CENTRO DE INGENIERÍA Y DESARROLLO INDUSTRIAL

Centro de Ingeniería y Desarrollo Industrial.

Proyecto industrial terminal:

Automatización de la prueba de emisiones conducidas del laboratorio EMC MABE

Para obtener la especialidad en:

"TECNÓLOGO EN MECATRÓNICA"

Presenta:

Alumno: Ing. Diego Fernando Torres Rubio.

Asesor externo: Ing. Angélica Chávez Rubio.

Recibí copia y acepte
24/02/2009

Querétaro de Arteaga, Querétaro, 24 de Febrero de 2009.

00 1527

CENTRO DE INGENIERÍA Y DESARROLLO INDUSTRIAL

Centro de Ingeniería y Desarrollo Industrial.

Proyecto industrial terminal:

Automatización de la prueba de emisiones conducidas del laboratorio EMC MABE

Para obtener la especialidad en:

"TECNÓLOGO EN MECATRÓNICA"

Presenta:

Alumno: Ing. Diego Fernando Torres Rubio.

Asesor externo: Ing. Angélica Chávez Rubio.

Recibí copia y acepte
22/08/2008

Querétaro de Arteaga, Querétaro, 22 de Agosto de 2008.

TÍTULO:

Automatización de la prueba de emisiones conducidas del laboratorio EMC MABE.

RESUMEN:

El presente trabajo tiene como propósito presentar, la forma de desempeñar una programación del protocolo GPIB, con el objetivo de obtener un producto final, que pueda ser utilizado para este protocolo. Dentro de lo cual exige una extensa investigación acerca del mismo, para la comprensión del campo de trabajo que se desea explotar.

Aquí pues será expuesta de una forma ordenada y concisa la metodología de trabajo utilizada, los recursos de los que se hizo uso y de las diferentes fuentes consultadas para un desarrollo exitoso, que si bien presenta un producto, se puede considerar en una segunda instancia para un segundo re-trabajo.

Así pues la organización del presente documento muestra un índice temático, un índice de ilustraciones y otro de tablas generadas. En seguida se despliega la justificación que dio lugar al presente desarrollo, también la fundamentación teórica en la que se basa todo el desarrollo, a continuación se contempla el desarrollo del documento, guiándonos hasta las conclusiones para cerrar con la bibliografía utilizada.

ÍNDICE TEMÁTICO:

ÍNDICE DE FIGURAS:

ÍNDICE DE TABLAS:

ÍNDICE DE ECUACIONES:

JUSTIFICACIÓN:

El laboratorio EMC MABE, es de los pocos de su tipo en el país, dentro del cual se realizan diferentes pruebas a equipos y accesorios de uso doméstico, como lo son lavadoras, secadoras, refrigeradores, entre otros. El propósito de estas evaluaciones, se encuentra dirigido a la protección del usuario, en primer término, la durabilidad y confiabilidad del dispositivo como segundo objetivo.

La ejecución de las distintas comprobaciones se realiza a través de equipos de última generación, y por ende con la mejor tecnología. Dicho lo anterior, se observó directamente el factor económico de cada uno de los equipos y métodos utilizados, donde, regularmente, el impacto de mayor costo, se lo caliza en el software especializado para la interacción de los dispositivos de prueba y adquisición de mediciones, a demás, no sólo es el precio del software el que impacta directamente a la empresa, también, el recurso humano, en este caso de los proveedores de la marca, para la instalación, puesta en marcha y mantenimiento de los dispositivos. Es en ésta observación, que el personal encargado del área, decidió evaluar la siguiente alternativa: realizar la integración de los equipos necesarios, así como del software que se utilizará, con mano de obra barata, que cuente con conocimientos en el área de la adquisición de señales, con el uso de tarjetas PCMIA-GPIB, bajo ambiente LabVIEW.

Actualmente, debido a la expansión del laboratorio, se consideró que era de vital importancia, separar el controlador de los equipos, es decir, realizar a parte la prueba de emisiones conducidas, donde los dispositivos que trabajan dentro de ese proceso son el ESPI-3 y LISN, ambos de la marca "Rohde&Schwarz", donde el primero de ellos es un analizador de frecuencias y el segundo es un acoplador de señales, que además realiza captura de emisiones de energía producidos por electrodomésticos. El objetivo de esta prueba es registrar todas las alteraciones, medidas en dB/μV, producidas por electrodomésticos, y que son transmitidas por las líneas de alimentación eléctrica hacia el exterior. Sin embargo, el fabricante solo ofrece una licencia por cada software que vende. Es en éste punto, donde la fabricación de un software, de bajo costo, que realice las mismas actividades que el del fabricante, se vuelve necesaria.

El desarrollo del software beneficiará directamente al departamento de EMC MABE, así como al personal encargado de la fabricación del programa. El primero de los sujetos obtendrá el producto genérico a muy bajo costo, con la precisión y velocidad del de fábrica, el otro sujeto adquirirá nuevos conocimientos, además de un proyecto que le ofrecerá una calificación final, para la obtención de su título de "TECNÓLOGO EN MECATRÓNICA".

OBJETIVO:

Desarrollo de un software para el control, adquisición de datos y generación de reportes de la prueba de emisiones conducidas del laboratorio EMC MABE.

OBJETIVOS ESPECÍFICOS:

- ❖ Observación, reconocimiento de los equipos de trabajo, y su documentación.
- ❖ Indagación del protocolo de comunicación GPIB, nociones básicas acerca de éste.
- ❖ Acopio de información acerca del protocolo GPIB para comunicación con dispositivos en el caso de ser necesario, conocimiento profundo del tema.
- ❖ Verificación del funcionamiento e interacción de trabajo de los equipos y su controlador.
- ❖ Lectura y análisis de la documentación de la tarjeta PCMIA-GPIB.
- ❖ Instalación y adquisición de licencia del software LabVIEW.
- ❖ Desarrollo y construcción del software y ambiente de trabajo para el proceso de emisiones conducidas.
- ➢ Utilización de la herramienta NI-SPY para la captura de comandos durante un proceso de trabajo normal de los dispositivos.
- ➢ Análisis de los comandos y rutinas de trabajo capturados.
- ➢ Investigación de la naturaleza de los datos adquiridos durante un proceso de trabajo. Conversión de éstos en caso de ser requeridos para su interpretación.
- ➢ Puesta en marcha de prototipo de software para los dispositivos en existencia.
- ➢ Adquisición y almacenamiento en una base de datos de la información adquirida para la generación de reportes y construcción de gráficas.
- ➢ Generación de trabajo final para la especialidad durante el transcurso del proyecto.

FUNDAMENTACIÓN TEÓRICA

1. INTRODUCCIÓN AL BUS DE COMUNICACIONES GPIB

GPIB, "General Purpose Interface Bus", también conocido como IEEE 488, en un método de comunicación entre instrumentos electrónicos, como lo pueden ser multímetros y osciloscopios. [1]

GPIB fue diseñado originalmente por Hewlett-Packard, quien le dio el nombre de HP-IB, para la conexión y control de instrumentos programables de la marca HP. Debido a su alta transferencia de datos, mayor a 1MBps, el GPIB ganó rápidamente popularidad para otras aplicaciones como redes de comunicación y control de periféricos. [1]

Ya en 1975 el IEEE, basándose en HP-IB, propone la norma IEEE 488/1975 y se renombra como GP-IB, en 1987 debido a carencias de la primera norma aparece la IEEE 488.2, la que sería analizada un año después para dar paso a un nuevo GPIB de alta velocidad (HS488). [2]

1. 1. ESTÁNDAR IEEE 488.1

El estándar IEEE 488.1 define los requisitos mecánicos, eléctricos y funcionales que deben cumplir los dispositivos para comunicarse.

Algunas de sus características son:

- Un máximo de 363 dispositivos conectados, pero solo 15 en funcionamiento simultáneo.
- Interconexión en línea, en estrella o mixta. Entre dos instrumentos no puede haber más de 2 metros de longitud del cable. La longitud total de todo el bus del sistema, no puede exceder 20 metros, en caso contrario usar técnicas de extensión.
- Transferencia asíncrona para adaptar velocidades. Velocidad máxima de 1 MBps, en nuevas versiones hasta 8 MBps.
- Compatibilidad TTL y lógica inversa.
- Capacidad de direccionamiento de hasta 31 direcciones, 0 – 30.
- Para gestionar la información hay tres tipos de elementos: receptor (listener), emisor (talker) y controlador (controller).
- El receptor sólo puede recibir datos cuando se le direcciona. Se permiten hasta 14 receptores activos simultáneamente.
- El emisor sólo transmite cuando es direccionado, y no se permite más de un emisor activo al mismo tiempo en el bus.
- No puede haber más de un controlador al mismo tiempo en el bus, pero se le puede cambiar a lo largo del tiempo. Este dispositivo es el encargado de asignar las funcionalidades al resto de instrumentos del bus. [2]

1. 1. 1. Cableado

Los cables son apantallados de 24 hilos acabados en ambos extremos con un conector doble macho-hembra de tipo Microribbon 57. En la figura 1, se aprecia que el conector tiene varios cables de tierra, porque cada uno, exceptuando el 24, forma un par trenzado con el cable que tiene en frente, ayudando a disminuir el ruido. Las líneas de datos (DIO1-DIO8) tienen un retorno común. Todas están apantalladas.

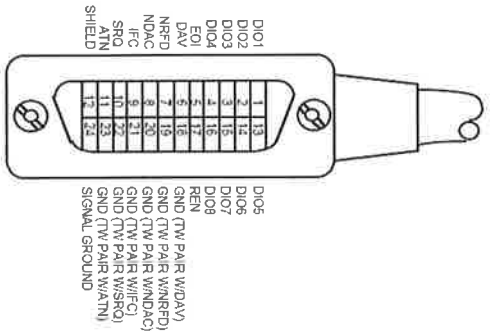


Figura 1, Líneas de conexión para GPIB. [1]

Las 16 líneas forman tres grupos de trabajo:

- ♣ Control de transferencia (handshake):
  - *DAV (Data valid):* Indica que el dato presente en *DIOX* es válido.
  - *NRFD (Not ready for data):* Indica la disposición de aceptar un dato.
  - *NDAC (Not data accepted):* Indica la aceptación de un dato.
- ♣ Control del bus:
  - *ATN (Attention):* El controlador muestra si envía comandos o datos.
  - *IFC (Interface clear):* Permite al controlador inicializar el bus.
  - *SRQ (Service request):* Ayuda a que algún dispositivo pida una solicitud de atención al controlador. El controlador se entera qué dispositivo produjo la solicitud al ejecutar un sondeo (*polling*) que puede ser serie o paralelo.
  - *REN (Remote enable):* El controlador habilita o deshabilita controles locales que tienen correspondencia remota.
  - *EOI (End or identify):* Permite que un emisor marque el final de un mensaje de varios bytes o junto con ATN para indicar que el controlador está ejecutando un sondeo.
- ♣ Datos bidireccionales:
  - *DIO1-DIO8:* son líneas bidireccionales que transmiten los datos codificados normalmente en ASCII de 7 bits. DIO1 es el dato menos significativo, DIO7 el más significativo y DIO8 el bit de paridad. [2]

1. 1. 2. Handshake

El protocolo de comunicación se asegura de que todos los receptores estén listos para recibir, mucho antes de que los emisores comiencen a transmitir con el propósito de coordinar la transferencia sobre el bus de datos, además ha de asegurar la integridad de los mismos.

La comunicación es asíncrona y se utilizan tres líneas del bus de control de transferencia en lugar de un reloj, que son la DAV, NRFD, NDAC. Éste procedimiento se usa para ajustar la velocidad de la transmisión al dispositivo más lento.

En la figura 2, se muestra un ejemplo del uso del bus de control de la transferencia, en la cual se marcan los instantes importantes.

