UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERIA Y ARQUITECTURA

ESCUELA DE INGENIERIA ELECTRICA

Diseño de procedimientos para control de instrumentos electrónicos.

PRESENTADO POR:

PEDRO BOANERGES PAZ ROMERO

PARA OPTAR AL TITULO DE:

INGENIERO ELECTRICISTA

CIUDAD UNIVERSITARIA, MARZO DE 2015

UNIVERSIDAD DE EL SALVADOR

RECTOR :

ING. MARIO ROBERTO NIETO LOVO

SECRETARIA GENERAL :

DRA. ANA LETICIA ZAVALETA DE AMAYA

FACULTAD DE INGENIERIA Y ARQUITECTURA

DECANO :

ING. FRANCISCO ANTONIO ALARCÓN SANDOVAL

SECRETARIO :

ING. JULIO ALBERTO PORTILLO

ESCUELA DE INGENIERIA ELECTRICA

DIRECTOR :

MSc. e ING. JOSÉ WILBER CALDERÓN URRUTIA

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERIA Y ARQUITECTURA

ESCUELA DE INGENIERIA ELECTRICA

Trabajo de Graduación previo a la opción al Grado de:

INGENIERO ELECTRICISTA

Título :

Diseño de procedimientos para control de instrumentos electrónicos.

Presentado por :

PEDRO BOANERGES PAZ ROMERO

Trabajo de Graduación Aprobado por:

Docente Asesor :

ING. JOSÉ ROBERTO RAMOS LÓPEZ

San Salvador, Marzo de 2015

Trabajo de Graduación Aprobado por:

Docente Asesor :

ING. JOSÉ ROBERTO RAMOS LÓPEZ

ACTA DE CONSTANCIA DE NOTA Y DEFENSA FINAL

En esta fecha, viernes 12 de diciembre de 2014, en la Sala de Lectura de la Escuela de Ingeniería Eléctrica, a las 10:00 horas, en presencia de las siguientes autoridades de la Escuela de Ingeniería Eléctrica de la Universidad de El Salvador:

1. Ing. José Wilber Calderón Urrutia
   Director — Firma: Wilber Calderón

2. Ing. Salvador de Jesús Germán
   Secretario — Firma:

Y, con el Honorable Jurado de Evaluación integrado por las personas siguientes:

1- Ing. José Roberto Ramos López — Firma:

2- Msc. e Ing. Carlos Osmín Pocasangre Jiménez

3- Ing. Walter Leopoldo Zelaya Chicas

Se efectuó la defensa final reglamentaria del Trabajo de Graduación:

Diseño de procedimientos para control de instrumentos electrónicos.

A cargo del Bachiller:

- Paz Romero, Pedro Boanerges

Habiendo obtenido en el presente Trabajo una nota promedio de la defensa final, de: 6.0

(SEIS PUNTO CERO)

AGRADECIMIENTOS

Agradezco primeramente a Dios por darme las fuerzas, la salud, la protección, el coraje, la sabiduría para seguir hasta el final y culminar esta meta y por poner en mi camino a todas las personas que de una u otra forma me ayudaron en este largo camino lleno obstáculos y retos, agradezco a mi familia que siempre ha estado pendiente de mi caminar en esta carrera, a mis amigos cercanos y lejanos que siempre me dieron una palabra de ánimo, de consuelo y apoyo para que yo no desistiera, al personal Administrativo y Docentes de la Escuela de Ingeniería Eléctrica por su paciencia y tolerancia.

Con todos estoy eternamente agradecido y deseo que se multiplique en ustedes todas las buenas intenciones, los deseos y la ayuda que me proporcionaron.

Bendiciones!!

Pedro Boanerges Paz Romero

ÍNDICE DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN.

En las actividades relacionadas con la ciencia y la tecnología surge siempre la necesidad de medir variables físicas ya sea de forma directa o indirecta para poder cualificar, cuantificar, controlar y tomar decisiones sobre un proceso. Un sistema de adquisición de datos es el instrumento del cual nos auxiliamos para obtener información de un determinado proceso. Estos sistemas de adquisición de datos pueden estar basados en instrumentos independientes o en sistemas modulares. Para el control de instrumentos independientes puede utilizarse el panel frontal de dichos instrumentos y hacerlo así de forma local y directa o se puede llevar a cabo de forma autónoma y remota mediante una computadora que permite la conexión de varios instrumentos a través de un bus de comunicación.

Hablando de instrumentos modulares, la interconexión entre los módulos del sistema se realiza a través de un bus de conexión local, es decir, el bus está embebido en todo el sistema lo que permite poder conectar más módulos a la aplicación.

En esta ocasión se abordará el control de instrumentos independientes a través de diferentes protocolos de comunicación, en particular se utilizaran los protocolos de comunicación GPIB y USBTMC. Se aplicarán estos protocolos de comunicación al multímetro de referencia Fluke 8508A (sólo GPIB), al generador de funciones AGEILENT 33210A(GPIB y USBTMC) y al osciloscopio AGILENT DSO1012A(sólo USBTMC) actualmente en uso en la EIE.

Cabe mencionar que el control de dichos instrumentos se llevará a cabo mediante el lenguaje de programación gráfico LabView y la librería para comunicación VISA(Virtual Instrument Software Architecture) que en conjunto permitirán el envío de comandos SCPI(Standard Commands for Programmable Instruments).

OBJETIVOS.

GENERALES:

- Diseñar procedimientos para completar las funciones remotas de control del Multímetro de referencia Fluke 8508A de la EIE y de otros sistemas de instrumentación que permiten el control vía GPIB, USBTMC y otros.

ESPECIFICOS:

- Complementar los procedimientos de control del Multímetro de Referencia Fluke 8508A.
- Complementar los procedimientos de control del generados de funciones AGILENT 33210A y divulgar esta tecnología en el curso Instrumentación Electrónica I que está siendo impartido durante el ciclo I 2014.
- Investigar la aplicabilidad de control vía instrumentos virtuales del osciloscopio AGILENT DSO1012A actualmente en uso en la EIE.

JUSTIFICACIÓN.

El laboratorio de la EIE cuenta con instrumentos de medida con la capacidad de ser controlados remotamente como lo son el osciloscopio AGILENT DSO1012A y el generador de funciones AGILENT 33210A y cuya capacidad no está siendo aprovechada, también se cuenta con un multímetro patrón Fluke 8508A que se utiliza en el campo de la metrología del cual no se han desarrollado todos los procedimientos para ser controlado remotamente. Todas estas capacidades en estos instrumentos pueden ser aprovechadas para ampliar la gama de conocimientos en la población estudiantil.

La realización de este trabajo atiende a la necesidad del conocimiento que nos permita la aplicación de procedimientos que permitan controlar remotamente los equipos (que lo permitan) existentes en el laboratorio de la EIE y con esto dar entrada a la cultura del control remoto de instrumentos de prueba y medida con el fin de una buena proyección profesional y así obtener un nivel y rendimiento más competitivo en el ámbito laboral para los estudiantes de esta y nuevas generaciones.

ALCANCES.

Se sumaran nuevos procedimientos para control remoto del multímetro de referencia Fluke 8508A así como también para el generador de funciones AGILENT 33210A, además con este último se expondrá como laboratorio demostrativo adicional en el curso de Instrumentación Electrónica I el uso del bus GPIB para control remoto de instrumentos. Se explorará las capacidades que tiene el osciloscopio AGILENT DSO1012A en cuanto a la comunicación en forma remota mediante el puerto USB utilizando la especificación USBTMC.

Finalmente se implementará la comunicación por medio de un cable GPIB entre el multímetro patrón y el generador de funciones con lo cual se mostrará las capacidades de dicho bus y todos controlados de forma remota por software.

En todos los casos, el alcance consiste en procedimientos funcionales desarrollados en LabView. Cabe mencionar que dichas aplicaciones serán documentadas adecuadamente, de manera que puedan ser continuadas por Trabajos de Graduación futuros en la EIE.

CAPITULO I: MARCO TEORICO

DESCRIPCIÓN GENERAL.

1.1 Control de instrumentos.

En muchas ocasiones la realización de una medición requiere la intervención de varios instrumentos, unos generaran estímulos sobre el dispositivo que se pretende medir y otros recogerán la respuesta a dichos estímulos. Este conjunto de instrumentos que hace posible la realización de la medición recibe el nombre de sistema de instrumentación. Todo sistema de instrumentación consta de instrumentos, un sistema de interconexión de los mismos y un controlador inteligente que gestiona el funcionamiento de todo el sistema y da las órdenes para que una medición se realice correctamente.

Los comienzos del control de instrumentos desde una computadora y de hecho de los sistemas de instrumentación, se sitúan a mediados de los años 60 cuando Hewlett Packard, desarrolló su bus para instrumentación HP-IB (Hewlett Packard Interface Bus) que permitía conectar su gama de instrumentos programables a una PC. Esta interfase ganó rápidamente gran popularidad y en 1975 fue aceptada como un estándar: el IEEE-488, de este protocolo se detallará más adelante en el documento.

En general para poder controlar un instrumento de prueba o medida o un grupo de ellos se seguirá el siguiente esquema ilustrado en la figura 1.1:

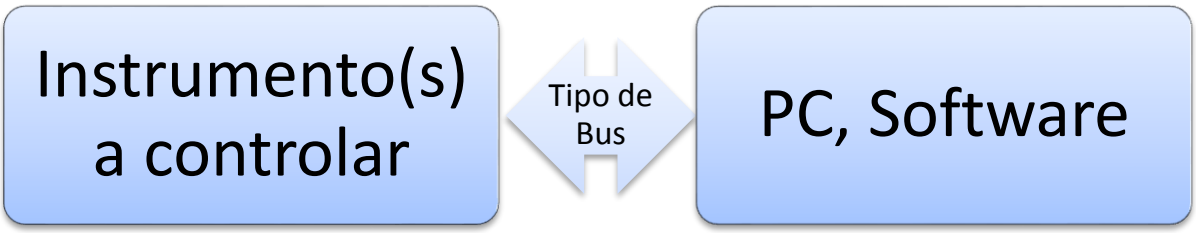

Figura 1-1 Diagrama General Para el Control de Instrumentos Electrónicos

- El bloque de Instrumentos, pueden todos aquellos instrumentos de prueba(Fuentes de Voltaje, Generadores de Señales, etc) que generan una señal eléctrica a su salida y todos los instrumentos de medida(Osciloscopios, Multímetros, Analizadores de Espectro, etc.) que puedan muestrear señales eléctricas, procesarlas y mostrarlas.

- El tipo de bus como se verá más adelante, es el puente que conecta al instrumento con la aplicación de software y el usuario final, y se selecciona de acuerdo a la aplicación que se requiera desarrollar, entre los buses más populares se pueden encontrar los siguientes:
  - **RS232**, Se trata de un antiguo estándar de comunicación serie que se diseñó para la comunicación entre un equipo terminal de datos (DTE), como un computador y un equipo de comunicaciones de datos (DCE), como un módem. Es un enlace de tipo full dúplex y punto a punto. La distancia máxima que abarca sin ningún circuito de ampliación es de aproximadamente 15m (en la práctica puede llegar a funcionar hasta con distancias de 100m) y su velocidad típica ronda los 19000 baudios aunque puede ser superior.
  - **RS422**, Se trata de otro estándar de comunicación serie full-dúplex que utiliza señales diferenciales. Es una interfaz muy apropiada para aplicaciones industriales que conectan un maestro con varios terminales. También permite la comunicación punto a punto entre dos nodos utilizando, generalmente, un par de cables trenzados para cada línea de señal (4 hilos). La transmisión en modo diferencial presenta como ventaja principal su inmunidad al ruido electromagnético (de modo común). Los unos y ceros lógicos se establecen en función de la diferencia de tensión entre ambos conductores. Permite establecer una mayor distancia de conexión, respecto a la interfaz serie RS-232, sobre todo, en entornos ruidosos como el industrial. Las distancias y frecuencias que se utilizan en este bus son de 1200 a 1500m a 100 Kbits/s o 50m a 10 Mbits/s. Las líneas deben cargarse en los extremos con resistencia de terminación de línea (120 _ generalmente). El propósito de la resistencia de terminación es prevenir la reflexión de los datos en el fin de línea.
  - **RS485**, Este enlace es uno de los más utilizados en la industria. Se trata de un enlace serie, con transmisión de señales en modo diferencial, multipunto. Está basado en la interfaz RS-422 pero utiliza sólo un par trenzado para la comunicación. Permite usar una topología de bus, conectando los dispositivos en paralelo a los dos conductores, aunque desde un punto de vista lógico puede después organizarse como un anillo, estrella u otro tipo. Presenta una alta inmunidad al ruido y se pueden crear redes multipunto maestro/esclavo de forma muy sencilla.

- **VXI**, Físicamente, VXI (VMEbus eXtensión for Instrumentation) consiste en un chasis plano posterior sobre el que se conectan unos módulos en forma de tarjetas enchufables; mediante este sistema puede configurarse una solución modular de instrumentación VXI se utiliza fundamentalmente cuando se necesita un sistema de adquisición de datos fiable, de altas prestaciones, con gran número de variables a capturar y con posibilidades de ampliación. En general, para la adquisición de pocos canales (hasta 20) una tarjeta de adquisición de datos puede ser suficiente. Sin embargo, para un número de canales superior (hasta 100) puede utilizarse un instrumento externo de adquisición independiente. Cuando las necesidades aumentan, VXI puede ser la mejor solución. En general, el coste de los módulos es menor que el de un instrumento independiente y su potencia es superior a la de una tarjeta de adquisición.
- **PXI, PCI y PCI-Express,** Otra alternativa de bus muy difundida para instrumentación modular es el PXI. PXI utiliza una variante del popular bus PCI, muy utilizado en los computadores personales, para la interconexión de los módulos. A diferencia del bus PCI, esta arquitectura dispone de unas características de sincronización avanzadas, pudiendo trasmitirse por el bus señales digitales (TTL) a alta velocidad e incluso señales analógicas entre los distintos módulos. En la actualidad, el más utilizado es el PCI-Express, la evolución del bus PCI, que es una reformulación radicalmente distinta de éste aunque mantiene totalmente la compatibilidad software. PCI-Express es un bus de red de comunicaciones de alta velocidad en distancias cortas y con baja latencia para interconectar dispositivos y tarjetas entre sí dentro de un chasis.
- **USB,** El diseño del USB tenía como objetivo eliminar la necesidad de adquirir tarjetas separadas para poner en los puertos bus ISA o PCI, y mejorar las capacidades plug & play permitiendo a esos dispositivos ser conectados o desconectados al sistema sin necesidad de reiniciar. Sin embargo, en aplicaciones donde se necesita ancho de banda para grandes transferencias de datos, los buses PCI o PCI-Express salen ganando. Igualmente sucede si la aplicación requiere de robustez industrial. A favor del bus USB, cabe decir que cuando se conecta un nuevo dispositivo, el servidor lo

enumera y agrega el software necesario para que pueda funcionar (esto dependerá ciertamente del sistema operativo que esté usando el computador).

- **GPIB**, El bus GPIB tuvo sus orígenes en el Hewlett-Packard Instrument Bus (HP-IB), que es un estándar bus de datos digital de corto rango desarrollado por Hewlett-Packard en los años 1970 para conectar dispositivos de prueba y medida (por ejemplo multímetros, osciloscopios, etc.) con dispositivos que los controlen como un ordenador. 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 es aproximadamente de 1 Mbps.

➢ El bloque del PC, Software hace referencia a los distintos tipos de lenguajes de programación que soportan la comunicación con instrumentos, algunos ejemplos de estos lenguajes son: LabView, LabVEE, Matlab, Visual Basic, C++, Python, SciLab, etc.

El presente trabajo desarrolla el control de instrumentos a través de los buses GPIB y USB que se explicarán con más detalle en los siguientes apartados.

1.2 Conceptos sobre BUS GPIB.

Uno de los estándares más importantes dentro de la instrumentación electrónica es la interfaz IEEE 488 [1], conocida ampliamente como GPIB, diseñada para integrar uno o más instrumentos a una computadora o controlador.

GPIB es un bus y un protocolo estándar para el control y comunicación con instrumentos de test y medida, como multímetro digitales, osciloscopios, generadores de señales, etc. Permite configurar sistemas automáticos en el laboratorio y en la industria con gran flexibilidad y eficacia.

El bus GPIB fue inventado por Hewlett Packard a finales de los años 1960. La finalidad era crear un bus fiable, principalmente diseñado para conectar computadoras e instrumentos en una configuración de red que tuviera las características requeridas por un equipo de medida.

El control remoto de los instrumentos es un aspecto relevante del bus, pero hay otros más importantes como el reconocimiento de recepción de datos ("data hardware handshake"), que otorga a las operaciones de fiabilidad; o la capacidad de respuesta en tiempo real.

El principal objetivo del bus GPIB consiste en gestionar la transferencia de información entre dos o más dispositivos. Antes de enviar los datos hacia los dispositivos (instrumentos conectados al bus) éstos deben configurarse de acuerdo con este protocolo de transmisión de información. Entre los parámetros relativos al protocolo se encuentra la asignación de direcciones a los instrumentos interconectados. La numeración del dispositivo, o asignación de su dirección, se efectúa desde el panel frontal o alterando la conexión de los puentes de su tarjeta interfaz, que suele ser accesible desde la parte posterior del instrumento.

El elemento controlador del equipo GPIB es único (generalmente la tarjeta controladora instalada en un PC, en cuyo caso se le asigna la dirección 0), supervisa todas las operaciones que se realizan en el bus, y determina el dispositivo que envía la información y el momento en que se realiza su envío. El controlador puede designar un sustituto si en un determinado momento no puede atender los requisitos de control. El nuevo controlador recibe el nombre de controlador activo.

El controlador asegura que no puede haber dos o más instrumentos enviando información al bus simultáneamente. Además, establece los dispositivos que permanecen en estado de recepción o escucha, ya que no todos los instrumentos están siempre interesados en captar la información del bus. Esta función la realiza "despertando" a los dispositivos en estado de "latencia" mediante una solicitud de reafirmación, y mediante órdenes que especifican los nuevos receptores y el nuevo emisor.

Cuando el proceso de transmisión-recepción ha finalizado, el controlador del equipo se asegura de que todos los receptores han recibido la información enviada al bus por el emisor mediante el "data hardware handshake" o control de transferencia de datos. Este protocolo permite asegurar la recepción de la información por parte de los dispositivos más lentos. Como consecuencia, el dispositivo más lento limita la velocidad de operación del equipo GPIB.

En resumen, se consideraran los siguientes elementos o conceptos más relevantes y específicos, involucrados en un equipo red de instrumentación mediante el protocolo GPIB:

- Controlador del equipo controlador activo.
- Dispositivos conectados al bus.
- Dispositivo fuente.
- Dispositivos destino.
- Comandos y funciones.

1.3 Historia y evolución del GPIB.

En septiembre de 1965, la compañía Hewlett-Packard (HP), en la actualidad la división de Test y Medida con el nombre "Agilent Technologies", ideó la posibilidad de interconectar sus instrumentos mediante un sistema de comunicaciones en el cual un instrumento pudiera *hablar* a otro y viceversa. Como resultado surgió el bus de interfaz Hewlett-Packard (HP-IB, ***H****ewlett* ***P****ackard* ***I****nterface* ***B****us*).

Debido a su elevada velocidad de transferencia (1 Mbyte/s nominal) y a su fiabilidad, este interfaz adquirió popularidad a pasos agigantados. Así, en abril de 1975, el HP-IB logró la aceptación del IEEE (*Institute of Electrical and Electronics Engineers*) mediante la publicación del estándar ANSI/IEEE 488-1975 "Interfaz Digital para Instrumentos Programables IEEE", el cual contenía especificaciones eléctricas, mecánicas y funcionales de un sistema de interfaz. En noviembre de 1978, se revisó el estándar, principalmente para realizar clarificaciones editoriales y agregar anexos y como resultado se obtuvo el documento identificado como IEEE 488-1978, este último se conoce como el estándar del bus de instrumentación de propósito general (GPIB, *General Purpose Instrumentation Bus*). La norma GPIB fue adoptada en Europa bajo la denominación IEC 625.1 por la Comisión Electrotécnica Internacional (IEC, *International Electrotechnical Commission*) y en enero de 1976, el Instituto Nacional de Estándares Americanos (ANSI, *American National Standard Institute*) publicó un estándar idéntico denominado.

Debido a que el documento original IEEE 488 no incluía normas para una sintaxis de preferencia y convenciones de formato, se continuó trabajando en la especificación para

mejorar el sistema de compatibilidad y configuración entre sistemas de pruebas. El resultado de este trabajo se aprueba por el IEEE en junio de 1987 generando así un nuevo estándar para dispositivos programables, el IEEE 488.2-1987, que incluye códigos, formatos, protocolos y comandos comunes, para utilizarse con el IEEE 488 (el cual fue renombrado como IEEE 488.1). El IEEE 488.2 no reemplaza el IEEE 488.1, muchos dispositivos todavía cumplen sólo con el IEEE 488.1 al definir un conjunto mínimo de habilidades para la interfaz de un dispositivo, un conjunto común de códigos y formatos para datos, un protocolo de mensajes para dispositivos, un conjunto genérico de comandos comúnmente necesitados por dispositivos y un nuevo modelo para generación de reportes de estado.

En resumidas cuentas el estándar IEEE 488 se forma del estándar IEEE 488.1, encargado de las especificaciones mecánicas, eléctricas y funcionales; y del estándar IEEE 488.2, el cual define la sintaxis, la estructura de los datos y la manera en que éstos se transmiten y representan por los dispositivos mediante código ASCII (*American Standard Code for Information Interchange*). La figura 1.2 muestra la evolución del estándar 488.

