo: gpib0, gpib1.
Logical Unit	Debe de ser el número de SICL este en único de acuerdo a la tarjeta y el rango puede variar entre 0 – 1000.

Tabla 6.2: Parámetro.

Nota: Si se desea instalar otra tarjeta se repiten los pasos.

Paso 5. Aprenderemos a comunicar los instrumentos con la PC, para esto necesitamos programación y podemos utilizar los comandos de VISA, VISA COM, SICL, o SCPI.

1. Se oprime en donde esta el icono IO en “interantive IO”.
2. Se escribe y envía uno de los comandos comunes *IDN (Identificación) y se oprime el botón query (o de enviar y recibir).
 - 2.1. Recibiremos una respuesta, la cual va a ser el nombre del instrumento conectado.
3. Se selecciona uno de los instrumentos conectados y se le envían algunos comandos específicos del instrumento que estamos utilizando y debemos recibir los datos deseados.
4. Para programar los instrumentos se pueden utilizar varios lenguajes de programación tales como Visual Basic, Visual C++, y Agilent VEE y también se pueden utilizar como el Labview de National Instruments, pero por ser de diferentes compañías se tiene que configurar un poco diferente.[1]

Ejemplo: ¿Cómo comunicarnos con una fuente de poder?

1. Conectar una fuente de poder de corriente directa, (Aligent Technologies E364xA).
2. Oprimir el icono IO
3. Aparece un cuadro de diálogo
4. Escribir *IDN
5. Oprimir botón Query
6. Aparece en el cuadro de diálogo las características de la fuente de poder
7. Escribir OUTP ON

8. Se habilitará la fuente
9. Escribir VOLT 5.0
10. Aparecerá el valor de 5 Volts en la fuente
11. Escribir CURR 1.0
12. Aparecerá el valor de 1 Ampere en la fuente
13. Escribir OUTP OFF
14. Se deshabilitará la fuente
15. Si desea realizar alguna otra función, checar el manual de usuario de la fuente de poder, para conocer más comandos específicos.

¿Cómo asignar direcciones GPIB a cada uno de los instrumentos?

- Generador de Funciones Sony-Tektronik AFG310
Shift+System(blue)
GPIB Address: 2.
- Multímetro HP-34401A
Shift+On/Off Menu <I/O Menu> 1:GPIB ADDR
(GPIB address is 3).
- Tektronik TDS 210 Dos canales digitales en tiempo real, osciloscopio.
Utility--Opciones--Configurar GPIB--Dirección--Recibe/Envía o
Desconectado.
- Thrurlby Thandar Instruments PL330TP
Por hardware de un conjunto de 8 jumpers en la parte trasera de la fuente programable (ver p.12 del manual de instrucciones".
p.12-instruction manual "The address is selected by the set of 8 dip switches on the rear panel".
p.12-instruction manual "If the REMOTE/LOCAL switch is in the REMOTE position the front panel controls will be inoperative and the instrument will respond to remote commands."

➤ Power Supply HP E3631A

I/O Config + HP-IB/488 haciendo girar el mando de control hacia la derecha + ADD 4 (para modificar la dirección se gira el mando de control hacia la dcha. o izda.) + salvar cambio -CHANGE SAVED

Como resultado obtenemos la comunicación de cualquier instrumento que utilice el bus GPIB con nuestra PC.

En el ejemplo con la fuente de poder obtuvimos 5 Volts y 1 Ampere que fueron los valores que enviamos desde la PC.

También habilitamos y deshabilitamos la fuente.

La descripción de la práctica es esencial, casi cualquier tipo de tarjeta se instala de manera similar a otras tarjetas. Lo importante es conocer las opciones para poder comunicar la PC con los instrumentos, ya que si utilizamos el programa de comunicación de la misma compañía de Agilent, es sencilla la instalación, pero si tenemos el programa de Nacional Instruments, la instalación tiende a complicarse, ya que utilizaremos programas como Labview, MatLab, C++, etc., con lo que tenemos que aprender a utilizar las herramientas VISA. [2]

Capítulo 7

COMO REALIZAR PRÁCTICAS DE LABORATORIO UTILIZANDO LA COMUNICACIÓN GPIB.

7.1 Adquisición de voltaje.

LabVIEW constituye un revolucionario sistema de programación gráfica para aplicaciones que involucren adquisición, control, análisis y presentación de datos. Las ventajas que proporciona el empleo de LabVIEW se resumen en las siguientes:

- Se reduce el tiempo de desarrollo de las aplicaciones al menos de 4 a 10 veces, ya que es muy intuitivo y fácil de aprender.
- Dota de gran flexibilidad al sistema, permitiendo cambios y actualizaciones tanto del hardware como del software.
- Da la posibilidad a los usuarios de crear soluciones completas y complejas.
- Con un único sistema de desarrollo se integran las funciones de adquisición, análisis y presentación de datos.
- El sistema está dotado de un compilador gráfico para lograr la máxima velocidad de ejecución posible.

- Tiene la posibilidad de incorporar aplicaciones escritas en otros lenguajes.

Utilizaremos una herramienta llamada VISA (Virtual Instrument Estándar Architecture): Especificación definida por Agilent y National Instrument es una librería que puede ser usada para desarrollar aplicaciones y drivers de I/O de forma que software de diferentes empresas puedan trabajar conjuntamente sobre el mismo sistema y que pueden ser instalados en conjunción con drivers VXIplug&play utilizando simultáneamente varios medios de comunicación (GPIB,VXI, RS232, LAN, etc) y en aplicaciones desarrolladas con diferentes lenguajes (C, C++, VisualBasic, etc.).

Esta herramienta se utiliza ya que debemos de saber que existen dos grandes compañías una es National Instruments y la otra es Agilent, cada una de éstas tiene su equipo y sus programas y cuando se trabaja con instrumentos y programas de la misma compañía es muy fácil hacer la comunicación, pero si se trabaja con el equipo de una compañía y el programa de otra, la comunicación se complica, pero no es imposible.[25]

Se realiza el diagrama de flujo antes de empezar con cualquier práctica (figura 7.1)

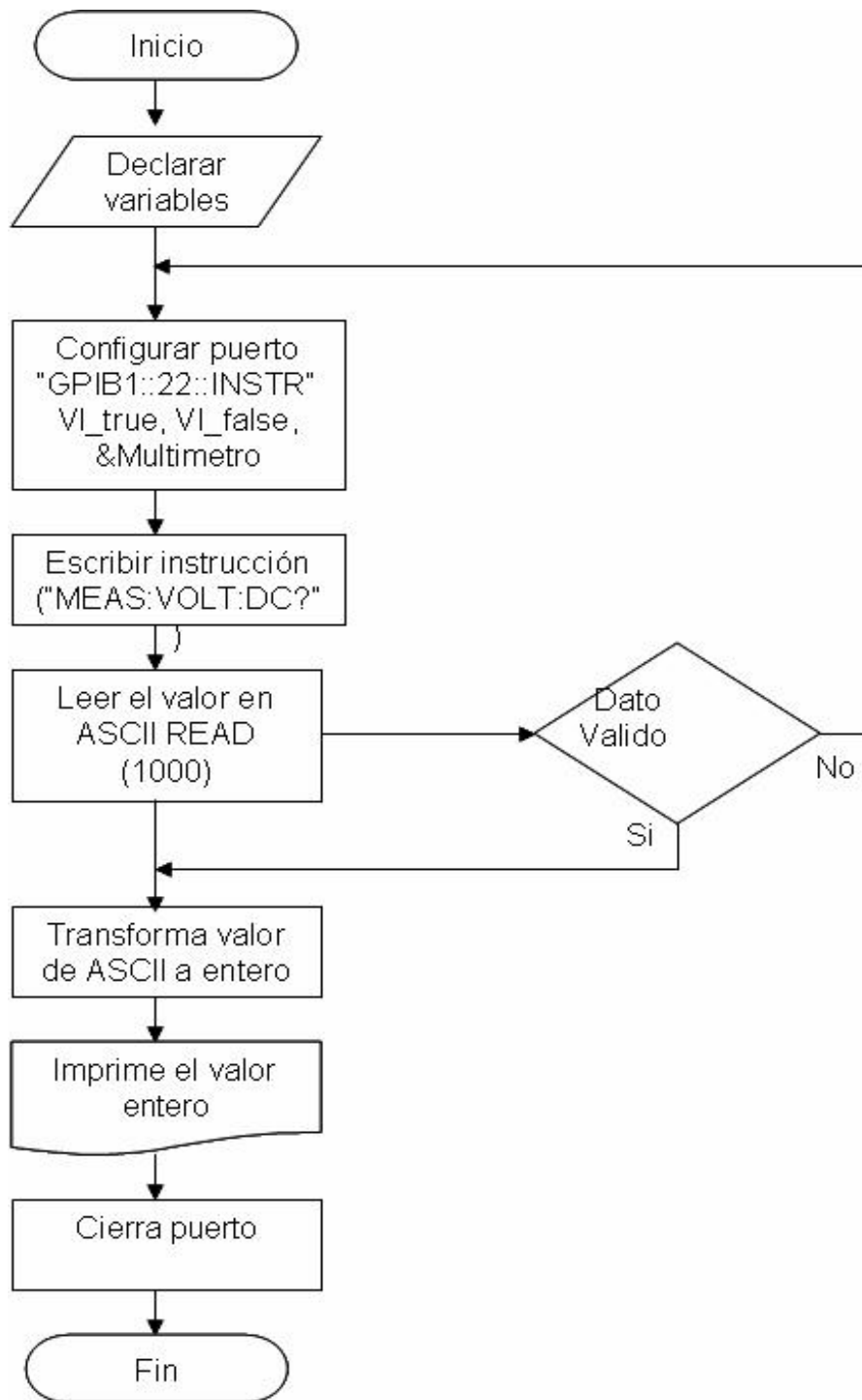


Figura 7.1. Diagrama de Flujo de cómo adquirir voltaje.

7.1.1 Pasos para adquirir voltaje.

1. Conectar el multímetro a la PC en el puerto GPIB.

- Entrada de la tarjeta (figura 7.2)



Figura 7.2: Tarjeta Agilent instalada en CPU.

- Conexión de GPIB (figura 7.3).

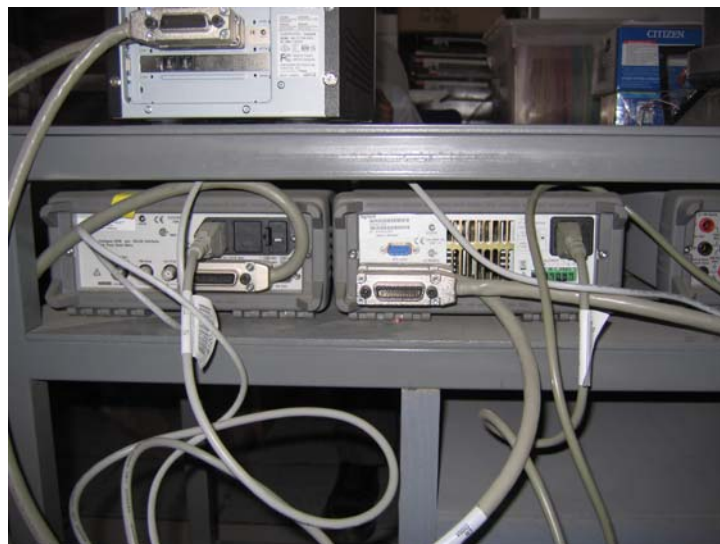


Figura 7.3: Conexión con instrumentos.

- Multímetro digital 34401A (figura 7.4)



Figura 7.4: Multímetro 34401A.

2. Primero se configura el multímetro (figura 7.5).

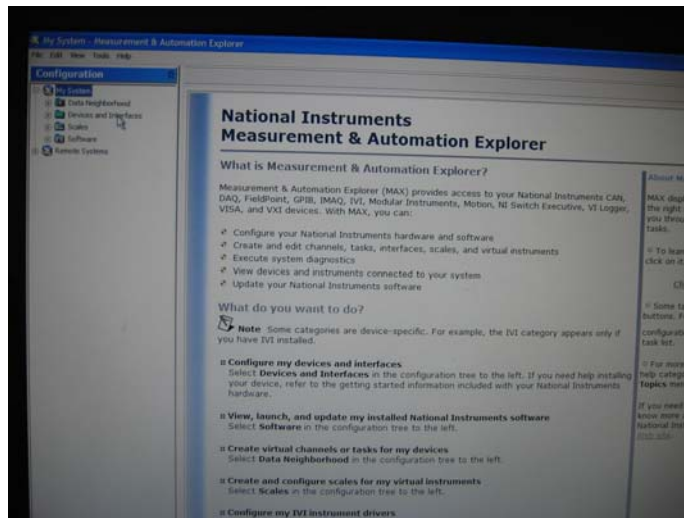


Figura 7.5: Configurar multímetro.

- Ir al icono Measurement & Automation Explorer.
- Se abre un cuadro de diálogo.
- Se selecciona Devices and Interfaces.
- Seleccionar Miscellaneous Visa Resources.
- Aparecerán todos los instrumentos de GPIB conectados.

- Seleccionar el instrumento con la dirección del multímetro, en este caso es GPIB0::22::INSTR.
- Salvar esta información.

3. Ejecuta el programa de Labview.

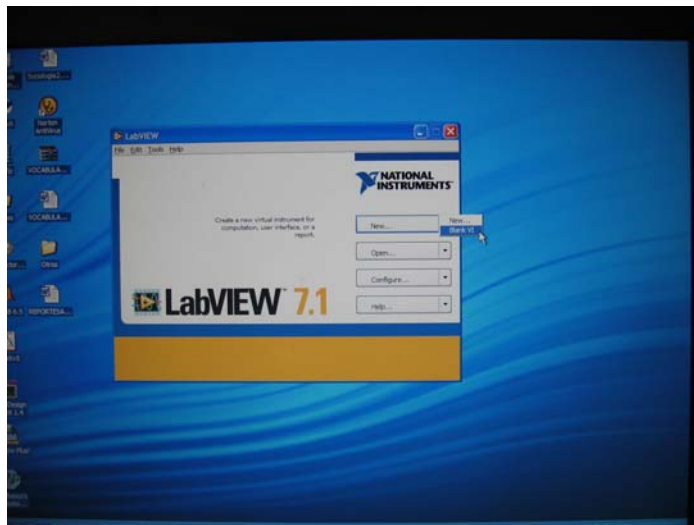


Figura 7.6: Abrir un VI en blanco.

- Abrir una página en blanco (figura 7.6).
- Ir al menú Windows y abrir el diagrama a bloques.

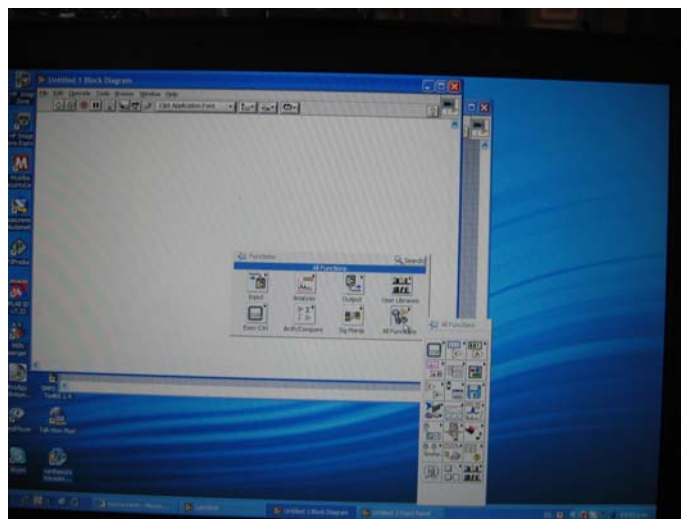


Figura: 7.7: Ventana de diagrama a bloques y funciones.

- Presionar el botón derecho del mouse (figura 7.7).
- Seleccionar All Functions.

- Se despliega un menú.
- Seleccionar Instrument I/O.
- Se despliega otro menú y selecciona VISA.
- Se despliega otro menú y selecciona VISA Advanced.
- Seleccionar VISA Open y posicionar la herramienta en la página del diagrama a bloques.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Control.
- Aparecerá un control en la página del diagrama a bloques.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Write posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Create Control.
- Y este aparecerá en la página del diagrama a bloques.
- Ir al menú de VISA y seleccionar VISA Read.
- El primer y último pin del lado izquierdo de VISA Read unirlos al primer y último pin de VISA Write del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Read posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Constant.
- Escribir en este un número, en este caso podemos poner 1000.
- Se posiciona el cursor en el segundo pin de VISA Read del lado derecho.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Indicator.
- Se posiciona el cursor en el tercer pin de VISA Read del lado derecho.

- Oprimir el botón derecho de mouse.
- Aparecerá un menú y seleccionar Indicador.
- Ir al menú All Functions.
- Seleccionar el menú String.
- Seleccionar el menú String/Number Conversión.
- Seleccionar Fract/Exp String to Number y posicionarlo en la página del diagrama a bloques.
- Conectar el segundo pin de VISA Read del lado derecho al primer pin de Fract/Exp String to Number del lado izquierdo.
- En el segundo pin de Fract/Exp String to Number posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Indicador.
- Por último abrir el menú VISA Advanced.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Read del lado derecho, respectivamente.

4. Ir al panel frontal y acomodar los ítems (figura 7.8).

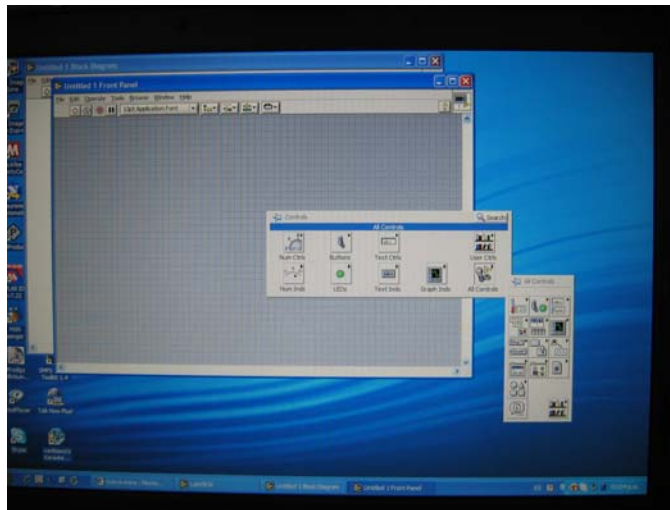


Figura 7.8: Ventana de panel frontal y funciones.

- Posicionar el cursor en la flecha del del Item que se llama VISA resource name.
- Se despliega una lista y debemos seleccionar, en donde dice GPIB.
- En el Item que dice write buffer escribir MEAS:VOLT:DC?.
- En el Item que dice read buffer va a registrar el valor real y este será dado en ascii cuando se realice el programa.
- En el Item number va a ser el valor en decimales.
- En el Item que dice return count va a ser el número de cadenas que este leyendo en programa en ese momento.