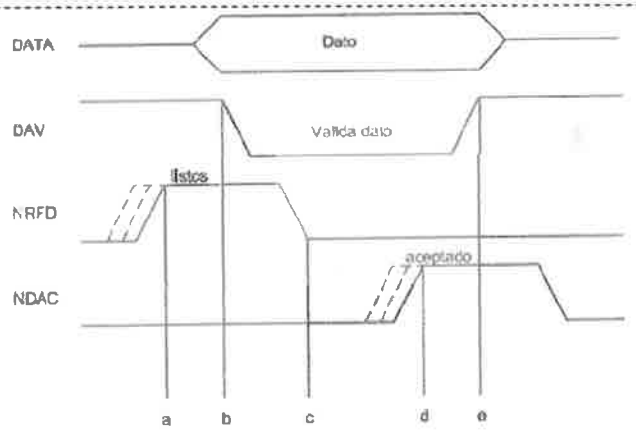


Figura 2, instantes clave del bus de control de la transferencia. [2]

a) Cuando un dispositivo esté diseccionado como receptor, éste intentará cambiar al estado de escucha. Si un dispositivo, que está conectado al bus no puede recibir el dato, la señal NRFD cambiará a "*TRUE*", lo que provoca que la tensión en la línea sea 0 V, sin importar el estado de los otros dispositivos. Cuando todos los dispositivos estén listos para recibir información entonces NRFD será "*FALSE*", por lo que el nivel de tensión en la línea puede ser alto.

b) Cuando el nivel de NRFD sube, el emisor se da por enterado de que todos los dispositivos están listos para la recepción, por lo que validará con la señal DAV los datos que previamente colocó en el bus, esto es, "*Data valid=True*", línea con nivel bajo.

c) El primer receptor leyó el dato y aplica un cambio en el estado de la línea NRFD, e intenta elevar la tensión de la NDAC, pero deberá esperar a que los demás receptores lean el dato.

d) Los receptores terminaron de leer el dato, entonces la línea NDAC subirá su nivel para confirmar la lectura.

e) El emisor quita el dato del bus y lo invalida mediante la línea DAV. Todas las líneas regresan al estado inicial, permitiendo repetir todo el proceso par un nuevo dato. [2]

1. 2. ESTÁNDAR IEEE 488.2

El IEEE 488.2 "*Codes, Formats Protocols and Common Commands For use with Standard 488.1-1987*" surgió de la necesidad de unir los criterios de distintos fabricantes. Establece un estándar en el formato de los mensajes, un conjunto común de comandos y una forma única para poder comprobar el estado de los instrumentos.

El formato de datos y sintaxis utilizados, IEEE 488.2 define el formado de números decimales, en punto flotante y cadenas, pudiendo ser una codificación ASCII de 7 bits o binaria de 8 bits.

El 488.2 tiene tres registros de estado, donde cada uno puede ser enmascarado con su propio registro de habilitación. En la figura 3 se muestran dichos registros.

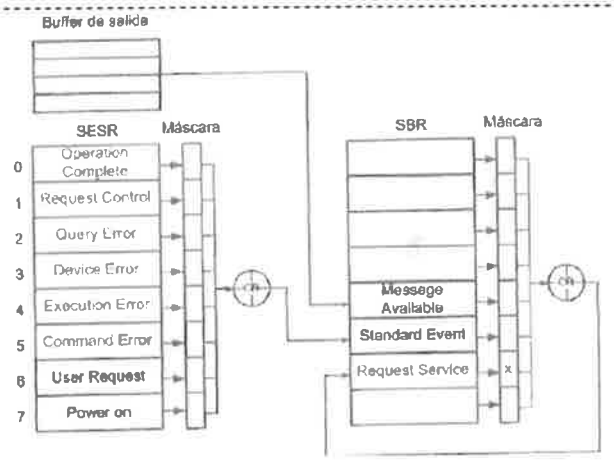


Figura 3, registros de estado.

La función de los tres registros es la siguiente:

- *Stand event status register (SESR):* Informa el estado de los eventos.
- *Questionable data:* Indica sucesos sobretensiones, sobrecorrientes, entre otros.
- *Status byte register (SBR):* Notifica si el equipo quiere enviar algún mensaje, pidió atención al controlador, entre otros.

1. 2. 1. Comandos de la norma 488.2

Los comandos que expone la norma 488.2 son comunes a todos los dispositivos, y se agrupan según su funcionalidad. La tabla 1, enuncia algunos de esos comandos. [2]

Comandos para datos del sistema: Solicitan información sobre el sistema del dispositivo.	
*IDN?	Identifica el fabricante, modelo, número de serie y firmware.
Operaciones internas: Está formado pro comandos relacionados con capacidades internas que puede tener el equipo, como reseteo, calibración o verificación.	
*CAL	Calibra el equipo.
*RST	Reseteo del dispositivo.
*TST	Autoverificación.
Estado y eventos: Controlan las estructuras de estado del equipo.	
*CLS	Borra el registro de estado del dispositivo.
*ESE	Fija un registro interno determinado (*Standard event status enable*).
*ESE?	Lee el valor del registro anterior.
*ESR?	Lee el valor del registro de incidencias (*Standard event status*).
*SER?	Lee el valor del registro SRER (*Service request enable register*).
*STB?	Lee el valor del registro (*Status byte*).
*SER	Fija un valor en un registro interno (*Service request enable*).
Sincronización.	
*OPC	Activa el bit de operación completa (bit 0 del registro *Status byte*).
*OPC?	Comprueba el estado del bit anterior.
*WAI	Obliga a completar todos los comandos anteriores antes de continuar procesando los comandos siguientes al *WAI.
Sondeo paralelo.	
*IST?	Petición del bit individual de estado.
Trigger. Definen como un dispositivo tiene que responder ante un mensaje GET.	
*DDT	Graba una secuencia de comandos que el equipo ejecutará cuando reciba el

	mensaje *GET* o el comando *TRG.
*TRG	Permite activar o disparar varios instrumentos simultáneamente.
Controlador.	
*PCB	Se le indica al controlador la dirección a la que deba enviar un mensaje TCT, que pasa la función de controlador a otro dispositivo.
Autoconfiguración.	
*DLF	Deshabilita la función de receptor.
Macros, estos comandos permiten al usuario definir nuevos comandos basándose en los ya disponibles.	
*DMC	Define una macro, es decir, una secuencia de comandos.
*EMC	Habilita o deshabilita las macros.
Almacenamiento de la configuración: guardan y restauran la configuración del equipo.	
*RCL	Restaura el estado del dispositivo desde una copia guardada previamente.
*SAV	Guarda el estado actual del dispositivo en memoria.

Tabla 1, comandos de la norma IEEE 488.2 [2]

1. 3. SCPI

En 1990 Hewlett-Packard formalizó la propuesta de estandarización de comandos para equipos programables, de ésta manera se define un lenguaje común para todos los equipos. Se trata de un lenguaje basado en comandos, consistente, *English – Like*, que es independiente del fabricante del equipo. En SCPI los comandos se agrupan en varios subsistemas o familias, que se identifican con un bloque funcional del instrumento. La figura 4 ilustra un modelo de bloques de un instrumento SCPI.

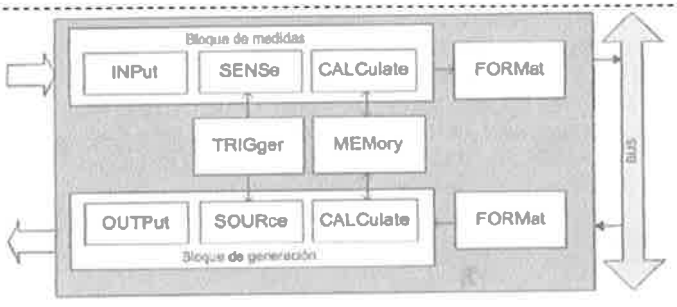


Figura 4, modelo de bloques de un instrumento SCPI.

Cada familia o subfamilia se caracteriza con una palabra clave. Las familias principales son: *CALCulate, CALibration, CONTrol, DIAGnostic, DISPlay, FORMat, HCOPy, INPut, INSTrument, MEMory, MMEMory, Output, PROGram, ROUTe, SENSe, SOURce, STATus, SYSTem, TEST, TRACe, TRIGger, UNIT* y *VXI.*

Las sintaxis de construcción de líneas de comandos deben tener las siguientes características:

- ♣ La familia raíz es la que comienza la línea de comando, seguida por las subfamilias. Entre las familias se coloca el carácter dos puntos ":" para separarlas, las letras en minúscula se pueden omitir. Los parámetros que deban tener los comandos se ponen al final separados de la última palabra por un espacio. El comando finaliza con un fin de línea.
  - *SYSTem:COMMunicate:SERial:BAUD 9600*
  - *SYST:COMM:SER:BAUD 9600*
- ♣ Para preguntar algún dato en específico a un dispositivo se utiliza el signo de interrogación "?".
  - *SYST:COMM:SER:BAUD?*
- ♣ Se pueden agrupar varios comandos de la familia en una misma línea, separando los comandos con el signo punto y coma ";". Si al ";" le sigue el símbolo ":" se indica que la escritura del siguiente comando empieza por una familia raíz.
  - *SYST:COMM:SER:BAUD 9600;:SYST:COMM:SER:BITS 8 [2]*

2. TRANSFERENCIA DE FORMAS DE ONDA

Muchos digitalizadores, como osciloscopios y analizadores, regresan formas de onda codificadas en una cadena ASCII o bien por una cadena binaria. Imaginando que se tratara de una misma forma de onda, la codificación en binario requiere menos memoria y es más rápida.

Tomando en consideración el siguiente ejemplo, imagine una forma de onda compuesta por 1024 puntos, cada uno tiene un valor entre 0 y 255. Usando una codificación ASCII solo se requerirán máximo 4 bytes para representar cada punto, esto es, 3 bytes como máximo para el valor del punto y 1 byte para el separador, como lo puede se una coma ",". Por lo que se necesitarán 4906 bytes (4*1024), más los bytes necesarios para cualquier encabezado y avance de línea, para la representación de una curva como cadena ASCII. En la figura 5 se observa un ejemplo de una curva en cadena ASCII.

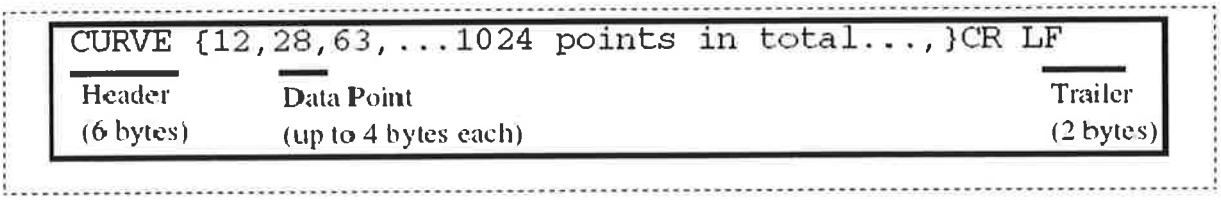


Figura 5, representación de una curva en cadena ASCII. [3]

Ahora, usando el mismo ejemplo, esa misma forma de onda solo requeriría 1024 bytes, más los bytes necesarios para cualquier encabezado y avance de línea, para la representación de una curva como cadena binaria. Si se usa la codificación binaria, solo se necesitará 1 byte para representar el punto, asumiendo que cada punto era un entero sin signo de 8 bits. La ilustración siguiente, figura 6, muestra un ejemplo de una curva en cadena binaria. [3]

Figura 6, representación de una curva en cadena binaria. [3]

2.1. CODIFICACIÓN ASCII

"*American Standard Code for Information Interchange (ASCII)*" es una codificación simbólica basada en el alfabeto inglés, regularmente utilizado en la representación de texto en computadoras, equipos de comunicación y muchos otros dispositivos que trabajan con texto. ASCII fue originalmente creado para el uso en telégrafos, y su primer uso comercial fue una impresora de códigos telegráficos, promovida por "*Bell data services*". La historia del ASCII se puede ver concretamente en la tabla 2.