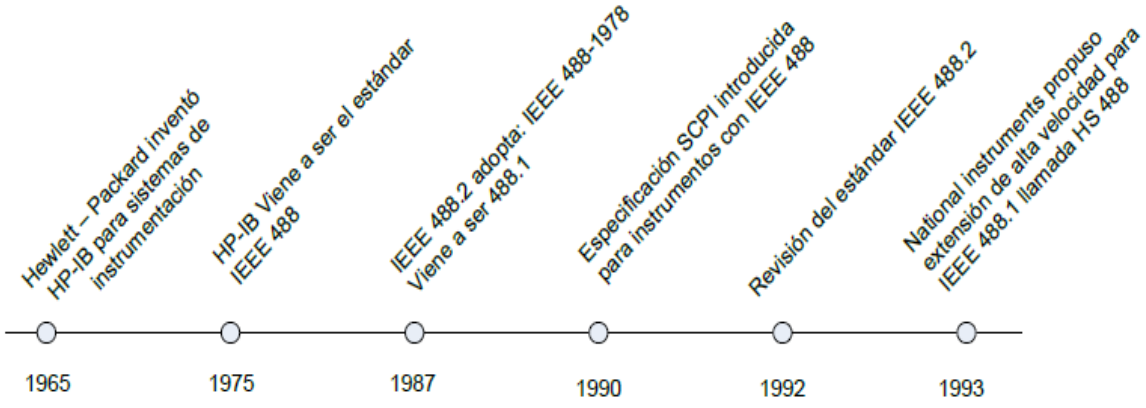

Figura 1-2 Evolución del estándar 488.

Hasta entonces, cada fabricante definía su propio conjunto de órdenes para cada instrumento, por ello en abril de 1990, un grupo de fabricantes de dispositivos de instrumentación electrónica anunciaron la especificación de Los Comandos Estándares para Instrumentación programable; SCPI por sus siglas en inglés (*Standard Commands for Programmable éInstruments*). El SCPI define un conjunto de órdenes comunes para instrumentos

electrónicos programables. Por lo tanto, el SCPI garantiza compatibilidad y configuración de sistema completas entre estos instrumentos. Ya no es necesario aprender un conjunto de comandos diferentes para cada instrumento en un sistema que cumple con el SCPI, y es fácil reemplazar un instrumento de un proveedor con el de otro proveedor. La Figura 1.3 muestra la relación entre los estándares IEEE 488 y SCPI.

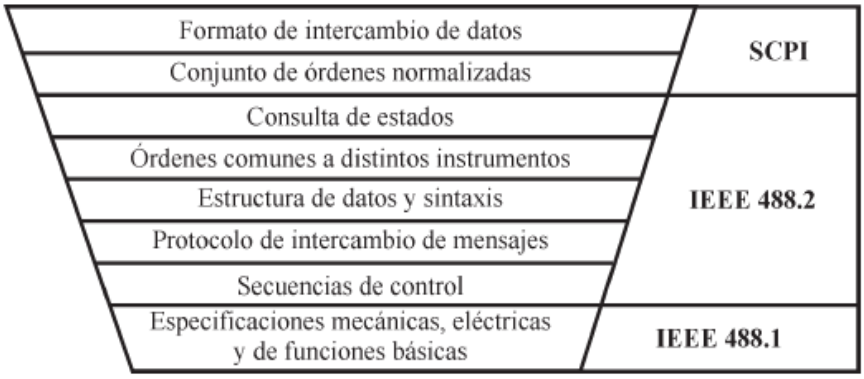

Figura 1-3. Relación entre los estándares IEEE 488.1, IEEE 488.2 y SCPI.

En 2003, el estándar IEEE 488.1 adoptó el protocolo GPIB de alta velocidad, conocido como HS488 y propuesto por la firma National Instruments. HS488 alcanza velocidades de transferencia de datos de hasta 8 MBps y es compatible con el estándar IEEE 488.1.

1.4 IEEE 488.1

El estándar IEEE 488.1 permite la interconexión de hasta 15 dispositivos activos en el bus, incluyendo el controlador, define la longitud máxima del bus, la cual no debe ser mayor a 20m, y recomienda la conexión de dispositivos al bus mediante cables GPIB de 2m. de longitud.

La máxima velocidad de transferencia de datos es de 1 MBps (Megabyte por segundo) y se reduce a 250 o 500 KBps (Kilobyte por segundo) a distancias de bus mayores a los 20 m.

Todos los instrumentos en el bus se conectan en paralelo mediante un conector hermafrodita, lo cual permite dos tipos de configuración, lineal o en estrella, como muestra la Figura 1.4.

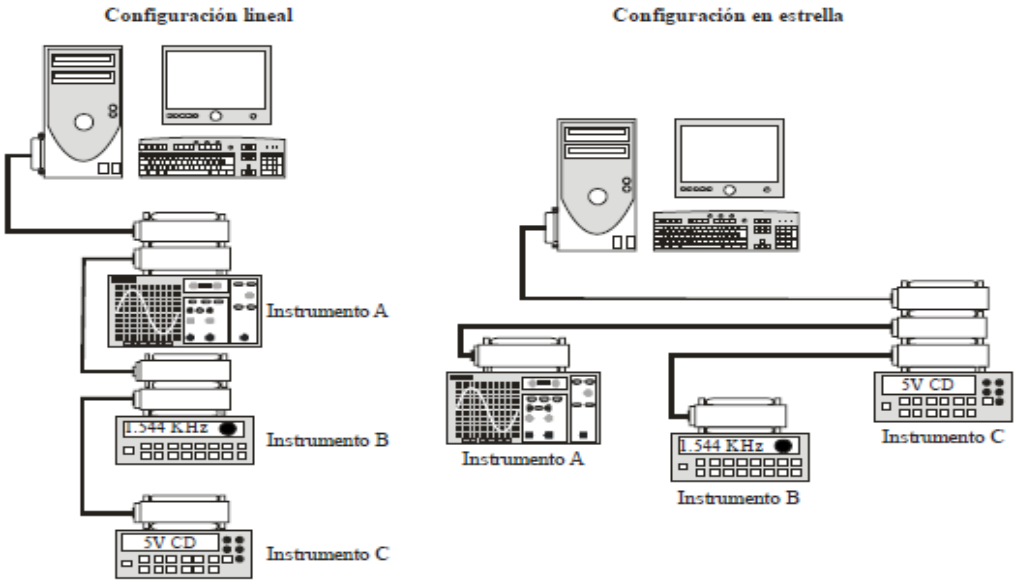

Figura 1-4. Configuración lineal y en estrella.

1.4.1 Especificaciones eléctricas.

El estándar IEEE 488.1 define un sistema basado en la lógica negativa, el cual utiliza un nivel lógico "0" para especificar un estado verdadero (*true*) y un nivel lógico "1" para especificar un estado falso (*not true*) (Tabla 1-1). Con ello se reduce la susceptibilidad de ruido en el estado verdadero y proporciona un estado falso sobre las líneas que no están conectas o en uso.

Nivel lógico Nivel eléctrico
0 > + 2 Volts
1 < + 0.8 Volts

Tabla 1-1. Niveles de voltaje IEEE 488.1.

1.4.2 Especificaciones mecánicas.

El estándar IEEE 488.1 especifica el tipo de conector utilizado para interconectar los dispositivos GPIB, el cual es un conector tipo americano o hermafrodita de 24 terminales (Figura 1.5).

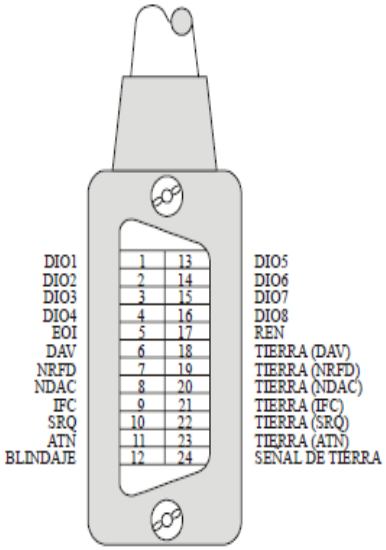

Figura 1-5. Configuración de terminales GPIB.

1.4.3 Especificaciones funcionales.

Los aspectos funcionales definen los tipos de instrumentos conectados al bus, las órdenes, datos, direcciones y funciones básicas; también considera los protocolos o mecanismos de transferencia de información y el mecanismo de reconocimiento.

1.4.3.1 Tipos de instrumentos GPIB.

El estándar IEEE 488.1 define tres tipos de instrumentos o dispositivos GPIB:

- *Emisor (Talker)*: dispositivo capaz de enviar información referente a su estado a través del bus.
- *Receptor (Listener)*: dispositivo que puede recibir órdenes y datos a través del bus cuando es direccionado.
- *Controlador (Controller)*: dispositivo encargado de controlar el sistema GPIB mediante la utilización de órdenes para el control del bus. El controlador GPIB puede ser tanto emisor como receptor, y generalmente es una PC.

Cualquier dispositivo puede ser un emisor o un receptor, pero no simultáneamente.

1.4.3.2 Funciones de la interfaz.

El estándar IEEE 488.1 específica diez funciones básicas que deben implementarse en todo dispositivo GPIB. Cada función cuenta con un nemónico de identificación, el cual consta de 1 a 3 letras para describir la interfaz (Tabla 1-2).

Nemónico	Capacidad	Instrumento
SH	Control de emisión	Emisores y receptores
AH	Control de recepción	
T/TE	Emisor/Emisor ampliado	
L/LE	Receptor/Receptor ampliado	
SR	Petición de servicio	
RL	Modo remoto o local	Controlador, emisores y receptores
PP	Sondeo paralelo	
DC	Inicialización	
DT	Disparo	
C	Controlador	

Tabla 1-2. Funciones básica del estándar IEEE 488.1.

1.4.3.3 Estructura del bus.

El GPIB es un canal de comunicación de dos vías y el flujo de los datos es en ambas direcciones (*full duplex*). El canal de comunicación que incluye cada dispositivo GPIB, carga los datos u órdenes en el bus utilizando 16 líneas (Figura 1.4):

- La información de datos y direcciones se maneja mediante palabras de 8 bits (byte u octeto), mediante las líneas DI0 a DI7.
- Las líneas DAV, NRFD y NDAC se utilizan para realizar la función de transferencia (*handshake*).
- Las cinco restantes proporcionan el manejo y el control del sistema GPIB.

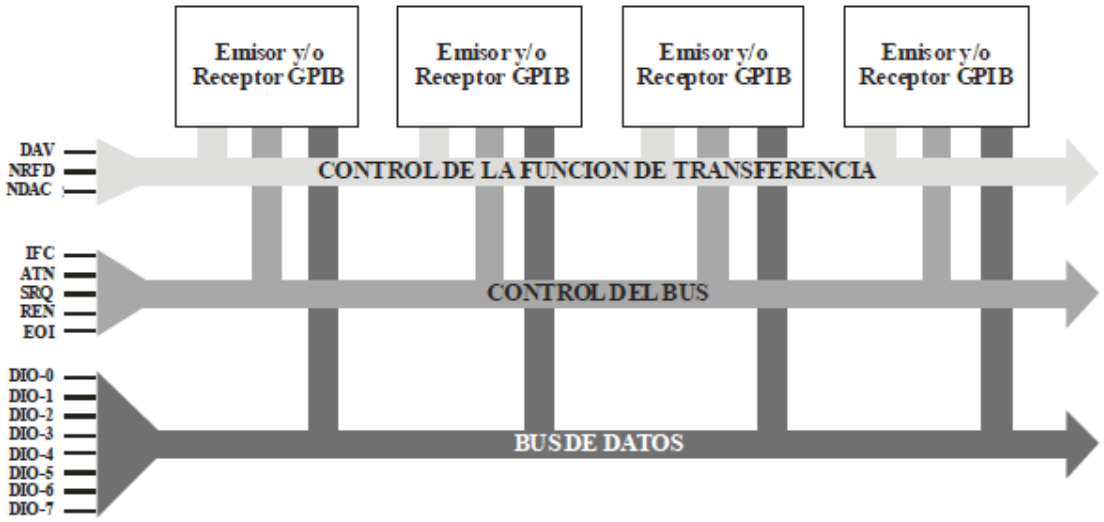

Figura 1-6. Estructura del bus.

1.4.3.4 Direcciones GPIB.

A cada dispositivo se asigna una dirección única, para ello el protocolo GPIB define un total de 31 direcciones llamadas direcciones primarias.

Se asigna una dirección primaria a cada dispositivo mediante la configuración de interruptores (*switches* o *jumpers*) localizados en la parte posterior (*rear panel*) de los dispositivos. Actualmente, los dispositivos configuran su dirección mediante herramientas software o desde su panel frontal (*front panel*).

Las direcciones válidas pueden tomar un valor de 0 a 30, se reserva la dirección número 31 para propósitos de control y la dirección 21 para controlar la dirección del emisor/receptor.

Las direcciones primarias especifican dos códigos de direcciones sobre las líneas de datos, éstas diferencian entre las direcciones de emisor y las direcciones de receptor definidas en los bits 6 y 7 del octeto de datos.

1.4.3.5 Sondeo.

El controlador GPIB cuenta con dos formas de conocer el estado actual de los dispositivos activos en el bus:

- *Sondeo serie (Serial Polling)*: se direcciona individualmente a cada dispositivo activo en el bus y éste regresa el octeto de estado, el cual indica su condición actual.
  La ventaja de este tipo de sondeo es que el controlador obtiene la identidad del dispositivo sondeado y su respectivo octeto de estado, y su principal desventaja es el tiempo que se requiere para realizar el sondeo de varios dispositivos.

- *Sondeo paralelo (Parallel Poll)*: esta forma de sondeo es más rápida que el sondeo serie, pero únicamente pueden ser sondeados hasta ocho dispositivos al mismo tiempo debido a que el controlador GPIB asigna a cada dispositivo un bit del octeto de estado.

1.5 IEEE 488.2

El estándar IEEE 488.2 fue desarrollado para eliminar los problemas y ambigüedades que presentaba el estándar original, por ejemplo:

- No definía un formato común para la representación de los datos.
- No definía un protocolo estándar de mensajes.
- No contaba con un conjunto de órdenes comunes a todos los dispositivos.
- El octeto de estado de cada dispositivo GPIB era diferente

Para solucionar tales problemas, el estándar IEEE 488.2 proporciona las siguientes soluciones:

- Define formatos de datos que van desde números decimales a cadenas de caracteres.
- Describe un protocolo de intercambio de mensajes, que indican al dispositivo cómo reaccionar cuando ocurren condiciones no establecidas.
- Define un conjunto de órdenes comunes para una comunicación uniforme.
- Define un modelo de informe de estado para los instrumentos GPIB.

1.5.1 Sintaxis y formato de datos.

El estándar IEEE 488.2 especifica el formato de los datos para cualquier tipo de mensaje que pueda ser enviado, incluyendo números y cadenas de caracteres, en el GPIB. Los tipos de números incluidos en el estándar son binario, octal y hexadecimal.

1.5.2 Órdenes comunes.

Todos los dispositivos GPIB realizan un conjunto de funciones comunes para comunicarse a través del bus e informar de su estado actual. Antes de definir el estándar IEEE 488.2, cada dispositivo utilizaba su propio conjunto de órdenes para realizar sus funciones.

Las órdenes comunes no son específicas de los instrumentos y algunas son opcionales. El siguiente conjunto de órdenes mostrados en la Tabla 1-3 asegura que todos los dispositivos conectados al bus se comuniquen adecuadamente:

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 registró 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.*
***OPC**	Operation Complete Command	- *Fija el bit de "Operación completa" del registro estándar.*
***OPC?**	Operation Complete Query	- *Responde con "1" si se han ejecutado ordenes previas.*
***OPT?**	Option Identification Query	- *Requiere la opción instalada en el equipo.*
***RCL**	Recall Command	- *Restaura el estado del equipo del registro save/recall.*
***RST**	Reset Command	- *Sitúa al equipo en el estado básico de referencia.*
***SAV**	Save Command	- *Almacena el estado actual en un registro save/recall.*
***SRE**	Service Request Enable Command	- *Habilita los bits del registro de habilitación de Byte de estado.*
***SRE?**	Service Request Enable Query	- *Requiere el contenido del registro SER de habilitación del byte de estado.*
***STB?**	Read Status Byte Query	

***TRG** ***TST?** ***WAI**	Trigger Command Selft-Test Query Wait-to-Continue Command	- *Requiere el estado del registro resumido del Byte de estado.* - *Arranca o dispara la operación del equipo de forma remota.* - *Requiere el resultado del autotest del equipo.* - *Espera a que se realicen todas las operaciones pendientes.*

Tabla 1-3. Ordenes Comunes a todos los Instrumentos.

1.6 USBTMC

USBTMC es acrónimo de **USB Test & Measurement Class.** USBTMC es un protocolo construido sobre USB y permite comunicación tipo GPIB con dispositivos USB. Desde el punto de vista del usuario, el dispositivo USB se comporta similar a como lo hace un dispositivo GPIB. Por ejemplo, usted puede emplear VISA Write para enviar el comando *IDN? y utilizar VISA Read para recibir una respuesta. El protocolo USBTMC soporta peticiones de servicio, disparos y otras operaciones específicas de GPIB.

USBTMC permite a los fabricantes de instrumentos actualizar la capa física de GPIB a USB manteniendo la compatibilidad de software con los programas ya existentes, tales como controladores de instrumentos y cualquier aplicación que utilice VISA. Esto es similar a lo que el protocolo VXI-11 provee para TCP/IP.

Es importante notar que no todos los dispositivos USB son compatibles con USBTMC. El fabricante del dispositivo debe agregar el soporte en el firmware del dispositivo para soportar USBTMC. Instrumentos tradicionales (tales como DMMs y Osciloscopios) con puertos USB probablemente soporten USBTMC. Refiérase a la documentación de los instrumentos para determinar si son compatibles con USBTMC.

Esta especificación establece los atributos compartidos, servicios comunes y formatos de datos para los dispositivos con una interfaz de prueba y medida compatible con USBTMC.

Los requisitos del protocolo y de interoperabilidad se establecen de manera que el software de aplicación pueda gestionar múltiples implementaciones basadas en esta especificación USBTMC.

1.7 VISA

Una de las alternativas de software de I/O para el control de instrumentos de uso más popular en entornos de instrumentación programable como VEE y LabVIEW, así como en lenguajes basados en C/C++ o BASIC es la herramienta VISA (*Virtual Instrument Software Architecture*). Esta define interfaces de comunicaciones con los instrumentos VXI, GPIB, RS232, USBTMC. Desarrollada por la empresa Alliance System VXI*plug&play*, su condición de estándar ha hecho de este software de I/O, un elemento de uso confiable, en el desarrollo de aplicaciones de control de instrumentos mediante protocolos de intercambio de mensajes como SCPI. Firmas inmersas profundamente en el ámbito de la instrumentación electrónica programable, como Agilent Technologies y National Instruments proporcionan soluciones de software de I/O basadas en la arquitectura VISA.

Agilent Technologies agrupa la arquitectura VISA (Agilent VTL), junto con un particular desarrollo llamado SICL en un software propietario de I/O (*Agilent IO Libraries Control*). Dicho software facilita la configuración de interfaces de I/O, tales como GPIB, VXI, LAN, etc., al igual que provee utilidades (librerías) que pueden ser usadas por diversos ambientes de desarrollo, para el control de instrumentos. Este software tiene la facultad de poder coexistir con otra herramienta VISA de distinto proveedor en un mismo controlador (PC), sin embargo esta característica puede traer consigo inconvenientes para algunos ambientes de desarrollo.

El software de I/O de National Instruments llamado NI-VISA, provee de igual forma, no solo herramientas para la verificación de la conectividad con el instrumento mediante VISA, sino también suministra código utilitario para desarrollos sobre diferentes ambientes.

Una nota relevante en el uso de software de I/O propietario, en la existencia de una dependencia con el hardware bajo operación, así el manejo de productos de interfaz de una firma particular requerirá el uso exclusivo de software de I/O de la misma firma. De esta manera una tarjeta de interfaz PCI/GPIB de Agilent Technologies requerirá para su manejo

exclusivamente el uso de software de I/O de Agilent. Afortunadamente esta dependencia se encuentra establecida en este nivel de software, dejando al software de aplicación libre de esta atenuante.

1.8 Ambiente de desarrollo de aplicaciones

Un ambiente de desarrollo de aplicaciones, dentro del campo de la instrumentación electrónica programable, se entiende como un lenguaje o entorno mediante el cual es posible el desarrollo de programas para el control y coordinación de todos los instrumentos de un sistema.

La elección del entorno bajo el cual es desarrollada una aplicación, independientemente de su índole, se encuentra definida tanto por los objetivos trazados, como por los medios con los que se cuentan. Aunque no existen una serie de características específicas para el software usado en el control de instrumentos, algunos criterios útiles en la elección de estos, son:

- *Conectividad con instrumentos*: los medios que un software presenta para el control de instrumentos y la forma en como estos son utilizados, determinara en gran medida el tiempo de desarrollo de la aplicación. Así, existirá una considerable diferencia entre el uso de una orden nativa (SCPI) donde un conocimiento del formato de intercambio de mensajes es necesario y la utilización de un controlador cuyo compilador asociado realiza la mayor parte de las operaciones, presentando al usuario objetos simples de manipulación.
- *Capacidades de análisis y presentación*: una vez que los datos han sido adquiridos desde un instrumento, se hace evidente el uso de algoritmos y funciones diseñadas especialmente para el análisis y procesamientos de señales. La inclusión de robustas capacidades de análisis y presentación facilitaran no solo generar, modificar, procesar y analizar la información obtenida, sino también el desarrollo de una interfaz gráfica de usuario (GUI, *Graphical User Interfaz*) que cumpla con los propósito de la aplicación.

Generalmente, el desarrollo de aplicaciones encaminadas al control de instrumentos, en base a algún protocolo de comunicaciones como GPIB, Serial, VXI, USBTMC dentro de un entorno de instrumentación programable, se lleva a cabo mediante herramientas de software propietarias, como lo son VEE de la firma Agilent Technologies o LabVIEW y LabWindows/CVI. Sin embargo, esta no es una condicionante, que evite que lenguajes de propósito específico como C/C++, BASIC, Java, Matlab, Scilab, Python gocen de una gran popularidad dentro del ámbito de la instrumentación programable.