5. **Correr el programa con la flecha superior izquierda del panel de control (figura 7.9).**

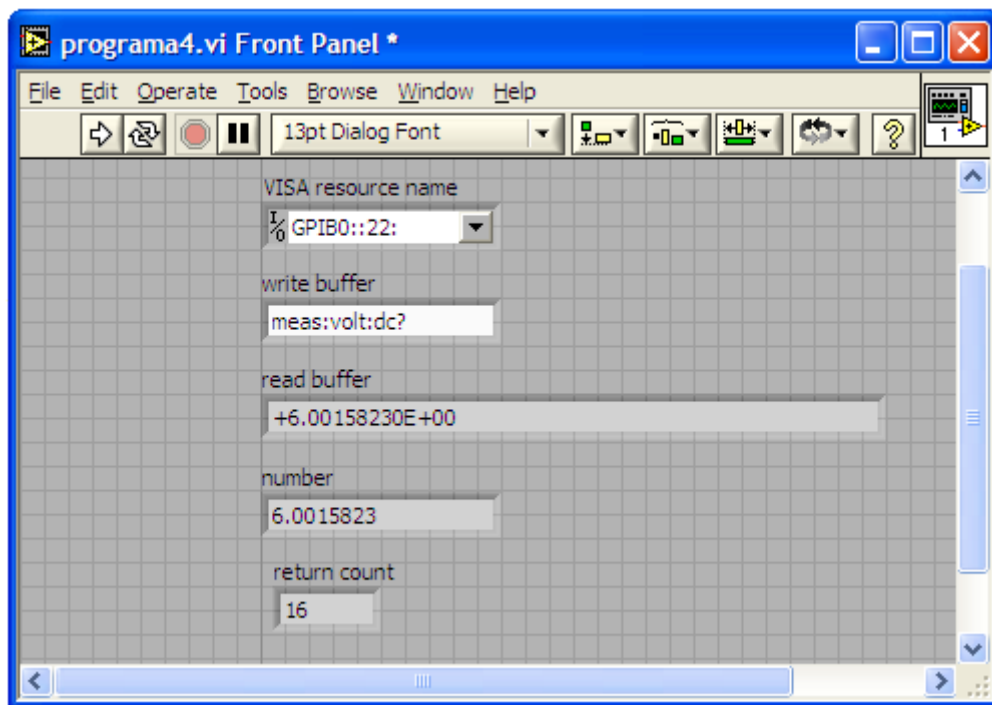


Figura 7.9: Panel frontal del programa LabView para adquirir voltaje.

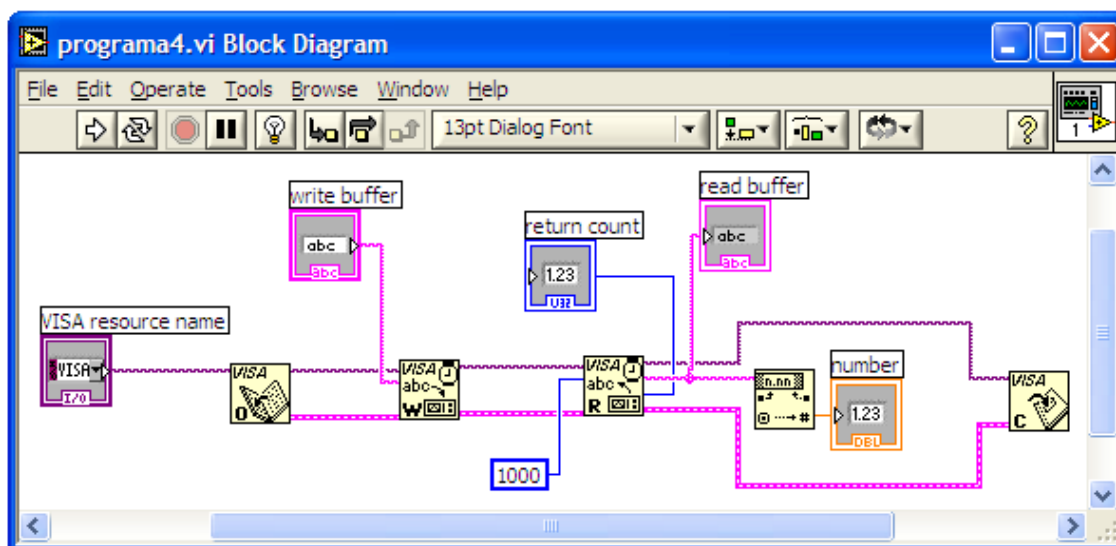


Figura 7.10: Diagrama a bloques del programa LabView para adquirir voltaje.

1. La herramienta VISA Open se utiliza para abrir la sesión de VISA (figura 7.10).

El control que se le pone al VISA Open es para poner la dirección de GPIB en donde esta conectado el multímetro.

2. La herramienta VISA Write se utiliza para escribir los comandos que necesitamos. En esta práctica utilizamos el comando “MEAS:VOLT:DC?”.

El control que se le pone al VISA Write es para ingresar el comando en el panel frontal (MEAS:VOLT:DC?).

3. La herramienta VISA Read es para recibir el valor, en este caso, de voltaje que el multímetro está leyendo.

La constante que se le pone al VISA Read es el número de cadenas o strings que queremos que lea.

En este caso se puede poner casi cualquier valor la diferencia es que en el número que está en read buffer nos dirá cuantas cadenas lee por lo que no es

necesario poner un número muy grande mientras sea mayor o igual al de read buffer

El indicador del lado izquierdo nos indica el número de cadenas que esta leyendo el programa en ese momento y con esto sabremos que rango se puede utilizar.

El indicador del lado derecho nos sirve para saber el valor real de voltaje y este se da en ascii. El valor en ascii es en notación científica.

4. El Fract/Exp String to number se utiliza porque el GPIB envía cadenas y esta herramienta es un convertidor de cadena a número.

El indicador que se le pone al Fract/Exp String to number es en donde podemos recibir el valor de voltaje.

5. La herramienta VISA Close nos sirve para cerrar la sesión de VISA.

6. Las conexiones de todos los VISA del primer y último pin es porque así reciben estos la dirección de GPIB y el error.

7.2 Adquisición de resistencia.

Al haber varios dispositivos conectados al bus, es necesario establecer una identificación particular para poder dirigirse a cada uno. Eso se hace a través de una dirección. Cada dispositivo contiene esa información almacenada en 7 bits. Los dos primeros indican si el dispositivo transmite o recibe (listen, talk). En los otros 5 se puede enumerar los dispositivos del 0 al 31. Teniendo en cuenta que hay dos bits reservados (uno de ellos para el ordenador), quedan 30 direcciones válidas para los equipos: del 1 al 30.

A un mismo ordenador puede conectarse varios buses GPIB, cada uno de ellos con su tarjeta correspondiente. Por eso, la identificación completa incluye también el número correspondiente a la tarjeta de interfaz. Por razones históricas, el número que se le da por defecto a la tarjeta (caso de que solo haya una) es el 7. Adicionalmente, en el caso de que se tengan instrumentos dentro de instrumentos (como es el caso de Mainframe VXI) se necesita una dirección secundaria de dos cifras. Por ejemplo la dirección 70913 indica la tarjeta número 7, dirección principal 9, dirección secundaria 13. En caso de instrumentos elementales, la dirección consta de tres cifras.

En cuanto a las características operacionales, quedaron definitivamente estipuladas con la norma IEEE 488.2. Se precisó en ella el formato permitido en los datos que se transmiten y el significado de los sufijos que pueden acompañarlos. También se establecieron unos comandos comunes que todos los aparatos deben aceptar, y que comienzan con un asterisco; por ejemplo todo instrumento debe aceptar el comando *IDN? Y devolver una cadena de caracteres con su identificación (modelo, etc.). Y se determinaron también algunas particularidades de la arquitectura de los instrumentos; en concreto el funcionamiento del Status Byte. En resumen todos los aparatos deben ser capaces de realizar ciertas operaciones internas básicas y comunicar su estado al bus de manera definida.[17]

El diagrama a de flujo se presenta en la figura 7.11.

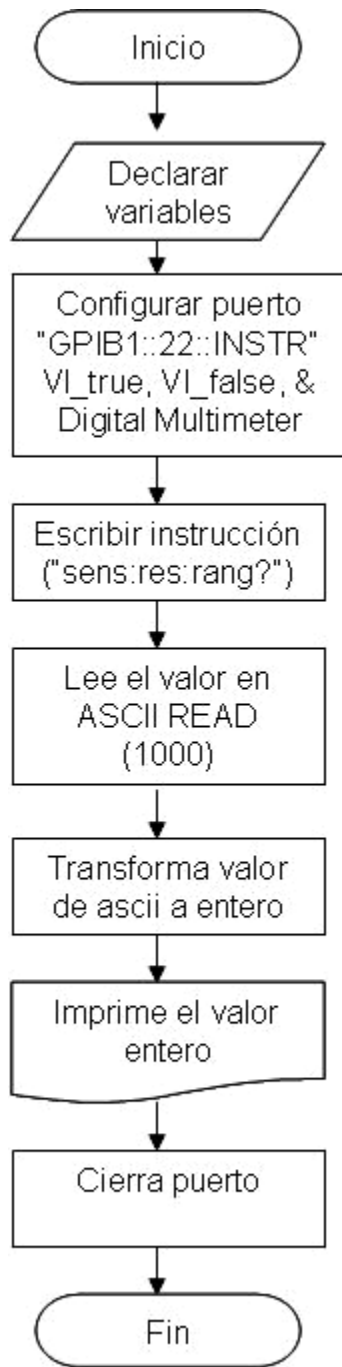


Figura 7.11: Diagrama de Flujo de cómo adquirir resistencia.

7.2.1 Pasos para adquirir resistencia.

- Las imágenes fotográficas de esta práctica se omiten, ya que se utiliza el mismo equipo que en la primer práctica y son de la figura 7.2 a la figura 7.8.

1. Conectar el multímetro a la PC en el puerto GPIB.

2. Primero se configura el multímetro digital.

- Ir al icono Measurement & Automation Explorer.
- Se abre un cuadro de diálogo.
- Se selecciona Devices and Interfaces.
- Seleccionar Miscellaneous Visa Resources.
- Aparecerán todos los instrumentos de GPIB conectados.
- Seleccionar el instrumento con la dirección del multímetro, en este caso es GPIB0::22::INSTR.
- Salvar esta información.

3. Ejecuta el programa de Labview.

- Abrir una página en blanco.
- Ir al menú Window y abrir el diagrama a bloques.
- Presionar el botón derecho del mouse.
- Seleccionar All Functions.
- Se despliega un menú.
- Seleccionar Instrument I/O.
- Se despliega otro menú y selecciona VISA.
- Se despliega otro menú y selecciona VISA Advanced.
- Seleccionar VISA Open y posicionar la herramienta en la página del diagrama a bloques.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.

- Seleccionar Create Control.
- Aparecerá un control en la página del diagrama a bloques.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Write posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Create Control.
- Y este aparecerá en la página del diagrama a bloques.
- Ir al menú de VISA y seleccionar VISA Read.
- El primer y último pin del lado izquierdo de VISA Read unirlos al primer y último pin de VISA Write del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Read posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Constant.
- Escribir en este un número en este caso podemos poner 1000.
- Se posiciona el cursor en el segundo pin de VISA Read del lado derecho.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Indicador.
- Se posiciona el cursor en el tercer pin de VISA Read del lado derecho.
- Oprimir el botón derecho de mouse.
- Aparecerá un menú y seleccionar Indicador.
- Ir al menú All Functions.
- Seleccionar el menú String.
- Seleccionar el menú String/Number Conversión.
- Seleccionar Fract/Exp String to Number y posicionarlo en la página del diagrama a bloques.
- Conectar el segundo pin de VISA Read del lado derecho al primer pin de Fract/Exp String to Number del lado izquierdo.

- En el segundo pin de Fract/Exp String to Number posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Indicador.
- Por último abrir el menú VISA Advanced.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Read del lado derecho, respectivamente.

4. Ir al panel frontal y acomodar los Ítems.

- Posicionar el cursor en la flecha del del Item que se llama VISA resource name.
- Se despliega una lista y debemos seleccionar en donde dice GPIB1::22::instr.
- En el Item que dice write buffer escribir SENS:RES:RANG?.
- En el Item que dice read buffer va a registrar el valor real y este será dado en ascii cuando se realice el programa.
- En el Item number va a ser el valor en decimales.
- En el Item que dice return count va a ser el número de cadenas que este leyendo en programa en ese momento.

5. Correr el programa con la flecha superior izquierda del panel de control (figura 7.12).

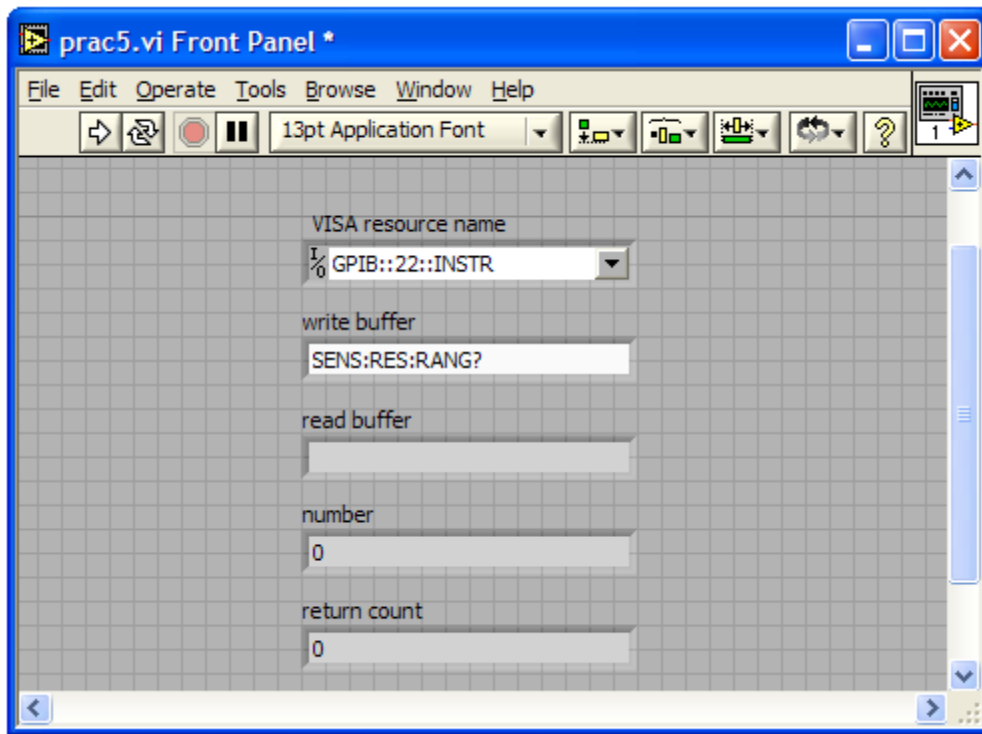


Figura 7.12: Panel frontal del programa LabView para adquirir resistencia.

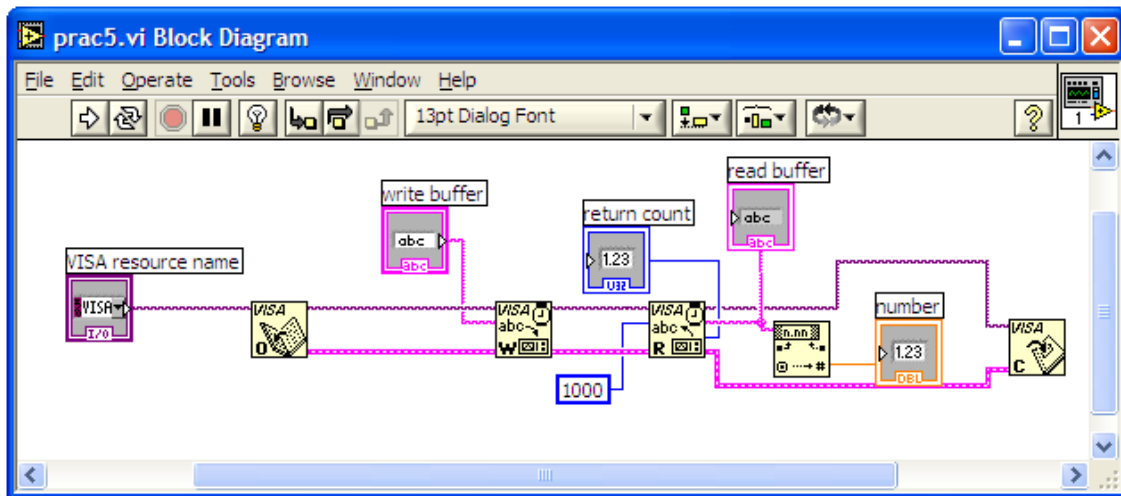


Figura 7.13: Diagrama a bloques del programa LabView para adquirir resistencia.

- 1. La herramienta VISA Open se utiliza para abrir la sesión de VISA (figura 7.13).**

El control que se le pone al VISA Open es para poner la dirección de GPIB en donde esta conectado el multímetro digital.

- 2. La herramienta VISA Write se utiliza para escribir los comandos que necesitamos. En esta práctica utilizamos el comando “SENS:RES:RANG?”.**

El control que se le pone al VISA Write es para ingresar el comando en el panel frontal (SENS:RES:RANG?).

- 3. La herramienta VISA Read es para recibir el valor, en este caso, de voltaje que el multímetro digital está leyendo.**

La constante que se le pone al VISA Read es el número de cadenas o strings que queremos que lea.

En este caso se puede poner casi cualquier valor la diferencia es que en el número que está en read buffer nos dirá cuantas cadenas lee por lo que no es necesario poner un número muy grande mientras sea mayor o igual al de read buffer.

El indicador del lado izquierdo nos indica el número de cadenas que esta leyendo el programa en ese momento y con esto sabremos que rango se puede utilizar.

El indicador del lado derecho nos sirve para saber el valor real de voltaje y este se da en ascii. El valor en ascii es en notación científica.

- 4. El Fract/Exp String to number se utiliza porque el GPIB envía cadenas y esta herrmienta es un convertidor de cadena a número.**

El indicador que se le pone al Fract/Exp String to number es en donde podemos recibir el valor de voltaje.

- 5. La herramienta VISA Close nos sirve para cerrar la sesión de VISA.**
- 6. Las conexiones de todos los VISA del primer y último pin es porque así reciben todos la dirección de GPIB y el error.**

7.3 Adquisición de Temperatura.

Existen tres tipos de dispositivos para medir temperatura estos son: termistores,

7.3.1 RTDs y termopares.

Compuesto de una mezcla sintetizada de óxidos metálicos, el termistor es esencialmente un semiconductor que se comporta como un "resistor térmico" con un coeficiente térmico de temperatura negativo de valor muy elevado.

Los termistores también se pueden encontrar en el mercado con la denominación NTC (Negative Temperature Coefficient) habiendo casos especiales de coeficiente positivo cuando su resistencia aumenta con la temperatura y se los denomina PTC (Positive Temperature Coefficient).

En algunos casos, la resistencia de un termistor a la temperatura ambiente puede disminuir en hasta 6% por cada 1°C de aumento de temperatura.

Esta elevada sensibilidad a variaciones de temperatura hace que el termistor resulte muy adecuado para mediciones precisas de temperatura, utilizándose ampliamente para aplicaciones de control y compensación en el rango de 150°C a 450°C.

Tres características importantes del termistor lo hacen extremadamente útil en aplicaciones de medición y control:

- a) Resistencia-temperatura.
- b) Voltaje-corriente.
- c) Corriente-tiempo.

