ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Find the *National Instruments GPIB-422CT* at our website: *Click HERE*

GPIB-422CT
User Manual

MicroGPIB®
RS-422 to IEEE-488 Controller

March 1991 Edition

Part Number 320115-01

© Copyright 1988, 1992 National Instruments Corporation.
All Rights Reserved.

National Instruments Corporate Headquarters
6504 Bridge Point Parkway
Austin, TX 78730-5039
(512) 794-0100
(800) IEEE-488 (toll-free U.S. and Canada)
Technical support fax: (512) 794-5678

Branch Offices:
Australia 03 879 9422, Belgium 02 757 00 20, Canada 519 622 9310,
Denmark 45 76 73 22, France 1 48 65 33 70, Germany 089 714 50 93,
Italy 02 48301892, Japan 03 3788 1921, Netherlands 01720 45761,
Norway 03 846866, Spain 91 896 0675, Sweden 08 984970,
Switzerland 056 45 58 80, U.K. 0635 523545

Contents

Chapter 1
Description of the GPIB-422CT 1-1
Introduction 1-1
Special Note for Macintosh Users 1-1
What Your Kit Contains 1-2
Optional Equipment 1-3
GPIB-422CT Specifications 1-4
The GPIB-422CT Front Panel 1-5
The GPIB-422CT Rear Panel 1-7
The RS-422 Port 1-8
The GPIB Port 1-9

Chapter 2
The GPIB-422CT Modes of Operation 2-1
Introduction 2-1
Choosing Between S Mode or G Mode 2-1

Chapter 3
Installation and Configuration in S Mode 3-1
Installation 3-1
Step 1: Inspection 3-1
Step 2: Verify the Voltage Requirement 3-2
Step 3: Configure the Operating Parameters 3-2
Set Configuration Switches 3-3
Default Settings for Configuration Switch 3-5
Step 4: Connect the Cables 3-6
Step 5: Power on the Unit 3-6

Chapter 4
Programming in S Mode 4-1
Programming Messages 4-1
Programming Message Format 4-1
Example of a Programming Message 4-1
Example of a Programming Message with Data String 4-2
How Messages are Processed 4-2
Function Names 4-2
Function Argument Delimiters 4-3
Abbreviations for Arguments 4-3

GPIB Address 4-3
Numeric String Argument 4-4
Status Information 4-4
Serial Port Error Handling 4-5
GPIB Read and Write Termination Method (END and EOS) ... 4-5
S Mode - Default Settings 4-6
List of S Mode Functions by Group 4-7
GPIB Functions 4-7
Serial Port Functions 4-10
General Use Functions 4-10
Alphabetical List of S Mode Functions 4-11

Chapter 5
S Mode Functions 5-1
Points to Remember 5-1
cac - Become Active Controller 5-3
caddr - Change the GPIB Address of the GPIB-422CT 5-4
clr - Clear Specified Device 5-6
cmd - Send GPIB Commands 5-8
echo - Echo Characters Received from Serial Port 5-11
eos - Change/Disable GPIB EOS Termination Mode 5-12
eot - Enable/Disable END Message on GPIB Writes 5-15
gts - Go from Active Controller to Standby 5-17
ibcl - Enter ibcl Operating System 5-19
id - Identify System 5-21
ist - Set or Clear Individual Status Bit 5-22
loc - Go to Local 5-23
onl - Place the GPIB-422CT Online/Offline 5-25
pct - Pass Control 5-26
ppc - Parallel Poll Configure 5-27
ppu - Parallel Poll Unconfigure 5-29
rd - Read Data 5-31
rpp - Request (Conduct) a Parallel Poll 5-34
rsc - Request System Control 5-36
rsp - Request (Conduct) a Serial Poll 5-38
rsv - Request Service/Set or Change Serial Poll Status Byte 5-41
sic - Send Interface Clear 5-42
spign - Ignore Serial Port Errors 5-44
sre - Set (or clear) Remote Enable 5-46
stat - Return GPIB-422CT Status 5-47
tmo - Change or Disable Time Limit 5-53
trg - Trigger Selected Device(s) 5-55
wait - Wait for Selected Event 5-57

wrt - Write Data 5-60
xon - Change Serial Port XON/XOFF Protocol 5-63

Chapter 6
Installation and Configuration in G Mode 6-1
Installation 6-1
Step 1: Inspection 6-1
Step 2: Verify the Voltage Requirement 6-2
Step 3: Configure the Operating Parameters 6-2
Set Configuration Switches 6-3
Example Settings for Configuration Switch 6-6
Step 4: Connect the Cables 6-7
Step 5: Power on the Unit 6-7