CAPÍTULO II: UNA BREVE INTRODUCCIÓN A LABVIEW.

2.1 Que es LabView?

LabVIEW, (Laboratory Virtual Instrument Engineering Workbench, por sus siglas en inglés), es un software de desarrollo basado en programación gráfica por la empresa National Instruments, destinado a desarrollar aplicaciones para instrumentación, ingeniería, ciencias, aplicaciones médicas, etc., que tiene una serie de opciones para conectarse a instrumentos electrónicos, con tarjetas de adquisición de datos, con sistemas de acondicionamiento y control y otros tipos de software como lo serían Microsoft Excel y Matlab. La versión utilizada para este proyecto es el LabView 2013.

Todos los programas desarrollados en LabVIEW se les llaman instrumentos virtuales "VI´s", porque aun siendo virtuales tienen las mismas funciones y programación de instrumentos reales.

Las principales características de los VI's son:

- Interface interactiva: Los VI´s contienen una interface muy interactiva de usuario, el usuario trabaja en el llamado panel frontal, ya que simula el panel de instrumentos, donde realmente va colocando los instrumentos virtuales necesarios para su programa. Esos instrumentos necesitan datos de entrada que pueden ser introducidos usando el teclado o el ratón para tener una visualización de los resultados en la pantalla de la computadora.

- Diagrama de bloques: Los VI´s, realizan su función de acuerdo a las instrucciones desde el diagrama de bloques (programación G), los instrumentos al seguir en un orden estas instrucciones llegan a una solución de problema mostrado en el panel frontal. Los dos paneles en la pantalla de la PC se visualizan como en la figura 2.1.

- La estructura de los VI´s son módulos independientes a la vez interconectados entre sí, para llevar a cabo la función indicada que permite realizar programas por niveles, en un orden estrictamente jerárquico, VI´s que a la vez pueden convenirse en sub-VI´s.

- Rápido: la programación en LabVIEW es mucho más rápida en comparación con las formas de programación tradicionales, lo que en programación tradicional nos llevaría, varias horas hasta días, en LabVIEW podríamos tener el programa completamente desarrollado en unas horas de trabajo serio, claro dependiendo del grado de complejidad del programa a desarrollar y de la experticia que tenga con el lenguaje, pero sin lugar a duda, hay un mejor aprovechamiento del tiempo.

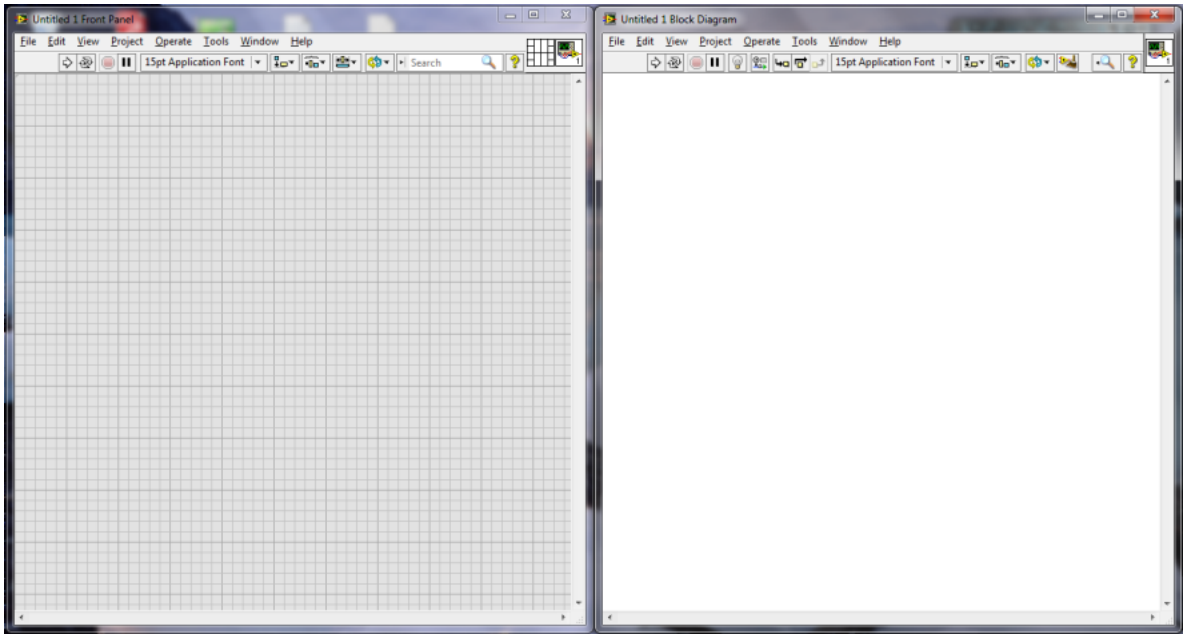

Figura 2-1. Panel Frontal

Estas características principales nos dan una idea generalizada de cómo funciona LabVIEW, así como las tareas del VI´s son interconectadas para que el programa funcione correctamente, también son divisibles en cualquier momento de edición del programa, solamente por parte del programador, es decir, en el diagrama de bloques. Si se va a desarrollar un programa relativamente complicado, tal vez convenga más dejarlo en términos de varios sub VI´s, para facilitar la manipulación del programa y en dado caso de tener que modificar algo, solo sea en el sub programa y no tocar el resto de código.

Al introducir las variables a utilizar éstas aparecen tanto en el panel frontal como en el diagrama de bloques, así podemos observar como es el flujo de datos y la presentación de

resultados, así mismo las gráficas y accesos directos, cumplen las mismas condiciones, no sucede lo mismo con las funciones específicas del diagrama de bloques.

Una gran ventaja de LabVIEW es que puede ser usado con muy poca experiencia en programación de cualquier tipo, pues utiliza métodos que llevan de la mano en el buen funcionamiento del programa y con los que están familiarizados la mayoría de los técnicos, ingenieros, científicos, etc. Por lo anterior descrito podemos ya deducir que un VI´s está compuesto por un panel frontal, que es para el usuario y un diagrama de bloques, donde trabaja el programador. Éstas a su vez cuentan con barra de herramientas que muestran los objetos e instrumentos, variables, controles e indicadores necesarios para implementar y desarrollar tareas.

Es la interface gráfica para el usuario que simula el panel de un instrumento real donde se colocan los controles e indicadores, permite la entrada y salida de datos, puede contener pulsadores, perillas, botones, gráficos y en general controles e indicadores. Se puede ver el panel en la figura 2.2.

Los controles (entradas) pueden ser booleanos, numéricos, strings, un arreglo matricial de éstos o una combinación de los anteriores dependiendo de la función necesaria o el estilo q le queramos dar; y los indicadores (salidas) pueden ser como para el caso de controles pero pudiéndolos visualizar como tablas, gráficos en 2D o 3D, browser, entre otros dependiendo también del tipo de información que vamos a demostrar.

2.2 Diagrama de bloques

El diagrama de bloques contiene la estructura de tareas gráfico del VI, posee funciones y estructuras en sus menús conectan las entradas con las salidas creadas en el panel frontal. En un diagrama se distinguen: los controles e indicadores del panel. Funciones y SubVI’s, que realizan tareas específicas, estos no aparecen en el panel frontal. Estructuras y líneas, que determinan el flujo de los datos en el programa (figura 2.3).

El flujo de datos va de izquierda a derecha en el panel de programación y está determinado por las operaciones o funciones que procesan los datos este flujo se puede observar con la función highlight, así es muy fácil ver paso a paso la ejecución del programa y nos ayuda a percatarnos de errores surgidos durante la ejecución que no podríamos ser capaces de ver de manera normal.

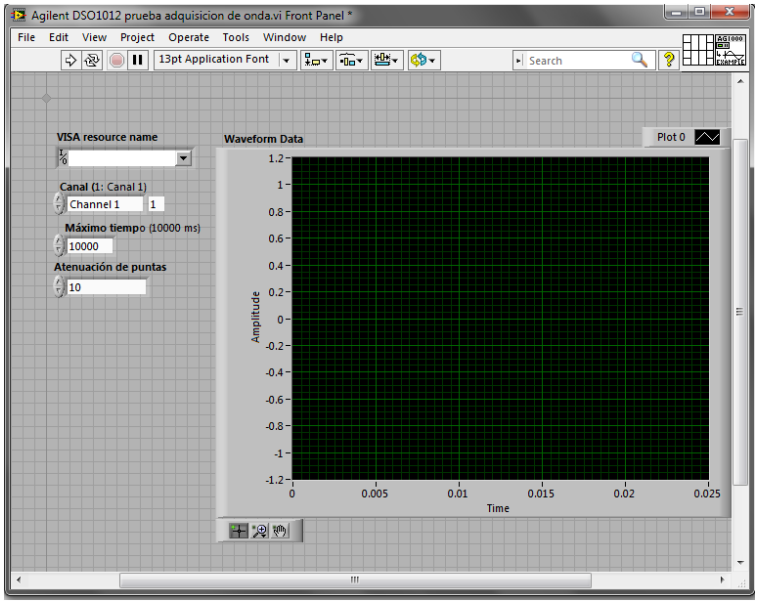

Figura 2-2. Panel Frontal

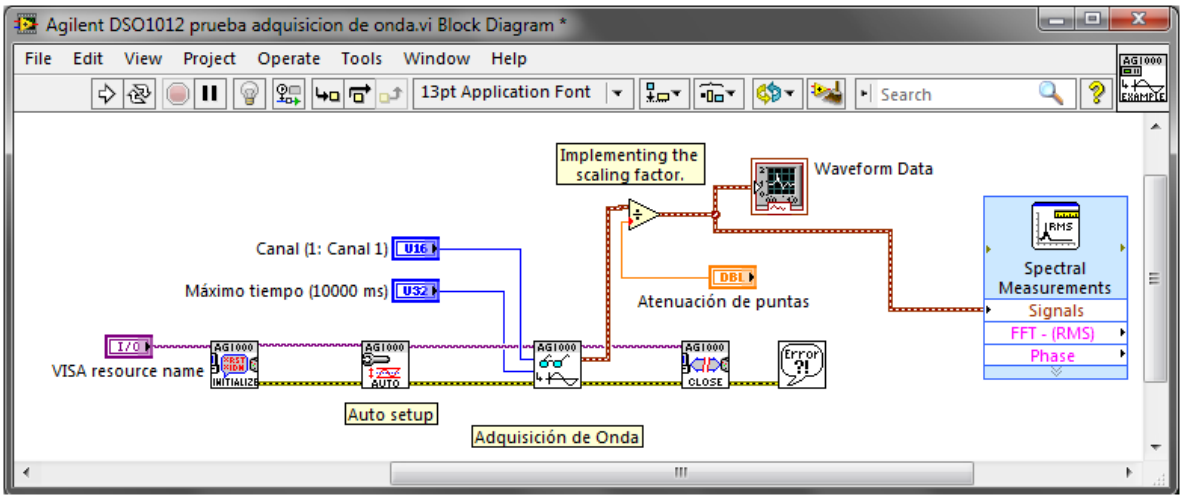

Figura 2-3. Diagrama de Bloques de un SubVI.

2.3 Paleta de Herramientas.

La paleta de herramientas, es una barra que contiene diferentes herramientas que nos dan la opción manipular en panel frontal, descritas a continuación y mostrada en la figura 2.4. La podemos tener en el panel siguiendo el menú View>>Tools palette, y activar y desactivar su ejecución automática.

Figura 2-4. Paleta de Herramientas.

Operación: Se simboliza con una manita, con la cual podemos manipular los valores de los controles en el panel frontal, se puede utilizar tanto en la edición como en la ejecución del VI Cuando lo que se va a modificar el texto o números, cambia el icono a una flecha.

Posición: Se simboliza por una flecha, su función es seleccionar, mover y modificar el tamaño de los objetos. Cambia el icono del puntero cuando pasa por encima de objetos que pueden modificar su tamaño.

Líneas: Los controles e indicadores se unen por medio de líneas que funcionan como cables virtuales que mandan la señal de los datos utilizados. Es así como se crea una secuencia lógica del diagrama de bloques y se crea el flujo de datos.

Menú desplegable: Permite obtener el menú de opciones de un objeto. Solo se selecciona el objeto y se da clic en el icono que se simboliza por un recuadro gris. Esta misma función se puede realizar haciendo un clic derecho del ratón sobre el objeto.

Desplazamiento: Se simboliza por una mano abierta y mueve todos los objetos dentro de la ventana activa, es decir, como si se desplazara a través de toda la pantalla, no solo mueve un objeto, mueve todos a la vez.

 **Punto de quiebre**: Se simboliza con un botón rojo, al seleccionarlo detiene automáticamente e instantáneamente la ejecución del programa en el diagrama, en la parte que se seleccione.

 **Punto de prueba**: Se simboliza por un botón amarillo y se coloca alguna línea de conexión para comprobar de forma temporal el valor que fluye a través de éste y que sea correcto.

 **Capturar color**: Se simboliza con un gotero, al posicionarlo en el objeto, toma el color de éste.

 **Colorear**: Se simboliza con un pincel. Al seleccionarlo se abre un cuadro donde escogemos el color y lo llevamos al objeto donde deseamos aplicarlo.

 **Texto**: Crea una caja para poder insertar texto, son las conocidas etiquetas o labels.

2.4 Paleta de Controles

Se utiliza únicamente en el panel frontal y contiene los objetos necesarios para crear una interface de entrada y salida de datos (controles e indicadores). Esta paleta se obtiene de la barra de menús con la opción View>Controls Palette, o haciendo clic derecho sobre el panel frontal. La paleta de controles se muestra en la figura 2.5

2.5 Paleta de Funciones.

Se usa únicamente en el diagrama de bloques y contiene todos los objetos para crear y editar el código fuente. Esta paleta se obtiene de la barra de menús con la opción View>> Functions Palette, o haciendo clic derecho en el diagrama. La paleta de funciones se muestra en la figura 2.6.

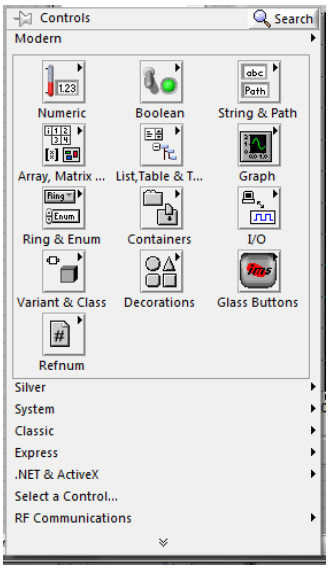

Figura 2-5. Paleta de Controles.

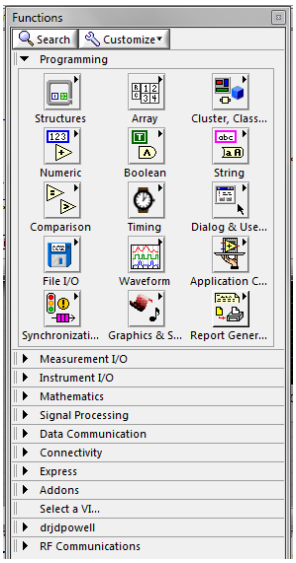

Figura 2-6. Paleta de Funciones.

Cada submenú de la paleta contiene funciones para distintas tareas, en estas paletas de funciones encontramos una gran gama de opciones para poder desarrollar nuestro programa. Habiendo casi cualquier función que imaginemos, según las necesidades de nuestro proyecto.

2.6 Tipo de datos.

Dependiendo de los objetivos de nuestra aplicación, utilizamos diversos tipos de datos en el diagrama de bloques existiendo un color característico para cada uno. (Booleanos: verde claro, Numéricos: azules-naranjos y los Alfanuméricos: rosados). Esto para poderlos identificar rápido y fácilmente en el diagrama de bloques y poder unir las terminales correctamente.

Booleano: El bit más significativo contiene al valor booleano. Si el bit toma un valor de 1, el valor del control o indicador será: true (verdadero) y si el bit toma un valor 0, toma el valor: false (falso). Para poder acceder al menú que nos muestra los diferentes tipos de datos damos un clic con el botón derecho del mouse sobre el área de trabajo del panel frontal nos aparecerá la paleta de controles (figura 2.7) y dar un clic con el botón derecho del mouse sobre el diagrama de bloques (figura 2.8) allí podemos hallar los diferentes controles e indicadores booleanos que posee LabVIEW.

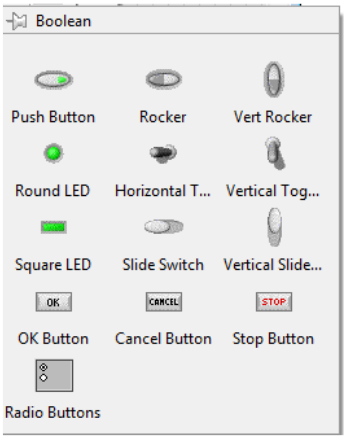

Figura 2-7. Datos para Panel Frontal

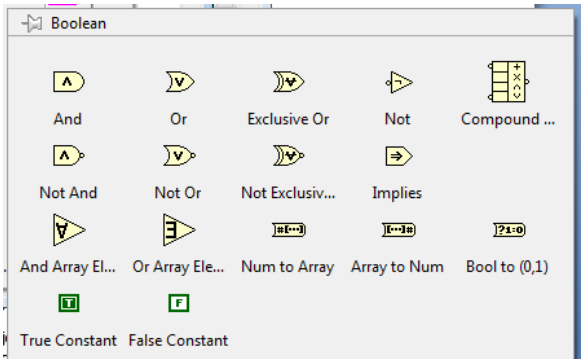

Figura 2-8. Datos para Diagrama de Bloques.

Numérico: Los datos numéricos que nos ofrece LabVIEW se clasifican en 12 representaciones para los controles e indicadores señalados por su respectivo color:

a) Números de tipo entero
b) Números de tipo sin signo.
c) Números de punto flotante
d) Números de tipo complejos simples, dobles y extendidos.

Colocando un control o indicador numérico en el diagrama de bloques o panel frontal (figuras 2.9 y 2.10), hacemos clic sobre él con el botón derecho del mouse y nos dirigimos a "Representación" donde podemos configurar el tipo de dato.

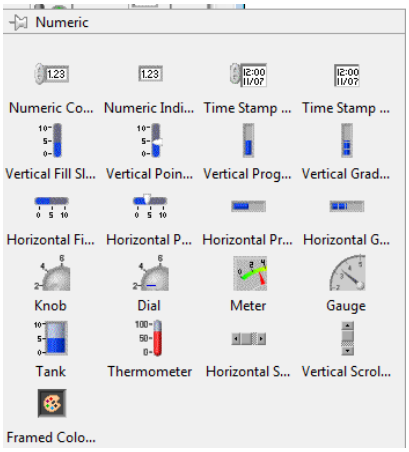

Figura 2-9. Números para Panel Frontal

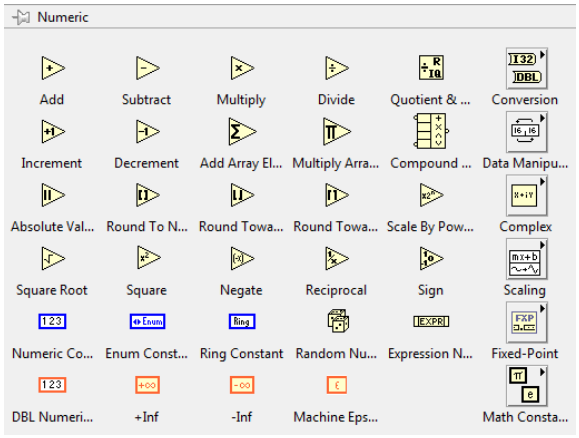

Figura 2-10. Números para Diagrama de Bloques.

Alfanuméricos: Es posible ingresar texto o números de manera alfanumérica en nuestro programa esto como la función de strings como si fuera un array unidimensional. La principal

función que desempeña esta opción es mandar a pantalla mensajes de texto, o caracteres de tipo numérico, para especificar nombres o jerarquizar los elementos.

Como mencionamos anteriormente, los datos alfanuméricos se visualizan en el panel frontal como etiquetas, tablas y entradas de texto (figura 2.11) mientras que en el diagrama de bloques son de color rosado (figura 2.12).

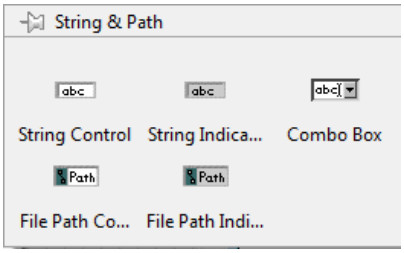

Figura 2-11. Alfanumericos Panel Frontal

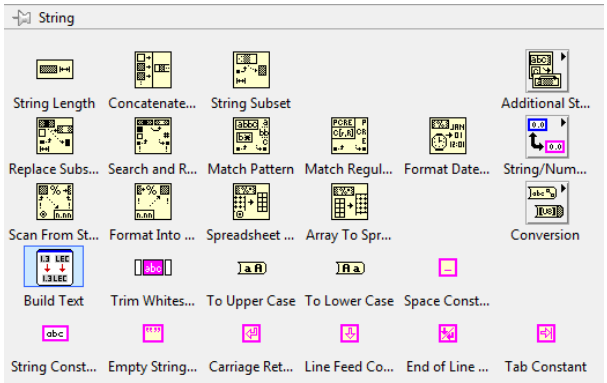

Figura 2-12. Alfanumericos Diagrama de Bloques.

2.7 Tecnicas de Unión

La estructura básica y clave para desarrollar un programa en LabVIEW, es que es un sistema de objetos interconectados entre ellos e independientes a la vez. Estos se conectan a través de líneas que salen de sus terminales dependiendo del tipo de función que realicen y que mandan las señales que generan o a las que están destinadas entre ellos.

Para poder unir dos objetos en el diagrama de bloques debemos identificar primero el tipo de terminales que tiene, es decir, si mandaran una señal o instrucción o la recibirán, se selecciona la herramienta de cableado de la tolos palette o automáticamente aparecerá el icono para esta función al posicionar el cursos en la terminal del icono, se da clic ahí, y se arrastra sin soltar hasta la terminar que recibirá esa señal.