Las características resistencia—temperatura, es que un termistor tiene coeficiente de temperatura de resistencia muy elevado y negativo, lo cual lo convierte en un TRANSDUCTOR DE TEMPERATURA IDEAL.[24]

En la característica voltaje-corriente se observa que la caída de voltaje a través de un termistor aumenta con el incremento de corriente hasta que alcanza un valor pico, más allá del cual la caída de voltaje decrece con el incremento de corriente.

En los termistores se observan relaciones de la resistencia con la temperatura que no son lineales, sino más bien de carácter exponencial. Para termistores comerciales comunes, se utiliza la fórmula de resistencia total.

$$R(T) = R(T = T_0) \cdot e^{x \cdot \left[\frac{1}{T} - \frac{1}{T_0} \right]} \quad \text{Resistencia total}$$

donde "R(T)" es la resistencia (en ohms, abreviado por la letra griega W) observada a temperatura "T", la que depende de un primer parámetro dado por la resistencia a una temperatura conocida "R(T=T₀)", típicamente 25°C, y de un segundo parámetro de ajuste "x". Como caso ejemplo, "x" valdrá del orden 4.000 y R (T=25°C) valdrá del orden 800 W.

En ambos casos (PT100 y termistores), dado que se mide resistencia, se debe cuidar, en primer lugar, que la resistencia de los cables de conexión (que también cambia con la temperatura) no incida sobre la medición y, en segundo lugar, que la corriente que circule por el elemento para medir su resistencia sea lo suficientemente pequeña como para calentar sólo mínimamente el propio sensor. A ese fin se utilizan sistemas de compensación que incorporan cables idénticos pero sin el sensor, en el brazo adyacente de un puente de Wheaston, amén de circular una corriente nula por el sensor. [15]

Los RTD son sensores de temperatura resistivos. En ellos se aprovecha el efecto que tiene la temperatura en la conducción de los electrones para que, ante un aumento de temperatura, haya un aumento de la resistencia eléctrica que presentan. Este aumento viene expresado como resistencia eléctrica:

$$R = R_0 \left[1 + AT + BT^2 - 100CT^3 + CT^4 \right] \quad \text{Resistencia eléctrica}$$

donde:

R es la resistencia a una temperatura de T°C

R₀ es la resistencia a 0°C

T es la temperatura

Este efecto suele aproximarse a un sistema de primer o segundo orden para facilitar los cálculos.

Un tipo de RTD son las Pt100. Estos sensores deben su nombre al hecho de estar fabricados de platino (Pt) y presentar una resistencia de 100ohms a 0°C. Son dispositivos muy lineales en un gran rango de temperaturas, por lo que suele expresarse su variación como resistencia de temperatura:

$$R = R_0(1 + \alpha(T^a - T_0^a)) \quad \text{Resistencia de temperatura}$$

Donde T_0 es una temperatura de referencia y R_0 es la resistencia a esa temperatura. Otros dispositivos similares son los Pt1000, con una resistencia de 1000ohms a 0°C .

Un termopar es un circuito formado por dos metales distintos que produce un voltaje siempre y cuando los metales se encuentren a temperaturas diferentes.

En electrónica, los termopares son ampliamente usados como sensores de temperatura. Son baratos, intercambiables, tienen conectores estándar y son capaces de medir un amplio rango de temperaturas. Su principal limitación es la exactitud ya que los errores del sistema inferiores a un grado centígrado son difíciles de obtener.[6]

El grupo de termopares conectados en serie recibe el nombre de termopila. Tanto los termopares como las termopilas son muy usados en aplicaciones de calefacción a gas.

En 1822 el físico estoniano Thomas Seebeck descubrió accidentalmente que la unión entre dos metales genera un voltaje que es función de la temperatura. Los termopares funcionan bajo este principio, el llamado efecto Seebeck. Si bien casi cualquier par de metales pueden ser usados para crear un termopar, se usa un cierto número debido a que producen voltajes predecibles y amplios gradientes de temperatura.

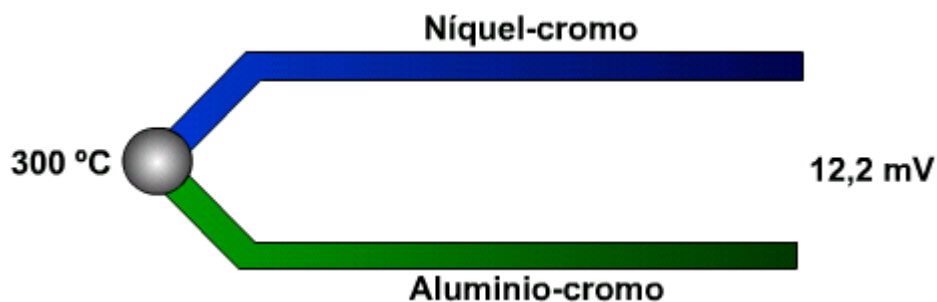


Figura 7.14: Termopar tipo K.

En la figura 7.14, es el termopar de tipo K producirá 12,2mV a 300°C. Desafortunadamente no es posible conectar un voltímetro al termopar para medir este voltaje porque la conexión a las guías del voltímetro hará una segunda unión no deseada. Para realizar mediciones precisas se debe compensar al usar una técnica conocida como compensación de unión fría (CUF).

La ley de los metales intermedios dice que un tercer metal introducido entre dos metales distintos de una unión de termopar no tendrá efecto siempre y cuando las dos uniones estén a la misma temperatura. Esta ley es importante en la construcción de uniones de termopares. Es posible hacer una unión termopar al estañar dos metales, ya que la estañadura no afectará la sensibilidad. En la práctica, las uniones termopares se realizan con soldaduras de los dos metales (por lo general con una carga capacitiva) ya que esto asegura que el desempeño no esté limitado al punto de fusión de una estañadura. Por lo general, la temperatura de la unión fría es detectada por un termistor de precisión en buen contacto con los conectores de salida del instrumento de medición. Esta segunda lectura de temperatura, junto con la lectura del termopar es usada por el instrumento de medición para calcular la temperatura verdadera en el extremo del termopar. Para aplicaciones menos críticas, la CUF es usada por un sensor de temperatura semiconductor. Al combinar la señal de este semiconductor con la señal del termopar, la lectura correcta puede ser obtenida sin la necesidad o esfuerzo de registrar dos temperaturas. La comprensión de la compensación de unión fría es importante; cualquier error en la medición de la temperatura de la unión fría terminará en el error de la temperatura medida en el extremo del termopar.[34]

La relación entre la temperatura y la generación de voltaje es un polinomio complejo (de 5° a 9° orden dependiendo del tipo de termopar).

Tipos de termopares:

- Tipo K (Cromo (Ni-Cr) / Aluminio (aleación de Ni-Al)): con una amplia variedad de aplicaciones, está disponible a un bajo costo y en una variedad de sondas. Tienen un rango de temperatura de $-200\text{ }^{\circ}\text{C}$ a $+1.200\text{ }^{\circ}\text{C}$ y una sensibilidad $41\mu\text{V}/^{\circ}\text{C}$ aprox.
- Tipo E (Cromo / Constantán (aleación de Cu-Ni)): No son magnéticos y gracias a su sensibilidad, son ideales para el uso en bajas temperaturas, en el ámbito criogénico. Tienen una sensibilidad de $68\mu\text{V}/^{\circ}\text{C}$.
- Tipo J (Hierro / Constantán): debido a su limitado rango, el tipo J es menos popular que el K. Son ideales para usar en viejos equipos que no aceptan el uso de termopares más modernos. El tipo K no puede usarse a temperaturas superiores a $760\text{ }^{\circ}\text{C}$ ya que una abrupta transformación magnética causa una descalibración permanente. Tienen un rango de -40°C a $+750^{\circ}\text{C}$.
- Tipo N (Nicrosil (Ni-Cr-Si / Nisil (Ni-Si)): es adecuado para mediciones de alta temperatura gracias a su elevada estabilidad y resistencia a la oxidación de altas temperaturas, y no necesita del platino utilizado en los tipos B, R y S que son más caros.
- Tipo B (Platino (Pt)-Rodio (Rh)): son adecuados para la medición de altas temperaturas superiores a $1.800\text{ }^{\circ}\text{C}$. El tipo B por lo general presentan el mismo resultado a $0\text{ }^{\circ}\text{C}$ y $42\text{ }^{\circ}\text{C}$ debido a su curva de temperatura/voltaje.
- Tipo R (Platino (Pt)-Rodio (Rh)): adecuados para la medición de temperaturas de hasta $1.600\text{ }^{\circ}\text{C}$. Su baja sensibilidad ($10\mu\text{V}/^{\circ}\text{C}$) y su elevado quitan su atractivo.
- Tipo S (Hierro / Constantán): ideales para mediciones de altas temperaturas hasta los $1.600\text{ }^{\circ}\text{C}$, pero su baja sensibilidad ($10\mu\text{V}/^{\circ}\text{C}$) y su elevado precio lo convierten en un instrumento no adecuado para el uso general. Debido a su elevada estabilidad, el tipo S es utilizado para la calibración universal del punto de fusión del oro ($1064,43\text{ }^{\circ}\text{C}$).

- Tipo T: es un termopar adecuado para mediciones en el rango de $-200\text{ }^{\circ}\text{C}$ a $0\text{ }^{\circ}\text{C}$. El conductor positivo está hecho de cobre y el negativo, de constantán.

Por otro lado, los termopares tipo B, R y S son los más estables, pero debido a su baja sensibilidad ($10\text{ }\mu\text{V}/^{\circ}\text{C}$ aprox.) generalmente son usados para medir altas temperaturas (superiores a $300\text{ }^{\circ}\text{C}$). [28]

Los termopares con una baja sensibilidad, como en el caso de los tipos B, R y S, tienen además una resolución menor. La selección de termopares es importante para asegurarse que cubren el rango de temperaturas a determinar. La descalibración es el proceso de alterar accidentalmente la conformación del cable del termopar. La causa más común es la difusión de partículas atmosféricas en el metal a los extremos de la temperatura de operación. Otras causas son las impurezas y los químicos del aislante difundiendo en el cable del termopar. Si se opera a elevadas temperaturas, se deben revisar las especificaciones del aislante de la sonda.

La salida de un termopar es una pequeña señal, así que es propenso a absorber ruido eléctrico. La mayoría de los instrumentos de medición rechazan cualquier modo de ruido (señales que están en el mismo cable o en ambos) así que el ruido puede ser minimizado al retorcer los cables para asegurarse que ambos recogen la misma señal de ruido. Si se opera en un ambiente extremadamente ruidoso, (Ej: cerca de un gran motor), es necesario considerar usar un cable de extensión protegido. Si se sospecha de la recepción de ruido, primero se deben apagar todos los equipos sospechosos y comprobar si las lecturas cambian.[12]

Los RTD, termistores y sensores de IC miden todos ellos temperaturas absolutas, pero el termopar mide solamente temperaturas relativas, y el motivo resulta obvio cuando pensamos en la conexión de un termopar a un voltímetro o a un sistema de adquisición de datos. Supongamos que estamos utilizando un termopar Tipo J,

que es el más normal y consiste en un hilo de hierro y otro de constatan (una aleación con un 45% de níquel y un 55% de cobre). ¿Qué ocurrirá cuando conectemos los dos hilos conductores de prueba, que probablemente sean de cobre? Que crearemos otros dos termopares, cada uno de los cuales aportará una tensión al circuito, con lo que tendremos tres termopares y tres temperaturas desconocidas.

La temperatura de referencia (T_{ref}) es una de las dos cantidades que necesitamos conocer para calcular T_x . La otra es V , que medimos con el sistema de adquisición de datos (o voltímetro). Aplicando la fórmula $V = (T_x - T_{ref})$, podemos calcular tensiones equivalentes para los dos valores de temperatura y a continuación restar para determinar el valor de T_x . Los coeficientes de Seebeck y las tensiones de salida resultantes son números pequeños (tabla 7.1), por lo que resulta difícil medir con exactitud tanto los niveles absolutos como los cambios relativos. En este punto el ruido eléctrico puede alterar la precisión de las medidas de temperatura. El acoplamiento magnético y electrostático se reduce utilizando cable de par trenzado, reduciendo al mínimo la longitud de los hilos conductores y permaneciendo alejado de campos magnéticos y eléctricos intensos. Por último, pero no por ello menos importante, se necesita instrumentación capaz de realizar medidas de bajo nivel limpias. [13]

Tipo de Termopar	a 0 °C	a 100 °C	Tensión de salida a 100 °C
B	-0,25 V/C	0,90 V/C	0,033 mV
E	58,7 V/C	67,5 V/C	6,32 mV
J	50,4 V/C	54,4 V/C	5,27 mV
K	39,5 V/C	41,4 V/C	4,10 mV
S	5,40 V/C	7,34 V/C	0,65 mV

Tabla 7.1: Coeficientes de Seebeck y tensiones de salida.

Un asunto adicional muy importante en el uso de termopares en la industria tiene que ver con la variación de la temperatura ambiente en las uniones frías. Esta es la situación: si supiéramos de antemano la temperatura de las uniones frías, entonces en lugar de relacionar la lectura del voltímetro con la diferencia de temperatura, se podría relacionarla con la temperatura de la unión caliente misma. Esto sería posible pues podríamos construir las tablas de temperatura contra voltaje para que reflejaran el hecho de que las uniones frías están a una cierta temperatura de referencia (como se le denomina) conocida.

Las ventajas y desventajas de los componentes utilizados para medir temperatura se muestran en la tabla 7.2.

	RTD	Termistor	Termopar
Ventajas	Más estable. Más preciso. Más lineal que Los Termopares.	Alto rendimiento Rápido Medida de dos hilos	Autoalimentado Robusto Económico Amplia variedad de formas físicas Amplia gama de temperaturas
Desventajas	Caro. Lento. Precisa fuente de alimentación. Pequeño cambio de resistencia. Medida de 4 hilos Autocalentable	No lineal. Rango de Temperaturas limitado. Frágil. Precisa fuente de alimentación. Autocalentable	No lineal Baja tensión Precisa referencia El menos estable El menos sensible

Tabla 7.2: Ventajas y Desventajas.

7.3.2 Pasos para adquirir temperatura.

Se realizará un programa para comunicar la PC con un multímetro de laboratorio, utilizando el programa Labview y las herramientas VISA.

En este programa podremos medir la temperatura de algún circuito, dispositivo o aparato que deseemos saber su temperatura.

Para realizar una medición de temperatura veraz, se necesita tener una referencia ya sea interna o externa. La interna es la referencia del mismo NanoVoltmetro y la externa es una medición con otro dispositivo.

Para esta parte de la práctica ocuparemos un termopar tipo K conectado a un termómetro de laboratorio y un termopar tipo T conectado al NanoVoltmetro. En este caso la referencia externa será el termómetro.

Primero en la parte superior de la resistencia se ponen las puntas de los dos termopares a la misma distancia y posteriormente se le unta la pasta a los dos termopares y a la resistencia. [27]

Después se conecta la fuente y empezamos a hacer una tabulación desde temperatura ambiente hasta 85°, esto se logra aumentando la corriente que pasa por la resistencia (tabla 7.3).

Termopar tipo T (mV)	Termopar tipo K (°C)
-0.0515	22°
0.0812	25°
0.2849	30°
0.3758	35°
0.5209	40°
0.6953	45°
0.8672	50°
1.0665	55°
1.2114	60°
1.3950	65°
1.6224	70°
1.7551	75°
1.9535	80°
2.1123	85°

Tabla 7.3: Tabulación de temperatura.

Nota: los valores de la tabla pueden ser diferentes, ya que estos dependen del medio ambiente y fenómenos atmosféricos.

Estos valores se grafican y después se tiene que linealizar la gráfica (figura 7.15). Se puede utilizar un software especializado para estos datos y se obtiene la siguiente gráfica que es la curva de calibración de temperatura y la ecuación lineal que relacione el voltaje contra la temperatura.

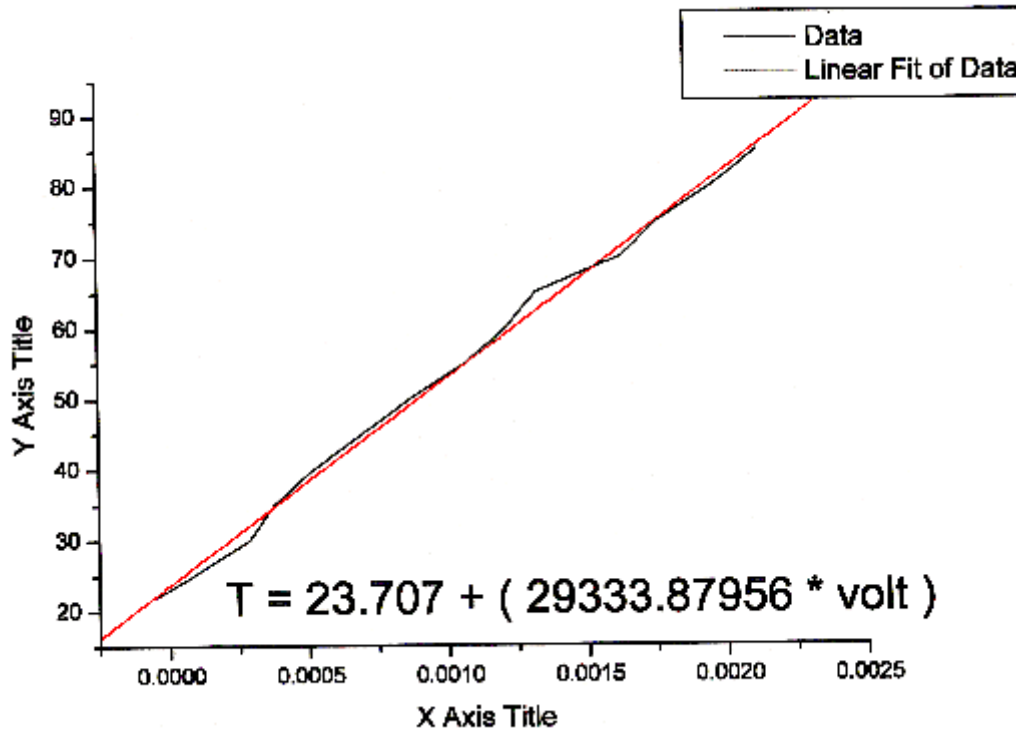


Figura 7.15: Linealización de la temperatura.

Con esta ecuación obtenemos el valor de temperatura y esta se utiliza en el programa.[5]

Después se hace el diagrama de flujo (figura 7.16).



Figura 7.16: Diagrama de Flujo de cómo adquirir temperatura.

- Las imágenes fotográficas de esta práctica se omiten ya que se utiliza el mismo equipo que en la primer práctica y son de la figura 7.2 a la figura 7.8.

1. Conectar el multímetro a la PC en el puerto GPIB.

2. Primero se configura el Nanovoltmetro digital.

- Ir al icono Measurement & Automation Explorer.
- Se abre un cuadro de diálogo.
- Se selecciona Devices and Interfaces.
- Seleccionar Miscellaneous Visa Resources.
- Aparecerán todos los instrumentos de GPIB conectados.
- Seleccionar el instrumento con la dirección del multímetro, en este caso es GPIB0::20::INSTR.
- Salvar esta información.