Chapter 7
Programming in G Mode 7-1
Programming Messages 7-1
Programming Message Format 7-1
Example of a Programming Message 7-1
How Messages are Processed 7-2
Function Names 7-2
Function Argument Delimiters 7-2
Abbreviation for Argument 7-2
Status Information 7-3
Communicating with the GPIB-422CT and the Serial Device 7-3
Address of the GPIB-422CT 7-3
Address of the Serial Device 7-3
Addressing Terminology 7-3
The GPIB-422CT and Serial Device as Listener 7-4
The GPIB-422CT and Serial Device as Talker 7-5
GPIB Read and Write Termination (END and EOS) 7-6
Serial Port Transmission 7-6
G Mode - Default Settings 7-7
List of G Mode Functions by Group 7-8
GPIB Configuration Functions 7-8
Serial Port Configuration Functions 7-9
General Use Functions 7-9
List of G Mode Functions in Alphabetical Order 7-10
Operation of the GPIB-422CT with a Serial Device 7-10
Serial Poll 7-11
Service Request 7-11
Parallel Poll 7-12

Trigger 7-12
Local 7-12
Control 7-12
Device Clear 7-13

Chapter 8
G Mode Functions 8-1
Points to Remember 8-1
echo - Echo Characters Received from Serial Port 8-3
eos - Change/Disable GPIB EOS Termination Mode 8-5
id - Identify System 8-7
onl - Place the GPIB-422CT Online/Offline 8-8
spign - Ignore Serial Port Errors 8-9
spset - Change Serial Port Parameters 8-10
srqen - Enable/Disable Setting of SRQ 8-12
stat - Return GPIB-422CT Status 8-14
xon - Change Serial Port XON/XOFF Protocol 8-20

Appendix A
Multiline Interface Messages A-1
Multiline Interface Messages A-2
Interface Message Reference List A-4

Appendix B
Status Information B-1
Status Bits B-1
GPIB Error Codes B-6
Serial Port Error Codes B-10

Appendix C
The Serial Connection C-1
RS-422 ..... C-1
Special Note for Macintosh Users C-2
Buffering and Handshaking C-3
Hardware Handshake C-4
XON/XOFF C-4

Appendix D
Operation of the GPIB D-1
Types of Messages D-1
Talkers, Listeners, and Controllers D-1
The Controller-In-Charge and System Controller D-3
GPIB Signals and Lines D-3

Data Lines D-4
Handshake Lines D-5
NRFD* (not ready for data) D-5
NDAC* (not data accepted) D-5
DAV* (data valid) D-5
Interface Management Lines D-5
ATN* (attention) D-5
IFC* (interface clear) D-6
REN* (remote enable) D-6
SRQ* (service request D-6
EOI* (end or identify) D-6
Physical and Electrical Characteristics D-6
Configuration Restrictions D-9
Related Document D-9

Appendix E
Common Questions E-1
S Mode E-1
G Mode E-3

Appendix F
Parallel Polling F-1
Operation F-1
Configuration F-2
The Parallel Poll F-3
Disabling Parallel Poll Response F-3
S Mode Example F-3

Appendix G
Setting Switches G-1

Appendix H
Sample Programs H-1
S Mode Sample Programs H-1
Using an HP 7475A Plotter with a Terminal H-2
Getting Ready to Program H-2
Programming Steps H-3
Step 1. stat Function H-3
Step 2. Serial Port Functions H-3
Step 3. GPIB Initialization Functions H-4
Step 4. Communicate with
rd and wrt Functions H-4
Using an HP 7475A Plotter with a Macintosh H-5

