Application Note:

AN2008-08-GPIB Rev 1.1

By Sara Nazemzadeh

IEEE-488.2 (GPIB) Software Configuration Application Note

Abstract:

This Application Note describes how to communicate with a Dow-Key CANBus switch using IEEE-488.2 (a.k.a. GPIB) interface. The main focus is on the software aspect of the protocol in terms of syntax and commands. Examples are shown to guide the user in using the syntax and to know the expected results.

The GPIB interface transfers data up to 1 Mbyte/s, which is slower compared to the Ethernet protocol. However, it can be extended to 1.8 Mbyte/s or up to 7.2 Mbyte/s by using, for instance, the **NI-GPIB-USB-B** or **NI-GPIB-USB-HS** -- a backward compatible extension cable provided by National Instruments -- using HS-488 protocol (a.k.a. IEEE-488.1).

Figure 1(a) shows the NI-GPIB-USB-B extended interface with a 26-pin D-sub connector at one end (attach it to the Dow-Key GPIB ribbon cable) and with USB 2.0 connector at the other end (connect it to the PC). Note that his GPIB extension cable is not provided by Dow-Key, but is recommended to use. More information on this can be found on National Instruments website http://www.ni.com or in this document:
http://www.ni.com/pdf/products/us/4gpib675-676.pdf

(a) (b)

(a) NI-GPIB-USB-B GPIB Extension Cable
(b) Full Package Provided by National Instruments

Figure 1: National Instruments NI-GPIB-USB-B GPIB Extension Interface Cable.

SCPI Register Model

Introduction to SCPI

This section describes a minimal register model that is required to be able to perform a safe handshaking between the controller and the instrument. In the instrument, a status system records various conditions and states othe the instrument in 2 registers. Each of the register groups is made up of several low-level registers called Condition Registers, Event Registers, and Enable Registers.

Condition Register

A condition register continuously monitors the state of the instrument. The bits in the condition register are updated in real time and the bits are not latched or buffered. This is a read-only register and bits are not cleared when you read the register. A query of a condition register returns a decimal value which corresponds to the binary-weighted sum of all bit set in that register.

Event Register

An event register latches the various events from the condition register. There is no buffering in this register; while an event bit is set, subsequent events corresponding to that bit are ignored. This is a read-only register. Once a bit is set, it remains set until cleared by a query command (such as *CLS). A query of this register returns a decimal value that corresponds to the binary-weighted sum of all bits in that register.

Enable Register

An enable register defines which bits in the event register will be reported to the Status Byte resister group. You can write to or read from an enable register. A *CLS command will not clear the enable register but it does clear all bits in the event register. To enable bit in the enable register to be reported to the Statue Byte register, you must write a decimal value that corresponds to the binary-weighted sum of the corresponding bits.

Bit Number	Decimal Value	Definitions
0	1	Free for manufacturer to assign
1	2	Free for manufacturer to assign
2	4	Free for manufacturer to assign
3	8	Free for manufacturer to assign
4	16	Data is available in the instruments output buffer
5 Message Available	32	One or more bits are set in the Standard Event Register (bits must be enabled)
6 Standard Event	64	One or more bits are set in the Status Byte Register (bits must be enabled)
7 Master Summary	128	Free for manufacturer to assign

Table 1: Bit definitions – Status Byte Register

The Status Byte Register

The Status Byte register reports conditions from the other registers. Data in the instruments output buffer is immediately reported on the "Message Available" bit (bit 4). Clearing an event register from one of the other registers will clear the corresponding bits in the Status Byte condition register. Reading all messages from the output buffer, including any pending queries, will clear the "Message Available" bit. To set the enable register mask and generate an SRQ (service request), you must write a decimal value to the register using the ***SRE** command. See *Table 1.*

The Standard Event register is cleared when:
- The ***CLS** command is executed.
- A query of the event register using the ***ESR?** Command.

The Standard Event Enable register is cleared when:
- The ***ESE 0** command is executed.