3. Ejecuta el programa de Labview .

- Abrir una página en blanco.
- Ir al menú Window y abrir el diagrama a bloques.
- Presionar el botón derecho del mouse.
- Seleccionar All Functions.
- Se despliega un menú.
- Seleccionar Instrument I/O.
- Se despliega otro menú y selecciona VISA.
- Se despliega otro menú y selecciona VISA Advanced.
- Seleccionar VISA Open y posicionar la herramienta en la página del diagrama a bloques.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Control.

- Aparecerá un control en la página del diagrama a bloques.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Write posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Create Control.
- Y este aparecerá en la página del diagrama a bloques.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin del primer VISA Write del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo del segundo VISA Write posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Create Control.
- Y este aparecerá en la página del diagrama a bloques.
- Ir al menú de VISA y seleccionar VISA Read.
- El primer y último pin del lado izquierdo de VISA Read unirlos al primer y último pin de VISA Write del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Read posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Constant.
- Escribir en este un número en este caso podemos poner 1000.
- Se posiciona el cursor en el segundo pin de VISA Read del lado derecho.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Indicador.
- Se posiciona el cursor en el tercer pin de VISA Read del lado derecho.
- Oprimir el botón derecho de mouse.
- Aparecerá un menú y seleccionar Indicador.

- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de la tercera VISA Write unirlos al primer y último pin de VISA Read del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Write posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Create Control.
- Y este aparecerá en la página del diagrama a bloques.
- Ir al menú All Functions.
- Seleccionar el menú String.
- Seleccionar el menú String/Number Conversión.
- Seleccionar Fract/Exp String to Number y posicionarlo en la página del diagrama a bloques.
- Conectar el segundo pin de VISA Read del lado derecho al primer pin de Fract/Exp String to Number del lado izquierdo.
- Ir al menú Functions y posicionarse el icono arith/compare.
- Seleccionar Formula e introducir la siguiente formula para obtener el valor de temperatura $23.707+(29333.87956*X1)$.
- El pin de X1 del lado izquierdo y conectar al segundo pin de Fract/Exp String to Number.
- Posicionar el cursor en el pin de Formula donde dice Result.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Indicador.
- Por último abrir el menú VISA Advanced.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Read del lado derecho, respectivamente.

4. Ir al panel frontal y acomodar los Ítems.

Posicionar el cursor en la flecha del Item que se llama VISA resource name.

- Se despliega una lista y debemos seleccionar en donde dice GPIB1::20::instr.
- En el Item que dice configurar CH2 escribir CONF:VOLT:DC (@FRON2).
- En el Item que dice Instrucción de voltaje escribir MEAS:VOLT:DC? (@FRON2).
- En el Item que dice Read voltaje va a registrar el valor real y este será dado en ascii cuando se realice el programa.
- En el Item que dice Cadenas va a registrar el número de cadenas que este leyendo en programa en ese momento.
- En el Item que dice Result va a registrar el valor de temperatura.
- En el Item que dice Permanecer en CH2 escribir ROUT:TERM (@FRON2).

5. Correr el programa con la flecha superior izquierda del panel de control (figura 7.17).

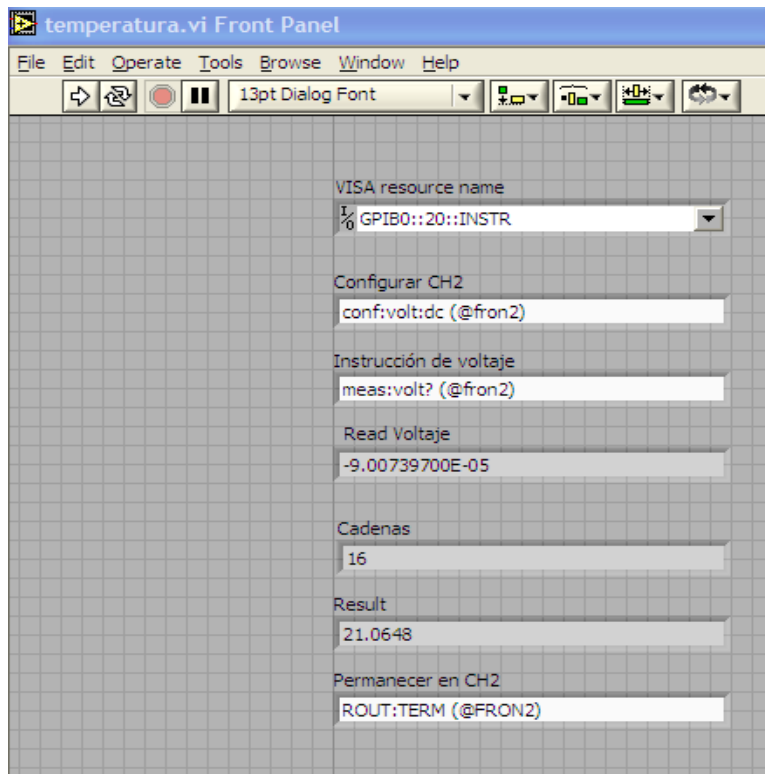


Figura 7.17: Panel frontal del programa LabView para adquirir temperatura.

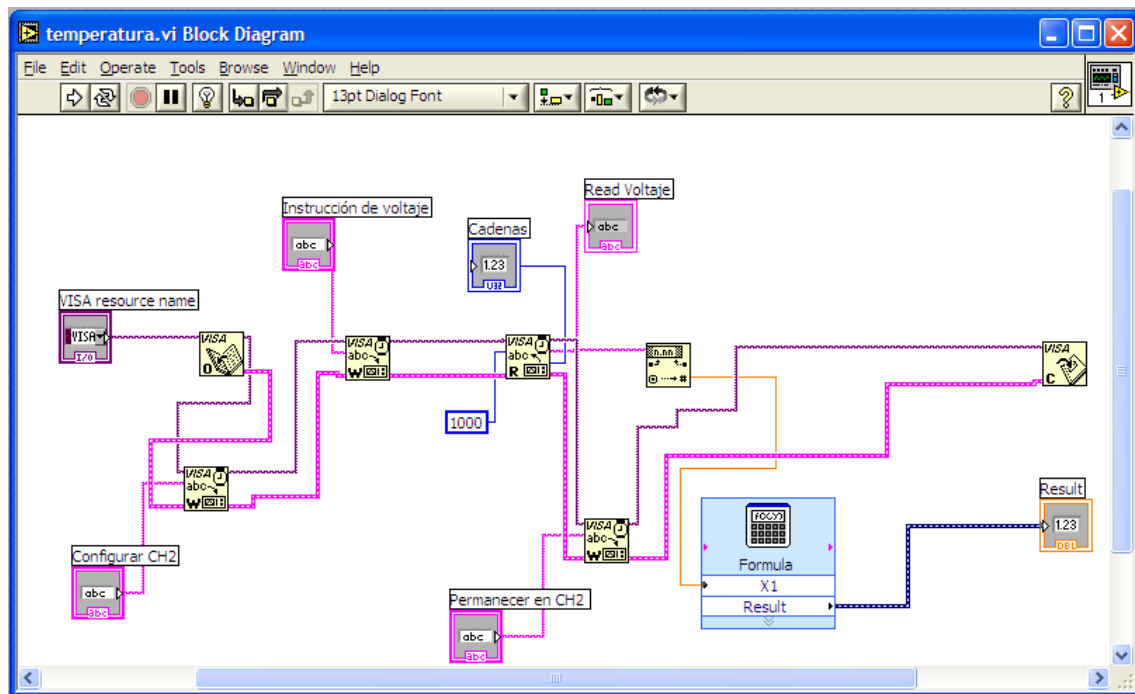


Figura 7.18: Diagrama a bloques del programa LabView para adquirir temperatura.

1. **La herramienta VISA Open se utiliza para abrir la sesión de VISA (figura 7.18).**

El control que se le pone al VISA Open es para poner la dirección de GPIB en donde esta conectado el Nanovoltmetro digital.

2. **La primer herramienta VISA Write se utiliza para escribir la configuración del canal 2, ya que si no se especifica el canal leerá el CH1.**

El control que se le pone al VISA Write es para ingresar el comando en el panel frontal (CONF:COLT:DC (@FRON2).

3. **La segunda herramienta VISA Write se utiliza para escribir los comandos que necesitamos. En esta práctica utilizamos el comando “MEAS:VOLT:DC?”.**

El control que se le pone al VISA Write es para ingresar el comando en el panel frontal (MEAS:VOLT:DC? (@FRON2)).

4. **La herramienta VISA Read es para recibir el valor, en este caso, de voltaje que el NanoVoltmetro digital está leyendo.**

La constante que se le pone al VISA Read es el número de cadenas o strings que queremos que lea.

En este caso se puede poner casi cualquier valor la diferencia es que en el número que está en Read buffer nos dirá cuantas cadenas lee por lo que no es necesario poner un número muy grande mientras sea mayor o igual al de Read buffer.

El indicador del lado izquierdo nos indica el número de cadenas que esta leyendo el programa en ese momento y con esto sabremos que rango se puede utilizar.

El indicador del lado derecho nos sirve para saber el valor real de voltaje y este se da en ascii. El valor en ascii es en notación científica.

5. La tercera herramienta VISA Write es para que la pantalla del NanoVoltmetro.

El control que se le pone al VISA Write es para ingresar el comando en el panel frontal (ROUT:TERM (@FRON2).

6. El Fract/Exp String to Number se utiliza porque el GPIB envía cadenas y esta herramienta es un convertidor de cadena a número.

7. La Formula es para transformar el valor de voltaje obtenido en temperatura

En el indicador que se le pone a la Formula se obtiene el valor de temperatura.

8. La herramienta VISA Close nos sirve para cerrar la sesión de VISA.

9. Las conexiones de todos los VISA del primer y último pin es porque así reciben todos la dirección de GPIB y el error.

7.4 Control de voltaje.

Voltaje es la fuerza que tiene un generador eléctrico. No es un valor absoluto sino la diferencia entre la carga eléctrica de los polos positivo y negativo del generador. Haciendo un símil, si tenemos dos recipientes llenos de agua, uno colocado a más altura que el otro, la fuerza con que cae el agua de uno a otro depende de la diferencia de altura existente entre ambos. Si la altura es pequeña esa caída de agua no será capaz de mover, por ejemplo, unas noria de un molino, pero si la altura es grande, sí. El nombre de voltaje es utilizado principalmente a nivel del público general. Existen otros dos sinónimos, diferencia de potencial y

fuerza electromotriz, que se usan principalmente a nivel académico e industrial, respectivamente. El voltaje se mide en voltios, en honor a Alessandro Volta.

La diferencia de potencial entre dos puntos (1 y 2) de un campo eléctrico es igual al trabajo que realiza dicho campo sobre la unidad de carga positiva para transportarla desde el punto 1 al punto 2.[8]

Es independiente del camino recorrido por la carga y depende exclusivamente del potencial de los puntos 1 y 2 en el campo; se expresa por la diferencia de potencial:

$$V_1 - V_2 = E \times r \quad \text{Diferencia de Potencial}$$

donde: $V_1 - V_2$ es la diferencia de potencial

E es la Intensidad de campo en newton/culombio

r es la distancia en metros entre los puntos 1 y 2

Igual que el potencial, en el Sistema Internacional de Unidades la diferencia de potencial se mide en voltios.[18]

Si dos puntos que tienen una diferencia de potencial se unen mediante un conductor, se producirá un flujo de corriente eléctrica. Parte de la carga que crea el punto de mayor potencial se trasladará a través del conductor al punto de menor potencial y, en ausencia de una fuente externa (generador), esta corriente cesará cuando ambos puntos igualen su potencial eléctrico.

Que dos puntos tengan igual potencial eléctrico no significa que tengan igual carga.

La carga no fluye a menos que exista una diferencia de potencial. Para que haya una corriente constante se necesita una bomba eléctrica adecuada que mantenga

la diferencia de potencial. Todo dispositivo que crea una diferencia de potencial se conoce como una fuente de voltaje.[3]

Las celdas o pilas secas, las pilas húmedas y los generadores son capaces de mantener un flujo constante. (Una batería no es otra cosa que dos o más celdas o pilas interconectadas).

Las pilas secas, las pilas húmedas y los generadores suministran energía que permite que las cargas se desplacen. En las pilas secas y en las húmedas la energía que se desprende de una reacción química que se lleva a cabo dentro de la pila se transforma en energía eléctrica. Los generadores por su parte convierten energía mecánica en energía eléctrica. La energía potencial eléctrica, sea cual sea el método empleado en su producción, está disponible en las terminales de la pila o generador. La energía potencial por coulomb de carga disponible para los electrones que se desplazan entre las terminales es el voltaje (llamado a veces fuerza electromotriz, o fem). El voltaje proporciona la “presión eléctrica” necesaria para desplazar los electrones entre las terminales de un circuito.[19]

Se realiza el diagrama de flujo (figura 7.19).

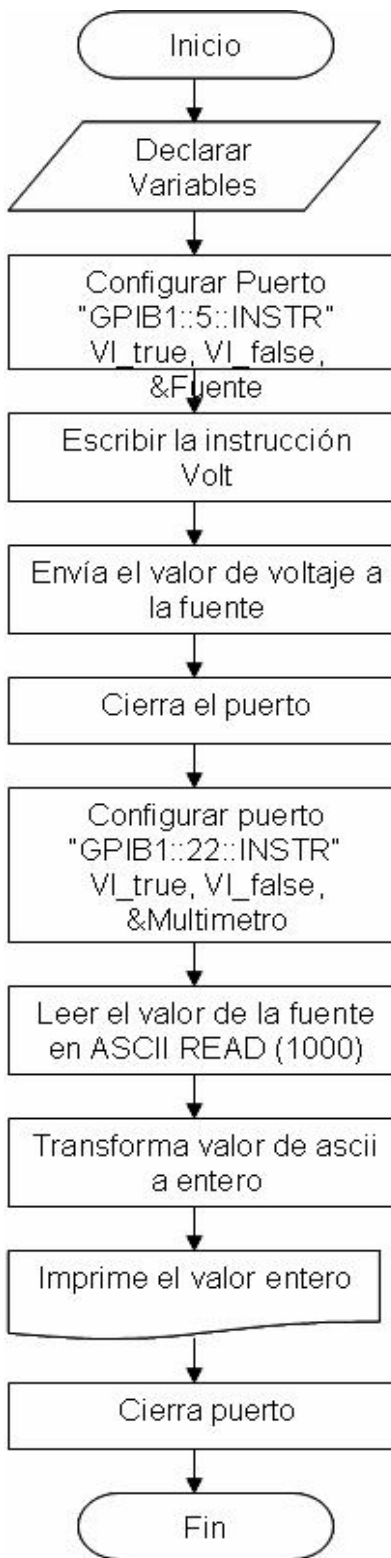


Figura 7.19: Diagrama de Flujo de cómo controlar el voltaje.

7.4.1 Pasos para controlar voltaje.

- Las imágenes fotográficas de esta práctica se omiten ya que se utiliza el mismo equipo que en la primer práctica y son de la figura 7.2 a la figura 7.8.

1. Conectar el multímetro a la PC en el puerto GPIB.

2. Primero se configura el multímetro.

- Ir al icono Measurement & Automation Explorer.
- Se abre un cuadro de diálogo.
- Se selecciona Devices and Interfaces.
- Seleccionar Miscellaneous Visa Resources.
- Aparecerán todos los instrumentos de GPIB conectados.
- Seleccionar el instrumento con la dirección del multímetro, en este caso es GPIB0::22::INSTR.
- Salvar esta información.

3. Ejecuta el programa de Labview.

- Abrir una página en blanco.
- Ir al menú Window y abrir el diagrama a bloques.
- Presionar el botón derecho del mouse.
- Seleccionar All Functions.
- Se despliega un menú.
- Seleccionar Structures.
- Se despliega otro menú y seleccionar Flat Sequense structure.
- Posicionar la herramienta en la página del diagrama a bloques.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Add Frame After y aparecerá otra estructura pegada a la primera.
- Seleccionar All Functions.
- Se despliega un menú.

- Seleccionar Instrument I/O.
- Se despliega otro menú y selecciona VISA.
- Se despliega otro menú y selecciona VISA Advanced.
- Seleccionar VISA Open y posicionar la herramienta dentro de la estructura.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Control.
- Aparecerá un control en la página del diagrama a bloques.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.
- Ir al menú All Functions y después al de String.
- Seleccionar la herramienta Format into String y unirla al segundo pin del lado izquierdo de VISA Write.
- En el tercer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Control.
- Aparecerá un control en la página del diagrama a bloques.
- En el cuarto pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Control.
- Aparecerá un control en la página del diagrama a bloques.
- En el pin de arriba posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Constant y escribir %s, %.4f.

- Abrir el menú VISA Advanced.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Write del lado derecho, respectivamente.
- Seleccionar VISA Open y posicionar la herramienta en la segunda estructura.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Control.
- Aparecerá un control en la página del diagrama a bloques.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Write posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Create Control.
- Y este aparecerá en la página del diagrama a bloques.
- Ir al menú de VISA y seleccionar VISA Read.
- El primer y último pin del lado izquierdo de VISA Read unirlos al primer y último pin de VISA Write del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Read posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Constant.
- Escribir en este un número, en este caso podemos poner 1000.
- Se posiciona el cursor en el segundo pin de VISA Read del lado derecho.
- Oprimir el botón derecho del mouse.

- Aparecerá un menú y seleccionar Indicator.
- Se posiciona el cursor en el tercer pin de VISA Read del lado derecho.
- Oprimir el botón derecho de mouse.
- Aparecerá un menú y seleccionar Indicator.
- Ir al menú All Functions.
- Seleccionar el menú String.
- Seleccionar el menú String/Number Conversión.
- Seleccionar Fract/Exp String to Number y posicionarlo en la página del diagrama a bloques.
- Conectar el segundo pin de VISA Read del lado derecho al primer pin de Fract/Exp String to Number del lado izquierdo.
- En el segundo pin de Fract/Exp String to Number posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Indicator.
- Por último abrir el menú VISA Advanced.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Read del lado derecho, respectivamente.

4. Ir al panel frontal y acomodar los Ítems.

- Posicionar el cursor en la flecha del ítem que se llama Dirección de la Fuente.
- Se despliega una lista y debemos seleccionar, en donde dice GPIB1::5::INSTR.
- En el ítem que dice Instrucción de la Fuente escribir VOLT.
- En el ítem que dice Voltaje podemos poner el valor de voltaje que deseemos.

- Posicionar el cursor en la flecha del Item que se llama Dirección del Multímetro.
- Se despliega una lista y debemos seleccionar, en donde dice GPIB1::22::INSTR.
- El Item que dice Leer Voltaje se pone la instrucción MEAS:VOLT:DC?.
- En el Item que dice Respuesta en ascii va a registrar el valor real y este será dado en ascii cuando se realice el programa.
- En el Item Respuesta del Multímetro va a ser el valor en decimales.
- En el Item que dice Cadenas va a ser el número de cadenas que este leyendo en programa en ese momento.