6 de Octubre de 1960	Comienzan las reuniones para la revisión del código con el subcomité ASA X3.2.
1963	Se publica la primera edición del estándar.
1967	Se realiza una nueva revisión.
1986	Se efectúa la más reciente actualización del código. Se realizan ajustes en el orden de los códigos, se agregan nuevos símbolos así como la secuencia "*ESCape*".

Tabla 2, breve historia del código ASCII.

En la actualidad ASCII se compone de 128 símbolos, 33 de los cuales no son imprimibles, en su mayoría códigos obsoletos de control, 94 imprimibles exceptuando el "espacio". [4]

Las tablas 3 y 4, ilustran la relación de conversión del código ASCII a las bases hexadecimal, decimal y el mensaje que puede representar.

Hex	Dec	ASCII	Msg
00	0	NUL	
01	1	SOH	GTL
02	2	STX	
03	3	ETX	
04	4	EOT	SDC
05	5	ENQ	PPC
06	6	ACK	
07	7	BEL	
08	8	BS	GET
09	9	HT	TCT
0A	10	LF	
0B	11	VT	
0C	12	FF	
0D	13	CR	
0E	14	SO	
0F	15	SI	
10	16	DLE	
11	17	DC1	LLO
12	18	DC2	
13	19	DC3	
14	20	DC4	DCL
15	21	NAK	PPU
16	22	SYN	
17	23	ETB	
18	24	CAN	SPE
19	25	EM	SPD
1A	26	SUB	
1B	27	ESC	
1C	28	FS	
1D	29	GS	
1E	30	RS	
1F	31	US	CFE

Hex	Dec	ASCII	Msg
20	32	SP	MLA0
21	33	!	MLA1
22	34	"	MLA2
23	35	#	MLA3
24	36	$	MLA4
25	37	%	MLA5
26	38	&	MLA6
27	39	'	MLA7
28	40	(	MLA8
29	41	)	MLA9
2A	42	*	MLA10
2B	43	+	MLA11
2C	44	,	MLA12
2D	45	-	MLA13
2E	46	.	MLA14
2F	47	/	MLA15
30	48	0	MLA16
31	49	1	MLA17
32	50	2	MLA18
33	51	3	MLA19
34	52	4	MLA20
35	53	5	MLA21
36	54	6	MLA22
37	55	7	MLA23
38	56	8	MLA24
39	57	9	MLA25
3A	58	:	MLA26
3B	59	;	MLA27
3C	60	<	MLA28
3D	61	=	MLA29
3E	62	>	MLA30
3F	63	?	UNL

Tabla 3, mensajes de ASCII. [5]

40	64	@	MTA0
41	65	A	MTA1
42	66	B	MTA2
43	67	C	MTA3
44	68	D	MTA4
45	69	E	MTA5
46	70	F	MTA6
47	71	G	MTA7
48	72	H	MTA8
49	73	I	MTA9
4A	74	J	MTA10
4B	75	K	MTA11
4C	76	L	MTA12
4D	77	M	MTA13
4E	78	N	MTA14
4F	79	O	MTA15
50	80	P	MTA16
51	81	Q	MTA17
52	82	R	MTA18
53	83	S	MTA19
54	84	T	MTA20
55	85	U	MTA21
56	86	V	MTA22
57	87	W	MTA23
58	88	X	MTA24
59	89	Y	MTA25
5A	90	Z	MTA26
5B	91	[	MTA27
5C	92	\	MTA28
5D	93	]	MTA29
5E	94	^	MTA30
5F	95	_	UNT

60	96	`	MSA0, PPE
61	97	a	MSA1, PPE, CFG1
62	98	b	MSA2, PPE, CFG2
63	99	c	MSA3, PPE, CFG3
64	100	d	MSA4, PPE, CFG4
65	101	e	MSA5, PPE, CFG5
66	102	f	MSA6, PPE, CFG6
67	103	g	MSA7, PPE, CFG7
68	104	h	MSA8, PPE, CFG8
69	105	i	MSA9, PPE, CFG9
6A	106	j	MSA10, PPE, CFG10
6B	107	k	MSA11, PPE, CFG11
6C	108	l	MSA12, PPE, CFG12
6D	109	m	MSA13, PPE, CFG13
6E	110	n	MSA14, PPE, CFG14
6F	111	o	MSA15, PPE, CFG15
70	112	p	MSA16, PPD
71	113	q	MSA17, PPD
72	114	r	MSA18, PPD
73	115	s	MSA19, PPD
74	116	t	MSA20, PPD
75	117	u	MSA21, PPD
76	118	v	MSA22, PPD
77	119	w	MSA23, PPD
78	120	x	MSA24, PPD
79	121	y	MSA25, PPD
7A	122	z	MSA26, PPD
7B	123	{	MSA27, PPD
7C	124	|	MSA28, PPD
7D	125	}	MSA29, PPD
7E	126	~	MSA30, PPD
7F	127	DEL	

Tabla 4, mensajes ASCII. [5]

Los mensajes de múltiple línea del código tienen un significado en la lengua inglesa, en la tabla 3, se muestran los que se propusieron como una extensión de los ya existentes para la norma IEEE 488.1, con el propósito de soportar el protocolo HS488 de alta velocidad.

CFE → *Configuration enable*	PPD → *Parallel Poll Disable*
CFG → *Configure*	PPE → *Parallel Poll Enable*
DCL → *Device clear*	PPU → *Parallel Poll Unconfigure*
GET → *Group execute trigger*	SDC → *Selected Device Clear*
GTL → *Go to local*	SPD → *Serial Poll Disable*
LLO → *Local Lockout*	SPE → *Serial Poll Enable*
MLA → *My Listen Address*	TCT → *Take Control*
MSA → *My Secondary Address*	UNL → *Unlisten*
MTA → *My Talk Address*	UNT → *Untalk*
PPC → *Parallel Poll Configure*	

Tabla 5, mensajes de múltiple línea, IEEE 488.1 - HS488. [5]

3. GPIB BAJO EL AMBIENTE LABVIEW

Los diferentes comandos que se desprenden de los protocolos antes mencionados, tienen una representación directa sobre el lenguaje gráfico de LabVIEW o en su defecto pueden ser introducidos como comando, o cadenas de caracteres. A continuación se muestran los iconos que emulan esos protocolos así como su explicación, nombrados de las figuras 7 – 12:

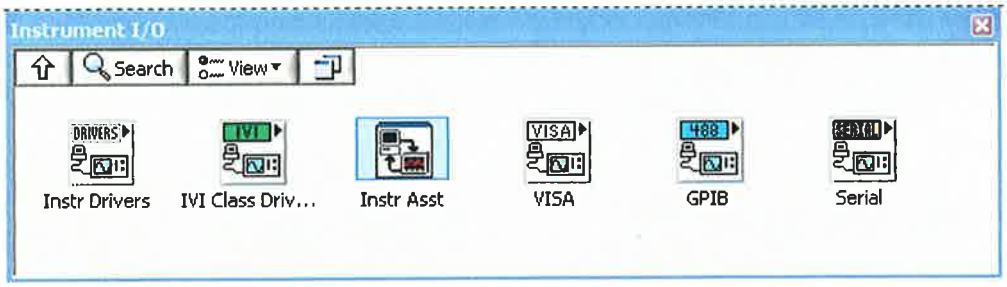


Figura 7, librería de entradas salidas en puertos. [2][6]

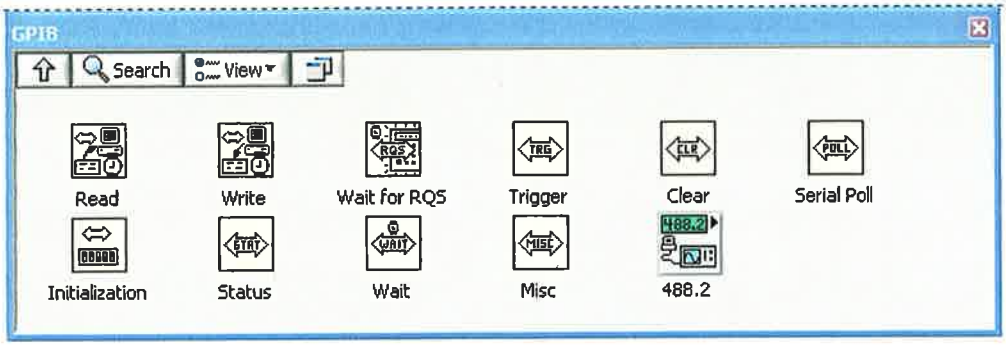


Figura 8, librería del protocolo de comunicación de uso general GPIB. [2][6]

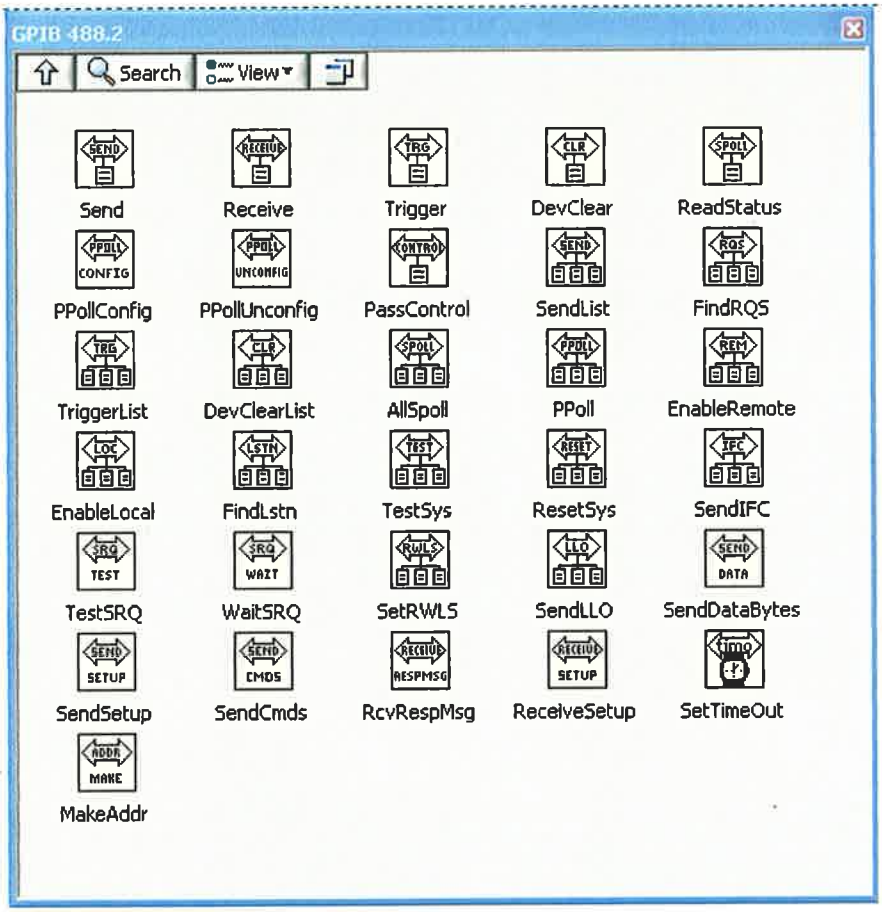


Figura 9, librería del protocolo de comunicación de uso específico GPIB. [2][6]

Figura 10, "*GPIB Read*", lee una cantidad definida de bytes, de un dispositivo previamente seleccionado por su dirección. Además se le puede indicar con cuál de sus tres modos puede finalizar: 0 para ningún final, 1 para añadir CR (0Dh \r) o 2 con el final LF (0Ah \n). [2][6]

Figura 11, "*GPIB Write*", escribe las cadenas de comandos en el dispositivo direccionado, el modo para finalizar la línea puede variar de ocho maneras distintas. [2][6]

Figura 12, "*GPIB Clear*", reinicia y manda a un dispositivo seleccionado al estado predeterminado de fábrica.[2][6]

Figura 13, "*GPIB Wait*", espera a que el dispositivo indicado pase a un cierto estado. El estado se indica con la misma codificación que el VI anterior. [2][6]

Figura 14, "*GPIB Misc*", ejecuta un comando de bajo nivel, dependiendo del tipo de comando se pondrá o no la dirección del dispositivo. [2][6]

4. UNIDADES DE TRABAJO PARA EL PROCESO DE EMC

Para el proceso correspondiente, la medición que se realiza sobre los equipos en prueba, le corresponde una unidad de medida en decibeles por micro volt (dBμV), la cual se describe en las siguientes ecuaciones, para las cuales se tienen también las consideraciones siguientes:

V → Voltaje, volts.