2.8 Estructuras

Una estructura en general es un nodo que controla el flujo de los datos de un programa y que se comporta bajo ciertas condiciones. LabVIEW cuenta, en orden, con las siguientes estructuras, siguiendo Funtions>>Programming>>Structures, figura 2.13

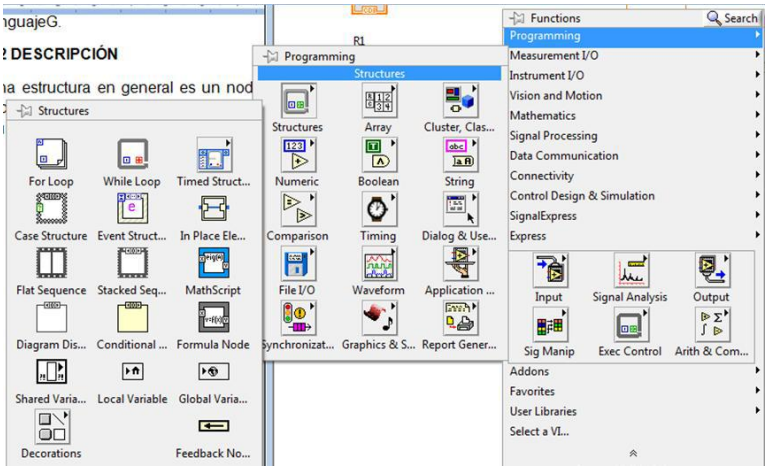

Figura 2-13. Menú de estructuras del Diagrama de Bloques

2.8.1 Estructura While Loop

La estructura While Loop es un ciclo que repite el sub diagrama que contiene en él hasta que una condición determinada se cumpla. En LabVIEW está representada de la forma que se muestra en la figura 2.14.

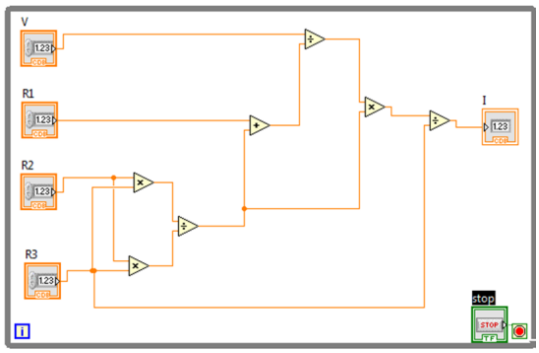

Figura 2-14. Estructura While Loop

Las instrucciones del ciclo se repetirán hasta que al terminal de condición llegue un valor true (verdadero). Si se desea cambiar la lógica del terminal de condición que es el icono azul en la esquina inferior izquierda, es decir, que el ciclo se repita tantas veces hasta que a éste

llegue un valor false (falso), solo se deberá hacer clic derecho en la terminal deseada y seleccionar la opción Stop If True del menú mostrado.

El terminal de iteración determina el número de veces que se ha ejecutado el programa dentro el ciclo y puede ser utilizado para visualización o para alguna operación dentro de la estructura que complemente a ésta. Ésta terminal varía desde 0 hasta N-1 donde N es el número de iteraciones realizadas por el ciclo.

2.8.2 Estructura For Loop

La estructura For Loop es un ciclo en donde se repite el diagrama que contiene un número definido de veces por una contante que se asigna en el icono de la parte superior izquierda con una constante. En LabVIEW está representada por el marco que se muestra en la figura 2.15.

Figura 2-15. Estructura For Loop

La terminal de iteración indica el número de veces que se ha ejecutado el ciclo. Varía desde 0 hasta N-1 donde N es el número total de iteraciones que realiza el ciclo. El control de iteraciones contiene el número de veces que se ejecutará el sub diagrama contenido en el ciclo.

Es muy frecuente que en las estructuras While Loop y For Loop se usen pasar datos entre iteraciones, es decir, que el resultado de ésta al final de un ciclo será la base para un nuevo ciclo. Para ello se utilizan los “shift registers”, están formados por un par de terminales en forma de flechas dentro de un recuadro que se adaptan a cualquier tipo de dato y que están

localizados a cada lado de los bordes de la estructura como se muestra en la figura 2.16, poniéndolos, con clic derecho sobre la estructura en la opción >>add shift register. La terminal derecha almacena el dato una vez concluya la iteración y le entrega el dato a la terminal de la izquierda para que sea utilizado en la próxima iteración. En la primera iteración, el sistema podría tomar un número que no sea correcto, por tanto se debe inicializar, desde afuera, con un valor constante conveniente del mismo tipo de la variable utilizada (figura 2.17).

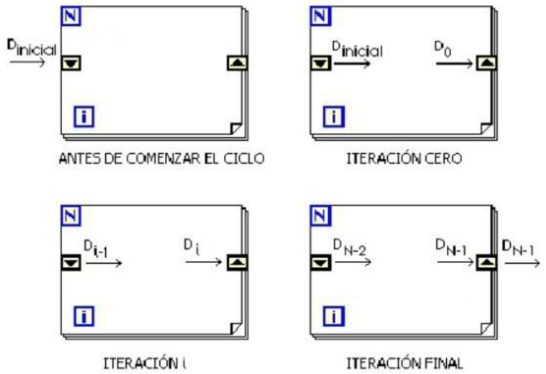

Figura 2-16. Secuencia Shift Register.

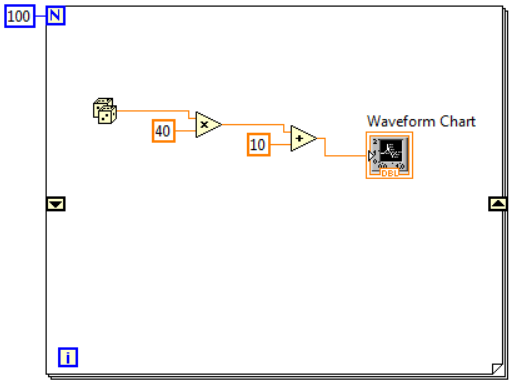

Figura 2-17. Shift Register.

2.8.3 Estructura Sequence

La estructura sequence tiene la apariencia de la figura 2.18 y permite ejecutar varios su diagramas de manera ordenada, controlada y jerarquizada por el programador, establecido por la programación. Esta estructura posee varios su diagramas denominados “frames” que se ejecutan en estricto orden y sólo es visible uno a la vez, se visualiza el que se ejecuta.

Hay que tener en cuenta que en LabVIEW esta estructura se ejecutará cuando se disponga de todos los datos de entrada, es decir, si falta algún dato de entrada la estructura no podrá llevar a cabo su función y por lo tanto no se podrá ejecutar.

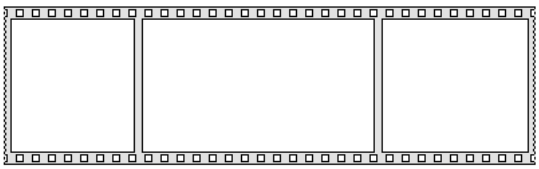

Figura 2-18. Estructura Sequence sin tareas dentro de sus frames.

2.8.4 Estructura Case

La estructura Case (figura 2.19) posee varios sub diagramas denominados casos (cases) de los cuales sólo se ejecuta uno dependiendo de la conducción de ejecución, se emplea cuando dos o más acciones alternativas dependen de una condición.

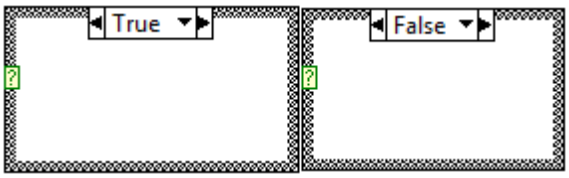

Figura 2-19. Estructura Case sin Tareas.

Dependiendo del tipo de variable asociada al terminal de selección la estructura se comportará como un IF o como un CASE. Si el valor cableado es booleano la estructura tendrá dos casos FALSE y TRUE, pero si es numérico o cadena la estructura podrá tener desde 2 hasta 65535 casos.

2.9 Graficadores

Es importante poder visualizar los resultados del programa desarrollado de una forma clara y fácil de interpretar, para esto LabVIEW cuenta con 14 diferentes tipos de graficadores que se encuentran en la paleta de controls>>modern>>graph, figura 2.20

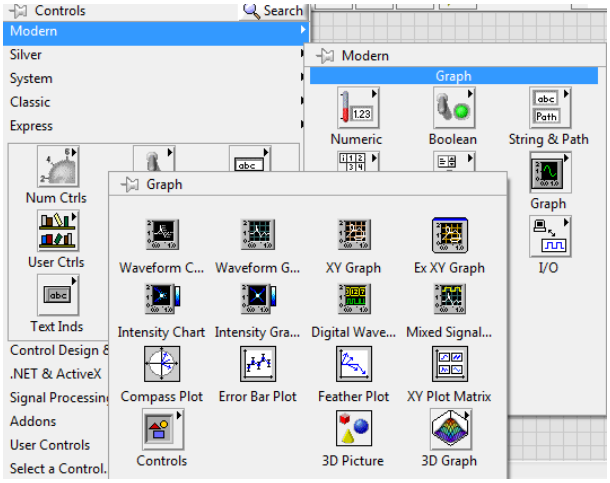

Figura 2-20. Menú de Graficadores.

Los gráficos en LabVIEW son un tipo diferente de indicador ya sea de dos o tres dimensiones, que permiten visualizar resultados, pero así como los indicadores normales, éstos pueden ser convertidos en controles en cualquier momento.

2.9.1 Gráfica Waveform Chart

El resultado de una gráfica tipo waveform chart mostrado en la figura 2.21 puede tomar diferentes formas al ser alambrado con diferentes tipos de variables de datos válidas para él, tales como:

- Escalares
- Vectores
- En función del tiempo
- Array
- Nodos

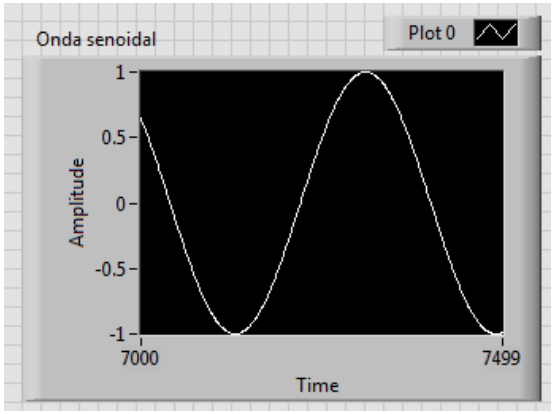

Figura 2-21. Waveform Chart.

2.9.2 Graficador Waveform Graph

Este tipo de gráfica está diseñado especialmente para graficar señales muestreadas, los datos estarán siempre referidos al eje X de manera continua. Un ejemplo de este tipo de gráfico lo vemos en la figura 2.21. Este tipo de gráficas es el utilizado en este proyecto.

Los tipos de datos que acepta son:

- Vectores
- WDT
- Array 2D

- Array de clúster
- Clústers para múltiples gráficas

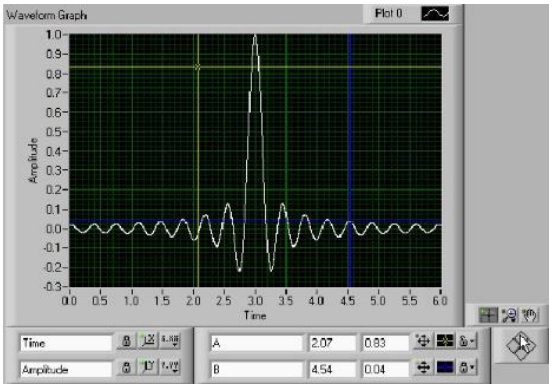

Figura 2-22. Waveform Graph.

2.9.3 Gráfica XY Graph

El gráfico resultado de una gráfica XY Graph, es una gráfica cartesiana de propósito general. La figura 3.62 muestra la gráfica y sus respectivos controles e indicadores.

Los tipos de datos aceptados por este graficador son:

Un registro conformado por dos vectores, el vector uno con los datos de X y el vector dos con los datos de Y. Un arreglo de registros. Cada registro está conformado por un valor X escalar y un valor Y escalar. Un arreglo de clústers. Cada clúster está conformado por un arreglo de datos [X] y un arreglo de datos [Y].

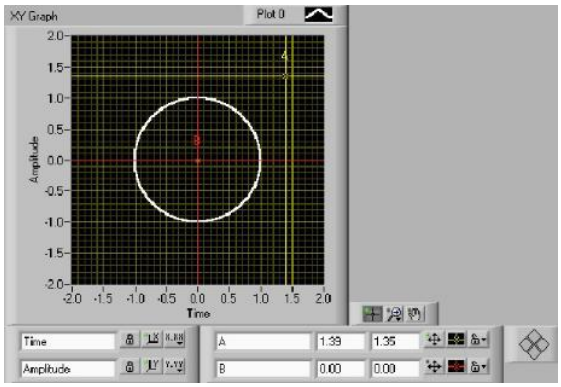

Figura 2-23. Waveform XY Graph.

2.10 SUBVI´S

En LabVIEW un SubVI´s o sub-programa es un VI que está siendo utilizado dentro de otro, los subVI permiten que una aplicación sea más simple, al hacer su diagrama de bloques más sencillo, permitiendo que aplicaciones extensas puedan ser divididas en varias tareas pequeñas, las que a su vez pueden ser divididas en otras tareas más pequeñas y así sucesivamente, donde finalmente podríamos tener un VI compuesto de básicamente solo SubVI's

El icono que identifica un VI, está ubicado en la parte superior derecha del panel frontal. Para editarlo se debe hacer clic derecho en él y seleccionar >>Edit Icon como se observa en la figura 2.24.

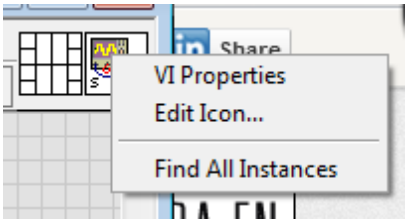

Figura 2-24. Icono que representa un SubVI.

Los conectores de un VI pueden enviar y recibir datos a un VI mayor cuando se esté utilizando como subVI. Para que esto sea posible primero habría que guardar el VI de manera normal pero modificando las terminales de conexión. El icono elaborado será entonces reemplazado por las terminales que LabVIEW espera que sean alambradas. Una vez asociado un conector con algún control o indicador, éste tomará el color del tipo de representación de la variable seleccionada.

Para adicionar un subVI en el diagrama de un VI, se siguen los siguientes pasos:

1. Para empezar el VI que se quiere utilizar ya deberá de estar previamente guardado en la carpeta correspondiente.
2. En el panel de diagramas del VI seleccione el menú >>Select a VI de la paleta de funciones.

3. La opción Select a VI, abrirá un cuadro de diálogo donde se puede seleccionar el VI que se requiere utilizar como subVI, como si se fuera a abrir ese VI.
4. Se busca y selecciona el VI deseado y luego se presiona Abrir.
5. Se ubica el cursor en el lugar del diagrama donde se desea ubicar el subVI, luego se hace un clic. Se observará el icono del subVI en el diagrama del VI en desarrollo.

CAPÍTULO III: ESTÁNDAR SCPI Y SUS CARACTERISTICAS

3.1 SCPI

Durante muchos años, los fabricantes de instrumentos trabajaron para estandarizar la interfaz mecánica y eléctrica entre instrumentos y computadoras. El surgimiento de la norma IEEE 488.1 en 1975, vino a dar en parte fin a esta tarea, e inició la búsqueda de la estandarización de los mensajes enviados sobre esta y otras interfaces, problema al cual, los fabricantes de instrumentos dieron como respuesta una gran variedad de órdenes para instrumentos, que forzaban a los usuarios a aprender un vocabulario para cada instrumento, así como a acrecentar el tiempo en el desarrollo, aprendizaje, mantenimiento y actualización de sistema. Aunque en 1987 el estándar IEEE 488.2 definió los roles de instrumentos y controladores en un sistema de medición, un esquema estructurado de comunicación y la manera de como enviar órdenes a un instrumento y como responder a ellas por parte del controlador. En general, no especificaba las ordenes o características que deberían ser implementadas en un instrumento, ocasionando que dos instrumentos similares, bajo este estándar manejaran conjuntos de ordenes diferentes.

Esfuerzos aislados fueron realizados con el objeto de alcanzar un modelo único para el diseño de instrumentos programables. En primera instancia por Hewlett Packard, ahora Agilent Technologies, quien desarrolló un conjunto de ordenes comunes basadas en los parámetros de las señales analizadas y un diagrama a bloques universal, para sus instrumentos. Este estándar interno fue el primero de su tipo en darse a conocer bajo el nombre de HP-SL y TMS posteriormente. Por otra parte, Tektronix examinó la posibilidad de almacenar y transportar mediciones de datos, así como compartir éstos entre diferentes instrumentos, concluyendo con la presentación de un formato de intercambio de datos (DIF, *Data Intercharge Format*), combinando los datos generados de señal con información contextual acerca de cómo es hecha la medición.

Finalmente, fue la conferencia tecnológica para miembros del IEEE (AUTOTESTCON, *Automated Test Conference*) de 1989, la que concentro a este grupo de compañías interesadas en el desarrollo del anhelado estándar. Así fue como ocho meses después, y tras la colaboración de nueve de estas compañías, surgió la norma SCPI (*Standard Commands for*

Programmable Instruments) y la consecuente creación de un consorcio bajo el mismo nombre, encargado de la regulación de este estándar.

Si bien el estudio de SCPI gira alrededor del conocimiento de los procedimientos empleados en la construcción de órdenes y las relaciones establecidas entre ellas, existen algunos otros argumentos que al margen de lo que se esperara representan la medula de la norma, tal como lo son el modelado y la clasificación de instrumentos. Aunado a esto, la norma SCPI destaca otros puntos de vital importancia en el desarrollo de cualquier aplicación relacionada con el control de instrumentos, tal es el caso del manejo de la estructura de registros de estado y el formato de intercambio de datos.

3.1.1 Objetivo de SCPI

La meta de SCPI es la reducción de tiempo en el desarrollo de aplicaciones, proporcionando un entorno consistente de programación para el control de instrumentos y manejo de sus datos. Este entorno consistente de programación es alcanzado mediante el uso de mensajes, respuestas y formatos de datos definidos para todos los instrumentos bajo SCPI.

Un entorno de programación consistente usa las mismas órdenes y parámetros para controlar instrumentos que presentan una funcionalidad similar. Estas órdenes y parámetros son enviados desde un controlador al instrumento usando interfaces como IEEE 488.1, bus VXI, RS-232C, USB, etc. Los instrumentos controlables mediante SCPI presentan gran flexibilidad en el uso de órdenes y parámetros.

La respuesta del instrumento al controlador puede ser información de datos o estado. El formato de respuesta de los instrumentos a un cuestionamiento particular se encuentra bien definido y reduce el esfuerzo de programación para comprender dicha información.

La consistencia en la programación mediante SCPI es manejada de dos maneras: vertical y horizontal.

La consistencia vertical define mensajes para una clase de instrumento, de esta forma se podrá usar una misma orden para leer voltaje de DC en diferentes multímetros. La

consistencia horizontal hace referencia al uso de una misma orden para controlar funciones similares a través de diferentes clases de instrumentos, así una orden de disparo podrá ser usado en osciloscopios, generadores de funciones u otro instrumento que realice esta función.

Una clave para alcanzar esta consistencia es la reducción de caminos para el control de funciones similares de un instrumento. La filosofía de SCPI manifiesta el uso de una misma orden SCPI para todas las funciones afines en un instrumento. SCPI emplea nombres estándares en la industria y términos que apoyan fabricantes y usuarios.

SCPI proporciona diferentes niveles para el control de instrumentos. Ordenes simples de medida proveen un control fácil y rápido. Así, mientras más detalladas sean las ordenes usadas se obtendrá un control más tradicional del instrumento.

SCPI está diseñado para ser expandido con nuevas definiciones de órdenes en el futuro sin causar problemas de programación. Un objetivo implícito en todos los nuevos instrumentos controlables mediante SCPI es mantener la compatibilidad de programas con instrumentos bajo SCPI ya existentes, sin embargo este hecho puede resultar erróneo. Así, la compatibilidad de un programa de prueba es asegurada hacia versiones de instrumentos SCPI posteriores, pero no así para anteriores.

Con el objeto de promover su uso y aceptación en diferentes ámbitos, SCPI se encuentra disponible públicamente para su implementación por cualquiera, sea o no miembro del consorcio SCPI.

El consorcio no da validez a documentos de investigación o documentación provisional, siendo únicamente estándares aprobados los puestos a disposición pública.

3.1.2 Intercambiabilidad de instrumentos

Uno de los principales fines perseguidos por SCPI es la capacidad de intercambio de instrumentos controlables bajo la norma, dentro de un sistema ATE. Sin embargo, SCPI no es un estándar que provee completamente intercambiabilidad entre instrumentos. SCPI ayuda a alcanzar esta intercambiabilidad mediante la definición de órdenes y respuestas de los

instrumentos, pero no define funcionalidad, precisión, resolución y conexiones entre dispositivos, argumentos necesarios para alcanzar una verdadera intercambiabilidad de instrumentos sin afectar los sistemas en hardware o software.

3.1.3 Ciclo de vida de SCPI

SCPI es un estándar "vivo". El surgimiento de nuevas tecnologías e instrumentos hace necesaria la adición constante de nuevas pautas y ordenes al estándar. La inclusión de nuevas órdenes puede ser propuesta tanto por miembros del consorcio SCPI, como por otras partes interesadas. Las propuestas aceptadas por el consorcio son publicadas y distribuidas a las compañías miembros para su uso inmediato. Las propuestas aprobadas son revisadas anualmente. Después de cada revisión anual, se publica una nueva versión del estándar SCPI. Todos los instrumentos bajo la norma SCPI pueden ser cuestionados mediante una orden específica, para determinar la versión (año) de la norma SCPI usada por estos.