5. Correr el programa con la flecha superior izquierda del panel de control (figura 7.20).

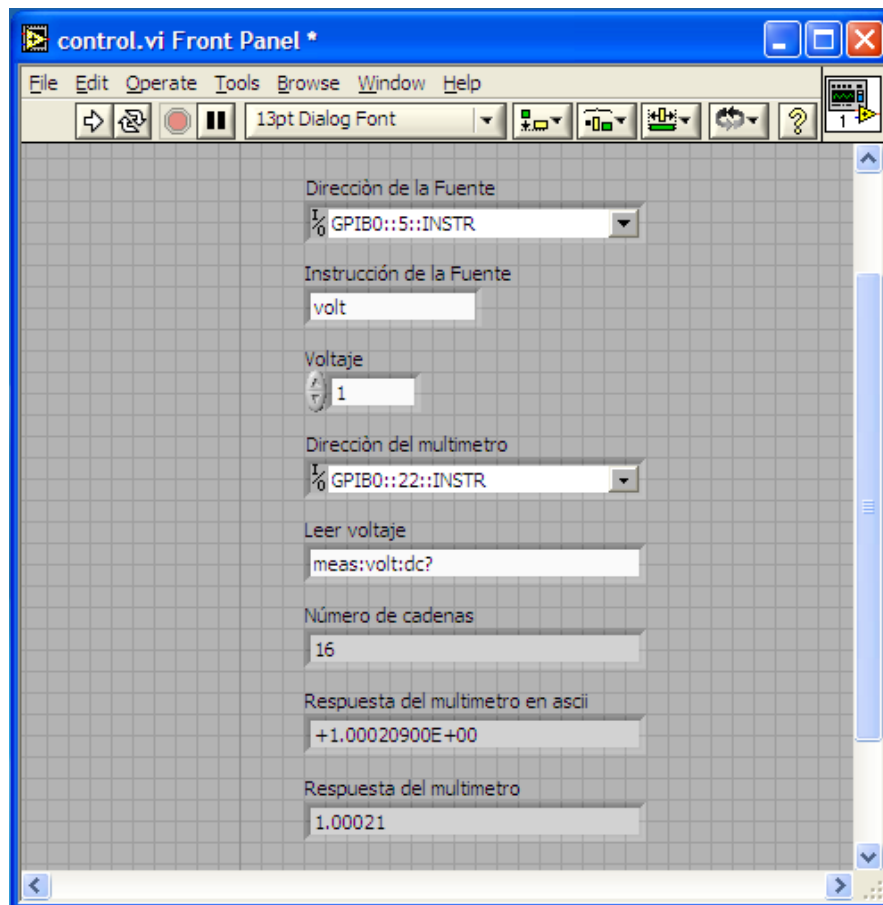


Figura 7.20: Panel frontal del programa LabView para controlar el voltaje.

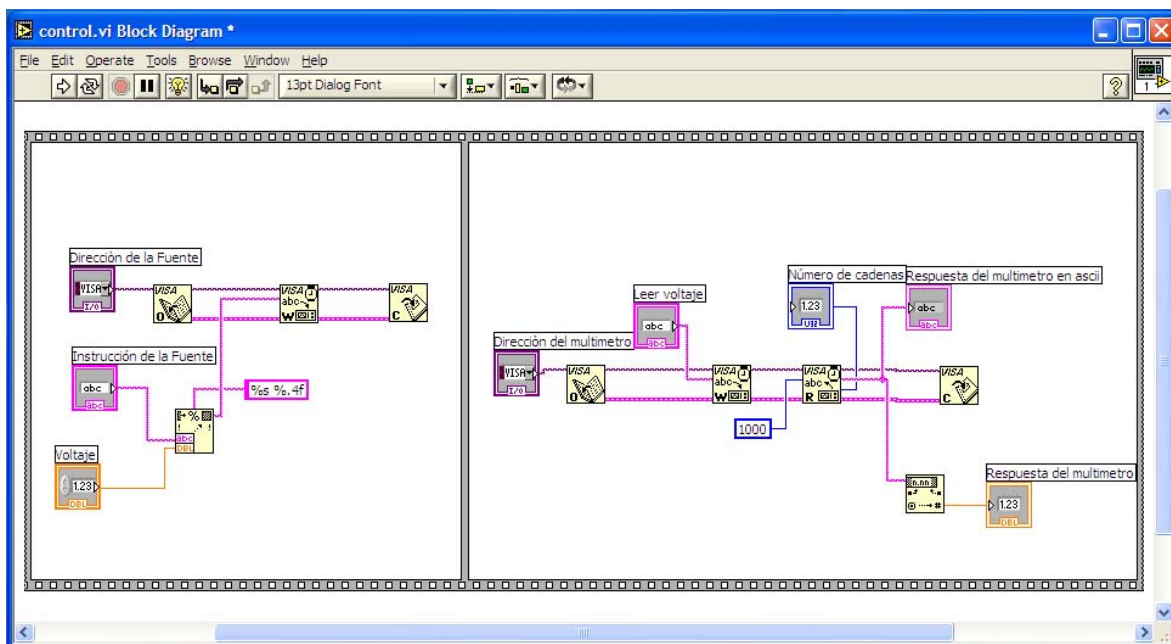


Figura 7.21: Diagrama a bloques del programa LabView para controlar el voltaje.

1. La herramienta Flat Sequence Structure nos sirve para hacer una sucesión de pasos, en este programa primero escribe desde la PC el valor de voltaje a la fuente y después el multímetro lee este valor para enviarlo de regreso a la PC (figura 7.21).

2. La herramienta VISA Open se utiliza para abrir la sesión de VISA.

El control que se le pone al VISA Open es para poner la dirección de GPIB en donde esta conectada la fuente.

3. La herramienta VISA Write se utiliza para escribir los comandos que necesitamos. En esta práctica utilizamos el comando “VOLT”.

3.1 El Format Into String es para poner las dos opciones la instrucción y el control para ir variándolo.

3.1.1 El primer control que se le pone al String es para ingresar el comando en el panel frontal (VOLT).

3.1.2 El segundo control es para variar el voltaje.

3.1.3 La constante que se le pone al String es para decirle que va a escribir una cadena de Bytes hasta con cuatro cifras significativas.

4. La herramienta VISA Close nos sirve para cerrar la sesión de VISA.

5. La herramienta VISA Open se utiliza para abrir la sesión de VISA, ya que en este caso se debe abrir una nueva sesión.

El control que se le pone al VISA Open es para poner la dirección de GPIB en donde esta conectado el multímetro.

6. La herramienta VISA Write se utiliza para escribir los comandos que necesitamos.

6.1 En el control pondremos el comando "MEAS:VOLT:DC?" para leer el valor de voltaje.

7. La herramienta VISA Read es para recibir el valor, en este caso, de voltaje que el multímetro está leyendo.

7.1 La constante que se le pone al VISA Read es el número de cadenas o strings que queremos que lea.

7.1.1 En este caso se puede poner casi cualquier valor la diferencia es que en el número que está en read buffer nos dirá cuantas cadenas lee por lo que no es necesario poner un número muy grande mientras sea mayor o igual al de read buffer.

7.1.2 El indicador del lado izquierdo nos indica el número de cadenas que esta leyendo el programa en ese momento y con esto sabremos que rango se puede utilizar.

7.2 El indicador del lado derecho nos sirve para saber el valor real de voltaje y este se da en ascii. El valor en ascii es en notación científica.

8 El Fract/Exp String to number se utiliza porque el GPIB envía cadenas y esta herramienta es un convertidor de cadena a número.

8.1 El indicador que se le pone al Fract/Exp String to number es en donde podemos recibir el valor de voltaje.

9 La herramienta VISA Close nos sirve para cerrar la segunda sesión de VISA.

10 Las conexiones de todos los VISA del primer y último pin es porque así reciben estos la dirección de GPIB y el error.

7.5 Control de velocidad de un motor.

Existen diferentes formas para medir la velocidad, las siguientes son algunas opciones:

7.5.1 Tacogenerador.

El tacogenerador es un dispositivo para medir la velocidad angular. Su funcionamiento es sencillo: convertir la energía rotacional del eje en cuestión en energía eléctrica, proporcional a la rotacional y que puede ser fácilmente medida.

Una posible configuración podría ser la que se ve en la figura 7.22.

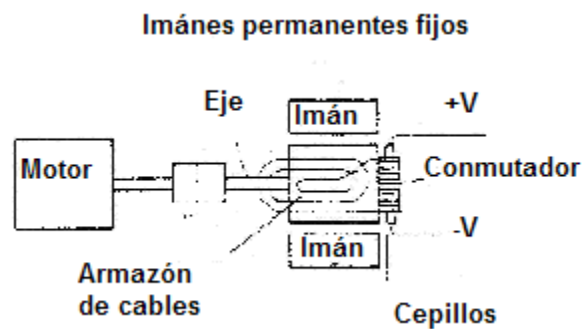


Figura 7.22. Partes de un tacogenerador.

Para generar la corriente a partir del giro se acopla al motor o eje que se va a medir, una espira situada dentro del campo magnético fijo (creada por los dos imanes). Al girar el motor la espira girará en el interior del campo magnético, lo que provocará una corriente eléctrica. Estos dispositivos pueden llegar a tener una precisión del 0.5%, por lo que pueden resultar una solución aceptable a la hora de medir la velocidad angular.

7.5.2 Convertidores de Frecuencia a Voltaje.

Los convertidores de frecuencia a voltaje son circuitos integrados que convierten un voltaje de entrada análogo en un tren de pulsos cuya frecuencia de salida es proporcional al nivel de entrada. Se utilizan en aplicaciones de conversión análoga a digital donde la velocidad no es un factor crítico, también opera como convertidores de frecuencia a voltaje y pueden ser utilizados como convertidores de señales digitales a análogas de baja frecuencia.

Dentro de los convertidores de señales de voltaje a frecuencia o de frecuencia a voltaje se encuentran: LM331 de National semiconductor, AD650 de Analog Devices, VFC32 de Burr Brown, XR4151 de Exar.[23]

Algunas aplicaciones de los convertidores de Frecuencia a Voltaje son:

Control de velocidad de motores.

Medición de flujo.

Demodulación de FM.

Transmisión de datos.

Aislamiento de sistemas.

Enlaces ópticos.

Interface de transductores con sistemas digitales.

Telemetría de FM de bajo costo.

Aislamiento de señales análogas.

Multiplexación análoga.

7.5.3 Frecuencímetro multifunción

Uno de los instrumentos básicos para el profesional o el aficionado a la electrónica es el frecuencímetro, los circuitos integrados más usados para este propósito han sido el ICM7216 y el ICM7226 que han sido usados por todos los que queríamos construir un frecuencímetro. Con el uso de un microcontrolador se consigue simplificar el circuito y a la vez ampliar las posibilidades de este, así no solo funciona como frecuencímetro sino que además es contador de impulsos, mide el periodo de una frecuencia, mide las revoluciones (tacómetro), puede generar una señal PWM (Pulse Width Modulation), puede servir también como generador de baja frecuencia y por si esto fuese poco aún puede hacer las funciones de un pequeño analizador de señal de un canal. En modo frecuencímetro puede sumar o restar un valor programado a la frecuencia leída, esto es necesario (como saben bien los aficionados a la radio) para visualizar la frecuencia de un receptor sumando o restando la frecuencia intermedia.[36]

7.5.4 Tacómetro.

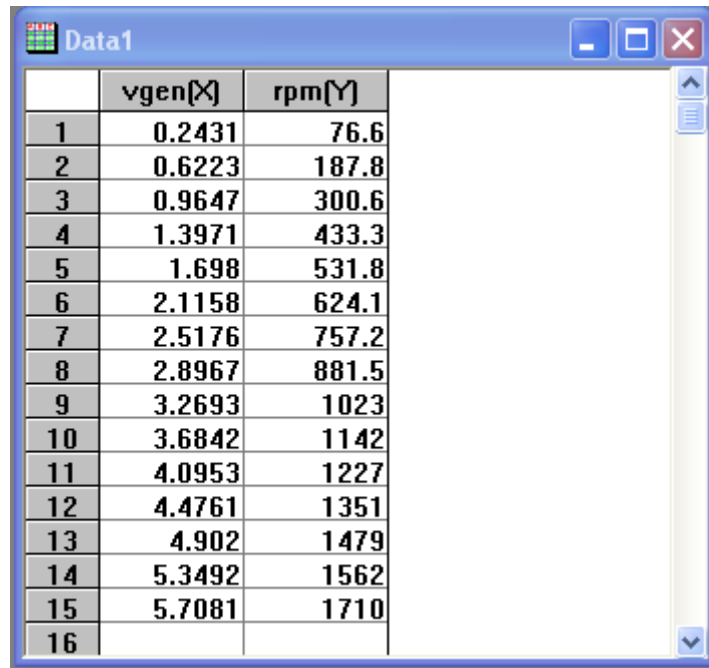
Aparato que mide el número de revoluciones de un eje. En automoción, designa en concreto al cuentarrevoluciones, es decir, el mecanismo que indica las revoluciones (giro completo) por minuto del cigüeñal de un motor.

7.5.5 Procedimiento y metodología.

Se realizará un programa para comunicar vía GPIB la PC con una fuente, donde introduciremos el valor de velocidad deseado; después el tacogenerador enviará un valor de voltaje proporcional y el multímetro de laboratorio leerá ese valor y lo enviará de regreso a la PC, utilizando el programa Labview y las herramientas VISA.

Para realizar una medición de velocidad veraz, se necesita tener una referencia ya externa, en este caso se utilizará un tacómetro; el cual medirá las revoluciones por minuto del motor y se comparará con el voltaje del tacogenerador.[14]

Se obtiene una tabulación (tabla 7.5), se grafica, y obtenemos la ecuación de linealización (figura 7.23), utilizando el software llamado Origin.



	vgen[X]	rpm[Y]
1	0.2431	76.6
2	0.6223	187.8
3	0.9647	300.6
4	1.3971	433.3
5	1.698	531.8
6	2.1158	624.1
7	2.5176	757.2
8	2.8967	881.5
9	3.2693	1023
10	3.6842	1142
11	4.0953	1227
12	4.4761	1351
13	4.902	1479
14	5.3492	1562
15	5.7081	1710
16		

Tabla 7.5: Tabulación de velocidad.

Después se realiza el diagrama a bloques (figura 7.24).

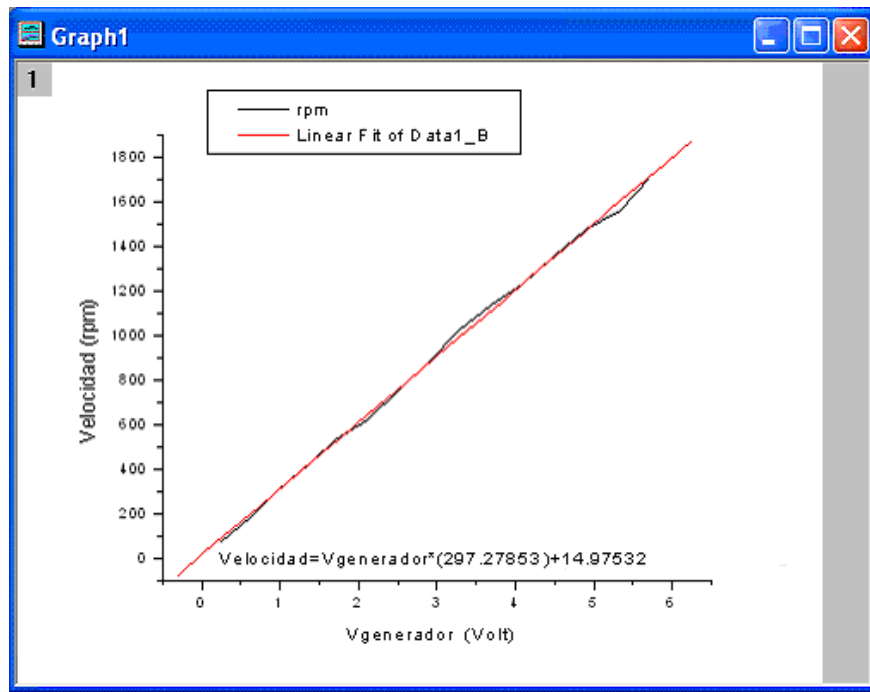


Figura 7.23: Gráfica de linealización.

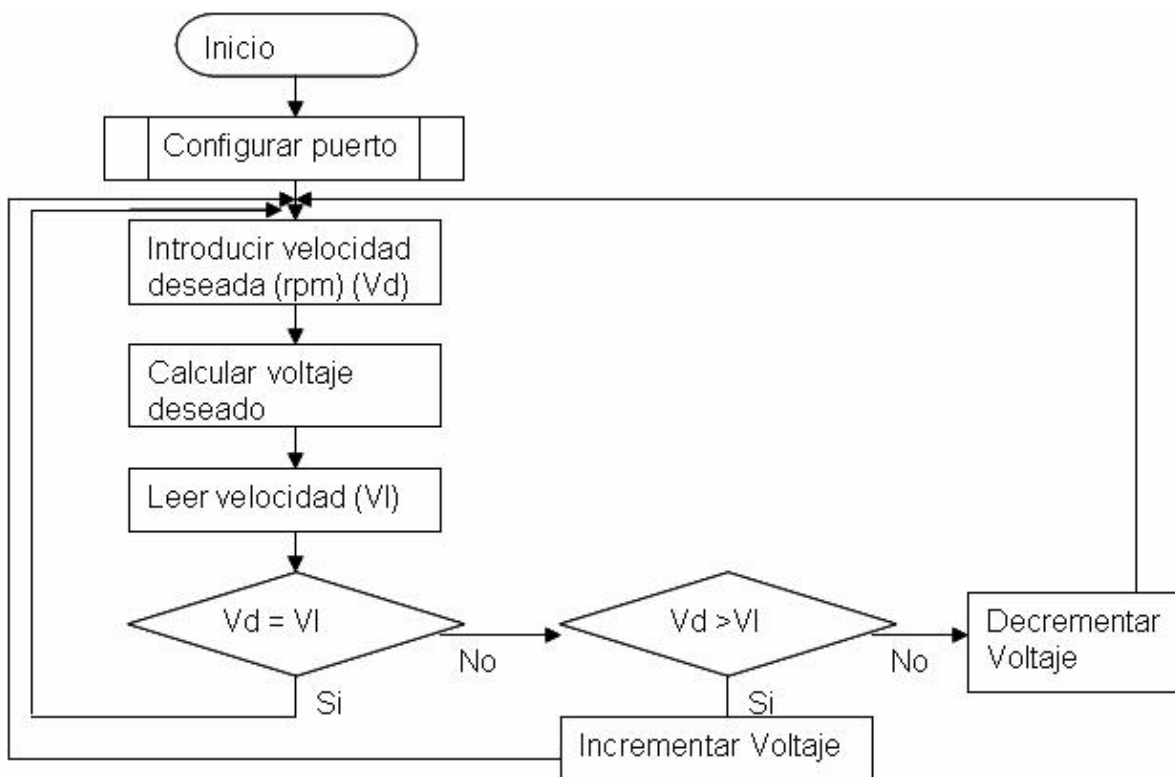


Figura 7.24: Diagrama de Flujo de cómo controlar el voltaje.

7.5.6 Pasos para controlar velocidad.

- Las imágenes fotográficas de esta práctica se omiten ya que se utiliza el mismo equipo que en la primer práctica y son de la figura 7.2 a la figura 7.8.