The Standard Event Register

The Standard Event Register reports different types of events that may occur in the instrument. Any or all of these conditions can be reported to the Standard Event summary bit through the enable register. To set the enable register mask, you must write a decimal value to the register using the ***ESE** command. See *Table 2.*

The Standard event register is cleared when:
- The ***CLS** command is executed.
- A query of the event register using the ***ESR?** Command.

The Standard Event enable register is cleared when:
- The ***ESE 0** command is executed.

Bit Number	Decimal Value	Definitions
0 Operation Complete	1	All commands prior to and including *OPC have been executed.
1	2	Free for manufacturer to assign
2 Query Error	4	The instrument tried to read the output buffer but it was empty. Or a new command line was received before a previous query has been read. Or bothe the input and output buffers are full.
3 Device Error	8	A self-test or calibration error occurred.
4 Execution Error	16	An execution error occurred.
5 Command Error	32	A command syntax error occurred.
6	64	Free for manufacturer to assign
7	128	Free for manufacturer to assign

Table 2: Bit definitions – Standard Event Register

SCPI Command Set

GPIB Translator Interface

Each GPIB message can be no longer than 170 characters. A message sent that exceeds 170 characters before EOI is detected, will be discarded.

A maximum of 6 commands can be sent in any single GPIB message as shown below:

Examples:

“**:SWIT1?**; **SWIT2?**; **SWIT3?**; **SWIT4?**; **SWIT5?**; **SWIT6?**””

This represents 6 commands in one GPIB message.

GPIB Address

GPIB address is set to 9 at the factory. This address can be changed to any value between 1 and 30. See the *System Command* section.

Command syntax structure

The format used to show command syntax is shown below:

[ROUTe**]:SWIT**ch**[**<id>**]:[VAL**ue**]**<number>

- Square brackets [ ] indicate optional keywords or parameters.
- Braces { } enclose parameter choices within a command string.
- Triangle brackets < > enclose parameters for which you must substitute a value.
- Vertical bar | separates multiple parameter choices.

The command syntax shows most commands as a mixture of upper and lower case letters. The upper case letters indicate the abbreviated spelling for the command. For shorter program lines, the abbreviated form is used. For better program readability, the long form is used. For example, in the above syntax statement, **ROUT** and **ROUTE** are both acceptable forms. Since both upper and/or lower case letters are acceptable, **ROUTE**, **rout** and **Rout** are all acceptable. Other forms, such as RO and ROU are not acceptable and will generate an error.

Command Separators

- A colon (:) is used to separate a command keyword from a lower level keyword.
- A blank space is used to separate a parameter from a command keyword.
- A comma (,) is used if a command requires more than one parameter.
- A semicolon (;) is used to combine commands from the same subsystem into one message string.
- A colon and a semicolon is used to link commands from different subsystems.

SCPI command terminators

The IEEE-488 EOI (end-or-identify) message must be sent with the last character of a command string. A command string sent to the instrument may be terminated with a <line feed> (“/n”) character.

“***IDN?”** – this will work if EOI is sent with the “?”
“***IDN?/n**” – this will work if EOI is sent with the “/n”

SCPI Command Set

Introduction

The SCPI language defines several different data formats to be used in program message and response messages.

Common Commands

Introduction

All SCPI instruments must implement common commands that the IEEE-488.2 standard defines. The following contains descriptions of a subset of these commands. See *Table 3* for list of commands.

**CLS*

Syntax
*CLS

Description
This command is used to clear the event register in all register groups

**ESE*

Syntax
*ESE <value>

Parameters

value	Decimal value which corresponds to the binary-weighted sum of the bits you wish to enable the register.

Description
Enable bits in the Standard Event Status Enable register.

**ESE?*

Syntax
*ESE?

Description
This query allows the user to determine the current contents of Standard Event Status Enable register.
The value returned corresponds to the binary-weighted sum of all bits enabled by the ***ESE** command.

**ESR?*

Syntax
*ESR?

Description
This query allows the user to determine the current contents of Event Status register. Reading the Event Status register clears it.
The status is returned as a decimal value which corresponds to the binary-weighted sum of all bits set in the register.

**IDN?*

Syntax
*IDN?