En general, una orden propuesta como una adición a la norma SCPI deberá como primera condición ser construida de acuerdo a las reglas de estilo y sintaxis establecidas por la norma. Estas reglas establecerán el carácter de la orden, la categoría en la que será puesta, y si existe o no una orden que permita el control de la misma funcionalidad.

SCPI está diseñado para crecer bajo un ambiente de compatibilidad. Para un programa de control de algún instrumento, esto significa que las adiciones hechas a SCPI no modificarán el significado de las ordenes existentes, por lo que este no necesitara ser rectificado, excepto por la inclusión de nuevas funcionalidades. Mientras que para un instrumento, esto significa que las extensiones hechas a la norma no harán obsoletos a los instrumentos ya existentes.

Es posible que una orden tenga que ser alterada o borrada para permitir que sean implementadas nuevas funcionalidades. Las propuestas de cambios que rompan el argumento de compatibilidad únicamente serán aceptadas si existe evidencia irrefutable de que los beneficios alcanzados por estas modificaciones son superiores.

Desde antes que un fabricante elija construir un instrumento controlable mediante SCPI con capacidades que no son cubiertas por la versión actual del estándar, este presumiblemente

presentara al consorcio de manera anticipada, el conjunto de pautas y nuevas órdenes que pretenda, formen parte de la norma SCPI, en acuerdo con el proceso de acreditación correspondiente.

3.1.4 Generación de mnemotécnicos

Cada mnemotécnico usado en la construcción de cabeceras de control de instrumento cuenta con una representación larga y otra corta, por lo que una cabecera de este tipo puede presentarse de estas dos maneras. El envío de una cabecera a un instrumento que no es exactamente de ninguna de las dos formas será causa de un error. El estándar IEEE 488.2 limita la longitud de una cabecera a doce caracteres, incluyendo cualquier sufijo que pueda incluirse. La cabecera en su forma larga es una simple palabra o una abreviatura de una frase. Una representación corta es una abreviatura de la forma larga de la cabecera. Con el objetivo de mantener un alto grado consistencia, SCPI define reglas para la generación de mnemotécnicos que conforman las cabeceras de control de instrumento.

Durante el presente documento, se emplea una notación especial para diferenciar la forma larga de una palabra clave de su representación corta. Así, la forma larga de una palabra será presentada en base a una combinación de letras mayúsculas y minúsculas, en donde la porción en letras mayúsculas corresponderá a su forma corta (INPut, SENSe, etc.). Esta notación es muy particular de todos aquellos documentos referentes a la norma SCPI, incluyendo la norma misma.

3.1.5 Generación de la estructura jerárquica

La definición de órdenes SCPI se encuentra ligada directamente al manejo de una estructura jerárquica también conocida como árbol. Los elementos de primer nivel en la estructura jerarquía y raíces de esta, son producto del modelado mismo de los instrumentos, y representan la palabra clave de cada agrupación de ordenes llamada subsistema. De esta forma las órdenes asociadas con un subsistema en particular son agrupadas bajo un nodo común en la jerarquía, análogamente a ramas conectadas a otra común. Cada nuevo nodo que origine una rama será caracterizado con un mnemotécnico, este nodo dará origen a nuevas

ramas y estas a otras sucesivamente, de acuerdo a la funcionalidad que se desea cubrir con la orden en desarrollo. Así, la lectura de una orden se hará siguiendo los nodos de cada rama generada a partir de la raíz del árbol o estructura jerárquica.

El procedimiento empleado permite a los mnemotécnicos que constituyen estas órdenes ser utilizados diversas veces para diferentes propósitos, sin que estos colisionen. De este modo, dos órdenes diferentes puedan hacer uso de un mismo mnemotécnico, poseyendo una independencia total una de otra.

La Figura 2.1 ejemplifica la construcción de ordenes SCPI bajo el nodo o subsistema SENSe en su nivel superior, nótese la existencia de dos nodos con mnemotécnicos iguales, que sin embargo no guardan relación alguna en la jerarquía. Así, la sintaxis de una orden para la modificación del ancho de banda de video en un instrumento, tendrá la forma siguiente:

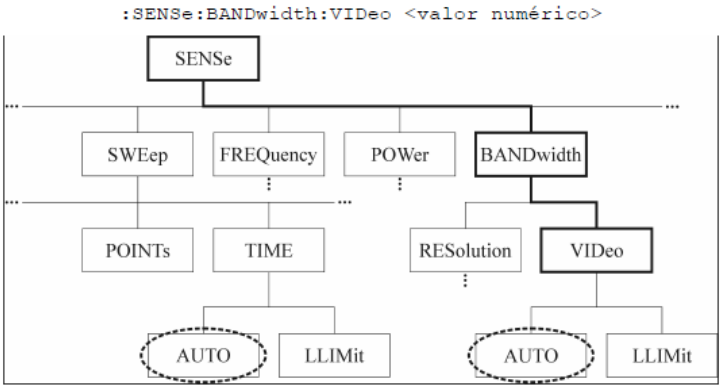

Figura 3-1. Esquema de estructura jerárquica SCPI.

3.2 Parámetros

SCPI usa las formas de parámetros descritas en la sección 7.7 del estándar IEEE 488.2, con algunas restricciones. Un parámetro indica el número y disposición de los argumentos en una orden.

Adicionalmente al uso de un tipo de parámetro particular, SCPI permite el uso de literales, o bien de una combinación parámetro-literal, como argumentos en una orden. Una literal es típicamente una palabra que enumera un argumento no descrito por algún parámetro definido

por el estándar. Los tipos de parámetros manejados por SCPI son: caracteres, numéricos, boleanos, unidades y expresiones.

3.2.1 Carácter

Las reglas empleadas en la construcción de este tipo de parámetros son las mismas que las usadas en la especificación de cabeceras de programa, sin embargo en muchos casos los estándares industriales toman preferencia sobre estas pautas. Algunos parámetros de este género son usados para predefinir parámetros de tipo numérico.

3.2.2 Numérico

Los elementos numéricos son usados para representar cualidades numéricas. Estos no son empleados para funciones de selección (uno de n). Cualquier número que exceda el valor de ±9.9 E 37 generara un error (-222, "Dato fuera de rango"). Los números usados por un instrumento son redondeados al más cercano valor que éste acepte, sin generar un error. Algunas expresiones de caracteres son definidas como formas especiales de números, estas son: DEFault, MINimun, MAXimun, UP, DOWN, NAN (*Not a Number*), INFinity, NINF (*Negative INFinity*). Las expresiones numéricas y etiquetas son consideradas también dentro de este tipo de parámetros.

3.2.3 Boleano

Un valor de 0 o 1 sin ningún atributo de unidad es asignado a este parámetro. Dentro de la documentación SCPI este tipo de parámetro son referenciados mediante la expresión ON/OFF. Las ordenes de pregunta cuya respuesta sea un parámetro boleano devolverán como respuesta un 1 o 0, nunca una expresión ON/OFF.

3.2.4 Unidades de medida y sufijos

Las unidades y sufijos usados por la norma SCPI obedecen a las especificaciones establecidas en la sección correspondiente del estándar IEEE 488.2. La industria, quien juega un papel importante en la determinación de algunos conceptos de la norma SCPI, es la causante de la expansión de las unidades de potencia y amplitud definidas por el estándar IEEE 488.2

(Watts, Volts, dBm y dBV), mediante la adición de unidades como uV, dBuV, dBuW y dBmV.

3.3 Instrumentos

Cualquier instrumento controlable mediante SCPI, dentro del ámbito normativo es citado como instrumento SCPI, de forma tal que dicha denominación sustente el hecho de pensar que el dispositivo en cuestión ha sido diseñado bajo los cánones que marca SCPI.

3.3.1 Modelo de un instrumento

El modelado de instrumentos en SCPI, es usado como un medio para conseguir la compatibilidad entre instrumentos. SCPI concierne por si mismo tres tipos de compatibilidad. La primera llamada compatibilidad vertical se da cuando dos instrumentos del mismo tipo tienen controles idénticos.

Por ejemplo, dos osciloscopios de diferente fabricante, donde ambos tienen los mismos controles para sus tiempos base, disparos, etc. son compatibles verticalmente.

La segunda forma de compatibilidad denominada horizontal, hace referencia a dos instrumentos capaces de hacer una misma medición, independiente de la técnica usada. De esta manera ambos instrumentos pueden usar la misma orden para hacer esta medición. Por ejemplo, un osciloscopio y un contador pueden llevar a cabo una medición del tiempo de subida para un pulso particular, mediante dos técnicas diferentes, mostrando una compatibilidad horizontal.

El tercer tipo de compatibilidad evocada como funcional se establece cuando dos instrumentos realizan una similar función mediante una misma orden. Por ejemplo, un analizador de espectros y un generador de RF que efectúan con diferente propósito, operaciones de barridos en frecuencia mediante ordenes idénticas, presentaran una compatibilidad funcional en esta área.

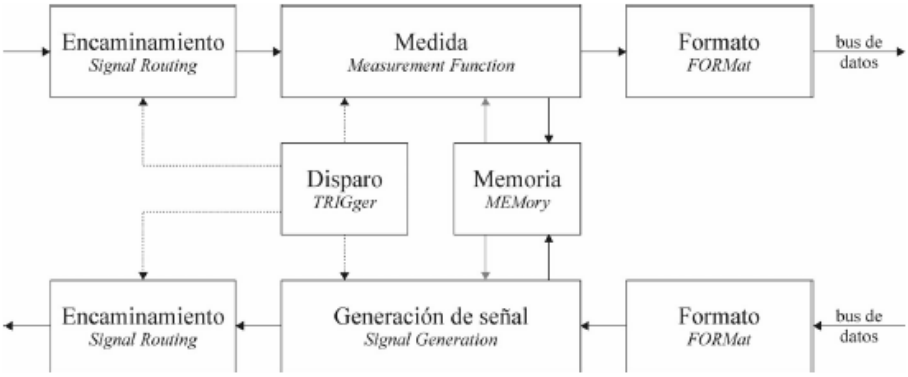

Figura 3-2. Modelo Básico de un Instrumento Programable.

La Figura 2.2 representa la manera en como es visto funcionalmente un instrumento y categorizado por SCPI. El flujo de datos es representado por líneas sólidas, mientras que el flujo de control es descrito por líneas punteadas. El propósito de esta categorización es proporcionar organización y consistencia entre las diversas órdenes disponibles por SCPI. El modelo define como los elementos del lenguaje deben ser agrupados y asignados en la estructura jerarquía SCPI, para la construcción de las órdenes. Las áreas de mayor impacto en el aspecto funcional del instrumento son consolidadas como bloques; cada uno de estos bloques se encuentra en un primer nivel jerárquico de la estructura SCPI.

El modelo describe el flujo de la medición y aplicación de las señales de datos a través del instrumento así como la administración del flujo de datos asociados con las órdenes, cuestionamientos, calibraciones, accesos a memoria y otras funciones relacionadas que no son incluidas en este modelo.

El modelo no define como un instrumento maneja o interpreta el formato de los datos. Los instrumentos únicamente implementaran los bloques que requieran.

3.4 Clases de instrumentos

La clasificación de instrumentos planteada por SCPI está basada tanto en la manipulación del modelo básico de instrumentos establecido por la norma, como en la funcionalidad presentada por el instrumento en cuestión. Los objetivos que esta tipificación persigue son:

- Reducir el tiempo en el desarrollo de productos basados en SCPI, guiando su diseño a través de un familiar punto de vista, como lo es la clase del instrumento.
- Alcanzar un alto grado de consistencia en implementaciones con la misma clase de instrumentos.

De manera sencilla la funcionalidad de un instrumento representa el que hace y como lo hace, así cada clase de instrumento expresara una funcionalidad en particular. A esta funcionalidad se le es asignado un mnemotécnico de acuerdo a las reglas estipuladas por SCPI. Estas palabras claves también definen funcionalidades adicionales.

3.5 Sintaxis y estilo

Los comandos en la norma SCPI se agrupan jerárquicamente en forma de árbol. En la raíz del árbol se encuentran los comandos que hacen referencia a los bloques que aparecen en el modelo de instrumento visto en la sección anterior. Cada uno de estos comandos se divide en un conjunto de ramas identificadas por palabras clave que a su vez identifican a funciones subordinadas al bloque raíz y así sucesivamente.

La notación utilizada es la siguiente:

- “:” indican el paso a un nivel jerárquico inferior. Para clarificar la notación también se ha utilizado identificación de los niveles inferiores.
- [ ] Palabras clave opcionales.
- < > Encierran el tipo de parámetro.
- “|” separa parámetros opcionales (solo se puede poner uno de ellos).
- “;” separa comandos que están en la misma línea (no cambia el nivel del último comando).
- “,” se usa para separar distintos parámetros dentro de un mismo nivel.
- “?” Indica que es un comando de consulta y que se espera una respuesta del equipo al que se envía. Es muy importante leer el dato solicitado; de no ser así al enviar otro comando se crea una situación de error en el instrumento.

SCPI se trata de un lenguaje basado en comandos e independiente de la conexión física, puede utilizarse sobre GPIB, USB, RS232, PCI, VXI, Ethernet, etc. En SCPI los distintos comandos se agrupan en varios subsistemas o familias, cada una de ellas identificada con un bloque funcional del Instrumento, lógicamente no todos los instrumentos tienen los mismos bloques. La agrupación se realiza de forma jerárquica empezando por la familia más genérica y particularizándose cada vez más. Cada familia o subfamilia está representada por un "keyword", de esta forma no hace falta memorizar cientos de mnemotécnicos. Algunas de las familias principales son: CALCulate, CALibration, CONTrol, DISPlay, MEMory, OUTPut, PROGram, SOURce, TRIGger, UNIT, etc.

Un ejemplo de comando es:

SYSTem:COMMunicate:SERial:BAUD 9600

Se puede ver como la familia raíz es SYSTem, luego van las subfamilias COMMunicate, SERial y BAUD. Entre las familias se pone en carácter ":" para separarlas. Las letras en minúscula se pueden omitir para conseguir mayor rapidez. Los parámetros que puedan tener los comandos van al final separados de la última "keyword" por un espacio. El comando acaba con un fin de línea.

En el caso de ejemplo anterior el comando sirve para poner la velocidad del puerto serie del equipo a 9600 baudios.

Otros comandos devuelven datos, por ejemplo, para preguntar al equipo la velocidad de comunicación del puerto serie se puede usar:

SYST:COMM:SER:BAUD?

Para separar los comandos se usa el carácter ";". Un ejemplo que equivale a los dos anteriores es:

SYST:COMM:SER:BAUD 9600; BAUD?

Si al ";" le sigue el carácter ":" se indica que la escritura del siguiente comando empieza por una familia raíz. Por ejemplo:

SYST:COMM:SER:BAUD 9600; :SYST:COMM:SER:BITS 8

Los comandos específicos definidos en la norma SCPI se dividen a su vez en dos grupos, los obligatorios y los opcionales. Los únicos bloques que son obligatorios son el de SYSTem y el de STATus.

Los comandos específicos definidos en la norma SCPI son los siguientes:

- MEASure	- INPut	- ROUTe	- TRACe|DATA
- CALCulate	- INSTrument	- SENSe	- TRIGger
- CALibration	- MEMory	- SOURce	- UNIT
- DIAGnostic	- MMEMory	- STATus	- VXI
- DISPlay	- OUPut	- SYSTem	
- FORMat	- PROGram	- TEST	

Los beneficios de SCPI es compatibilidad, esto es interoperabilidad entre diferentes instrumentos. El mismo comando que realiza una cierta función en un instrumento realizará exactamente la misma función en un instrumento completamente diferente, siempre y cuando ambos compartan esta capacidad. Un programa diseñado para controlar cierto tipo de instrumento, como un generador de funciones, funcionará con un generador de funciones de un diferente vendedor con pocos o ningún cambio.

CAPITULO IV: DESARROLLO DEL CONTROL Y LA INTERFAZ GRÁFICA DE USUARIO PARA EL GENERADOR DE SEÑALES AGILENT 33210A.

4.1 Generador de Funciones Agilent 33210A.

Este capítulo estará destinado a la explicación del desarrollo del control y la interfaz del generador de señales Agilent 33210A.

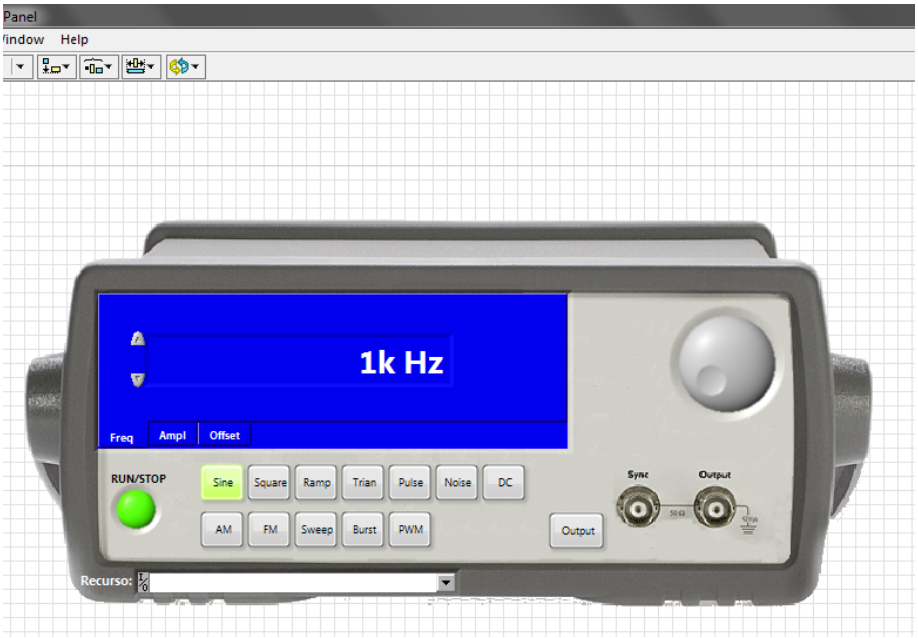

Figura 4-1. Generador de Señales Virtualizado

El control de dicho instrumento se hace bajo la plataforma de software LabView y utilizando el puerto USBTMC para la comunicación, este generador en particular tiene 3 puertos de comunicación los cuales son: GPIB, USBTMC y LAN, se utiliza para esta aplicación en particular el puerto USBTMC debido al costo de del cable USB, el cable GPIB es mucho más costoso!!.

El generador controlado por el código fuente de la figura 3.2, como se puede observar el diagrama está dividido en dos grandes partes, la primera corresponde a la etapa de comunicación e inicialización de variables, la segunda parte es la etapa de procesamiento de eventos producidos por el panel del generador virtual, la parte estética del generador quedará en segundo plano pues solo es la imagen de fondo con un pequeño retoque pictográfico con un software de edición de imágenes, nos centraremos en las funciones que se pueden generar a través de elementos de LabView y los comandos SCPI propios del generador.

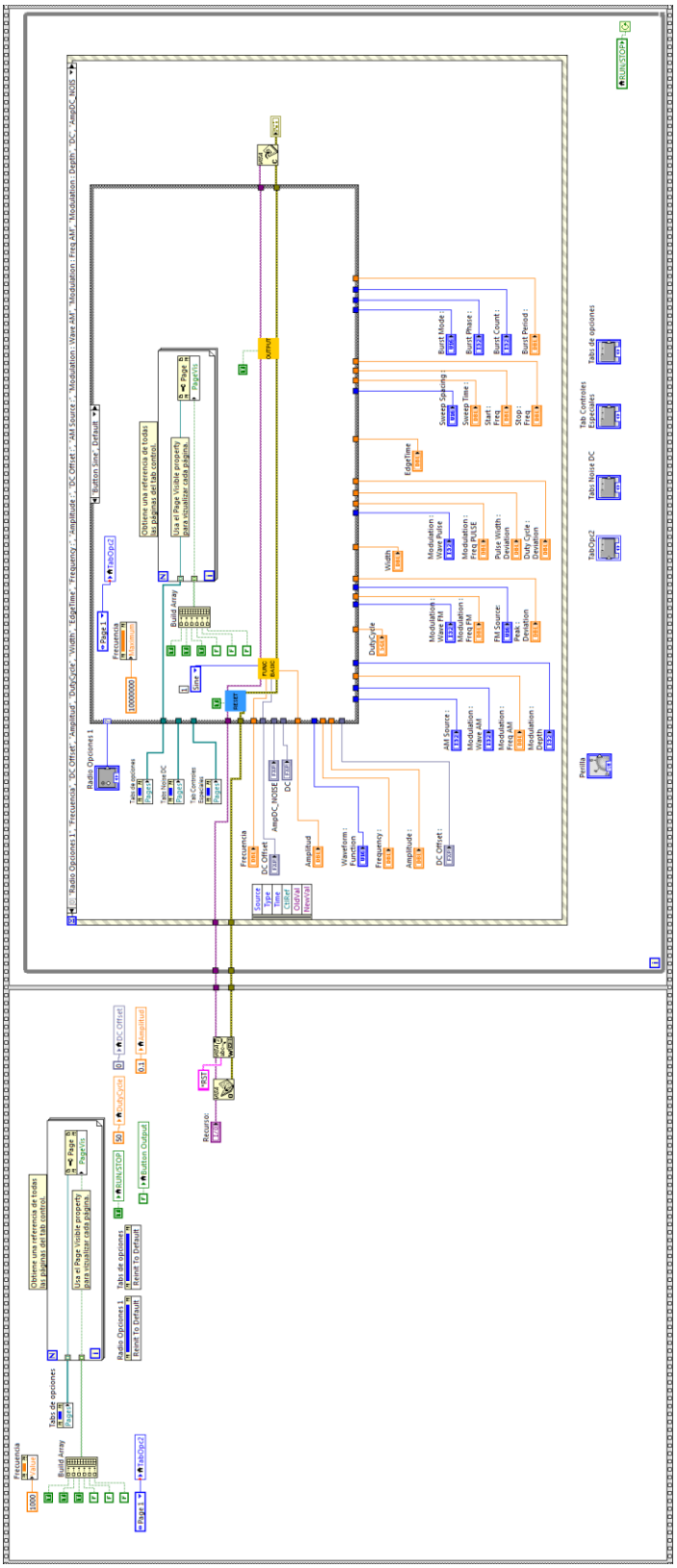

Figura 4-2. Diagrama de bloques para el generador.