R → Resistencia, Ohms.

P → Potencia (Watts, dBm, mW o W).

1mW → 1E-3 Watts.

1μV → 1E-6 Volts.

dB → Radio decibel ($\log_{10}$).

dBm → Radio decibel de Watts sobre miliwatt, descrito en la ecuación 1.

$$dBm = 10 \times \log_{10}\left(\frac{P}{1mW}\right)$$

Ecuación 1, representación de los parámetros que componen la unidad dBm.

dBμV → Radio decibel de Volts sobre microvolt, descrito por la ecuación 2.

$$dB\mu V = 20 \times \log_{10}\left(\frac{V}{1\mu V}\right)$$

Ecuación 2, descripción de los parámetros que conforman la unidad dBμV.

Para una impedancia de 50 Ohms en el sistema de medición, se tendrá la siguiente relación de conversión, presentada en la ecuación 3:

$$dB\mu V = dBm + 107dB$$

Ecuación 3, relación de conversión entre las unidades dBm y dBμV, para una impedancia de 50Ω.

La transformación descrita por la ecuación 3, se resuelve según lo muestran las ecuaciones 4 a 7:

Sea:

$$V = \sqrt{P \times R}$$

Ecuación 4, relación de composición del voltaje, con las componentes potencia y resistencia.

Y utilizando la ecuación 1, se despejará para la potencia (P) mostrada en la ecuación 5:

$$P = 1mW\left(10^{\left[\frac{dBm}{10}\right]}\right)$$

Ecuación 5, despeje de la potencia a partir de la ecuación 4, para obtenerla de una medición con unidad dBm.

Sustituyendo la ecuación 5 en la 4 y recordando que R = 50Ω se obtiene la ecuación 6:

$$V = \sqrt{\left[1mW\left(10^{\left[\frac{dBm}{10}\right]}\right)\right]\times 50\Omega}$$

Ecuación 6, sustitución de la potencia en la ecuación 4.

Modificando la ecuación 2 introduciendo la 6 en ella se tiene la ecuación 7:

$$dB\mu V = 20\times\log_{10}\left(\frac{\sqrt{\left[1mW\left(10^{\left[\frac{dBm}{10}\right]}\right)\right]\times 50\Omega}}{1\mu V}\right)$$

Ecuación 7, transformación a dBμV a partir de una medición en dBm.

En la ecuación 7, se obtiene la relación adecuada para la transformación de unidades de dBm a dBμV, donde las aproximaciones, al realizar varias transformaciones, dan un parámetro de 107, que separan a una unidad de otra, como lo presenta la ecuación 3.

Ejemplo: Para una medición de 50 dBm, obtener su transformación a dBμV utilizando la ecuación 3 y comprobar el resultado con la aplicación de la ecuación 7.

Usando la ecuación 3, se tendrá:

$$dB\mu V = 50+107dB = 157dB\mu V$$

Comprobando con la ecuación 7, se tiene:

$$dB\mu V = 20\times\log_{10}\left(\frac{\sqrt{\left[1mW\left(10^{\left[\frac{50}{10}\right]}\right)\right]\times 50\Omega}}{1\mu V}\right) = 156.99dB\mu V \approx 157dB\mu V$$

[7].

PROCEDIMIENTO

El proceso siguiente, fue planificado para la integración del proyecto "Automatización de la prueba de emisiones conducidas del laboratorio EMC de MABE TYP" y fue desarrollado desde el punto de vista de los objetivos específicos.

- ❖ Observación, reconocimiento de los equipos de trabajo, y su documentación.
- ❖ Indagación del protocolo de comunicación GPIB, nociones básicas acerca de éste.
- ❖ Acopio de información acerca del protocolo GPIB para comunicación con dispositivos en el caso de ser necesario, conocimiento profundo del tema.
- ❖ Verificación del funcionamiento e interacción de trabajo de los equipos y su controlador.
- ❖ Lectura y análisis de la documentación de la tarjeta PCMIA-GPIB.
- ❖ Instalación y adquisición de licencia del software LabVIEW.
- ❖ Desarrollo y construcción del software y ambiente de trabajo para el proceso de emisiones conducidas.
- ➢ Utilización de la herramienta NI-SPY para la captura de comandos durante un proceso de trabajo normal de los dispositivos.
- ➢ Análisis de los comandos y rutinas de trabajo capturados.
- ➢ Investigación de la naturaleza de los datos adquiridos durante un proceso de trabajo. Conversión de éstos en caso de ser requeridos para su interpretación.
- ➢ Puesta en marcha de prototipo de software para los dispositivos en existencia.
- ➢ Adquisición y almacenamiento en una base de datos de la información adquirida para la generación de reportes y construcción de gráficas.
- ➢ Generación de trabajo final para la especialidad durante el transcurso del proyecto.

5. ANALISIS DEL PROCESO

Es necesario aclarar que de todas las pruebas llevadas acabo en el área de "Compatibilidad electromagnética" en MABE T&P, la que ocupó todos los esfuerzos, fue la de Emisiones conducidas, que a su vez fue tratada en pequeñas partes, para poder llegar a una solución, que, en una primera instancia, diera una solución total o parcial, según la observación de los interesados del área.

5. 1. PROCESO DE EMISIONES CONDUCIDAS

El proceso de emisiones conducidas, es una técnica utilizada para la medición de las alteraciones electromagnéticas; producidas debido al uso de instrumentos eléctrico-electrónicos, las cuales son inducidas de manera directa a la línea eléctrica del hogar. Dichas perturbaciones son medidas en escala de μV/dB y se encuentran normalizadas por las diferentes políticas de los países a los cuales los productos MABE tienen acceso.

Para la medición de estos parámetros se requiere del uso de tres equipos y un software. Dichos instrumentos son un analizador de frecuencias modelo ESPI-3 y un sistema de acoplo de señal llamado LISN, los cuales son controlados desde un ordenador, todo a través

de un software llamado EMC32, todos los equipos y software, de la marca "Rohde&Schwarz".

Teniendo en cuenta lo anterior y según las instrucciones de los interesados de la empresa, la parte que interesaba atacar fue la interacción de un software genérico, que lograra hacer comunicación y control de los instrumentos inmiscuidos en el proceso.

En la ilustración 15, se muestra el analizador de frecuencias, la ilustración 16 da a conocer el acoplador de señal LISN, la imagen 17 ilustra el ordenador utilizado para el control de los equipos. En la ilustración número 18 se puede observar una prueba regular del proceso de emisiones conducidas. En la imagen 19 se puede apreciar el software utilizado para el control y medición de la prueba.

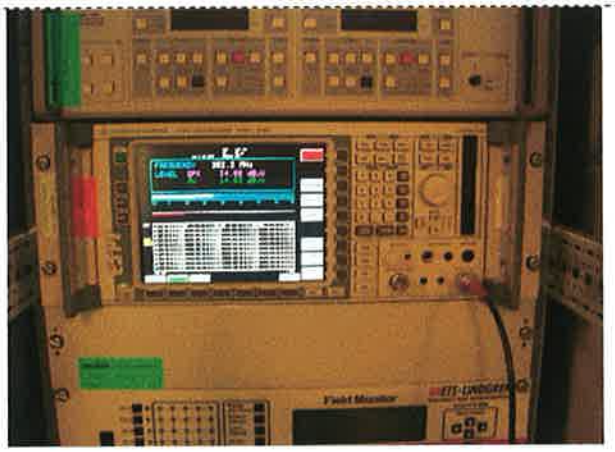


Figura 15 Analizador de frecuencias ESPI-3.

Figura 16 Acoplador de señal LISN.

Figura 17 Ordenador para el control del sistema.

Figura 18 Prueba regular.

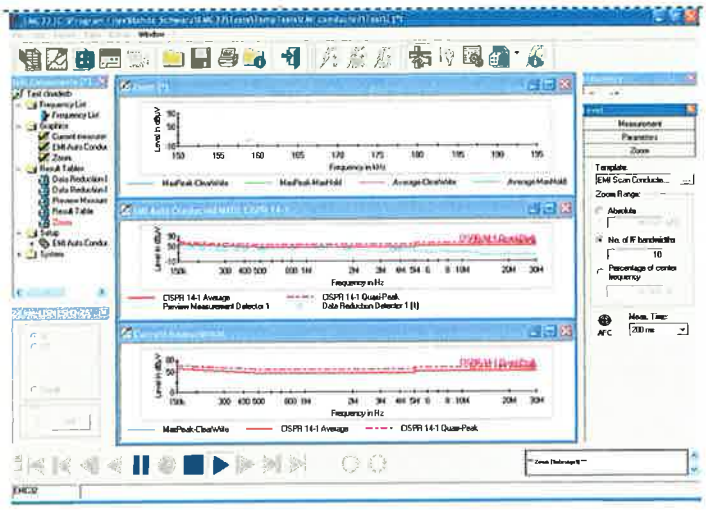


Figura 19 Software de control EMC32.

6 RECONOCIMIENTO DE DISPOSITIVOS

Es importante notificar que, dentro de cualquier integración de equipos, es necesario el realizar la identificación de los aparatos que interactuaran, esto es, asegurarse de que éstos serán debidamente identificados dentro del sistema de trabajo, para lo cual los sistemas NI tienen una gran ventaja, ya que con la ayuda de la herramienta "Measurement & Automation" es posible visualizar los elementos que actúan dentro de una red de control y adquisición de datos. Además en el supuesto de que los controladores adecuados para cada uno de los equipos, no se halla instalado, la herramienta nos permite verificar si es que se encuentran debidamente registrados en el sistema.

En la imagen 20 se visualiza el direccionamiento de "Measurement & Automation", además la ilustración 21 muestra el software durante la identificación de dispositivos compatibles o de la marca NI.

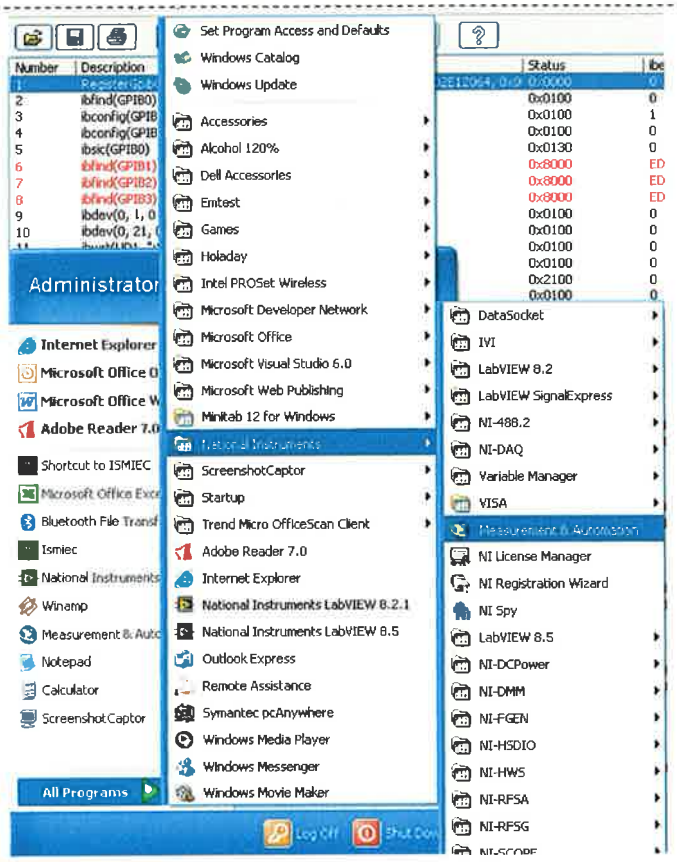


Figura 20 Dirección de Measurment & Automation.