Result
A string is returned which consists of the following parts:

Mnemonic	Name
*CLS	Clear Status Command
*ESE	Standard Event Status Enable Command
*ESE?	Standard Event Status Enable Query
*ESR?	Standard Event Status Register Query
*IDN?	Identification Query
*OPC	Operation Complete Command
*OPC?	Operation Complete Query
*RST	Reset Command
*SRE	Service Request Enable Command
*SRE?	Service Request Enable Query
*STB	Read Status Byte Query
*WAI	Wait-to-Continue Command

Table 3: Common Commands

Vendor,model,serial number,firware revision

vendor	Name of the vendor
description	Instrument description
serial number	Serial number of the instrument (3-digit number)
firmware revision	Revision state of the instrument firmware

Example
"*IDN?"

Result
"DOW-KEY,AUTOCONFIG,101,R8"

**OPC*

Syntax
*OPC

Description
This command causes the device to generate the operation complete message in the Event Status register when all pending operations have been finished.

**OPC?*

Syntax
*OPC?

Description
This query returns an ASCII character "1" when all pending operations have been finished.

Result
ASCII character "1".

Example
"SYST:PRE;*OPC?"

Result
"1"

**RST*

Syntax
*RST

Description
This command performs a device reset.
This will set the instrument to a default factory configuration

**STB?*

Syntax
*STB?

Description
Query the Status Byte register:
This command is similar to a Serial Poll but is processed like any other instrument command. This command retruns the same result as a Serial Poll but the Master Summary bit is not cleared if a Serial Poll has occurred.

Result

stb	Decimal value which corresponds to the binary-weighted sum of all bits set in the register.

**SRE*

Syntax
*SRE <*enable_value*>

Parameters

Enable_value	Value that corresponds to the binary-weighted sum of the bits you wish to enable in the register.

Description
Enable bits in the Status Byte Enable register.
To enable bits in the Status Byte Enable register, you must write a decimal value that corresponds to the binary-weighted sum of the bits you whish to enable in the register.

**SRE?*

Syntax
*SRE?

Description
The ***SRE?** query returns a decimal value which corresponds to the binary-weighted sum of all bits enabled by the ***SRE** command.

Result
Returns a decimal value which corresponds to the binary-weighted sum of all bits enabled by the ***SRE** command.

Example
"*SRE?"

Result
"16"

*WAI

Syntax
*WAI

Description
This command prevents the instrument from executing any further commands or queries until the current command has been processed.

System Commands

System Commands

The following contains the system commands of SCPI that the GPIB control is compatible with.

SYST:ERR?

Syntax
SYSTem:ERRor?

Description
Query the instrument's error queue:
A record of up to N errors is stored in the instrument's error queue. Errors are retrieved in first-in first-out (FIFO) order. The first error returned is the first error that was stored. Each additional error up to N is read by N subsequent queries (one for each error). For this translator board, the N=10. The error queue has to be read until no more errors are returned, otherwise the error status is not cleared.

Result
String with the following syntax:
code,message
code Numeric value with the error code (0 if no error).
message string with error message.

Example
"**SYST:ERR?**"

Result was "-1,"INVALID CHARACTER"", check for more errors.

*"***SYST:ERR?***"*
Result was "-2"INPUT BUFFER OVERFLOW"", check for more errors.

*"***SYST:ERR?***"*
Result was "-3"TOO MANY COMMANDS"", check for more errors.

*"***SYST:ERR?***"*
Result was "-4"SYNTAX ERROR"", check for more errors.

*"***SYST:ERR?***"*
Result was "-5"DATA OUT OF RANGE"", check for more errors.

*"***SYST:ERR?***"*
Result was "-6"ILLEGAL PARAMETER VALUE"", check for more errors.

*"***SYST:ERR?***"*
Result was "-7"INPUT BUFFER UNDERFLOW"", check for more errors.

*"***SYST:ERR?***"*
Result was "-8"MATRIX SOCKET NOT AVAIL"", check for more errors.

*"***SYST:ERR?***"*
Result was "-0"NO ERROR"", No more errors, error queue is empty.