1. Conectar el multímetro a la PC en el puerto GPIB.

2. Primero se configura el multímetro.

- Ir al icono Measurement & Automation Explorer.
- Se abre un cuadro de diálogo.
- Se selecciona Devices and Interfaces.
- Seleccionar Miscellaneous Visa Resources.
- Aparecerán todos los instrumentos de GPIB conectados.
- Seleccionar el instrumento con la dirección del multímetro, en este caso es GPIB0::22::INSTR.
- Salvar esta información.

3. Ejecuta el programa de Labview.

- Abrir una página en blanco.
- Ir al menú Window y abrir el diagrama a bloques.
- Presionar el botón derecho del mouse.
- Seleccionar Exec Ctrl.
- Se despliega un menú.
- Seleccionar Case Structure.
- Posicionar la herramienta en la página del diagrama a bloques.
 - Cuando el case es True hacer lo siguiente.
 - Oprimir el botón derecho del mouse.
 - Presionar el botón derecho del mouse.
 - Seleccionar Exec Ctrl.
 - Se despliega un menú.
 - Seleccionar While loop.

- Posicionar la herramienta en la página del diagrama a bloques dentro de la herramienta Case Structure.
- Presionar el botón derecho del mouse.
- Seleccionar Exec Ctrl.
- Se despliega un menú.
- Seleccionar While Loop.
- Posicionar la herramienta en la página del diagrama a bloques dentro de la herramienta Flat Sequence.
- Oprimir el botón derecho del mouse.
- Seleccionar All Functions.
- Se despliega un menú.
- Seleccionar Instrument I/O.
- Se despliega otro menú y selecciona VISA.
- Se despliega otro menú y selecciona VISA Advanced.
- Seleccionar VISA Open y posicionar la herramienta dentro de la estructura.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Constant.
- Aparecerá una constante y seleccionar la dirección de la fuente.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.
- Posicionarse en el segundo pin del VISA Write.
- Oprimir el botón derecho del mouse y seleccionar create Constant y escribir OUTPUT ON.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.

- Ir al menú All Functions y después al de String.
- Seleccionar la herramienta Format into String y unirla al segundo pin del lado izquierdo de VISA Write.
- En el tercer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Constant y escribir Volt.
- Aparecerá un control en la página del diagrama a bloques.
- En el cuarto pin posicionar el cursor y crear un Indicator.
- En el pin de arriba posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Constant y escribir %s, %.4f.
- Oprimir el botón derecho del mouse y seleccionar el menú All Functions.
- Se despliega otro menú y seleccionar Arith/Compare.
- Seleccionar la herramienta Fórmula dar doble clic encima y se despliega un cuadro de diálogo.
- Escribir la ecuación $(X1+162.99595)/233.06429$.
- Posicionar el cursor en el pin X1 de la herramienta fórmula.
- Oprimir el botón derecho del mouse y seleccionar Create Control.
- Unir el cuarto pin de Format to String al pin Result de la Fórmula.
- Abrir el menú VISA Advanced.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Write del lado derecho, respectivamente.
- Posicionar el cursor en la herramienta Flat Sequence y seleccionar Add Frame after.

- Seleccionar VISA Open y posicionar la herramienta en la segunda estructura.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Constant.
- Aparecerá una constante y escribir la dirección del multímetro.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Write posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Create Constant y escribir MEAS:VOLT:DC?
- Y este aparecerá en la página del diagrama a bloques.
- Ir al menú de VISA y seleccionar VISA Read.
- El primer y último pin del lado izquierdo de VISA Read unirlos al primer y último pin de VISA Write del lado derecho, respectivamente.
- En el segundo pin del lado izquierdo de VISA Read posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Constant.
- Escribir en este un número, en este caso podemos poner 1000.
- Se posiciona el cursor en el segundo pin de VISA Read del lado derecho.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú y seleccionar Indicator.

- Se posiciona el cursor en el tercer pin de VISA Read del lado derecho.
- Oprimir el botón derecho de mouse.
- Aparecerá un menú y seleccionar Indicator.
- Ir al menú All Functions.
- Seleccionar el menú String.
- Seleccionar el menú String/Number Conversión.
- Seleccionar Fract/Exp String to Number y posicionarlo en la página del diagrama a bloques.
- Conectar el segundo pin de VISA Read del lado derecho al primer pin de Fract/Exp String to Number del lado izquierdo.
- Oprimir el botón derecho del mouse y seleccionar Arith/Compare.
- Aparecerá un menú y seleccionar Fórmula.
- Escribir dentro la siguiente ecuación $X1*297.273+14.975$.
- Oprimir el botón derecho del mouse y seleccionar Sig Manip.
- Aparecerá un menú y seleccionar From DDT.
- Unir el pin Result con el de From DDT.
- Posicionar el cursor en el pin del lado derecho de From DDT.
- Oprimir el botón derecho del mouse y seleccionar Create Indicator.
- Por último abrir el menú VISA Advanced.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Read del lado derecho, respectivamente.
- Posicionar el cursor en Flat Sequence.
- Oprimir el botón derecho del mouse y seleccionar Add Frame Alter.
- Ir al menú Arith/Compare, se despliega un menú y seleccionar Comparison.
- Seleccionar la herramienta de igual que.
- Comparar la velocidad deseada con la velocidad leída.

- Unir el pin derecho de la herramienta igual que al signo de interrogación del Case.
- Cuando el Case es True no se hace nada y ahí termina el programa.
- Si el Case es False.
- Abrir otra Case Structure.
- Ir nuevamente al menú Comparison y elegir la herramienta mayor que.
- Unir el pin derecho de la herramienta mayor que al signo de interrogación del Case.
- Comparar nuevamente la velocidad deseada con la leída.
- Si el Case es True Oprimir el botón derecho del mouse y seleccionar Arith/Compare.
- Aparecerá un menú y seleccionar Fórmula y escribir la siguiente ecuación dentro $((X1*100)+1)/100$.
- Unir la Velocidad leída a X1.
- Oprimir el botón derecho del mouse y seleccionar Arith/Compare.
- Aparecerá un menú y seleccionar Fórmula y escribir la siguiente ecuación dentro $(X1-14.9753)/297.27853$.
- Unir Result a X1.
- Seleccionar VISA Open.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Constant y seleccionar la dirección de la fuente.
- Aparecerá una constante en la página del diagrama a bloques.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.
- El primer y último pin del lado izquierdo de VISA Write unirlos al primer y último pin de VISA Open del lado derecho, respectivamente.

- Seleccionar Format into String y unir el primer pin del lado derecho de este con el segundo pin del lado izquierdo de VISA Write.
- En el Tercer pin de Format into String crear una Constante y escribir Volt.
- El cuarto pin unirlo a Result de la segunda fórmula.
- En el pin de arriba de Format into String crear una Constante y escribir %s %.4f.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Read del lado derecho, respectivamente.
- Si el Case es False.
- Oprimir el botón derecho del mouse y seleccionar Arith/Compare
- Aparecerá un menú y seleccionar Fórmula y escribir la siguiente ecuación dentro $((X1*100)-1)/100$.
- Unir la Velocidad leída a X1.
- Oprimir el botón derecho del mouse y seleccionar Arith/Compare.
- Aparecerá un menú y seleccionar Fórmula y escribir la siguiente ecuación dentro $(X1-14.9753)/297.27853$.
- Unir Result a X1.
- Seleccionar VISA Open.
- En el primer pin del lado izquierdo posicionar el cursor.
- Oprimir el botón derecho del mouse.
- Aparecerá un menú.
- Seleccionar Create Constant y seleccionar la dirección de la fuente.
- Aparecerá una constante en la página del diagrama a bloques.
- Ir nuevamente al menú de VISA y seleccionar VISA Write.

- El primer y último pin del lado izquierdo de VISA Write unirlo al primer y último pin de VISA Open del lado derecho, respectivamente.
- Seleccionar Format into String y unir el primer pin del lado derecho de este con el segundo pin del lado izquierdo de VISA Write.
- En el Tercer pin de Format into String crear una Constante y escribir Volt.
- El cuarto pin unirlo a Result de la segunda fórmula.
- En el pin de arriba de Format into String crear una Constante y escribir %s %.4f.
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlo al primer y último pin de VISA Read del lado derecho, respectivamente.
- Si el Case es False.
 - Seleccionar VISA Open y posicionar la herramienta dentro de la estructura.
 - En el primer pin del lado izquierdo posicionar el cursor.
 - Oprimir el botón derecho del mouse.
 - Aparecerá un menú.
 - Seleccionar Create Constant.
 - Aparecerá una constante y seleccionar la dirección de la fuente.
 - Ir nuevamente al menú de VISA y seleccionar VISA Write.
 - El primer y último pin del lado izquierdo de VISA Write unirlo al primer y último pin de VISA Open del lado derecho, respectivamente
 - En el segundo pin del lado izquierdo de VISA Write posicionar el cursor.
 - Oprimir el botón derecho del mouse.

- Aparecerá un menú y seleccionar Create Constant y escribir Volt 0.0
- Seleccionar VISA Close.
- El primer y último pin del lado izquierdo de VISA Close unirlos al primer y último pin de VISA Read del lado derecho, respectivamente.

4. Ir al panel frontal y acomodar los Ítems.

- Oprimir el botón derecho del mouse y seleccionar buttons.
- Seleccionar Vertical Toggle Switch y posicionarlo en el panel frontal, pero a su vez aparecerá un icono en el Diagrama a bloques.
- Ir al diagrama a bloques, seleccionar este icono y posicionarlo fuera del Case principal y unir el pin al signo de interrogación del Case.
- En el Item que dice Voltaje podemos ver el resultado del voltaje que se necesita para la velocidad deseada.
- En el Item que dice Velocidad deseada podemos poner el valor en rpms que deseemos obtener.
- En el Item que dice Velocidad leída podemos ver la velocidad en que se encuentra el motor en ese momento.
- El botón de Stop es para parar el programa.
- El Switch es para parar el motor cuando se finalice el programa.

5. Correr el programa con la flecha superior izquierda del panel de control (figura 7.25).

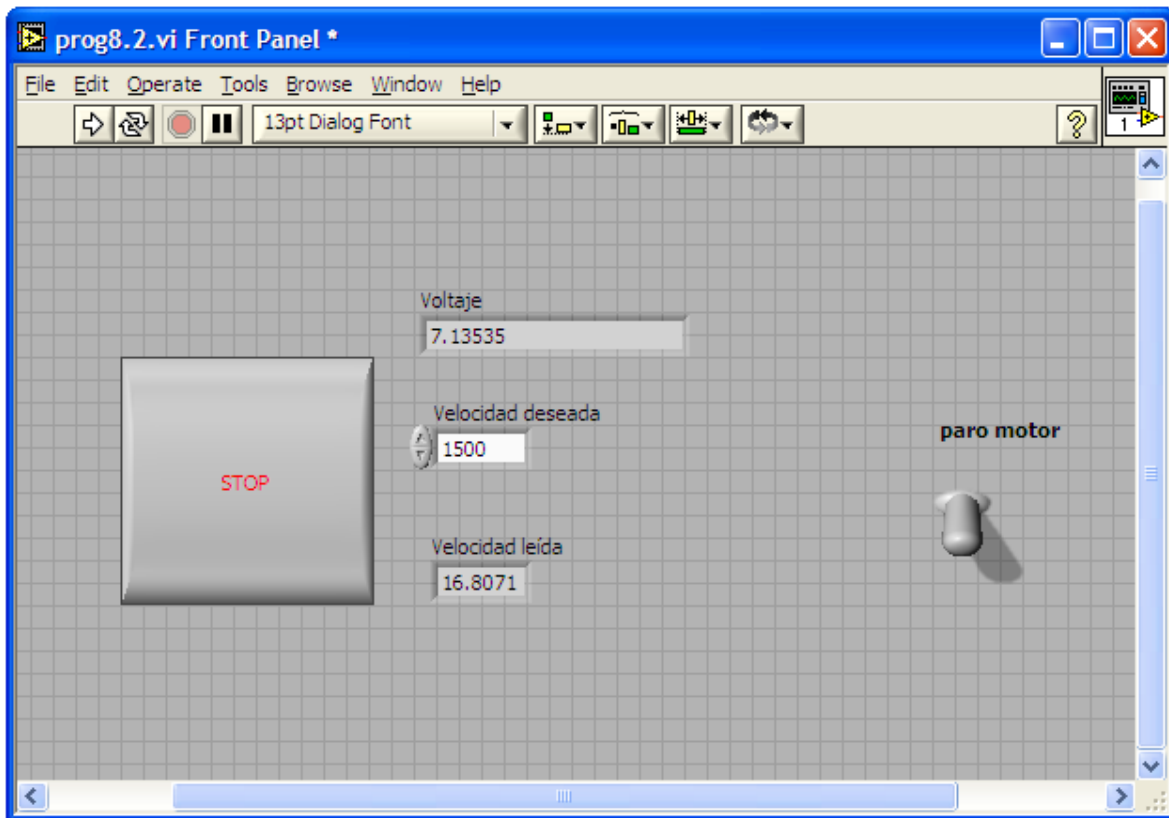


Figura 7.25: Panel frontal del programa LabView para el control de velocidad de un motor.

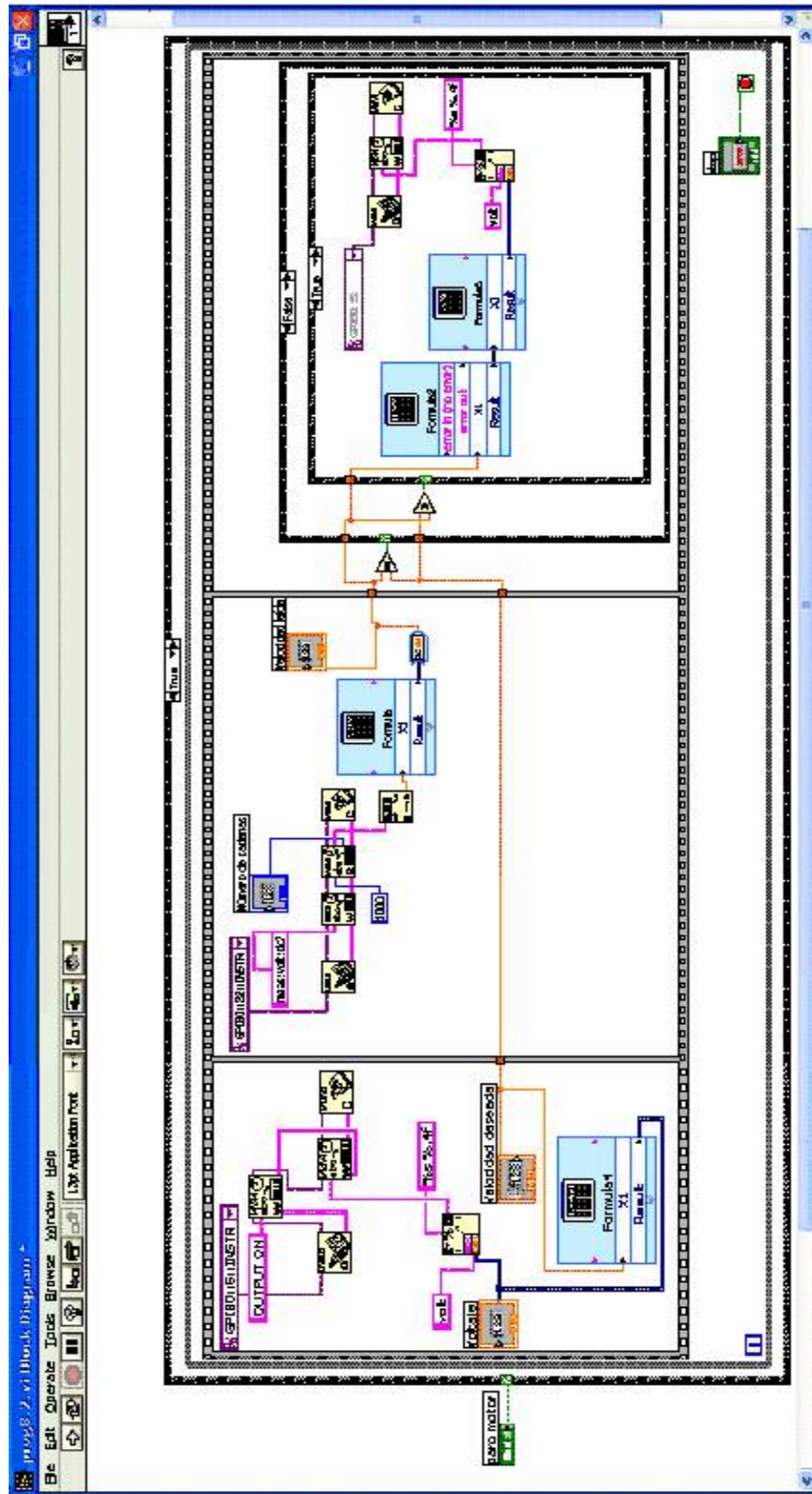


Figura 7.26: Diagrama a bloques del programa LabView para el control de velocidad de un motor.

1.3 Y un VISA Close para cerrar la sesión de VISA.

- 2. En el Case True se utiliza la herramienta While Loop para que se cicle el programa.**
- 3. Se utiliza también la herramienta Flat Sequence es se ocupa para realizar sucesiones en el programa.**
- 4. La herramienta VISA Open se utiliza para abrir la sesión de VISA.**

4.1 La constante que se le pone al VISA Open es para poner la dirección de GPIB en donde esta conectado la fuente.

- 5. La primer herramienta VISA Write se utiliza para escribir las instrucciones requeridas.**

En la constante se escribe "OUTPUT ON", para que se encienda la fuente.

- 6. La segunda herramienta VISA Write se utiliza para escribir las instrucciones que se necesitan. En esta práctica se escribirá "VOLT".**
- 7. El Format into String es para convertir números o instrucciones a cadenas para que puedan entender los aparatos.**

7.1 La constante del lado izquierdo es para escribir la instrucción VOLT.

7.2 La constante de arriba es para decir que se va a leer una serie de números con cuatro cifras significativas.

7.3 El cuarto pin del lado izquierdo se conecta al Result de la Fórmula y el indicador mostrará el valor de voltaje que se necesita para la velocidad deseada.

7.4 En la herramienta de Fórmula se ingresa la ecuación $(X1+162.99595)/233.06429$, se utiliza para obtener el valor de voltaje.

8 La herramienta VISA Close cierra la sesión de VISA y se pasa a la segunda parte de Flat Sequense.

9 Se utiliza nuevamente VISA Open para abrir una nueva sesión de VISA, pero en este caso en la constante se selecciona la dirección del multímetro digital.

10 El VISA Write se utiliza para ingresar la instrucción deseada en este caso “MEAS:VOLT:DC?” .

11 La herramienta VISA Read es para recibir el valor de voltaje, en este caso, del tacogenerador que es proporcional al de la fuente.

11.1 La constante que se le pone al VISA Read es el número de cadenas o strings que queremos que lea.

11.1.1 En este caso se puede poner casi cualquier valor la diferencia es que en el número que está en Read buffer nos dirá cuantas cadenas lee por lo que no es necesario poner un número muy grande mientras sea mayor o igual al de Read buffer.

11.1.2 El indicador (Oculto) del lado izquierdo nos indica el número de cadenas que esta leyendo el programa en ese momento y con esto sabremos que rango se puede utilizar.