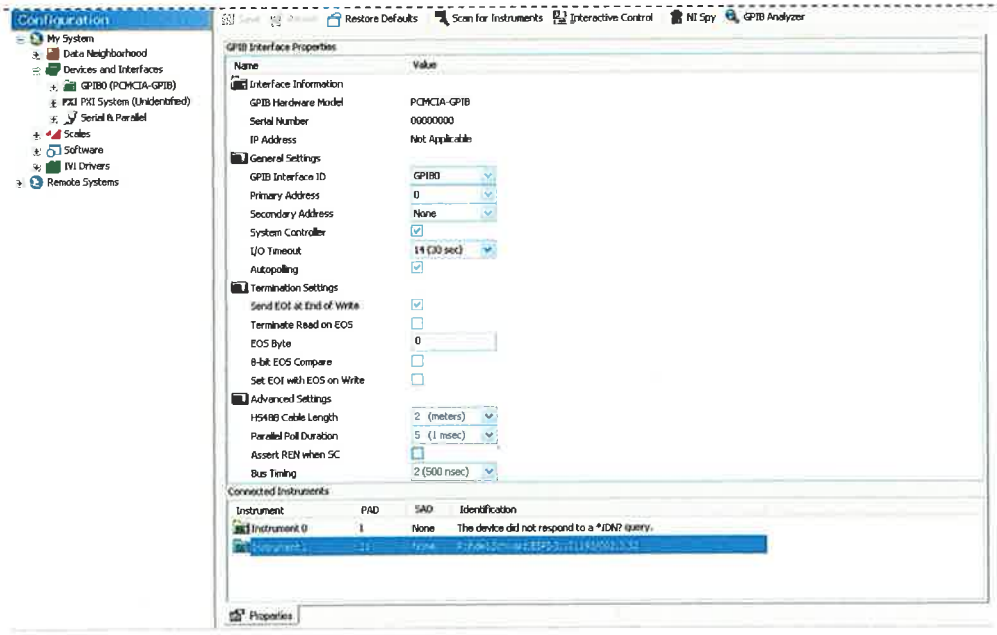


Figura 21 Corrida de la herramienta para identificación.

7 NI-SPY

Tomando en cuenta, que el patrón de trabajo del software e integración que realizó el fabricante, podía ser repetido, con el propósito de tener mayor seguridad en el funcionamiento de los equipos, por lo que según la investigación realizada, se acudió al uso del software NI-SPY de National Instruments, ya que, a través de el es posible observar de manera estructurada y sin pérdidas de información, todas y cada una de las líneas de programación utilizadas para el control de los instrumentos, lo que nos dio como resultado, un código capturado, que sería re-ejecutado con la ayuda de cualquier lenguaje de programación o ambiente, que debido a que la tarjeta que se utilizaría pertenece a NI, fue necesario utilizar LABView.

En la ilustración número 22 se aprecia el software NI-SPY, con el cual es posible realizar capturas de códigos que sean utilizados durante el control de sistemas GPIB, esta herramienta es ejecutada en paralelo.

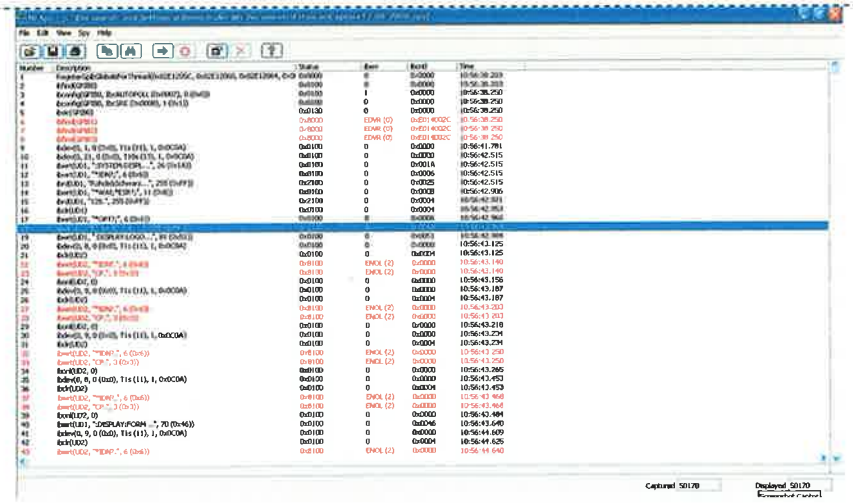


Figura 22 Herramienta NI-SPY para GPIB.

RESULTADOS

8 CÓDIGO CAPTURADO CON NI-SPY

Como resultado de la captura y análisis de las diferentes acciones que realiza el software EMC-32, se dividió el código de forma que pudiera ser identificado como una serie de procedimientos para el correcto funcionamiento del sistema en su conjunto, además de la aclaración de algunos de los diferentes comandos del protocolo GPIB, con la intención de tener una idea mejor de lo que se ejecutaba, dando como resultado la tabla 6.

Tabla 6 Categorización de comandos capturados

ORIGINAL	LAST VERSION
SELF CHECK	SELF CHECK
ibsic(GPIB0) *Makes the GPIB interface CIC (controller in charge)*	loc 21
:SYSTEM:DISPLAY:UPDATE ON; *Enable the update of all display elements.*	Sic
*WAI; *Wait until all commands have executed, and returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.*ESR?; *Queries the status of the contents of the event status register.*	SYSTEM:DISPLAY:UPDATE ON;
ibclr(UD1) *Clear a specific device.*	*WAI; *ESR?;
*OPT?; *Option identification query.*	*OPT?;
:DISPLAY:LOGO OFF;WINDOW2:TEXT 'Hola que viene';:DISPLAY:CMAP6:PDEFINED YELLOW; *It is only for configuring appearance of text.*	DISPLAY:LOGO OFF; WINDOW2:TEXT '32'; DISPLAY:CMAP6:PDEFINED GREEN;
:DISPLAY:FORMAT SPLIT; *Switches the display to 2 measurement*	DISPLAY:FORMAT SPLIT;

windows. :OUTPUT:UPORt #B00011111; *This command sets the control lines of the user ports. In manual operation, the control lines are represented by softkeys PORT 0 to 7.* :DISPLAY:FORMAT SINGLE; *Switches the display to 1 measurement window.*	OUTPUT:UPOR1 #B00011111; DISPLAY:FORMAT SINGLE;
:DISPLAY:LOGO ON;WINDOW1:TEXT 'Ventana 1';:DISPLAY;WINDOW2:TEXT 'Ventana 2'; *It is only for configuring appearance of text.*	DISPLAY:LOGO ON; WINDOW1:TEXT 'pRoBaNdO'; DISPLAY; WINDOW2:TEXT 'PrObAnDo';
ibloc(UD1) *Go to local*	loc 21
ibonl(UD1, 0) *Place the interface board online or offline.*	SYSTEM:DISPLAY:UPDATE ON;
:SYSTEM:DISPLAY:UPDATE ON; *Enable the update of all display elements.*	*WAI; *ESR?;
*WAI; *Wait until all commands have executed, and returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.*ESR?; *Queries the status of the contents of the event status register.*	*OPT?;
ibclr(UD1) *Clear a specific device.*	DISPLAY:LOGO OFF; WINDOW2:TEXT '32'; DISPLAY:CMAP6:PDEFINED GREEN;
*OPT?; *Option identification query.*	FREQUENCY LEVEL
:DISPLAY:LOGO OFF;WINDOW2:TEXT 'Hola que	Sic

va';:DISPLAY:CMAP6:PDEFINED BLUE; *It is only for configuring appearance of text.*	
FREQUENCY LEVEL	sre 1
ibsic(GPIB0) *Makes the GPIB interface CIC (controller in charge)*	cmd .
ibsre(GPIB0, 1) *Set or clear the Remote Enable (REN) line.*	DISPLAY:LOGO ON; WINDOW1:TEXT 'PrUeBa ToRdF'; DISPLAY:WINDOW2:TEXT 'bY tOrDf';
ibcmd(GPIB0, ".", 1 (0x1)) *Send GPIB commands.*	loc 21
:DISPLAY:LOGO ON;WINDOW1:TEXT 'vEnTaNa 1';:DISPLAY:WINDOW2:TEXT 'VeNtAnA 2'; *It is only for configuring appearance of text.*	SYSTEM:DISPLAY:UPDATE ON;
ibloc(UD1) *Go to local.*	*WAI; *ESR?;
ibonl(UD1, 0) *Place the interface board online or offline.*	*OPT?;
:SYSTEM:DISPLAY:UPDATE ON; *Enable the update of all display elements.*	DISPLAY:LOGO OFF; WINDOW2:TEXT '32'; DISPLAY:CMAP6; PDEFINED GREEN;
*WAI; *Wait until all commands have executed, and returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.*ESR?; *Queries the status of the contents of the event*	TABLE MANAGER PART 1

status register.	
ibclr(UD1) *Clear a specific device.*	DISPLAY:FORMAT SPLIT; OUTPUT:UPOR1 #B00011101; DISPLAY:FORMAT SINGLE;
*OPT?; *Option identification query.*	DISPLAY:FORMAT SPLIT; OUTPUT:UPOR1 #B00011101; DISPLAY:FORMAT SINGLE;
:DISPLAY:LOGO OFF;WINDOW2:TEXT 'Hola que va';:DISPLAY:CMAP6:PDEFINED BLUE; *It is only for configuring appearance of text.*	*CLS;
TABLE MANAGER 1	*ESE 49; *SRE 32;
:DISPLAY:FORMAT SPLIT; *Switches the display to 2 measurement windows.* :OUTPUT:UPORt #B00011101; *This command sets the control lines of the user ports. In manual operation, the control lines are represented by softkeys PORT 0 to 7.* :DISPLAY:FORMAT SINGLE; *Switches the display to 1 measurement window.*	MMEMORY:LOAD:AUTO 1,'FACTORY'; *RST; *OPC;
:DISPLAY:FORMAT SPLIT; *Switches the display to 2 measurement windows.* :OUTPUT:UPORt #B00011101; *This command sets the control lines of the user ports. In manual operation, the control lines are represented by softkeys PORT 0 to 7.* :DISPLAY:FORMAT SINGLE;	TABLE MANAGER PART 2