Explanation of Error Queue

"-1"INVALID CHARACTER""
Invalid character detected. Would be caused by "ROUTE:SWITCH2 Y". The software is expecting a switch position number and you have entered a letter instead. Hence 'Y' is an invalid character. Solution: check your syntax in the offending message.

"-2"INPUT BUFFER OVERFLOW""
The output buffer can store up to 8 messages. If there ends up being more than 8 messages in the output buffer, this error will occur. Solution: always read back messages after you ask the matrix for a reply.

"-3"TOO MANY COMMANDS""
More than 8 separate stacked commands were sent in the same single message string. Solution: send less than 8 at a time.

"-4"SYNTAX ERROR""
Would be caused by "RUOTE:SWITCH2 4". Solution: check for spelling.

"-5"DATA OUT OF RANGE""
Would be caused by trying to set a switch outside of its parameters. If you tried to set a 6 position switch to position 7 this would occur. Solution: check the switch parameters.

“-6”ILLEGAL PARAMETER VALUE””
This message is set if the GPIB address or MAC address is an invalid one. Sending SYSTEM:GPIBADDRESS 32 would generate this message since numbers between 1-30 are accepted for a valid GPIB address. Solution: check your syntax.

“-7”INPUT BUFFER UNDERFLOW””
This happens if you try to read out a message from the matrix and there is no message there to be read. Solution: always do a write than read.

“-8”MATRIX SOCKET NOT AVAIL””
This occurs when you open a TCP socket, then do not properly close it and then try to open a new TCP socket. Only 1 connection is allowed at a time. Solution: ensure that all old TCP sockets are closed properly before proceeding.

SYST:GPIBADDRESS?

Syntax
SYSTem:GPIBADDRESS?

Description
Reads the GPIB address. This command returns the current GPIB address of the instrument.

Result
Current GPIB address string

Example
“SYSTem:GPIBADDRESS?”

Result
“9”

SYST:GPIBADDRESS n

Syntax
SYSTem:GPIBADDRESS n

Description
Allows the user to set the instruments GPIB address. Valid addresses are 1-31.

Example
“SYSTem:GPIBADDRESS 12”

Result
GPIB address is set to 12

Switch Command Set

Switch [Module] Command Set

The following contains the switch [module] commands of SCPI which the GPIB control is compatible with.

SWITch[<id>]:[VALue]

Syntax
[ROUTe**]:SWIT**ch**[**<id>**]:[VAL**ue**]** <number>

Description
This command is used to control the position of the switches. The switch specified by the numeric suffix <id> is set to position <number>. Switch positions are specified in a 0 to N fashion, therefore legal values for <number> are from 0 (open state) to the maximum number of position for the switch. For example, a SP8T switch has 8 positions, 1 thru 8.

Setting switch x to position n

x = switch address (can be a number from 1-255).

n = position to set and must be within the switches parameter.

(Example: SP8T valid positions are 0 thru 8 only).

Examples:
- **ROUTE:SWITCH**x n
- **ROUT:SWITCH**x n
- **ROUTE:SWIT**x n
- **ROUT:SWIT**x n
- **:SWITCH**x n
- **:SWIT**x n
- **ROUTE:SWITCH**x**:VALUE** n
- **ROUTE:SWITCH**x**:VAL** n
- :**SWIT**x**:VAL** n

- ***ROUTE:SWITCH5 4***
  (Sets switch 5 to position 4)
- ***ROUTE:SWITCH5 0***
  (Sets switch 5 to position 0 or open state)

Requesting Switch x current position

x = switch address (can be a number from 1-255).

Example

- **ROUTE:SWITCH**x**?**
- **ROUT:SWIT**x**?**
- **:SWIT**x**?**

- **ROUTE:SWITCH5?**
  (Requests the current position of switch 5)

Technical Support

If the need of technical support is required, please contact Dow-Key Microwave Corporation.

Toll Free: (800) 266-3695
Local: (805) 650-0260

Dow-Key® Microwave
CORPORATION
A DOVER COMPANY