Se hace uso de SubVIs para poder hacer el código más legible y de fácil mantenimiento, cabe mencionar que en este proyecto se respetaron todos los límites que tiene el generador los cuales son descritos en el "*hp_33210a_Service_Guide*", las funciones generadas por el instrumento son:

- Senoidal
- Cuadrada
- Rampa
- Triangular
- Pulsos
- DC
- Ruido
- Señal con modulación AM/FM
- Barrido de Frecuencia. (Sweep)
- Ráfaga (Burst)
- Y PWM

Antes de mostrar los subVIs que hacen cada una de estas funciones veremos algunos de los comandos SCPI del generador por ejemplo:

```
FUNCtion {SINusoid|SQUare|RAMP|PULSe|NOISe|DC|USER}
FUNCtion?
```

De la teoría del capítulo III podemos intuir que FUNCtion puede uno solo de los parámetros encerrados en llaves debido a que están separados por el carácter "|", y que en la siguiente línea se está haciendo una consulta porque tiene un símbolo de interrogación.

4.2 Construyendo comandos SCPI con Labview

Para poder construir comandos que sean entendidos por los instrumentos se hace intensivo el manejo de "Strings" o cadenas para poder enlazar cada comando, por ejemplo en la figura 3.2 podemos observar cómo se concatenan los diferentes comandos para poder generar las funciones básicas del instrumento, en este apartado no se profundiza sobre las funciones específicas de cada bloque solamente se describen las conexiones de los comandos SCPI en LabView para poder generar las señales, para mayor información y detalles específicos de cada función en el generador se debe consultar el *"Agilent 33210A User's Guide"*:

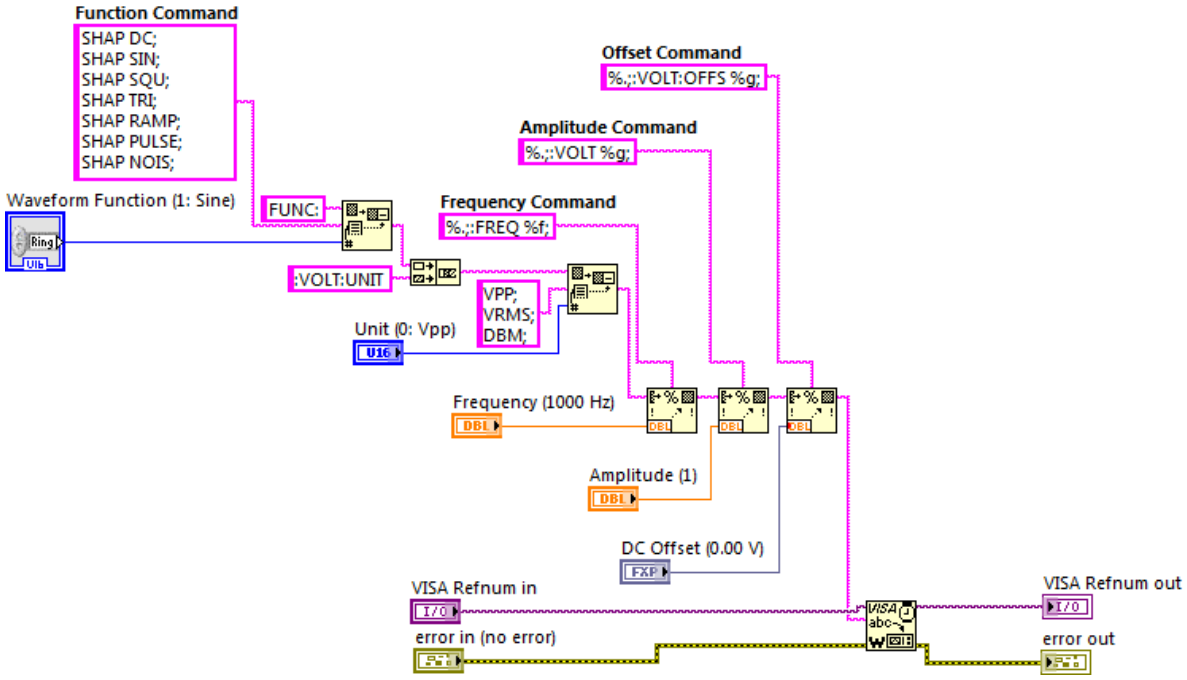

Figura 4-3. Código LabView para generar las señales básicas.

Cada comando excepto los tres últimos tienen la sintaxis pura SCPI, los tres últimos están combinados con formatos de los VI de Labview, específicamente el carácter "%" y el "%g" que Labview utiliza para dar formato a las cadenas de texto.

4.3 Diagramas de bloque en LabView de las funciones del generador de señales.

Ahora pasaremos a mostrar cada subVI con su respectivo diagrama de bloques:

SubVI encargado de dar Reset Maestro al instrumento.

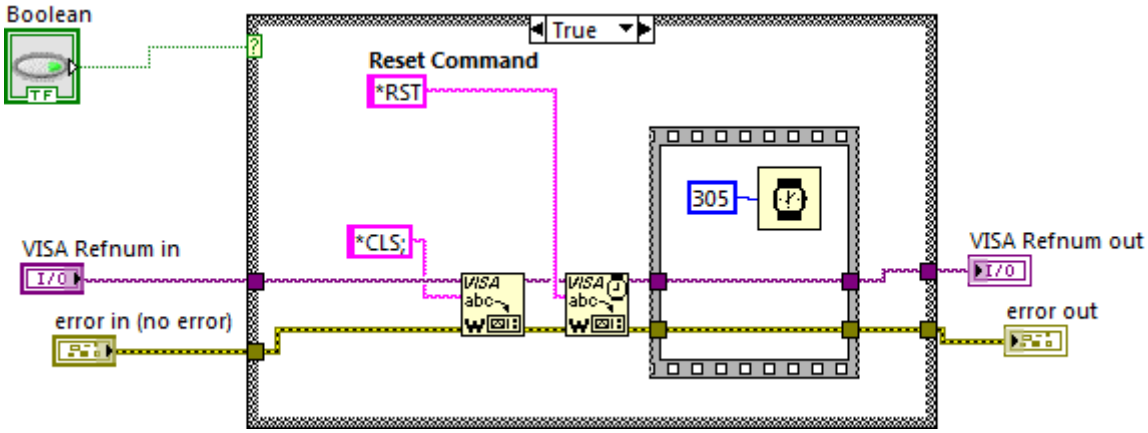

Figura 4-4. Diagrama de bloques del SubVI RESET.

Antes de dar la orden de RESET primero se limpia el status con *CLS;.

SubVI encargado de encender o apagar la salida del generador.

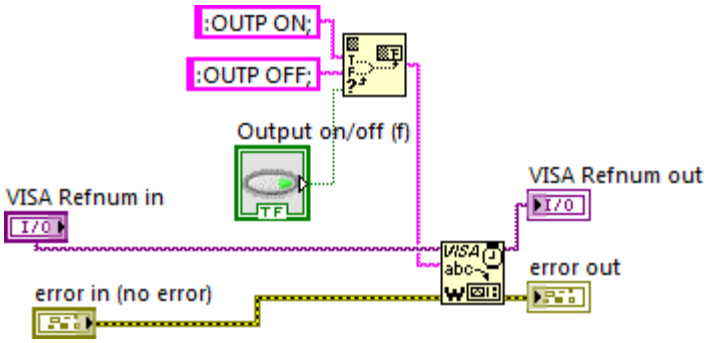

Figura 4-5. Diagrama de bloques del SubVI OUTPUT.

FUNC BASIC SubVI encargado de generar las funciones básicas del generador.

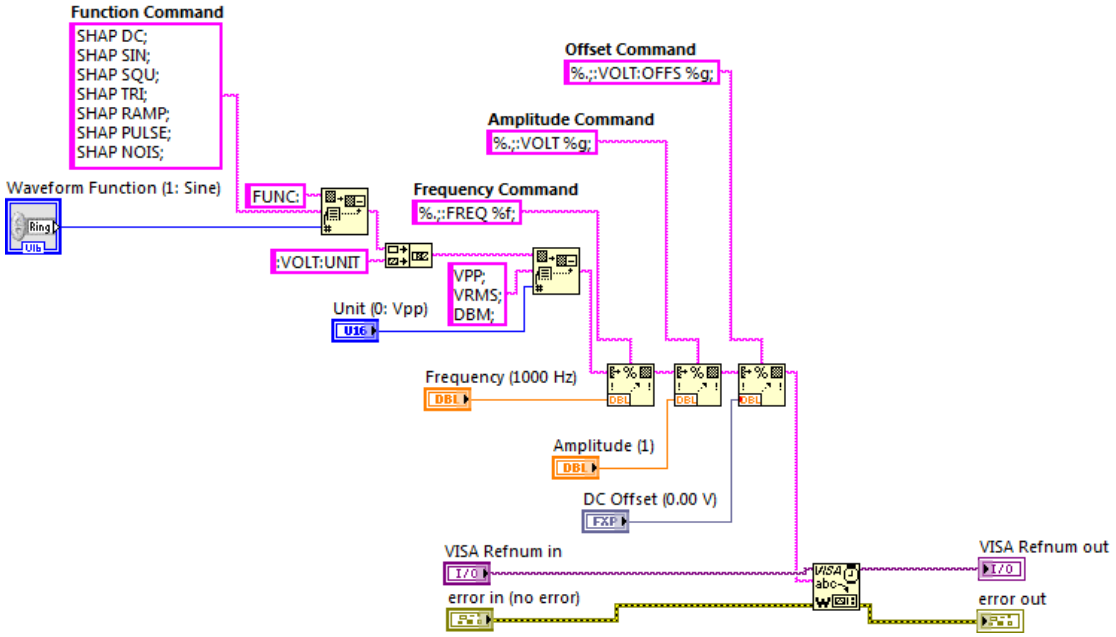

Figura 4-6. Diagrama de bloques del subvi FUNC BASIC.

FUNC PULSE SubVI encargado de generar la función PULSE del generador de señales:

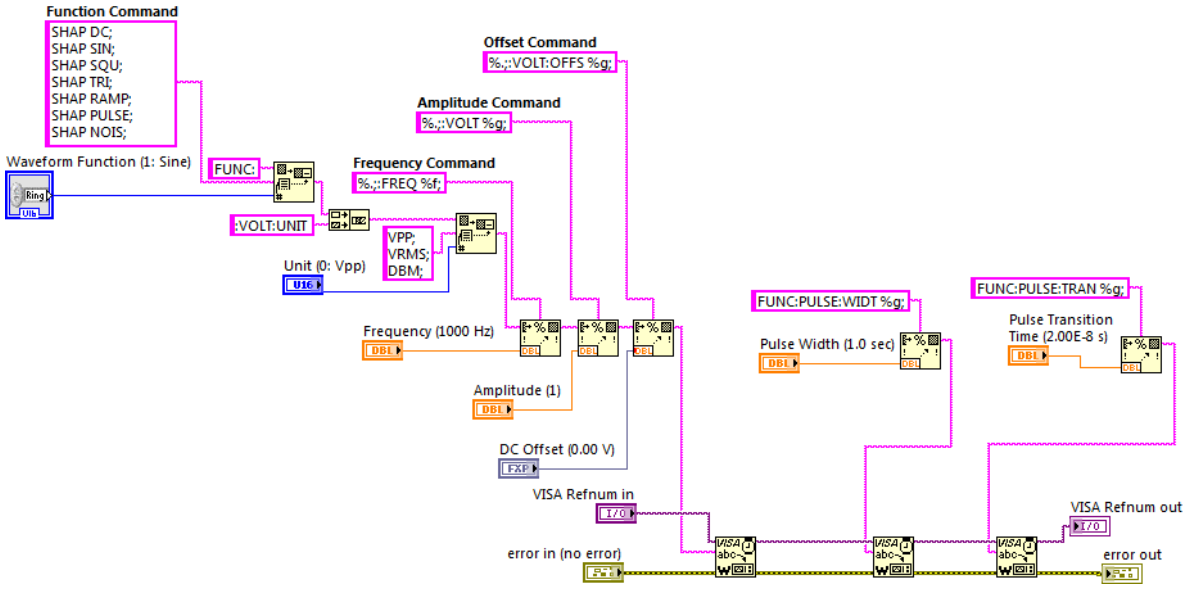

Figura 4-7. Diagrama de bloques del subvi FUNC PULSE.

AM MODUL SubVI encargado de hacer la modulación AM

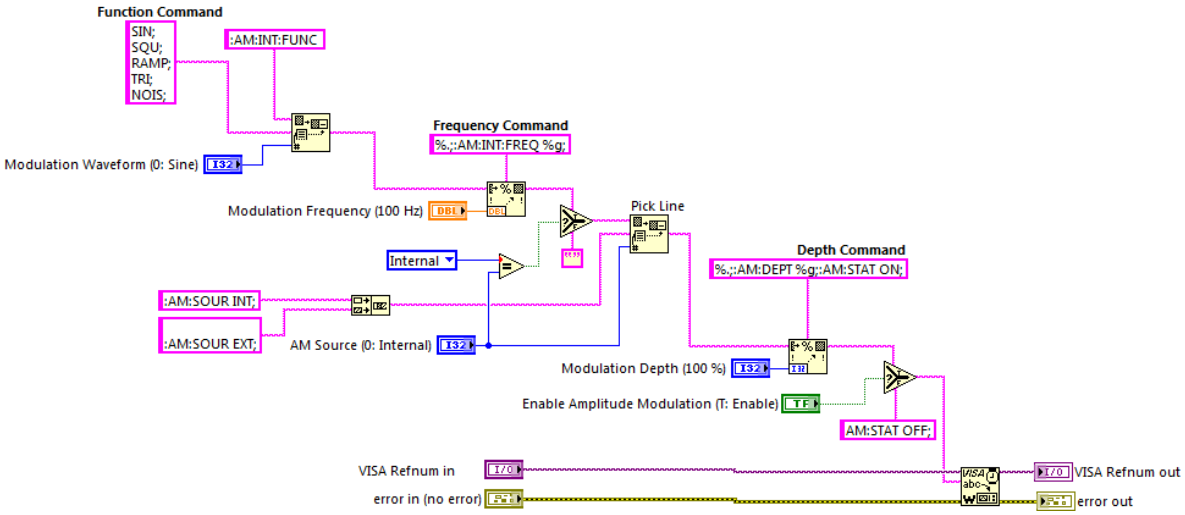

Figura 4-8. Diagrama de bloques del subvi AM MODUL.

SETUP CARRIER SubVI encargado de setear valores para la portadora de la señal.

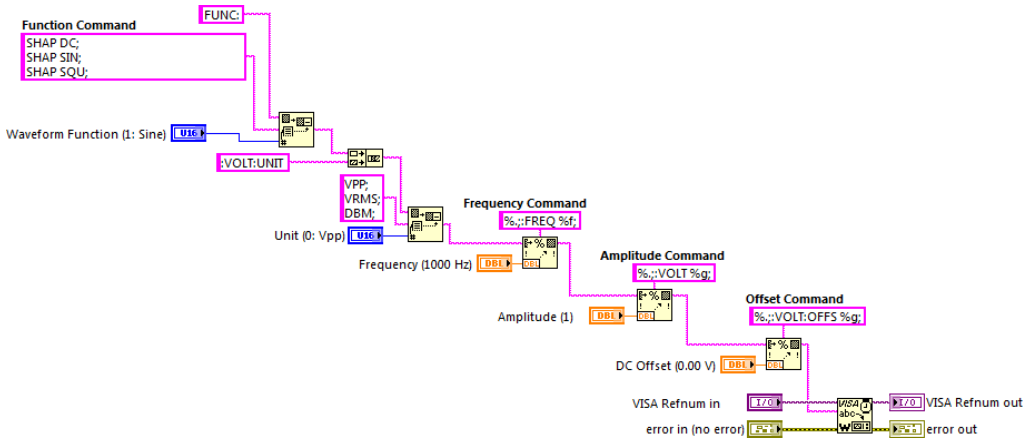

Figura 4-9. Diagrama de bloques del subvi SETUP CARRIER.

FM MODUL SubVI encargado de generar señales Moduladas en Frecuencia.

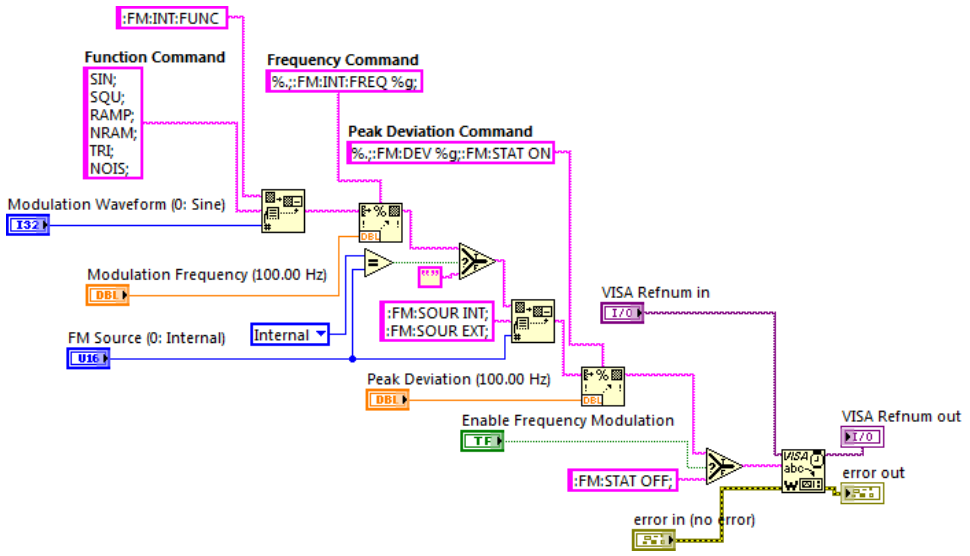

Figura 4-10. Diagrama de bloques del subvi FM MODUL.

FUNC SWEEP SubVI encargado de generar un barrido en frecuencia

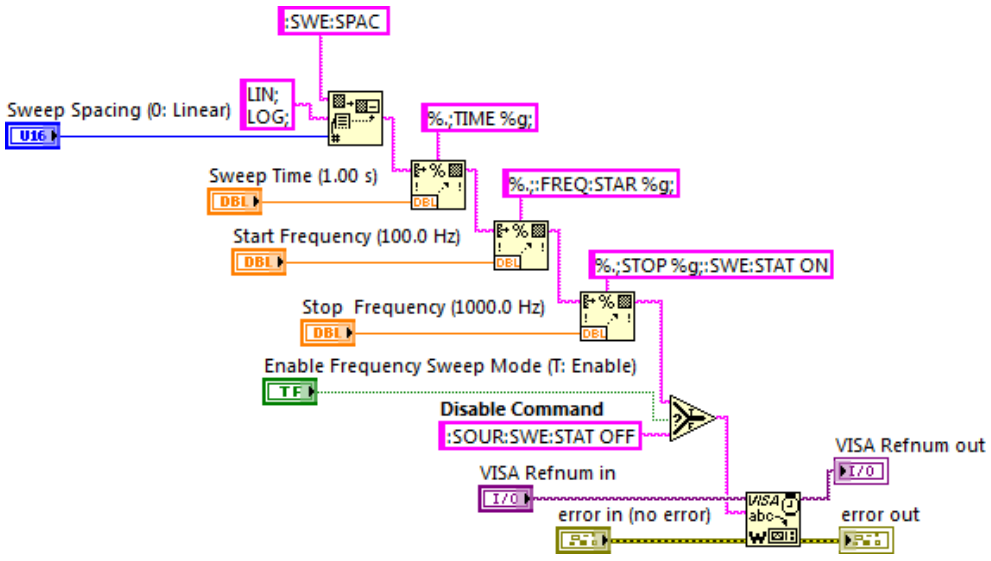

Figura 4-11. Diagrama de bloques del subvi FUNC SWEEP.

FUNC BURST SubVI encargado de generar señales en modo de disparos en el generador de señales.

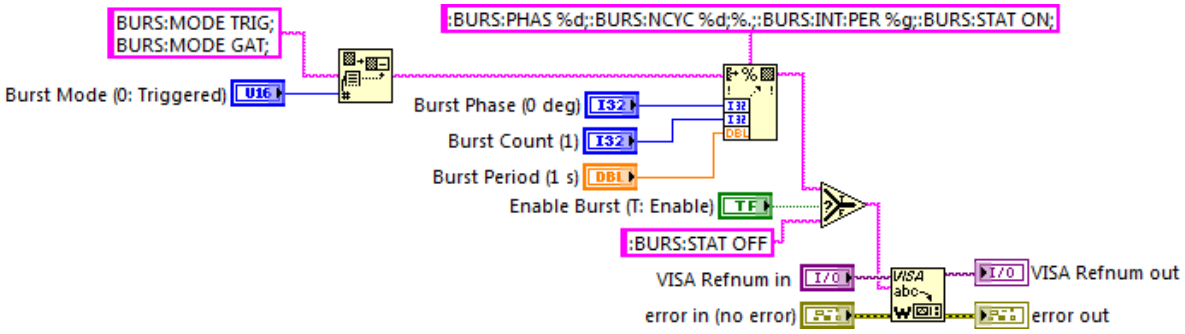

Figura 4-12. Diagrama de bloques del subvi FUNC BURST.

FUNC PWM SubVI encargado de generar ondas con Modulación de Ancho de Pulso (PWM).