Switches the display to 1 measurement window.	
*CLS; *CLEAR STATUS sets the status byte (STB), the standard event register (ESR) and the EVENt-part of the QUEStionable and the OPERation register to zero. The command does not alter the mask and transition parts of the registers. It clears the output buffer.*	*ESR?;
*ESE 49; *EVENT STATUS ENABLE, sets the bits of the event status enable register* *SRE 32; *Service Request Enable*	SYSTEM:DISPLAY:UPDATE ON;
:MMEMORY:LOAD:AUTO 1,'FACTORY'; *This command defines which device setting is automatically loaded after the device is switched on. The contents of the file are read after switching on the device and used to define the new device state. FACTORY denotes the data set previously in the instrument.* *RST; *RESET, resets the device* *OPC; *OPERATION COMPLETE sets bit 0 in the event status register when all preceding commands have been executed. This bit can be used to initiate a service request.*	OUTPUT:UPOR1:STAT ON;
TABLE MANAGER 2	DISPLAY:LOGO OFF; WINDOW2:TEXT '32'; DISPLAY:CMAP6:PDEFINED GREEN;
ibrsp(UD1, 96 (0x60)) *Conduct a*	DISPLAY:FORMAT SPLIT;

serial poll. Return a serial poll byte. To empty the queue use the ibrsp function (The verb queue means to form a line, and to wait for services. Queue is also the name of this line). It returns the first queued response.	OUTPUT:UPOR1 #B00011101; DISPLAY:FORMAT SINGLE;
*ESR?; *Queries the status of the contents of the event status register.*	INSTRUMENT RECEIVER; *OPC;
:SYSTEM:DISPLAY:UPDATE ON; *Enable the update of all display elements.*	**TABLE MANAGER PART 3**
:OUTPUT:UPOR1:STAT ON; *This command switches the control line of the user ports between INPut and OUTPut. The user port is switched to OUTPut with parameter ON, to INPut with OFF.*	*ESR?;
:DISPLAY:LOGO OFF;WINDOW2:TEXT 'Hola que va';:DISPLAY:CMAP6:PDEFINED BLUE; *It is only for configuring appearance of text.*	DISPLAY:FORMAT SINGLE;
:DISPLAY:FORMAT SPLIT; *Switches the display to 2 measurement windows.* :OUTPUT:UPORt #B00011101; *This command sets the control lines of the user ports. In manual operation, the control lines are represented by softkeys PORT 0 to 7.* :DISPLAY:FORMAT SINGLE; *Switches the display to 1 measurement window.*	STATUS:OPERATION:ENABLE 65535;
:INSTRUMENT RECEIVER; *This command switches between the operating modes by means of text*	INITIATE2:CONTINUOUS OFF;

parameters. EMI receiver. *OPC; *OPERATION COMPLETE sets bit 0 in the event status register when all preceding commands have been executed. This bit can be used to initiate a service request.*	
TABLE MANAGER 3	BANDWIDTH:AUTO OFF;
ibrsp(UD1, 96 (0x60)) *Conduct a serial poll. Return a serial poll byte. To empty the queue use the ibrsp function (The verb queue means to form a line, and to wait for services. Queue is also the name of this line). It returns the first queued response.*	SCAN:RANGES 1;
*ESR?; *Queries the status of the contents of the event status register.*	DISPLAY:TRACE:X:SPACING LOGARITHMIC;
:DISPLAY:FORMAT SINGLE; *Switches the display to 1 measurement window.*	FORM ASC;
:STATUS:OPERATION:ENABLE 512; *0 to 65535 This command sets the bits of the ENABle section of the STATus:OPERation register. The ENABle register electively enables the individual events of the associated EVENt section for the summary bit in the status byte.*	TRACE:FEED:CONTROL ALWAYS;
:INITIATE2:CONTINUOUS OFF; *Switches the sequence in screen B to single scan/sweep.*	*WAI; *ESR?;
:BANDWIDTH:AUTO OFF; *Switches off the coupling of the IF bandwidth to the 'frequency range (receiver mode).*	SCAN1:START 150000 HZ; FREQUENCY:START 150000 HZ;
:SCAN:RANGES 1; *Sets the number*	SCAN1:STOP 30000000 HZ;

of ranges to 1	FREQUENCY:STOP 30000000 HZ;
:DISPLAY:TRACE:X:SPACING LOGARITHMIC; *This command toggles between linear and logarithmic display in receiver mode.*	BANDWIDTH:FILTER 6; SCAN1:BANDWIDTH:RESOLUTION 9 KHZ;
:FORMAT:DATA REAL; *This command specifies the data format for the data transmitted from the instrument to the control PC. See also TRACE:DATA? The trace data in ASCII format (ASCII FILE EXPORT) is exported via the "MMEM:STORe:TRACe" command. P. 270*	SWEEP:SPACING LOGARITHMIC; SCAN1:STEP 1
:TRACE:FEED:CONTROL ALWAYS; *This command switches block data transmission during a scan on and off. The block size depends on the scan time; the trace number is not evaluated.*	:SENSE2:DETECTOR1 POSITIVE;:SENSE2:DETECTOR2 AVERAGE;:DISPLAY:WINDOW2:TRACE2:MODE WRITE;
*WAI; *Wait until all commands have executed, and returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.*ESR?; *Queries the status of the contents of the event status register.*	SCAN1:TIME 500 MS
:SCAN1:START 150000 HZ; *This command defines the start frequency of the selected receiver scan range.* :FREQUENCY:START 150000 HZ; *This command defines the start frequency of the overall scan in receiver mode. It defines the start frequency of the sweep in analyzer mode.*	SCAN1:INPUT:GAIN:STATE OFF;

:SCAN1:STOP 30000000 HZ; *This command defines the stop frequency of the selected receiver scan range.* :FREQUENCY:STOP 30000000 HZ; *This command defines the stop frequency of the overall scan in receiver mode. It defines the stop frequency of the sweep in analyzer mode.*	SCAN1:INPUT:ATTENUATION:AUTO ON;
:BANDWIDTH:FILTER 6;:SCAN1:BANDWIDTH:RESOLUTION 9 KHZ;	INPUT:ATTENUATION:PROTECTION OFF;
:SWEEP:SPACING LOGARITHMIC; *This command switches between linear, logarithmic and automatically selected linear step size in the receiver. In the receiver mode, this parameter has no impact on the graphical representation of the frequency axis. In the analyzer, the command toggles between linear and logarithmic sweep. The frequency axis is represented linearly or logarithmically depending on the sweep.* :SCAN1:STEP 1 HZ; *This command defines the step size for the frequency of the selected receiver scan range. From 1 to 100 Hz.*	DISPLAY:TRACE:Y:BOTTOM -3; TOP 100; BOTTOM -10; TOP 90;
:SENSE2:DETECTOR1 POSITIVE; *This command switches on the detector for the data acquisition in the selected trace and the indicated measurement window. The POSitive or NEGative detector only displays the positive or the negative peak value.* :SENSE2:DETECTOR2 AVERAGE; *The AVERage detector displays the*	SENSE2:DEMOD OFF;

power average value at each test point. :DISPLAY:WINDOW2:TRACE2:MODE WRITE; *This command defines the type of display and the evaluation of the traces in the selected measurement window. WRITE corresponds to the Clr/Write mode of manual operation.*	
:SCAN1:TIME 500 MS; *This command defines the measurement time of the receiver subscan.*	SYSTEM:SPEAKER:VOLUME 0E-2;
:SCAN1:INPUT:GAIN:STATE OFF; *This command switches on or off the preamplifier in the selected receiver scan range.*	*WAI; *ESR?;
:SCAN1:INPUT:ATTENUATION:AUTO ON; *This command switches on or off the autoranging function in the selected receiver scan range.*	TRACE:FEED:CONTROL ALWAYS; *SRE 128; INITIATE2;
:INPUT:ATTENUATION:PROTECTION OFF; *This command defines whether the 0 dB position of the attenuator is to be used in manual or automatic adjustment.*	WAVE FORM PETITION
:DISPLAY:TRACE:Y:BOTTOM -3; *This command defines the minimum grid level in the current unit for the scan display in the receiver mode.* TOP 100; *This command defines the maximum grid level in the current unit for the scan display in the receiver mode.* BOTTOM -10; *This command defines the minimum grid level in the current unit for the scan display in the receiver mode.* TOP 90; *This command defines the maximum grid level in the current unit for the scan display in the*	*ESR?;

receiver mode.	
:SENSE2:DEMOD OFF; *This command selects the type of analog demodulation.*	STATUS:OPERATION?;
:SYSTEM:SPEAKER:VOLUME 0E-2; *This command sets the volume of the built-in loudspeaker for demodulated signals. Minimum volume is set by 0 and maximum volume by 1.*	TRACE? SCAN;
*WAI; *Wait until all commands have executed, and returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.*ESR?; *Queries the status of the contents of the event status register.*	TABLE MANAGER PART 4
:TRACE:FEED:CONTROL ALWAYS; *This command switches block data transmission during a scan on and off. The block size depends on the scan time; the trace number is not evaluated.* *SRE 128; *Service Request Enable* :INITIATE2; *The command initiates a new sweep in the indicated measurement window. In receiver mode with SINGLE selected, the R&S ESPI performs a single scan and stops at the end frequency. With CONTINUOUS selected, the scan is performed continuously until it is deliberately stopped. In analyzer mode, with Sweep Count > 0 or Average Count > 0, this means a restart of the indicated number of measurements. With trace functions MAXHold, MINHold and AVERage, the previous results are reset on*	*WAI; *ESR?;

*restarting the measurement. In single sweep mode, synchronization to the end of the indicated number of measurements can be achieved with the command *OPC, *OPC? Or *WAI. In continuous-sweep mode, synchronization to the sweep end is not possible since the overall measurement never ends.*	
WAVE FORM PETITION	*CLS; *SRE 0;
ibrsp(UD1, 192 (0xC0)) *Conduct a serial poll. Return a serial poll byte. To empty the queue use the ibrsp function (The verb queue means to form a line, and to wait for services. Queue is also the name of this line). It returns the first queued response.*	DISPLAY:FORMAT SPLIT; OUTPUT:UPOR1 #B00011110; DISPLAY:FORMAT SINGLE;
*ESR?; *Queries the status of the contents of the event status register.*	DISPLAY:FORMAT SPLIT; OUTPUT:UPOR1 #B00011110; DISPLAY:FORMAT SINGLE;
:STATUS:OPERATION?; *This command queries the contents of the EVENt section of the STATus:OPERation register. The contents of the EVENt section are deleted after readout.*	*CLS;
:TRACE? SCAN; *This command transfers trace data from the control computer to the instrument, the query reads trace data out of the instrument. The associated measurement window is selected with the numeric suffix of TRACe<1|2>. TRAC TRACE1,"+A$*	*ESE 49; *SRE 32;

(A$: data list in the current format) "TRAC? TRACE1"	
ibeos(UD1, 0x0000) *Configure the End-Of-String termination mode or character. It is used to enable, disable, or configure the EOS modes.*	*WAI; *ESR?;
ibeos(UD1, 0x0C0A) *Configure the End-Of-String termination mode or character. It is used to enable, disable, or configure the EOS modes.*	TRACE:FEED:CONTROL ALWAYS; *SRE 128; INITIATE2;
TABLE MANAGER 4	ENDING
*WAI; *Wait until all commands have executed, and returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.*ESR?; *Queries the status of the contents of the event status register.*	*WAI; *ESR?;
*CLS; *CLEAR STATUS sets the status byte (STB), the standard event register (ESR) and the EVENt-part of the QUEStionable and the OPERation register to zero. The command does not alter the mask and transition parts of the registers. It clears the output buffer.* *SRE 0; *Service Request Enable.*	*CLS; *SRE 0;
:DISPLAY:FORMAT SPLIT; *Switches the display to 2 measurement windows.* :OUTPUT:UPORt #B00011110; *This command sets the control lines of the user ports. In manual operation, the control lines are represented by softkeys PORT 0 to 7.* :DISPLAY:FORMAT SINGLE;	sre 0

Switches the display to 1 measurement window.	
:DISPLAY:FORMAT SPLIT; *Switches the display to 2 measurement windows.* :OUTPUT:UPORt #B00011110; *This command sets the control lines of the user ports. In manual operation, the control lines are represented by softkeys PORT 0 to 7.* :DISPLAY:FORMAT SINGLE; *Switches the display to 1 measurement window.*	sre 1
*CLS; *CLEAR STATUS sets the status byte (STB), the standard event register (ESR) and the EVENt-part of the QUEStionable and the OPERation register to zero. The command does not alter the mask and transition parts of the registers. It clears the output buffer.*	DISPLAY:LOGO ON; WINDOW1:TEXT ''; DISPLAY:WINDOW2:TEXT '';
*ESE 49; *EVENT STATUS ENABLE, sets the bits of the event status enable register.* *SRE 32; *Service Request Enable*	loc 21
*WAI; *Wait until all commands have executed, and returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.*ESR?; *Queries the status of the contents of the event status register.*	off 21
:TRACE:FEED:CONTROL ALWAYS; *This command switches block data transmission during a scan on and off. The block size depends on the scan time; the trace number is not*	

evaluated. *SRE 128*; Service Request Enable* :INITIATE2; *The command initiates a new sweep in the indicated measurement window. In receiver mode with SINGLE selected, the R&S ESPI performs a single scan and stops at the end frequency. With CONTINUOUS selected, the scan is performed continuously until it is deliberately stopped. In analyzer mode, with Sweep Count > 0 or Average Count > 0, this means a restart of the indicated number of measurements. With trace functions MAXHold, MINHold and AVERage, the previous results are reset on restarting the measurement. In single sweep mode, synchronization to the end of the indicated number of measurements can be achieved with the command *OPC, *OPC? Or *WAI. In continuous-sweep mode, synchronization to the sweep end is not possible since the overall measurement never ends.*	
END	
*WAI; *Wait until all commands have executed, and returns the contents of the event status register in decimal form (0 to 255) and subsequently sets the register to zero.*ESR?; *Queries the status of the contents of the event status register.*	
*CLS; *CLEAR STATUS sets the status byte (STB), the standard event register (ESR) and the EVENt-part of the QUEStionable and the OPERation	

register to zero. The command does not alter the mask and transition parts of the registers. It clears the output buffer. *SRE 0; *Service Request Enable.*	
ibsre(GPIB0, 0) *Set or clear the Remote Enable (REN) line.*	
ibsre(GPIB0, 1) *Set or clear the Remote Enable (REN) line.*	
:DISPLAY:LOGO ON;WINDOW1:TEXT 'bYe';:DISPLAY:WINDOW2:TEXT 'ByE';	
ibloc(UD1) *Go to local.*	
ibonl(UD1, 0) *Place the interface board online or offline.*	

9 GENERACIÓN DE SUBVIS CON LABVIEW

Una vez identificadas las acciones que generaba el código capturado, fue ejecutada cada una de esas partes, utilizando para ello las librerías disponibles dentro del entorno de NI, por lo que teniendo conocimiento de la existencia de una librería especializada para el manejo del protocolo GPIB, dentro del ambiente LABView, se generaron los subVIs "IDN", "Selfcheck", "Writting & Reading", "Frequency Level", "Table Manager Part 1", "Table Manager Part 2", "Table Manager Part 3", "Waveform Petition", "Table Manager Part 4" y "END".