11.2 El indicador (Oculto) del lado derecho nos sirve para saber el valor real de voltaje y este se da en ascii. El valor en ascii es en notación científica.

12 El Fract/Exp String to Number se utiliza porque el GPIB envía cadenas y esta herramienta es un convertidor de cadena a número.

13 La Formula es para transformar el valor de voltaje obtenido en la velocidad leída.

13.1 El Convert from Dinamic Data como su nombre lo dice convierte el valor en un dato dinámico.

13.2 En el indicador que se le pone a la Formula se obtiene el valor de Velocidad leída.

14 La herramienta VISA Close nos sirve para cerrar la sesión de VISA y pasa a la tercera parte de Flat Sequense.

15 La herramienta “igual que” se utiliza para comparar la velocidad deseada con la velocidad leída y se abre un Case Structure para tener las dos opciones.

15.1 Si el Case es True se sale del programa y pide una nueva velocidad.

15.2 Si el Case es False Compara si la velocidad deseada es mayor que la leída.

16 Se abre nuevamente una Case Structure para tener dos opciones.

16.1 Si el Case es True se tiene que incrementar la velocidad leída para obtener la deseada.

16.1.1 Se introduce la siguiente ecuación en la primer Fórmula $((X1*100)+1)/100$, para incrementar y en la segunda Fórmula se introduce la siguiente ecuación $(X1-14.9753)/297.27853$ para cambiar de rpms a volts.

16.1.2 Se abre nuevamente una sesión de VISA con VISA Open y en la constante se selecciona la dirección de la fuente.

16.1.3 En el Format into String se le introduce la instrucción deseada, en este caso "VOLT" y en el cuarto pin el resultado de la segunda fórmula, para convertirlos en cadenas y la constante de arriba es para leer una serie de números con cuatro cifras significativas.

16.1.4 En el VISA Write se le manda una secuencia de cadenas desde el Format into String.

16.1.5 Se cierra la sesión de VISA con el VISA Close, se sale del programa y espera un nuevo valor de velocidad.

16.2 Si el Case es False se tiene que decrementar la velocidad leída para obtener la deseada.

16.2.1 Se introduce la siguiente ecuación en la primer Fórmula $((X1*100)-1)/100$, para decrementar la velocidad leída y se repiten los pasos del Case True.

17 Las conexiones de todos los VISA del primer y último pin es porque así reciben toda la dirección de GPIB y el error.

18 En el Panel Frontal el botón de Stop es para parar el programa.

19 El Swith es para detener el motor.

20 El Ítem de voltaje indica el voltaje requerido para la velocidad deseada.

21 El Ítem de la velocidad deseada es en donde ingresamos el valor en rpms deseado.

22 El Ítem de la velocidad leída indica la velocidad que esta leyendo en ese momento.

Conclusiones

GPIB sigue siendo el estándar dominante. Esto se debe a:

- (a) El hecho de que ninguno de los competidores se haya impuesto como predominante.
- (b) La rapidez con la que cambia la tecnología PC, en contraste con la de los instrumentos de medida, que tienen un tiempo de vida más largo.
- (c) El elevado coste de dichos instrumentos, que les confiere una gran inercia para actualizarse.

Por el momento, muchos fabricantes ofrecen adaptadores que permiten conectar instrumentos GPIB, una red Ethernet, operando de una forma más o menos transparente para el programador. Asimismo, el estándar VISA ha simplificado considerablemente el desarrollo de software, al hacer, hasta cierto punto, 'transparente' al bus físico de conexión.

Los instrumentos de medida de gama alta mantienen abiertas todas las puertas para la conexión. Es el caso por ejemplo de los osciloscopios de la serie Infiniium de Agilent, que incorporan de serie dos puertos USB, dos puertos PS/2, GPIB y conexión a Ethernet. El inconveniente: su precio, más parecido al de un coche que a un osciloscopio.

Este equipo ya está vigente en el Centro de Investigaciones Avanzadas de Ingeniería Industrial y no tendremos que realizar ningún gasto, el material ya presentado nos da a conocer a la perfección la implementación del bus y equipo a utilizar, nos permite conectar otros dispositivos y es compatible con diferentes tipos de marcas y equipos de software, la utilización es sencilla para que cualquier estudiante a nivel licenciatura pueda implementarlo.

En el Instituto de Ciencias Básicas e Ingeniería contamos con varias asignaturas, las cuales tienen un deficiente manual y uso de prácticas de laboratorio, este equipo nos permite realizarlas en menor tiempo con mayor exactitud ya que no existe el error humano en las lecturas.

Las prácticas expuestas en este documento son comprobables y realizables.

Referencia

- [1] Guía de instalación y configuración de la tarjeta Agilent 82350B PCI GPIB Interface.
- [2] Manual de usuario de Agilent Technologies E364xA Single Output DC Power Supplies.
- [3] Manual de usuario de la fuente Agilent E3644A
- [4] Manual de usuario del multímetro Agilent 34401A
- [5] Manual de usuario del Nanovoltmetro digital 34420A
- [6] http://cipres.cec.uchile.cl/~iq54a/apuntes/instrumentos/03_t_stores/t_stores.html
- [7] <http://digital.ni.com/worldwide/latam.nsf/b63ef100ab4b5df486256425006883b7/e8519bf5930117cd86256da6005c3cf5?OpenDocument>
- [8] http://es.wikipedia.org/wiki/Diferencia_de_potencial
- [9] <http://es.wikipedia.org/wiki/GPIB>
- [10] <http://es.wikipedia.org/wiki/LabVIEW>
- [11] http://es.wikipedia.org/wiki/Protocolo_de_red
- [12] <http://es.wikipedia.org/wiki/RTD>
- [13] <http://es.wikipedia.org/wiki/Termopar>
- [14] <http://motor.terra.es/motor/servicio/definicion.cfm?id=317>
- [15] <http://proton.ucting.udg.mx/temas/control/memo/MEMO.html>
- [16] <http://proton.ucting.udg.mx/tutorial/>
- [17] <http://www.addlink.es/productos.asp?pid=268>
- [18] <http://www.angelfire.com/empire/seigfrid/FuenteDevoltaje.html>
- [19] <http://www.ciencia-ficcion.com/glosario/v/voltaje.htm>
- [20] http://www.ctr.unican.es/asignaturas/instrumentacion_5_IT/GPIB1_04.pdf
- [21] http://www.ctr.unican.es/asignaturas/instrumentacion_5_IT/GPIB2_04.pdf
- [22] <http://www.datsi.fi.upm.es/docencia/Arquitectura/ieee.pdf>
- [23] <http://www.dccia.ua.es/dccia/inf/asignaturas/ROB/optativos/Sensores/internos.html>

- [24] <http://www.gmelectronica.com.ar/catalogo/pag69.html>
- [25] http://www.gte.us.es/~galvan/IE_4T/Tutorial%20de%20Labview.pdf
- [26] <http://www.iai.csic.es/users/fseco/teaching/gpib.pdf>
- [27] <http://www.metas.com.mx/guiametas/La-Guia-MetAs-02-07-TC.pdf>
- [28] <http://www.monografias.com/trabajos3/transductores/transductores.shtml>
- [29] http://www.sapiensman.com/medicion_de_temperatura/termistores.htm
- [30] <http://www.tracnova.com/control%20gpib.htm>
- [31] <http://www.tracnova.com/inter%20gpib1.htm>
- [32] <http://www.tracnova.com/inter%20gpib2.htm>
- [33] <http://www.tracnova.com/inter%20gpib3.htm>
- [34] <http://www.tracnova.com/inter%20gpib4.htm>
- [35] <http://www.uib.es/depart/dfs/GTE/staff/jfont/InstrETI/norma488.pdf>
- [36] <http://www.virtual.unal.edu.co/cursos/sedes/manizales/4040003/lecciones/cap4lecc7.h>

Anexo

LABVIEW

LabVIEW constituye un revolucionario sistema de programación gráfica para aplicaciones que involucren adquisición, control, análisis y presentación de datos. Algunas ventajas que proporciona el empleo de LabVIEW son:

- Reduce el tiempo de desarrollo de las aplicaciones al menos de 4 a 10 veces, ya que es muy intuitivo y fácil de aprender.
- Dota de gran flexibilidad al sistema, permitiendo cambios y actualizaciones tanto del hardware como del software.
- Da la posibilidad a los usuarios de crear soluciones completas y complejas.
- Es un sistema de desarrollo que integra funciones de adquisición, análisis y presentación de datos.
- Este está dotado de un compilador gráfico para lograr la máxima velocidad de ejecución posible.
- Tiene la posibilidad de incorporar aplicaciones escritas en otros lenguajes.

Es un entorno de programación destinado al desarrollo de aplicaciones, similar a los sistemas de desarrollo comerciales que utilizan el lenguaje C o BASIC. Sin embargo, se diferencia de dichos programas en un importante aspecto: los citados lenguajes de programación se basan en líneas de texto para crear el código fuente del programa, mientras que emplea la programación gráfica o lenguaje G para crear programas basados en diagramas de bloques.

Para el empleo de LabVIEW no se requiere gran experiencia en programación, ya que se emplean iconos, términos e ideas familiares a científicos e ingenieros, y se apoya sobre símbolos gráficos en lugar de lenguaje escrito para construir las aplicaciones. Por ello resulta mucho más intuitivo que el resto de lenguajes de programación convencionales.

Posee extensas librerías de funciones y subrutinas. Además de las funciones básicas de todo lenguaje de programación, incluye librerías específicas para la adquisición de datos, control de instrumentación VXI, GPIB y comunicación serie, análisis presentación y guardado de datos.

También proporciona potentes herramientas que facilitan la depuración de los programas.

FUNCIONAMIENTO LABVIEW

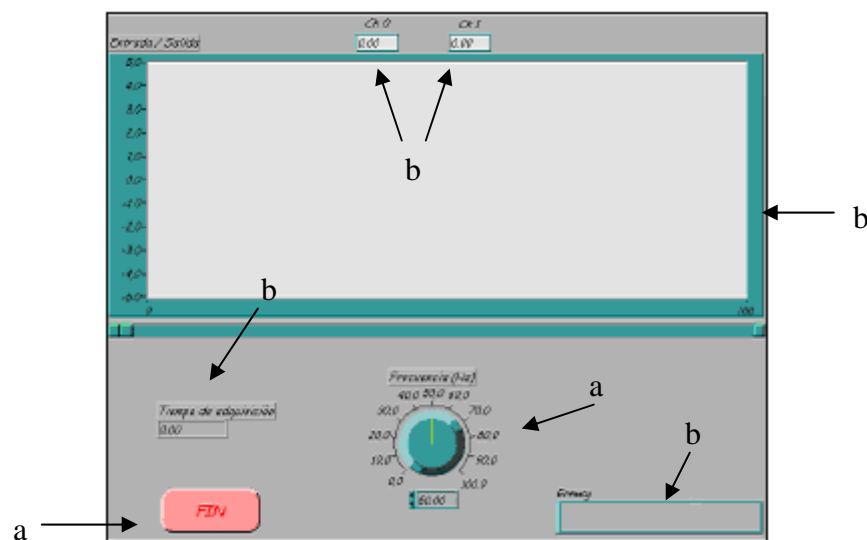
Los programas desarrollados mediante LabVIEW se denominan *Instrumentos Virtuales* (VIs), porque su apariencia y funcionamiento imitan los de un instrumento real. Sin embargo son análogos a las funciones creadas con los lenguajes de programación convencionales. Tienen una parte interactiva con el usuario y otra parte de código fuente, y aceptan parámetros procedentes de otros VIs.

Todos tienen un panel frontal(A) y un diagrama de bloques(B). Las paletas(C) contienen las opciones que se emplean para crear y modificarlos. A continuación se procederá a realizar una somera descripción de estos conceptos.

A) Panel Frontal

Se trata de la interfaz gráfica del VI con el usuario. Esta interfaz recoge las entradas procedentes del usuario y representa las salidas proporcionadas por el programa. Está formado por una serie de botones, pulsadores, potenciómetros, gráficos.

Cada uno de ellos puede estar definido como un control (a) o un indicador (b). Los primeros sirven para introducir parámetros al VI, mientras que los indicadores se emplean para mostrar los resultados producidos, ya sean datos adquiridos o resultados de alguna operación.



Panel frontal de LabView.

Nota: “a” es la función y “b” son las terminales (control e indicador)

B) Diagrama de bloques

Constituye el código fuente del VI. En este es donde se realiza la implementación del programa del VI para controlar o realizar cualquier procesamiento de las entradas y salidas que se crearon en el panel frontal.

Incluye funciones y estructuras integradas en las librerías que incorpora LabVIEW. En el lenguaje G las funciones y las estructuras son *nodos* elementales. Son análogas a los operadores o librerías de funciones de los lenguajes convencionales.

Los controles e indicadores que se colocaron previamente en el Panel Frontal, se materializan en el diagrama de bloques mediante los terminales. A continuación se presenta un ejemplo de lo recién citado:

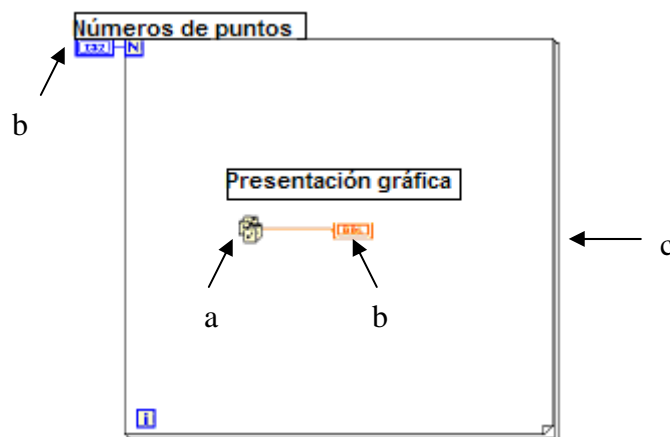


Diagrama de bloques de LabView.

- (a) Función.
- (b) Terminales (control e indicador).
- (c) Estructura.

Se construye conectando los distintos objetos entre sí, como si de un circuito se tratará. Este bus une terminales de entrada y salida con los objetos correspondientes, y por ellos fluyen los datos.

LabVIEW posee una extensa biblioteca de funciones, entre ellas, aritméticas, comparaciones, conversiones, funciones de entrada/salida, de análisis.

Las estructuras, similares a las declaraciones causales y a los bucles en lenguajes convencionales, ejecutan el código que contienen de forma condicional o repetitiva (bucle for, while, case,...).

Los cables son las trayectorias que siguen los datos desde su origen hasta su destino, ya sea una función, una estructura, un terminal, etc. Cada cable tiene un color o un estilo diferente, lo que diferencia unos tipos de datos de otros.[16]

C) Paletas.

Las paletas de LabVIEW proporcionan las herramientas que se requieren para crear y modificar tanto el panel frontal como el diagrama de bloques. Existen las siguientes paletas:

Paleta de herramientas (Tools palette)

Se emplea tanto en el panel frontal como en el diagrama de bloques. Contiene las herramientas necesarias para editar y depurar los objetos tanto del panel frontal como del diagrama de bloques.



Paleta de herramientas de LabView.

Las opciones que presenta esta paleta son las siguiente:



Operating tool – Cambia el valor de los controles.



Positioning tool – Desplaza, cambia de tamaño y selecciona los objetos.



Labeling tool – Edita texto y crea etiquetas.



Wiring tool – Une los objetos en el diagrama de bloques.



Object Pop-up Menu tool – Abre el menú desplegable de un objeto.



Scroll tool – Desplaza la pantalla sin necesidad de emplear las barras de desplazamiento.



Breakpoint tool – Fija puntos de interrupción de la ejecución del programa en VIs, funciones y estructuras.



Probe tool – Crea puntos de prueba en los cables, en los que se puede visualizar el valor del dato que fluya por dicho cable en cada instante.



Color Copy tool – Copia el color para después establecerlo mediante la siguiente herramienta.



Color tool – Establece el color de fondo y el de los objetos.

Paleta de controles (Controls palette)

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



Paleta de Controles de LabView.

El menú Controls de la ventana correspondiente al panel frontal contiene las siguientes opciones:



Numeric – Para la introducción y visualización de cantidades numéricas.



Boolean – Para la entrada y visualización de valores booleanos.



String & Table – Para la entrada y visualización de texto.



List & Ring – Para visualizar y/o seleccionar una lista de opciones.



Array & Cluster – Para agrupar elementos.



Graph – Para representar gráficamente los datos.



Path & RefNum – Para gestión de archivos.



Decorations – Para introducir decoraciones en el panel frontal. No visualizan datos.



User Controls – Para elegir un control creado por el propio usuario.



ActiveX – Para transferir datos y programas de unas aplicaciones a otras dentro de windows.



Select a Control – Para seleccionar cualquier control.

Al seleccionar objetos desde el menú Controls estos aparecen sobre el panel frontal, pueden colocarse donde convenga, y además tienen su propio menú desplegable que permite la configuración de algunos parámetros específicos de cada tipo de control.

Paleta de funciones (functions palette)

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



Paleta de funciones de LabView.

Para seleccionar una función o estructura concretas, se debe desplegar el menú Functions y elegir entre las opciones que aparecen. A continuación se enumeran todas ellas, junto con una pequeña definición.



Structures – Muestra las estructuras de control del programa, junto con las variables locales y globales.



Numeric – Muestra funciones aritméticas y constantes numéricas.



Boolean – Muestra funciones y constantes lógicas.



String – Muestra funciones para manipular cadenas de caracteres, así como constantes de caracteres.



Array – Contiene funciones útiles para procesar datos en forma de vectores, así como constantes de vectores.



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



Comparison – Muestra funciones que sirven para comparar números, valores booleanos o cadenas de caracteres.



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



File I/O – Muestra funciones para operar con ficheros.



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



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



Data Acquisition – Contiene a su vez un submenú donde puede elegirse entre distintas librerías referentes a la adquisición de datos.



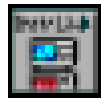
Analysis – Contiene un submenú en el que se puede elegir entre una amplia gama de funciones matemáticas de análisis.



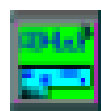
Tutorial – Incluye un menú de VIs que se utilizan en el manual LabVIEW Tutorial.



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



Instrument drivers – En él se muestran los drivers disponibles de distintos instrumentos.



User Libraries – Muestra as librerías definidas por el usuario. En este caso, la librería mostrada contiene los drivers de la tarjeta de adquisición de datos de Advantech.



Application control – Contiene varias funciones que regulan el funcionamiento de la propia aplicación en ejecución.



Select a VI – Permite seleccionar cualquier VI para emplearlo como subVI.

PROGRAMACIÓN EN LABVIEW

Con el entorno gráfico de programación de LabVIEW se comienza a programar a partir del panel frontal.

En primer lugar se definirán y seleccionarán de la paleta de controles todos los controles (entradas que dará el usuario) e indicadores (salidas que presentará en pantalla el VI) que se emplearán para introducir los datos por parte del usuario y presentar en pantalla los resultados.

Una vez colocados en la ventana correspondiente al panel frontal todos los objetos

necesarios, debe pasarse a la ventana Diagram (menú Windows > Show Diagram), que es donde se realiza la programación propiamente dicha (diagrama de bloques). Al abrir esta ventana, en ella se encuentran los terminales correspondientes a los objetos situados en el panel frontal, dispuestos automáticamente por LabVIEW.