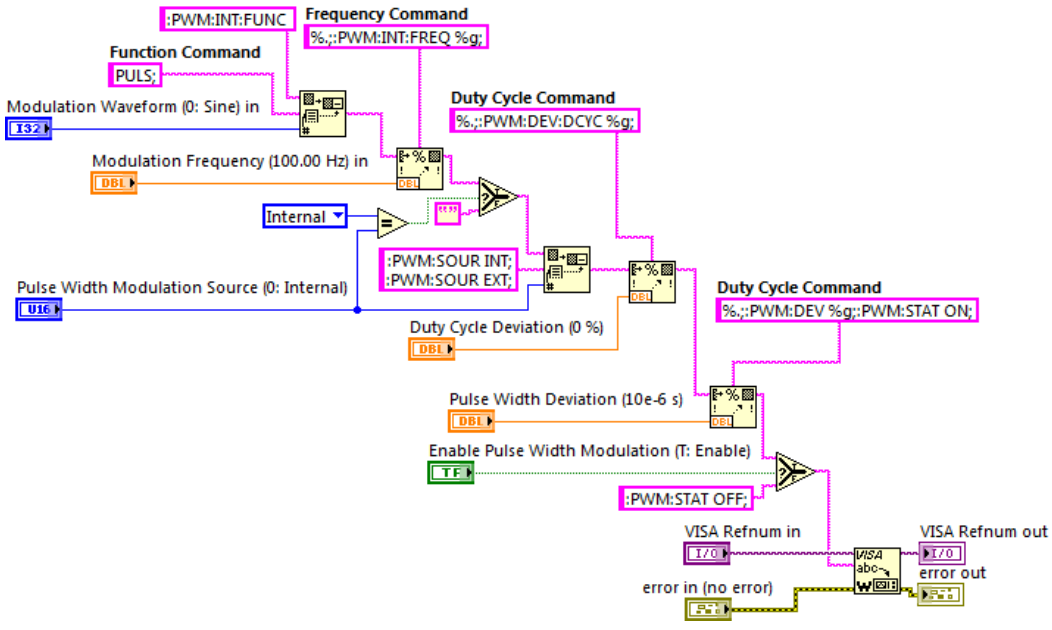

Figura 4-13. Diagrama de bloques del subvi FUNC PWM.

CAPITULO V: DESARROLLO DEL CONTROL E INTERFAZ GRÁFICA DE USUARIO PERSONALIZADA PARA EL OSCILOSCÓPIO AGILENT DSO1012A.

5.1 Osciloscopio DSO1012A.

Este capítulo estará destinado a la explicación del desarrollo del control y la interfaz personalizada del Osciloscopio, además se demostrará la factibilidad de comunicación con dicho instrumento implementando las funciones básicas del mismo, también tiene la capacidad de exportar datos como imagen y como archivo en Excel:

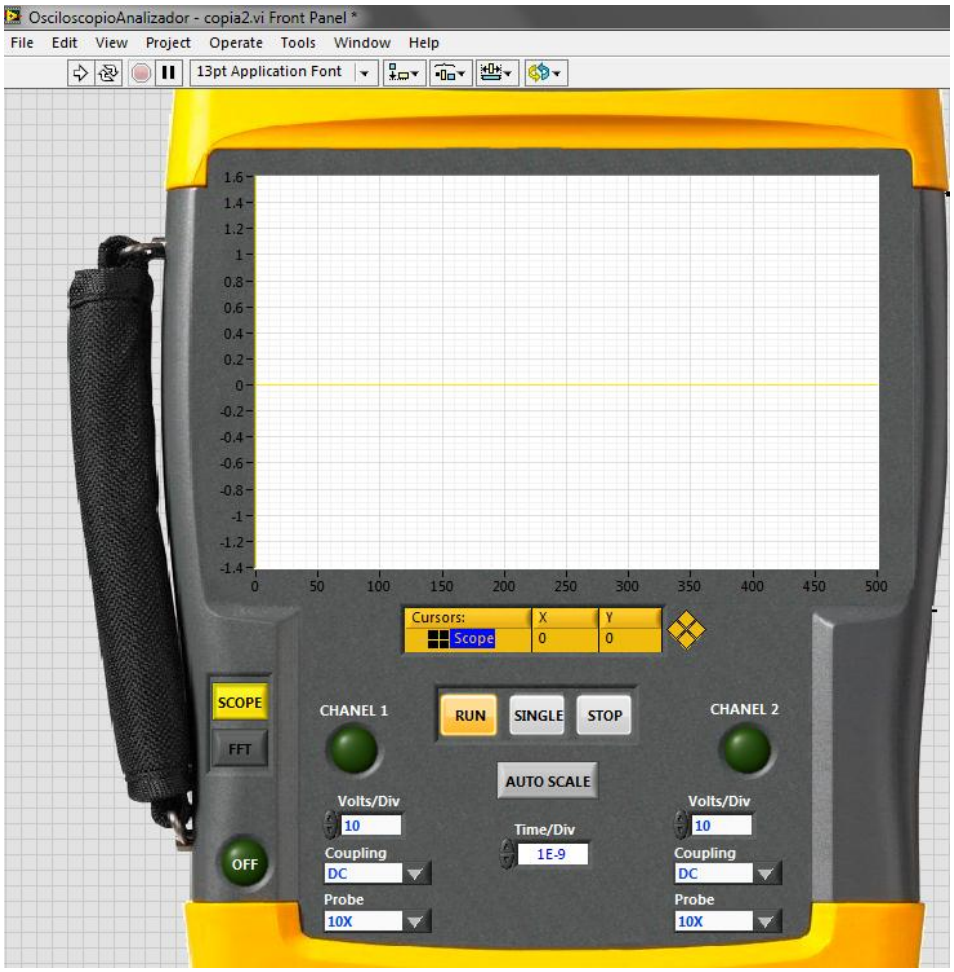

Figura 5-1. Interfaz Gráfica para reproducir funciones del osciloscopio DSO1012A.

El control de dicho instrumento al igual que el generador de señales se hace bajo la plataforma de software LabView y utilizando el puerto USBTMC para la comunicación.

Se desarrolló una interfaz diferente de la del osciloscopio original por un tema de sencillez y para mostrar otras capacidades del entorno de programación Labview, además se mejoró el

cálculo de la FFT, esto debido a que la forma de onda que muestra el osciloscopio no es tan clara ni tan exacta.

El osciloscopio controlado por el código fuente de la figura 4.3, como se puede observar el diagrama está dividido en tres partes, la primera corresponde a la etapa de comunicación, la segunda es de inicialización de variables, y la tercera parte es la etapa de procesamiento de eventos producidos por el panel del osciloscopio virtual, la parte estética de la aplicación es un poco más dinámica pero quedará en segundo plano pues solo es la imagen de fondo con un pequeño retoque pictográfico con un software de edición de imágenes y algo de movimiento gracias a los nodos de propiedad de ubicación de un TabControl de labview, nos centraremos en las funciones que se pueden hacer a través de elementos de LabView y los comandos SCPI propios del osciloscopio.

5.2 Funciones principales del diagrama de bloques del osciloscopio.

Los bloques más importantes están en la tercera parte del código, allí se encuentran dos lazos uno de ellos While Loop que controla todos los eventos producidos por la interacción con la interfaz gráfica y el segundo es un Timed Loop que tiene la tarea de adquirir y procesar los datos que el osciloscopio tiene guardados en memoria, la figura 4.2 muestra con más detalle el lazo While Loop:

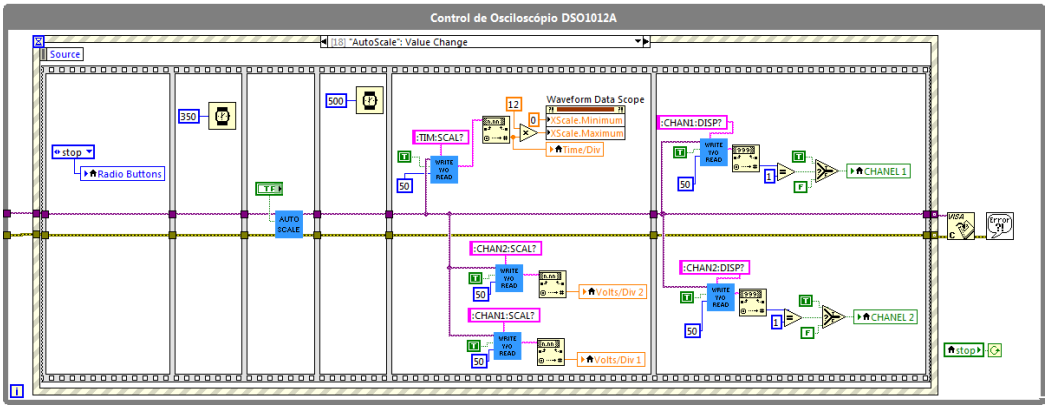

Figura 5-2. Lazo While Loop con un Case Event en el medio.

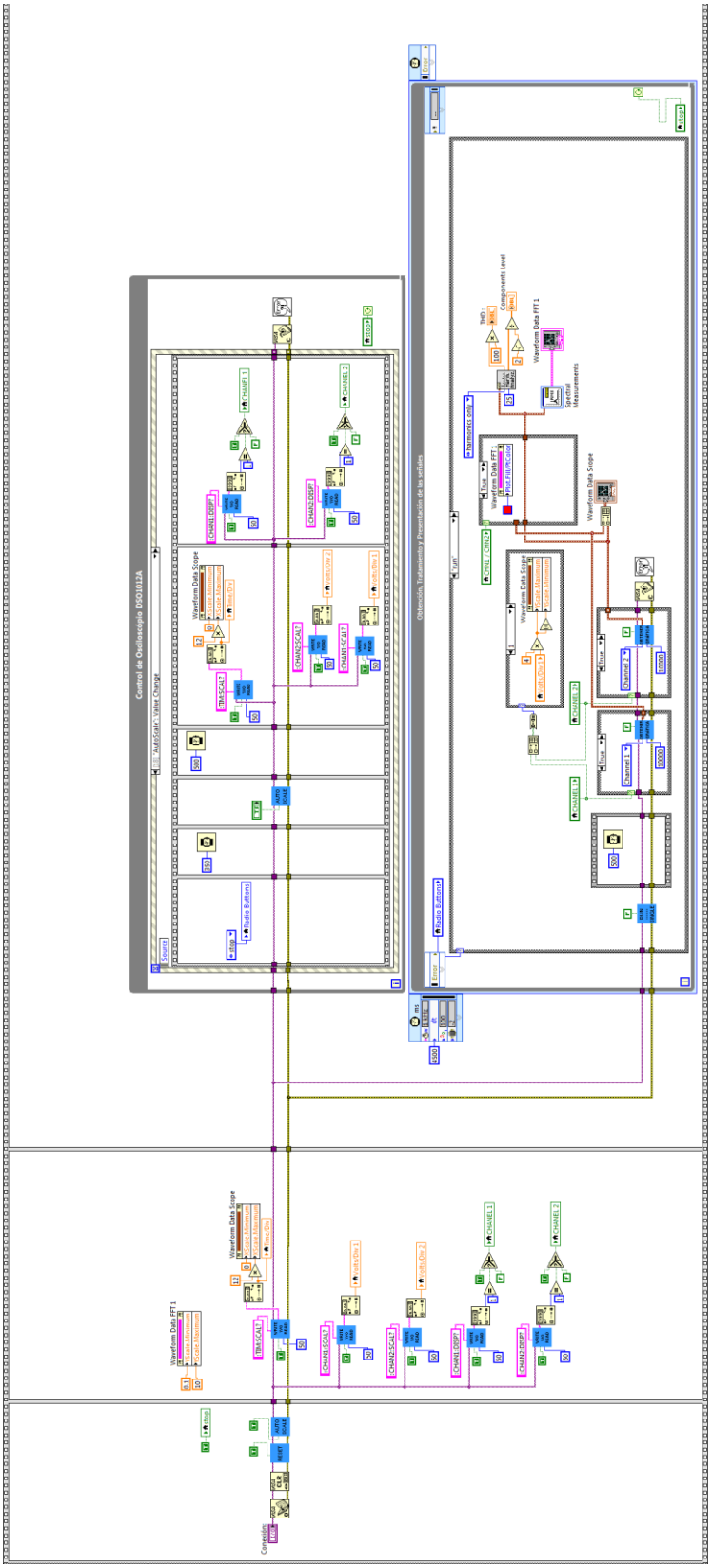

Figura 5-3. Diagrama de Bloques del osciloscopio.

El Case Event ayuda en gran manera a detectar un evento disparado por algún elemento particular y de esa forma se está seguro que solo las funciones relacionadas con ese evento se ejecuten.

La figura 4.4 muestra el lazo Timed Loop que encierra las rutinas de adquisición, procesamiento y presentación de datos:

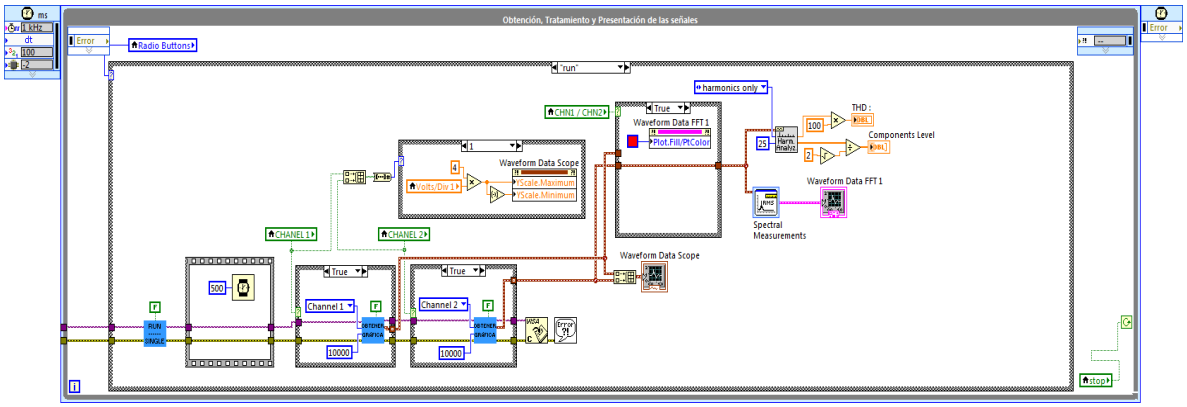

Figura 5-4. Timed Loop con una estructura de casos en el medio.

En esta etapa se grafican los datos y se calcula la FFT con un VI especializado de LabView, además contiene al SubVI más importante, que es el de obtener gráficas, gracias a eso se puede acceder a la memoria del osciloscopio y poder extraer los datos allí guardados, este VI extrae los datos en forma binaria y los convierte en datos que son graficables.

5.3 SubVI's encargados de implementar las funciones básicas del osciloscopio.

A continuación se presentan los SubVI`s que realizan las funciones básicas en el osciloscopio:

SubVI encargado de enviar el comando de autoescala para los canales del osciloscopio:

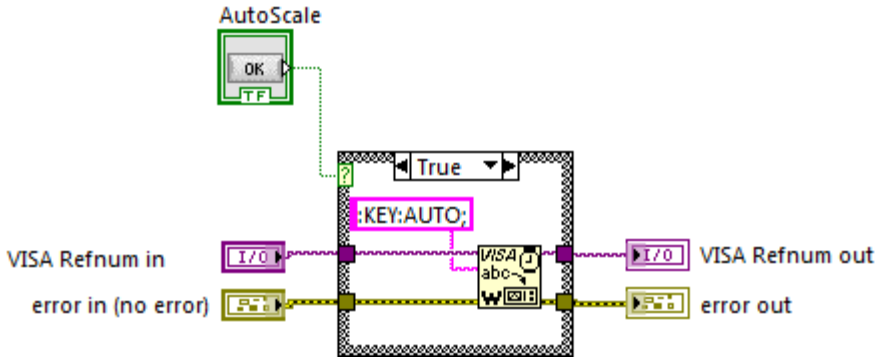

Figura 5-5. El comando Auto implementado con LabView.

Este SubVI se encarga de establecer el tiempo por división en el osciloscopio, utiliza el comando SCPI "TIM:SCAL #", donde el número es un valor entero que comience con los números 1,2 y 5, y lo hace en décadas, es decir, toma valores como: "20ns, 50ns, 100ns, 200ns, 500ns, 1us, 2us, … 1s, 2s, 5s,"

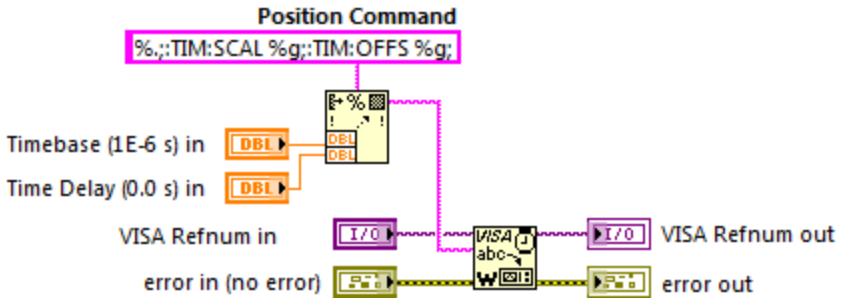

Figura 5-6. Diagrama de bloques que implementa el Subvi CONFIG TIEMPO DIVISION.

CONFIG CANAL Este SubVI configura el canal deseado con los valores de volts/div, acoplamiento y atenuación del canal deseado, los valores de volts/div son enteros que al igual que en el SubVI de tiempo pro división toma valores que comienzan por "1, 2 y 5" para este caso particular los valores que puede tomar están entre este rango "2mv, 5mv, 10mv, … 1, 2, 5, 10".

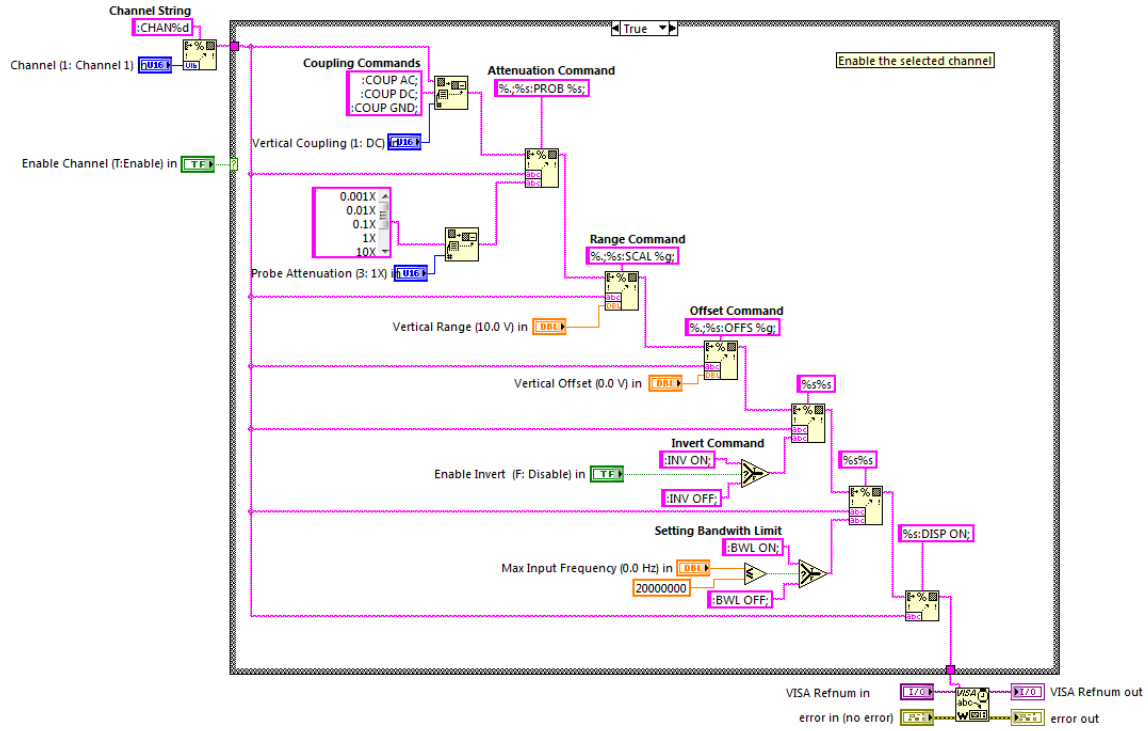

Figura 5-7. Diagrama de bloques que implementa el Subvi CONFIG CANAL

MESUR MENTS SubVI encargado de obtener las mediciones del osciloscopio, por ejemplo, Vpp, Vrms, Vmax, Vmin, Tfall, Trise, etc. Son 22 mediciones en total las que puede realizar el osciloscopio y se pueden obtener enviando el comando específico de la medición que se requiere, por ejemplo para obtener el valor rms se envía el comando "MEAS: VRMS ?" el signo de interrogación significa que se está consultando un valor calculado por el hardware del osciloscopio, para mayor detalle de cada comando de consulta se debe revisar el *"Agilent 1000 Series Oscilloscopes Programmer's Guide"* que explica con más profundidad los comandos que se envían al osciloscopio. La figura 4.8 muestra la conexión estándar para los primeros 18 comandos debido a que solo se necesita un canal para poder obtenerlos, la figura

4.9 muestra la conexión para los restantes 4 comandos que necesita tener los dos canales habilitados.