9.1. SOFTWARE PROTOTIPO

Con el propósito de realizar una comunicación exitosa, y un control adecuado de los dispositivos, se llegó a la construcción de un software en base a LABView, el cual, utilizando las líneas de comando capturadas con el NI-SPY, se encargaría de ejecutar de forma adecuada y en tiempo todo el programa, como resultado se obtuvo lo ilustrado en las imágenes 23 y 24.

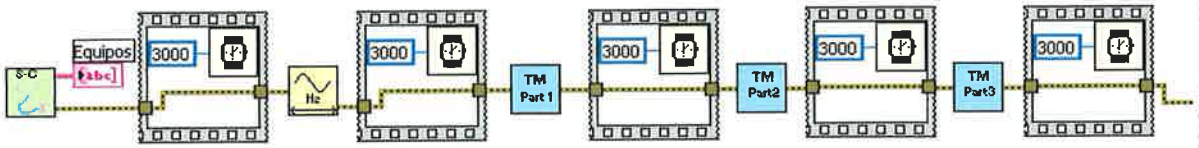


Figura 23 Software parte 1.

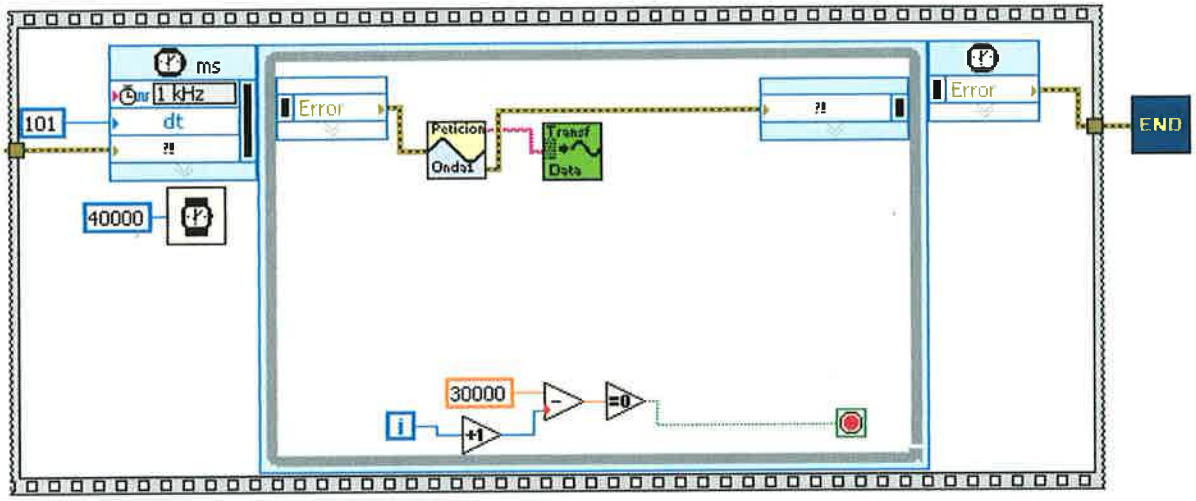


Figura 24 Software parte 2.

9.1. SUBVI "IDN"

El SubVI "IDN" dentro de toda la ejecución de programa, es el encargado de solicitar la identificación del dispositivo que se va a controlar, esto se realiza para tener seguridad en cuál dispositivo se comunica. En toda la aplicación de programa solo se le utiliza una sola vez, específicamente al inicio, y realmente se encuentra almacenado y ejecutado en el "Selfcheck", la ilustración 25 muestra el "IDN" y en la 26 su logo de identificación.

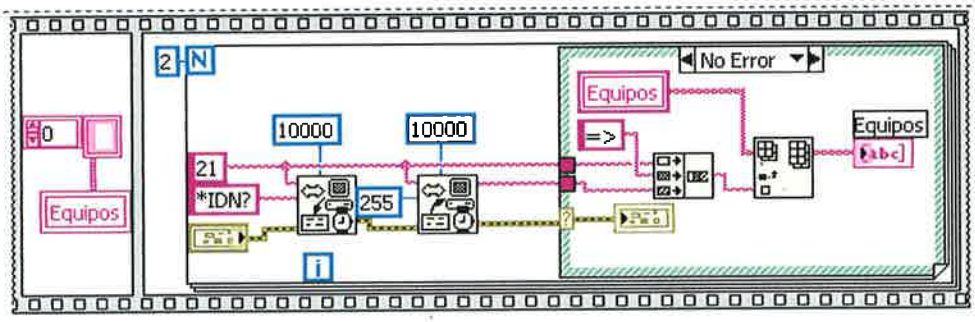


Figura 25 Conjunto del "IDN".

Figura 26 Logo de "IDN".

9.2. SUBVI "SELFCHECK"

Éste es el punto inicial de todo el programa, en el se verifica que todos los dispositivos se encuentren habilitados para el trabajo, donde les solicita la identificación, dirección de los mismos y muestra al usuario el resultado de la conexión, en este caso el nombre del dispositivo identificado, la figura 27 nos muestra los bloques que lo conforman y la número 28 el logo que se le relaciona.

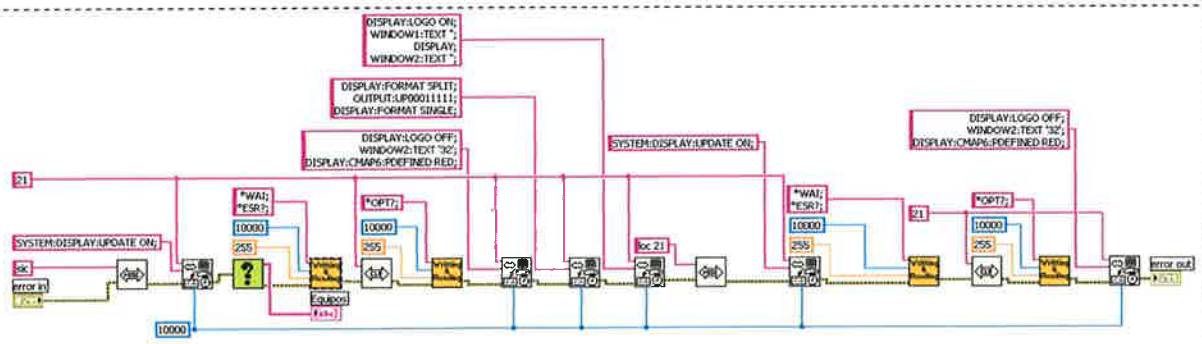


Figura 27 Bloques "Selfcheck".

Figura 28 Logo "Selfcheck".

El código que se transmite desde éste bloque puede ser observado en la tabla 6, mostrada en páginas anteriores.

9.3. SUBVI "WRITTING & READING"

En algunas ocasiones el programa solita no solo la escritura de comandos hacia el dispositivo, sino además la lectura de la respuesta que éste pueda dar a dicha petición, que en su momento es utilizada por el programa del fabricante, sin embargo para la aplicación desarrollada, se considero necesario agregar éstas solicitudes para respetar la forma de trabajo del dispositivo. En las figuras 29 y 30 se aprecia el bloque de comandos y el logo de identificación, respectivamente.

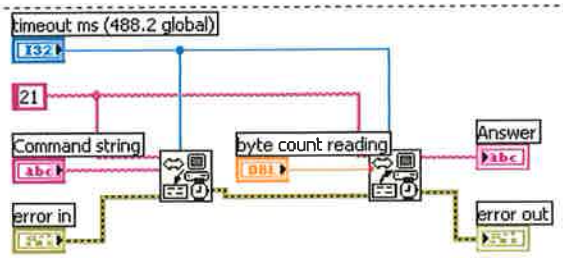


Figura 29 Bloques para lectura y escritura.

Figura 30 Logo de identificación para "Writting & Reading".

9.4. SUBVI "FREQUENCY LEVEL"

Es en este bloque en el que se realizan los ajustes para la medición, se configuran los límites inferiores y superiores en el dispositivo, además de realizar la configuración de las ventanas y la forma en la que se visualizaran las formas de onda, o si es que éstas serán rotuladas con algún título. En las imágenes 31 y 32 se identifican los bloques que le conforman y su logo de identificación.

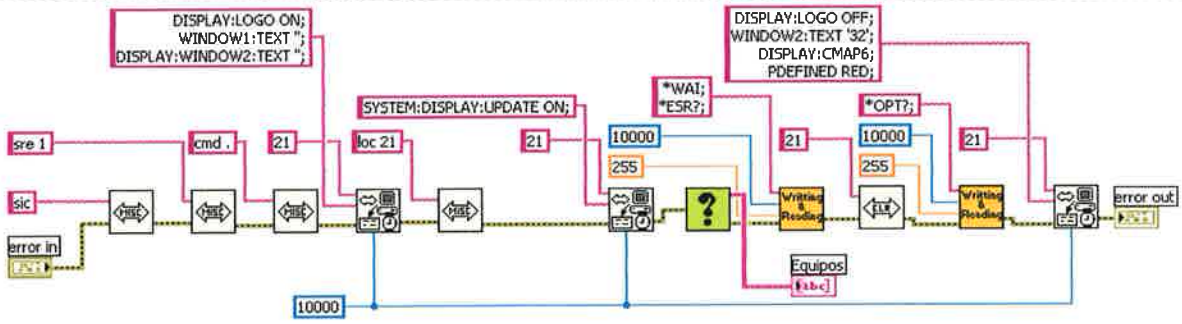


Figura 31 Bloques que conforman el "Frequency Level".

Figura 32 Imagen de identificación del "Frequency Level".

El código de programación GPIB que se introduce en estos bloques se puede observar en la tabla 6, en páginas previas.

9.5. SUBVI "TABLE MANAGER, PART 1"

Los paneles de comandos llamados "Table Manager, Part 1-3" son elementos encargados de preparar al dispositivo para la inicialización de la medición, también preparan las visualizaciones en la pantalla del analizador de frecuencia, además ordenan en donde se realizará el almacenamiento de los datos medidos, y qué tipo de configuraciones extra se le darán al dispositivo y a los datos que se trabajaran, ya que los datos pueden ser referidos al sistema de adquisición en forma binaria o en código ASCII, todo dependerá de la cantidad o tamaño del dato que se desee trabajar. Cabe aclarar que el "Table Manager, Part 4" es utilizado para la ejecución de un segundo conjunto de pruebas, las cuales toman como referencia la línea neutra de la alimentación de corriente alterna, por lo que vuelve a realizar una configuración del dispositivo. Las ilustraciones 33 a 40 muestran los diferentes cuadros de introducción de comandos utilizados dentro de los "Table Manager", así como su logo de identificación.