Se deben ir situando las funciones, estructuras, etc. que se requieran para el desarrollo del programa, las cuales se unen a los terminales mediante cables.

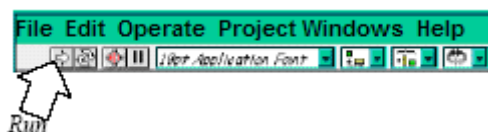
Para facilitar la tarea de conexión de todos los terminales, en el menú “Help” puede elegirse la opción “Show Help”, con lo que al colocar el cursor del ratón sobre un elemento aparece una ventana con información relativa a éste (parámetros de entrada y salida).

Además, si se tiene seleccionado el cursor de cableado, al situar éste sobre un elemento se muestran los terminales de forma intermitente.

EJECUCIÓN DE UN VI

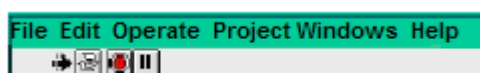
Una vez se ha concluido la programación del VI se debe proceder a su ejecución. Para ello la ventana activa debe ser el panel frontal (si se está en la ventana del diagrama de bloques, se debe seleccionar la opción Show Panel del menú Window).

Una vez situados en el panel frontal, se pulsará el botón de **run**, situado en la barra de herramientas.



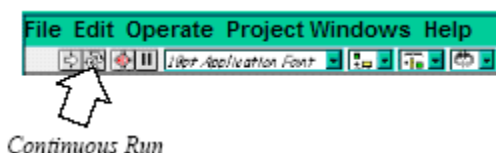
Barra de herramientas mostrando el botón Run.

El programa comenzará a ejecutarse. Mientras dura la ejecución del mismo, la apariencia del botón de Run es la que se muestra a continuación:



Se muestra el cambio del botón Run mientras se ejecuta el programa.

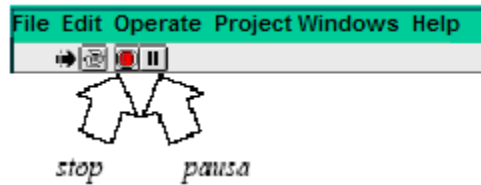
De este modo el programa se ejecutará una sola vez. Si se desea una ejecución continua, se pulsará el botón situado a la derecha del de Run (Continuous Run). Si durante el funcionamiento continuo del programa se vuelve a pulsar el citado botón, se finalizará la última ejecución del mismo, tras lo cual el programa se parará.



Barra de herramientas mostrando el botón Continuous Run.

Para finalizar la ejecución de un programa se puede operar de dos formas. La primera, y la más aconsejable, es emplear un botón en el panel frontal del VI, cuya pulsación produzca la interrupción del bucle de ejecución de la aplicación. La segunda forma de detener la ejecución del VI es pulsando el botón de pausa o el de stop. La diferencia entre ambos es que si se pulsa stop, la ejecución del programa finaliza inmediatamente, mientras que si se pulsa pausa, se produce

una detención en el funcionamiento del programa, retomándose su ejecución una vez se vuelve a pulsar el mismo botón.



Barra de herramientas mostrando el botón Stop.

ESTRUCTURAS

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

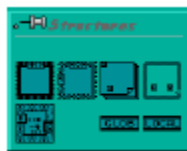


Figura 1.10: Tipos de estructuras.

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

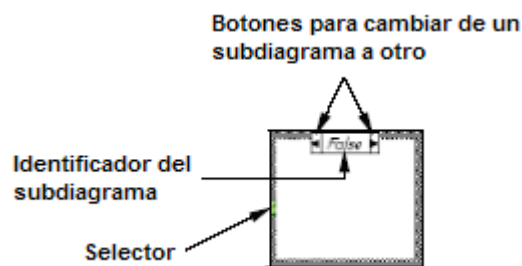
Un subdiagrama es una colección de nodos, cables y terminales situados en el interior del rectángulo que constituye la estructura. El For Loop y el While Loop

únicamente tienen un subdiagrama. El Case Structure y el Sequence Structure, sin embargo, pueden tener múltiples subdiagramas, superpuestos como si se tratara de cartas en una baraja, por lo que en el diagrama de bloques únicamente será posible visualizar al tiempo uno de ellos. Los subdiagramas se construyen del mismo modo que el resto del programa. Las siguientes estructuras se hallan disponibles en el lenguaje G.

Case Structure

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

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



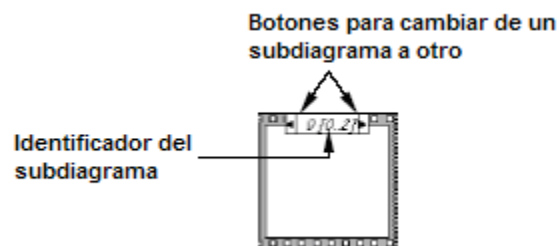
Case Structure.

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

Sequence Structure

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

También poseen un identificador del subdiagrama mostrado en su parte superior, con posibilidad de avanzar o retroceder a otros subdiagramas gracias a las flechas situadas a ambos lados del mismo.



Sequence Structure.

Esta estructura secuencia la ejecución del programa. Primero ejecutará el subdiagrama de la hoja (frame) nº0, después el de la nº 1, y así sucesivamente.

Para pasar datos de una hoja a otra se pulsará el botón derecho del ratón sobre el borde de la estructura, seleccionando la opción Add sequence local.



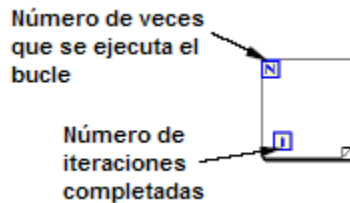
Forma de realizar dos Sequence Structure.

For Loop



Es el equivalente al bucle for en los lenguajes de programación convencionales.

Ejecuta el código dispuesto en su interior un número determinado de veces.



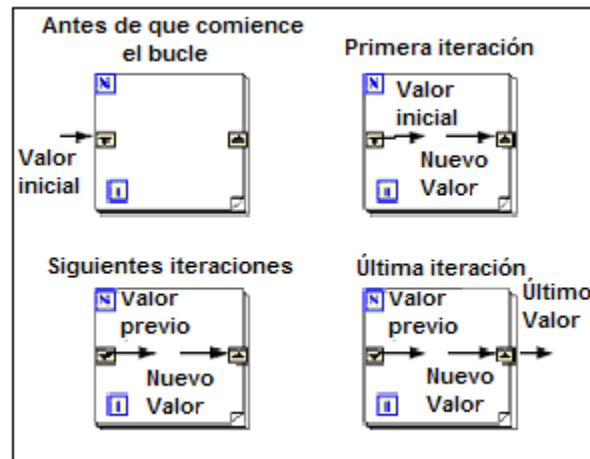
For Loop.

Ejecutar el bucle for es equivalente al siguiente fragmento de código:

```
For i = 0 o N - 1
Ejecutar el subdiagrama del interior del bucle
```

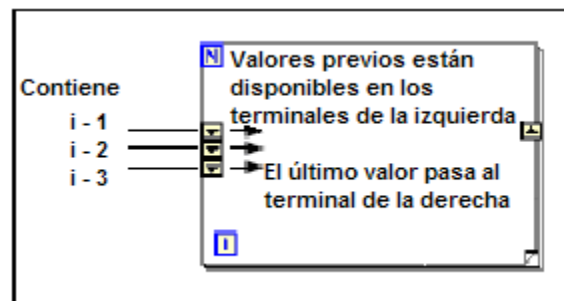
Código que realiza el For Loop.

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



Funcionamiento de For Loop.

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

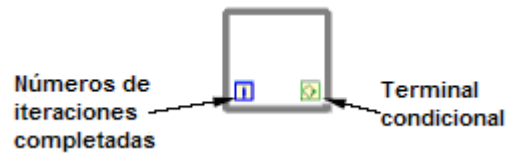


Registro de iteraciones.

While Loop



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



While Loop.

El bucle while es equivalente al código siguiente:

```

Do
Se ejecuta lo que hay en el interior del bloque
While terminal conditional is true

```


Código que realiza el While Loop.

El programa comprueba el valor de lo que se halle conectado al terminal condicional al finalizar el bucle. Por lo tanto, el bucle siempre se ejecuta al menos una vez.

Con esta estructura también se pueden emplear los shift registers para tener disponibles los datos obtenidos en iteraciones anteriores (es decir, para memorizar valores obtenidos).

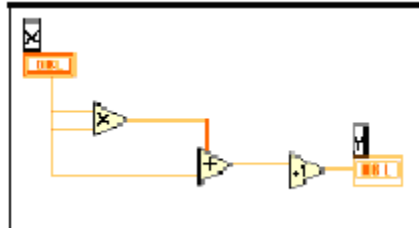
su empleo es análogo al de los bucles for, por lo que omitiré su explicación.

Formula Node

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

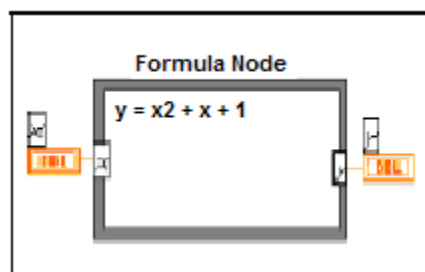
$$y = x^2 + x + 1$$

Empleando bloques pertenecientes al lenguaje G quedaría:



Ejemplo de la ecuación $y = x^2 + x + 1$.

Si se utiliza la formula node, se obtiene:



Formula Node.

Para definir una fórmula mediante esta estructura, se actuará del siguiente modo:

- En primer lugar, se deben definir las variables de entrada y las de salida. Para ello, se pulsa con el botón derecho del ratón sobre el borde de la formula node. A continuación se seleccionará Add Input o Add Output, según se trate de una entrada o una salida, respectivamente. Aparecerá un rectángulo, en el que se debe escribir el nombre de la variable (se distingue entre mayúsculas y minúsculas). Todas las variables que se empleen deben estar declaradas como entradas o salidas. Las que se empleen como variables intermedias se declararán como salidas, aunque posteriormente no se unan a ningún bloque posterior.

- Una vez definidas las variables a emplear, se escribirán la o las fórmulas en el interior del recuadro (para ello se emplea la labeling tool). Cada fórmula debe finalizar con un “;”.
- Los operadores y funciones que se pueden emplear se explican en la ayuda de LabVIEW, y son los que se muestran a continuación:

Operadores:						
asignación	=					
condicional	?:					
OR lógico						
AND lógico	&&					
relacionales	==	!=	>	<	>=	<=
aritméticos	+	-	*	/	^	
Funciones:						
abs acos acosh asin asinh atan atanh ceil cos csoh cot						
csc exp						
expml floor getexp getman int intrz ln lnp1 log log2						
max min mod rand ram sec sgn sin sinc sinh sqrt tan						
tanh						

Operadores y funciones.

La sintaxis de una expresión incondicional es la siguiente:

<expresión condicional> ? <texpresión>: <fexpresión>

Expresión incondicional.

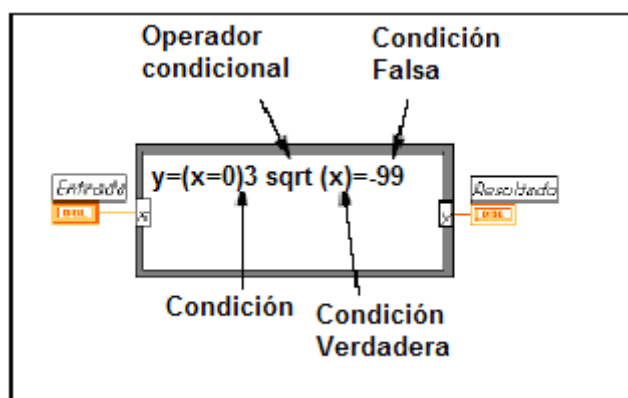
Si el valor lógico de la expresión condicional es true se ejecutará texpresión. Si, por el contrario, fuese false, lo que se aplicará será fexpresión

Como ejemplo considérese el siguiente fragmento de código:

```
if (x >= 0) then  
y = sqrt (x)  
else  
y = -99  
end if
```

Fragmento de código para utilizar Formula Node

Se puede implementar este fragmento de código empleando un formula node, tal y como se muestra en la siguiente figura:



Ejemplo en Formula Node.

Glosario

ANSI: El Instituto Nacional Estadounidense de Estándares (ANSI, por sus siglas en inglés: American National Standards Institute) es una organización sin ánimo de lucro que supervisa el desarrollo de estándares para productos, servicios, procesos y sistemas en los Estados Unidos. ANSI es miembro de la Organización Internacional para la Estandarización (ISO) y de la Comisión Electrotécnica Internacional (International Electrotechnical Commission, IEC). La organización también coordina estándares del país estadounidense con estándares internacionales, de tal modo que los productos de dicho país puedan usarse en todo el mundo.

Asíncrono: Es un componente de frecuencia en una firma de vibración, que no es un múltiplo entero de la velocidad de revolución o un armónico de la misma.

Flanco: En una señal digital, se denomina flanco a la transición del nivel bajo al alto (flanco de subida) o del nivel alto al bajo (flanco de bajada).

Bus: Es un subsistema que transfiere datos o electricidad entre componentes del ordenador dentro de un ordenador o entre ordenadores. A diferencia de una conexión punto-a-punto, un bus puede conectar mediante lógica varios periféricos utilizando el mismo conjunto de cables. Es el canal por el que circula información electrónica en forma de bits. El ancho de bus es el número de bits transmitidos simultáneamente por el bus.

CEI: La Comisión Electrotécnica Internacional (CEI) (IEC por sus siglas inglesas) es una organización de normalización en los campos eléctrico, electrónico y tecnologías relacionadas. Numerosas normas se desarrollan conjuntamente con la ISO (normas ISO/IEC).

Conector DIN: Es un conector que fue originariamente estandarizado por el Deutsches Institut für Normung (DIN), la organización de estandarización alemana.

Existen estándares DIN para una gran cantidad de diferentes conectores, por lo tanto el término "Conector DIN" por si sólo es ligeramente ambiguo ya que no identifica a un conector particular a menos que se le añada un número de documento que ilustre del tipo de conector en particular

Commodore pet/cbm: Es un programa de comunicación que enlaza dos o más equipos a una computadora personal.

Ethernet: Es el nombre de una tecnología de redes de computadoras de área local (LANs) basada en tramas de datos. El nombre viene del concepto físico de ether. Ethernet define las características de cableado y señalización de nivel físico y los formatos de trama del nivel de enlace de datos del modelo OSI. Ethernet se refiere a las redes de área local y dispositivos bajo el estándar IEEE 802.3 que define el protocolo CSMA/CD, aunque actualmente se llama Ethernet a todas las redes cableadas que usen el formato de trama descrito más abajo, aunque no tenga CSMA/CD como método de acceso al medio.

FireWire: El IEEE 1394 o FireWire o i.Link es un estándar multiplataforma para entrada/salida de datos en serie a gran velocidad. Suele utilizarse para la interconexión de dispositivos digitales como cámaras digitales y videocámaras a ordenadores.

IEEE: Corresponde a las siglas de The Institute of Electrical and Electronics Engineers, el Instituto de Ingenieros Eléctricos y Electrónicos, una asociación técnico-profesional mundial dedicada a la estandarización, entre otras cosas. Es la mayor asociación internacional sin fines de lucro formada por profesionales de las nuevas tecnologías, como ingenieros eléctricos, ingenieros en electrónica, ingenieros en sistemas e ingenieros en telecomunicación.

Instrumento virtual: Consiste en una computadora del tipo industrial, o una estación de trabajo, equipada con poderosos programas (software), hardware

económico, tales como placa para insertar, y manejadores (drives) que cumplen, en conjunto, las funciones de instrumentos tradicionales.

Interfaz: es la parte del programa informático que permite el flujo de información entre varias aplicaciones o entre el propio programa y el usuario. Metafóricamente se entiende la Interfaz como conversación entre el usuario y el sistema (o entre el usuario y el diseñador).

ISA: Industry Standard Architecture (en inglés, Arquitectura Estándar Industrial), casi siempre abreviado ISA, es una arquitectura de bus creada por IBM en 1980 en Boca Raton, Florida para ser empleado en los IBM PCs.

LabVIEW: Es un sistema de programación gráfica para aplicaciones que involucren adquisición, control, análisis y presentación de datos.

Nodo: Son los círculos numerados que forman parte del diagrama de red y representan el principio y el fin de las actividades que intervienen en el proyecto.

PCI: Peripheral Component Interconnect ("Interconexión de Componentes Periféricos") consiste en un bus de ordenador estándar para conectar dispositivos periféricos directamente a su placa base. Estos dispositivos pueden ser circuitos integrados ajustados en ésta (los llamados "dispositivos planares" en la especificación PCI) o tarjetas de expansión que se ajustan en conectores. Es común en PCs, donde ha desplazado al ISA como bus estándar, pero también se emplea en otro tipo de ordenadores. La especificación PCI cubre el tamaño físico del bus, características eléctricas, cronómetro del bus y sus protocolos. El grupo de interés especial de PCI (PCI Special Interest Group)

Parser: Es un programa que reconoce si una o varias cadenas de caracteres forman parte de un determinado lenguaje.

Plotter: Un plotter o trazador gráfico es un dispositivo de impresión conectado a una computadora, y diseñado específicamente para trazar gráficos vectoriales ó

dibujos lineales: planos, dibujos de piezas, etc. Efectúa con gran precisión impresiones gráficas que una impresora no podría obtener. Son usados en varios campos, tales como ambientes científicos, la ingeniería, el diseño, la arquitectura, etc. Muchos son monocromáticos, pero los hay de cuatro colores e incluso hay modelos que llegan a poseer hasta ocho colores.

Slot: Ranura, en español. Se trata de cada uno de los alojamientos que tiene la placa madre en los que se insertan las tarjetas de expansión.

Tarjeta PCMCIA: Es un dispositivo normalmente utilizado en computadoras portátiles para expandir las capacidades de este. Estas tarjetas reciben su nombre del estándar PCMCIA (estándar) (Personal Computer Memory Card International Association) Asociación Internacional de Tarjetas de Memoria para Computadoras Personales. Esta asociación de industria de fabricantes de hardware para ordenadores o computadoras portátiles encargada de la elaboración de estándares). Se usan para ampliar capacidades en cuanto a: memoria, disco duro, tarjeta de red, capturadora de radio y tv, puerto paralelo, puerto serial, puerto USB, etc.

Tri-state: Control para las líneas del bus GPIB

TTL: Acrónimo Inglés de Transistor-Transistor Logic o "Lógica Transistor a Transistor". Tecnología de construcción de circuitos electrónicos digitales, en los que los elementos de entrada de la red lógica son transistores, así como los elementos de salida del dispositivo.

USB: El Bus de Serie Universal (USB, de sus siglas en inglés Universal Serial Bus) es una interfaz que provee un estándar de bus serie para conectar dispositivos a un ordenador personal (generalmente a un PC). Un sistema USB tiene un diseño asimétrico, que consiste en un solo servidor y múltiples dispositivos conectados en serie para ampliar la gama de conexión, en una estructura de árbol utilizando concentradores especiales. Se pueden conectar

hasta 127 dispositivos a un sólo servidor, pero la suma debe incluir a los concentradores también, así que el total de dispositivos realmente usables es algo menor.