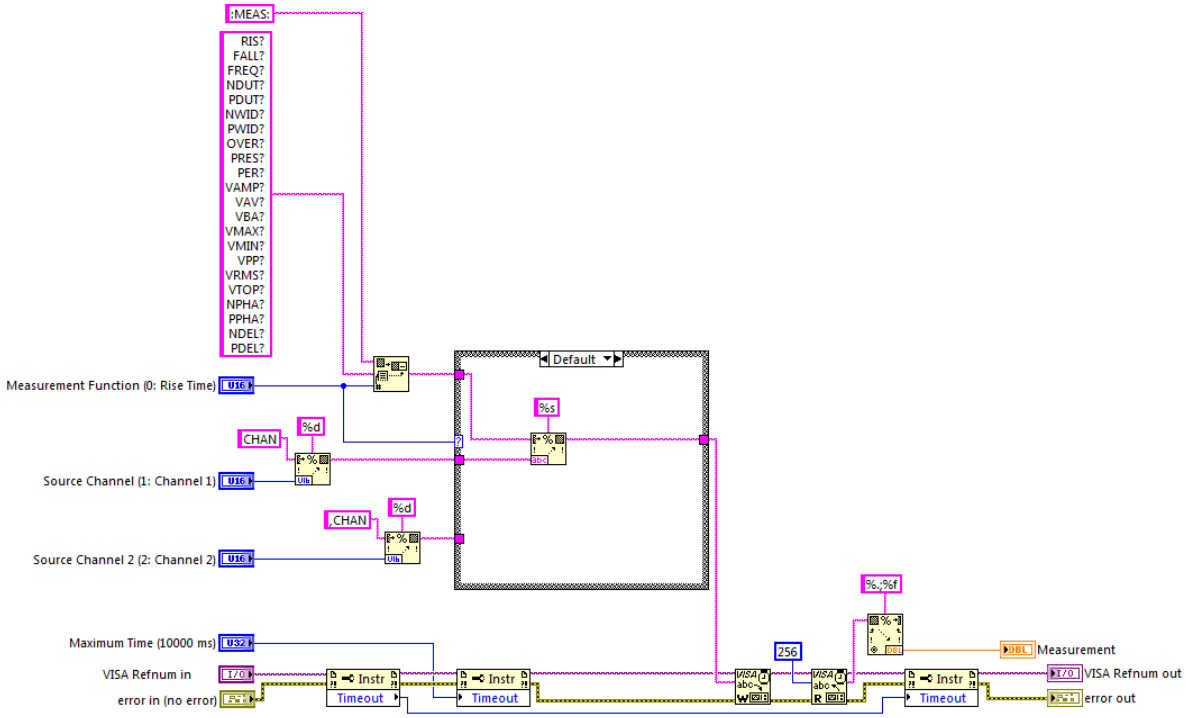

Figura 5-8. Diagrama de bloques para el Subvi MESURMENTS para los 18 primeros comandos.

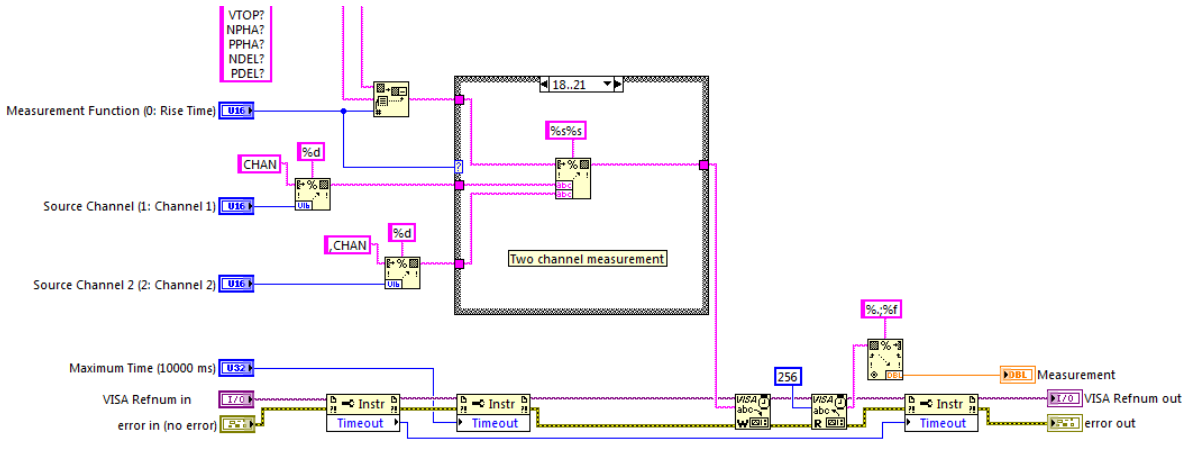

Figura 5-9. Diagrama de bloques para el Subvi MESURMENTS para los 4 restantes comandos.

Es el SubVI más importante de t odo el software porque es el encargado de obtener los datos desde la memoria interna del osciloscopio.

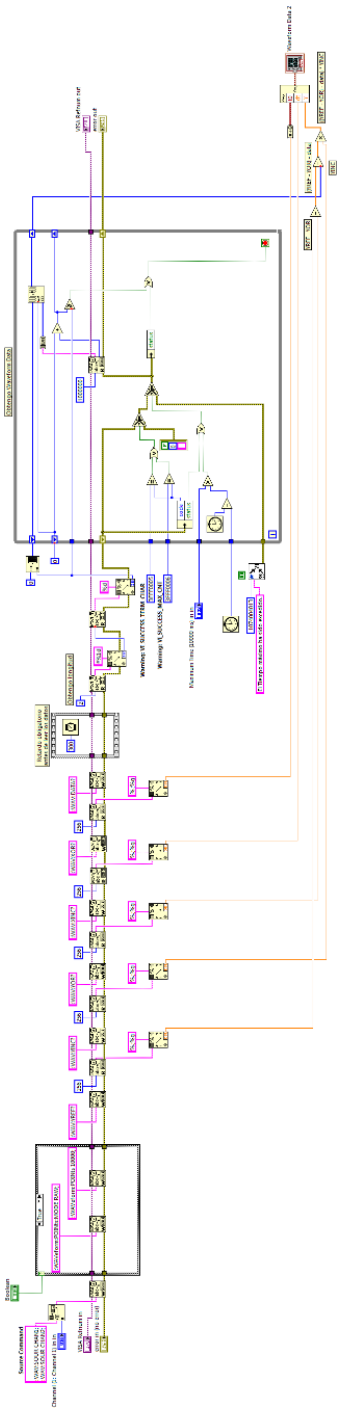

Figura 5-10. Diagrama de bloques para el Subvi OBTENER GRAFICA.

SubVI encargado de establecer el modo de adquisición de datos del osciloscopio, ya sea en modo continuo “RUN” o una sola medición “SINGLE”.

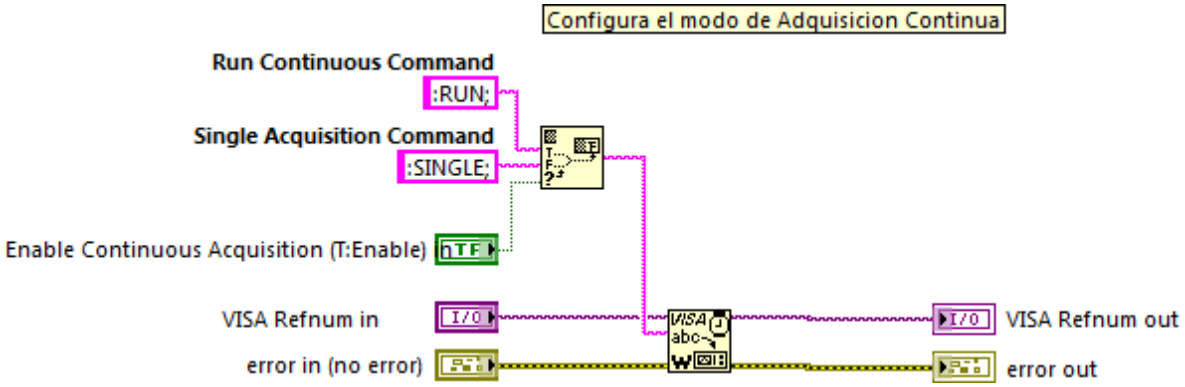

Figura 5-11. Diagrama de bloques para el Subvi RUN/SINGLE.

SubVI que sirve para escribir cualquier comando o leer cualquier respuesta en el osciloscopio, este subvi simplemente escribe o lee sobre el bus a través de las funciones VISA “Write” o “Read”, el selector datos sirve para elegir la función a realizar, “true” selecciona la operación de lectura y un “false” selecciona la operación de escritura.

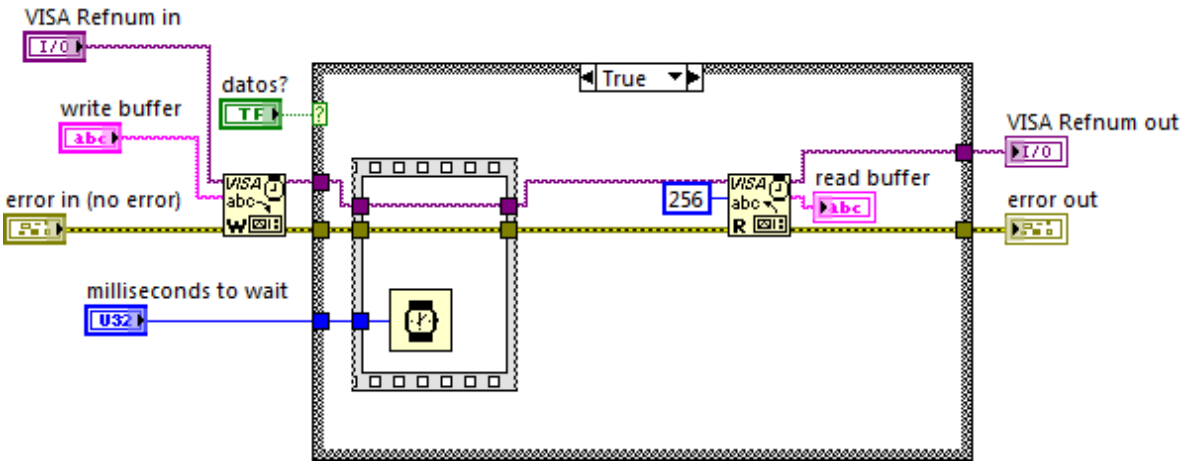

Figura 5-12. Diagrama de bloques del Subvi WRITE Y/O READ.

El cálculo de la FFT se realiza con la porción de código que muestra la figura 4-12, este código utiliza dos VI propios de LabView para generar las componentes espectrales a graficar y tabular . La figura 4-13, muestra el resultado de la FFT aplicado a un canal que tiene una onda cuadrada a su entrada.

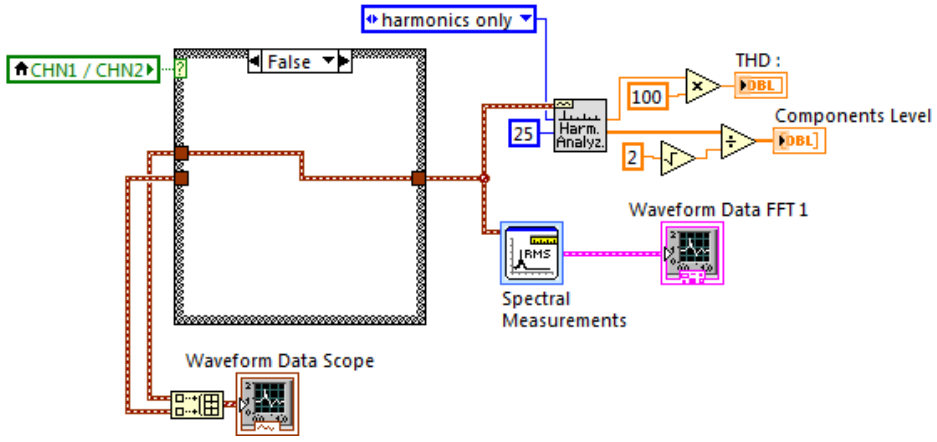

Figura 5-13. Diagrama de bloques que calcula la FFT a partir de la forma de onda obtenida.

Figura 5-14. Resultado de aplicar la FFT a una forma de onda cuadrada de entrada.

CONCLUSIONES

- Es imperante conocer el lenguaje de programación que servirá como plataforma de desarrollo para poder hacer uso eficiente de los comandos SCPI, además de eso se debe dominar la sintaxis de los comandos SCPI.

- Es necesario respetar la secuencia de ejecución de los comandos SCPI en un instrumento que no utiliza la concurrencia a la hora de servir o ejecutar funciones del mismo.

- No todas las funciones de un instrumento se pueden ejecutar mediante comandos SCPI, esto es debido a que el fabricante hace funciones específicas para necesidades específicas, por lo tanto no debe ser extraño no encontrar un comando en particular para dicha función.

- El protocolo GPIB seguirá siendo un referente en el mercado del control de instrumentos electrónicos a pesar de ser una tecnología de más de 50 años, esto debido a su robustez eléctrica y mecánica, facilidad de uso, amplia bibliografía, un gran abanico de lenguajes de programación con el cual tiene compatibilidad.

- La especificación USBTMC si bien no sustituirá al GPIB se está abriendo camino debido a sus ventajas como mayor velocidad y cable más económico, sin embargo el GPIB sigue siendo el favorito para aplicaciones en ambientes rústicos y ruidosos debido a su construcción.

REFERENCIAS BIBLIOGRÁFICAS

- [1] ANSI/IEEE Std 488-1978 1983 Standard Digital Interface for Programmable Instrumentation. ANSI/IEEE Std 488-1978. IEEE Std 728-1982. The Institute of Electrical and Electronics Engineers.
- [2] Diseño e implementación de un procedimiento de Medición de calibración de un multímetro patrón de 8.5 dígitos utilizando la comunicación por el BUS GPIB y el Estándar IEEE-488.2, Tesis de Ingeniería, UES, 2013.
- [3] González, A.: Estudio del protocolo IEEE 488 mediante el desarrollo de una herramienta de simulación, Tesis de Licenciatura, UTM, abril 2003.
- [4] Universal Serial Bus Test and Measurement Class Specification (USBTMC), Revision 1.0, April 14, 2003.
- [5] Agilent 33210A User Guide.
- [6] Agilent 1000 Series Oscilloscopes Programmer's Guide.
- [7] LabView User Manual, 2013.
- [8] Universal Serial Bus Specification, Revision 2.0, April 27, 2000, http://www.usb.org
- [9] ANSI X3.4-1986, American National Standard Code for Information Interchange Coded Character Set –7-bit, http://www.ansi.org
- [10] USB Test and Measurement Class USB488 subclass specification, Revision 1.0, http://www.usb.org.
- [11] Standard Commands for Programmable Instruments (SCPI), Reference Manual.
- [URL1] VISA Specification, http://www.vxipnp.org
- [URL2] http://www.agilent.com "Página electrónica de la empresa Agilent Technologies Incorporated"
- [URL3] http://www.ni.com "History of GPIB", National Instruments.
- [URL4] http://www.ivifoundation.org "Página del consorcio SCPI", SCPI Consortium.

ANEXO A

A.1-. Procedimientos Adicionales para el Multímetro Patrón FLUKE 8508A.

A continuación se muestran los procedimientos adicionales para completar el código de control para el multímetro patrón 8508A.

En esta sección solo se presenta el diagrama de bloques de cada unidad funcional, para tener una idea más amplia del multímetro patrón Fluke 8508A se debe consultar el Trabajo de Graduación "*Diseño e implementación de un procedimiento de medición de calibración de un multímetro patrón de 8.5 dígitos utilizando la comunicación por el bus GPIB y el estándar IEEE-488.2*", en ese trabajo se puede consultar sobre el funcionamiento del código en labview para controlar el multímetro, también se recomienda leer "*8508A Users Manual*" para comprender de mejor forma los comandos SCPI involucrados en los procedimientos de medición.

La figura A-1 muestra el diagrama de bloques para la medición de Corriente AC:

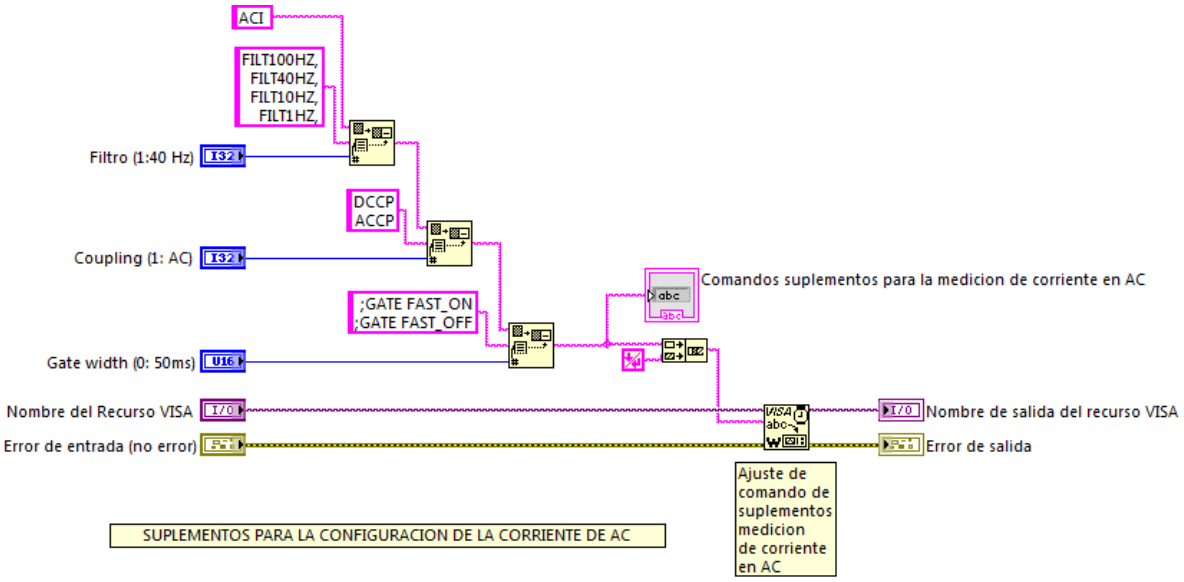

Figura A-1. Diagrama de bloque para la medición de Corriente Alterna.

La figura A-2 muestra el diagrama de bloques para la medición de Corriente AC:

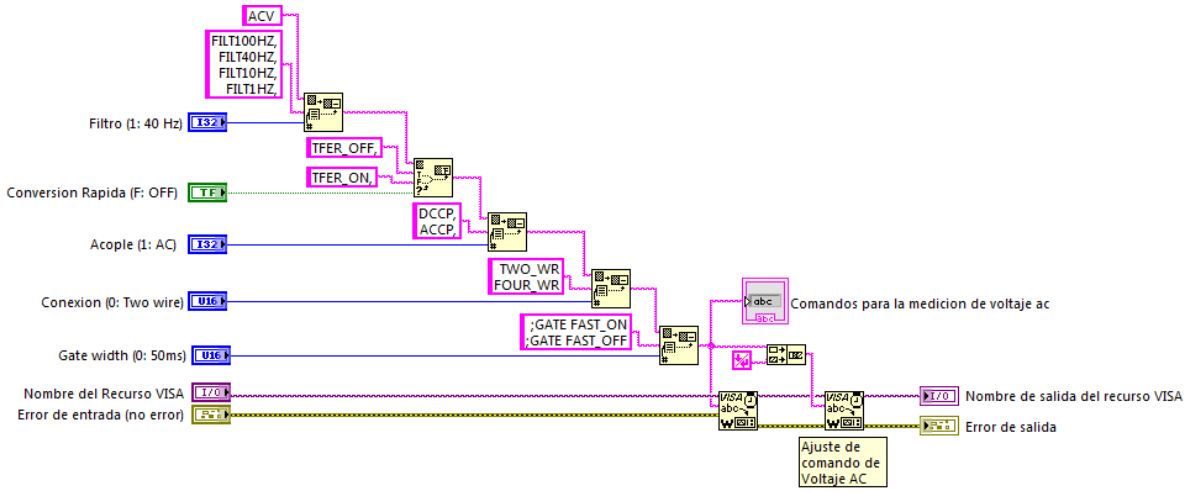

Figura A-2. Diagrama de bloques para el subvi de medición de Voltage Alterno.

La figura A-3 muestra el diagrame de bloques para controlar la medición de temperatura con el multímetro de referencia, cabe mencionar que este multímetro funciona con PTR de gran precisión, por tanto, es necesario tener una PTR para poder obtener datos correctos, si se utiliza en su lugar una RTD los datos que arrogaría la medición serían erróneos.

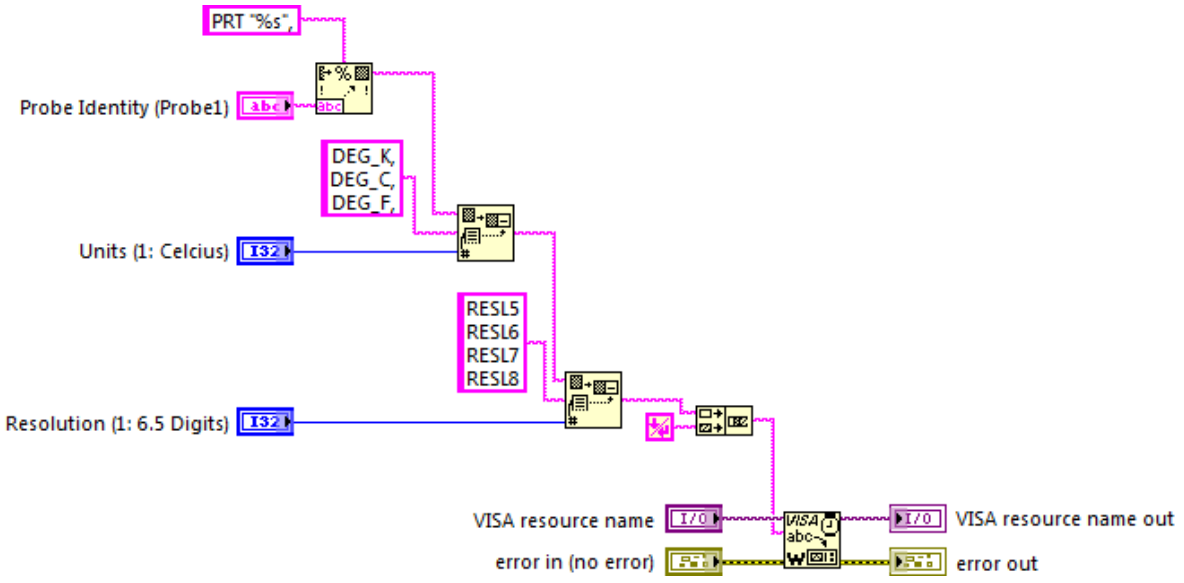

Figura A-3. Diagrama de bloques para el subvi de medición de Temperatura.