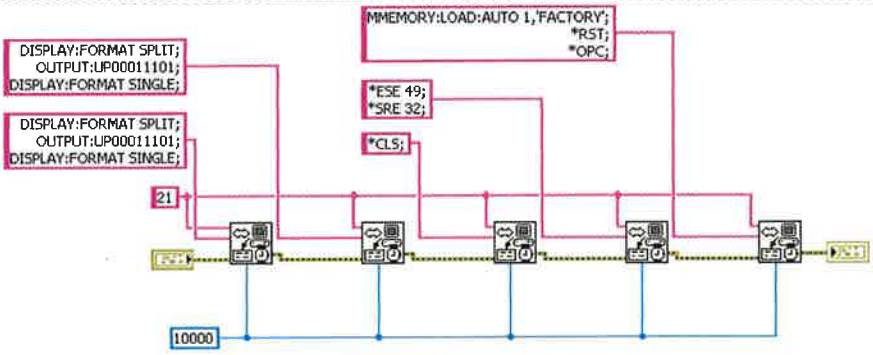


Figura 33 Bloques para el "TM PART1".

Figura 34 Logo "Table Maneger, Part 1".

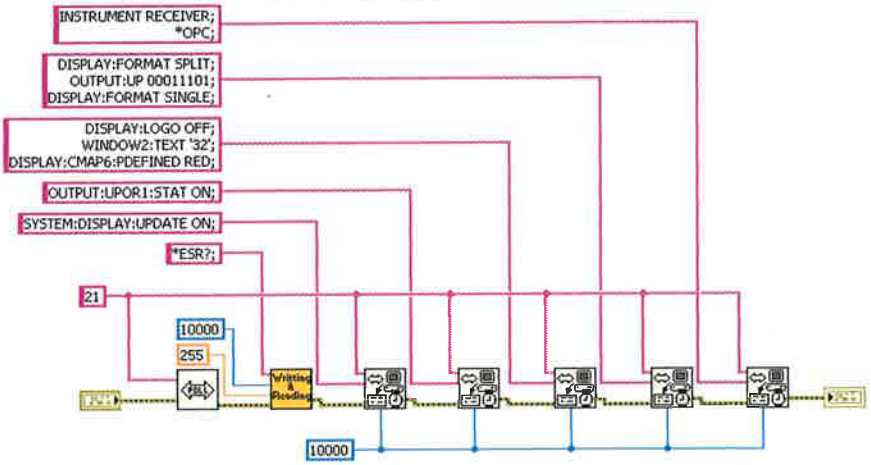
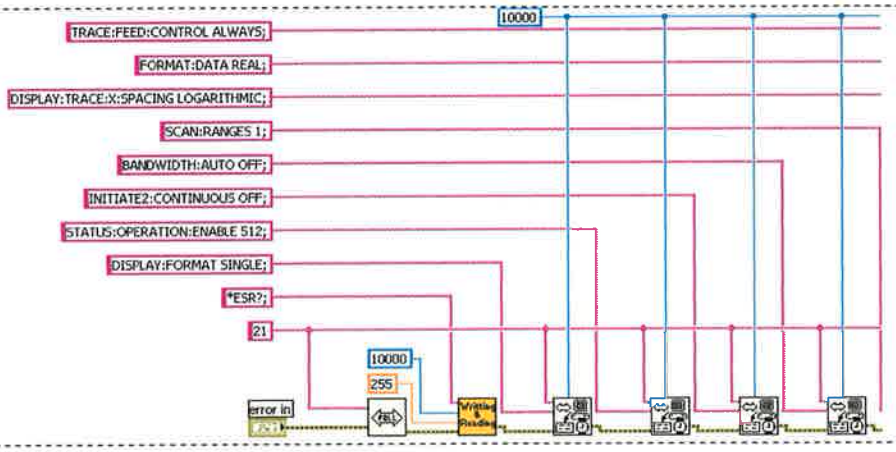


Figura 35 Coplejo de comandos para el "Table Manager, Part 2".

Figura 36 Identificador de "TM, Part2".



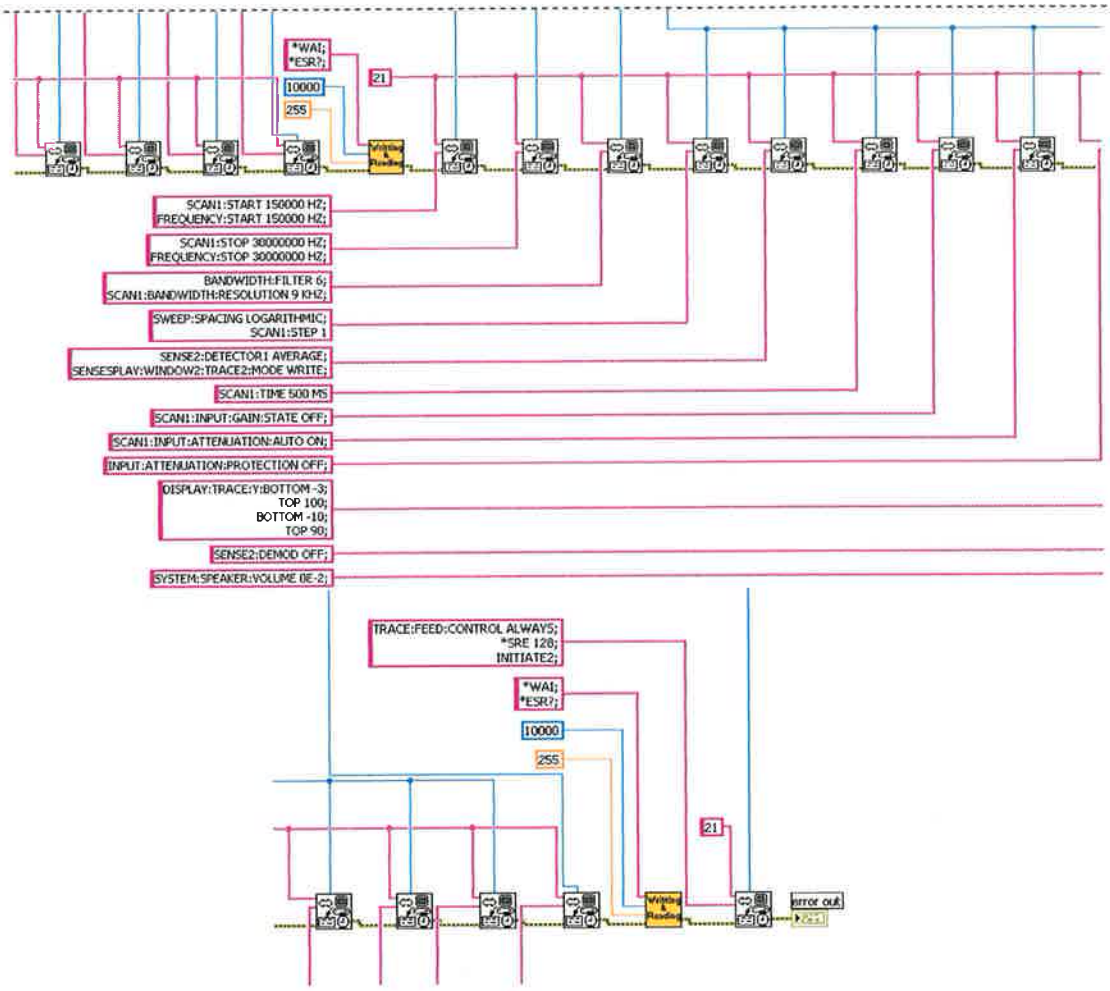


Figura 37 Conjunto de "Table Manager, Part 3" dividido en partes.

Figura 38 Icono identificador de "TM, Part3".

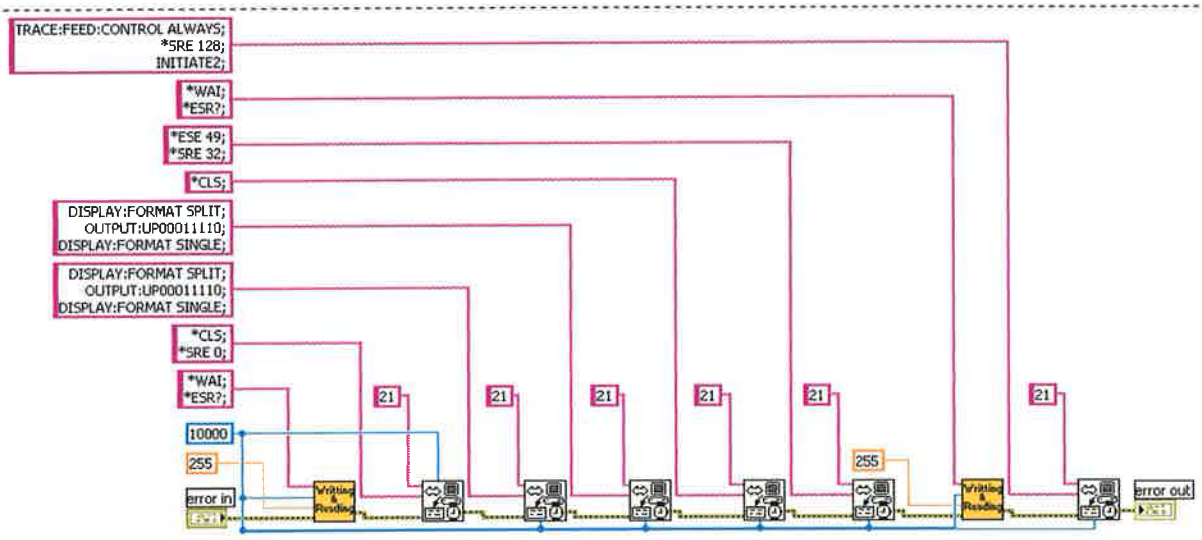


Figura 39 Bloques dedicados al "Table Manager, Part 4".

Figura 40 Identificador "TM, Part 4".

El conjunto de comandos totales, utilizados para las acciones de los "Table Manager" se puede consultar en la tabla 6, para una mejor ilustración de los mismos.

9.6. SUBVI "WAVEFORM PETITION"

Se puede considerar una de las partes del programa, ya que, éste conjunto de comandos son los encargados de la petición de los datos que serán registrados dentro del controlador, ésta rutina se ejecuta constantemente durante un periodo de tiempo, lo que ofrece como resultado un vector o conjunto de datos que pueden ser procesados de la manera que se desee. En la figura número 41 se aprecia el bloque de comandos así como su icono de identificación en la 42.

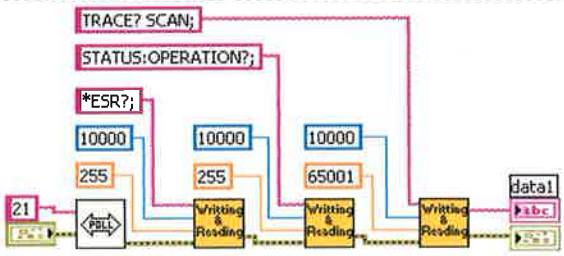


Figura 41 Petición de datos.

BIBLIOGRAFÍA

[1] National Instruments, NI-488.2 Software Reference Manual for MS-DOS. Part number 370889A-01. June 1997 edition.

[2] Lajara Vizcaíno, José Rafael y Pelegrí Sebastiá, José (2007). LabVIEW Entorno gráfico de programación, LabVIEW 8.20 y versiones anteriores. Alfaomega-Marcombo, México.

[3] National Instruments, LabVIEW Tutorial Manual. Part number 320998A-01. January 1996 edition.

[4] http://en.wikipedia.org/wiki/Ascii

[5] National Instruments, NI-488.2 Function Reference Manual for Windows. Part number 370936A-01. February 1999 Edition.

[6] National Instuments, LabVIEW software, V. 8.2.0.1.

[7] http://www.microvolt.com/table.html