Getting Ready to Program H-5
Programming Steps H-6
Step 1. stat Function H-6
Step 2. Serial Port Functions H-6
Step 3. GPIB Initialization Functions H-6
Step 4. Communicate with rd and wrt Functions H-7
Programming a Tektronix 2445 Oscilloscope from an Apple IIc H-8
Getting Ready to Program H-8
Programming Steps H-9
Step 1. stat Function H-9
Step 2. Serial Port Functions H-9
Step 3. GPIB Initialization Functions H-9
Step 4. Communicate with rd and wrt Functions H-9
G Mode Sample Programs H-10
Using a Serial HP 7475A Plotter with a GPIB Controller H-11
Getting Ready to Program H-11
Programming Steps H-12
Step 1. stat Function H-12
Step 2. GPIB Initialization Functions H-12
Step 3. Serial Port Initialization Functions H-12
Step 4. Communicate with Plotter H-13
IBM PC (with GPIB-PC) to Serial Printer H-14
Getting Ready to Program H-14
Programming Steps H-15
Step 1. stat Function H-15
Step 2. GPIB Initialization Functions H-15
Step 3. Serial Port Initialization Functions H-15
Step 4. Communicate with the Printer H-15

Index I-1

Figures

Figure 1-1. The GPIB-422CT 1-1
Figure 1-2. The GPIB-422CT Front Panel 1-5
Figure 1-3. The GPIB-422CT Rear Panel 1-7
Figure 1-4. The RS-422 Connector and Signal Designations 1-8
Figure 1-5. The GPIB Connector and Signal Designations 1-9

Figure 2-1. Personal Computer Controlling a GPIB Plotter 2-2
Figure 2-2. A Serial Printer Connected to a GPIB Controller 2-2

Figure 3-1. Standard RS-422 Configuration (Factory Setting) 3-2
Figure 3-2. Macintosh Serial Port Configuration 3-3
Figure 3-3. S Mode Switch Settings 3-3
Figure 3-4. Factory Default Settings for Switch U22 3-5

Figure 6-1. Standard RS-422 Configuration (Factory Setting) 6-2
Figure 6-2. Macintosh Serial Port Configuration 6-3
Figure 6-3. G Mode Switch Settings 6-3
Figure 6-4. Example Settings for Switch U22 6-6

Figure D-1. GPIB Cable Connector D-4
Figure D-2. Linear Configuration of the GPIB Devices D-7
Figure D-3. Star Configuration of GPIB Devices D-8

Figure G-1. S mode, 7 data bits, 1 stop bit, even parity, 300 baud G-1
Figure G-2. S mode, 8 data bits, 1 stop bit, even parity, 19200 baud. G-2
Figure G-3. S mode, 8 data bits, 1 stop bit, no parity, 9600 baud G-3

Figure H-1. S mode, 8 data bits, 1 stop bit, no parity, 19200 baud H-2
Figure H-2. S mode, 7 data bits, 1 stop bit, no parity, 9600 baud H-5
Figure H-3. S mode, 8 data bits, 1 stop bit, no parity, 9600 baud H-8
Figure H-4. G mode, primary GPIB address 2 ..... H-11
Figure H-5. G mode, primary GPIB address 18 ..... H-14

Tables

Table 1-1. Electrical Characteristics 1-4
Table 1-2. Environmental Characteristics 1-4
Table 1-3. Physical Characteristics 1-5
Table 1-4. LED Descriptions 1-6

Table 3-1. Configuration Parameters for Switches 1 through 5 3-4
Table 3-2. Configuration Parameters for Switches 6 through 8 3-5

Table 4-1. Serial Port Characteristics 4-6
Table 4-2. GPIB Characteristics 4-6
Table 4-3. I/O Functions 4-7
Table 4-4. Bus Management Functions 4-7
Table 4-5. GPIB Initialization Functions 4-8
Table 4-6. Serial Poll Functions 4-8
Table 4-7. Low-Level Controller Functions 4-9
Table 4-8. Parallel Poll Functions 4-9
Table 4-9. Serial Port Initialization Functions 4-10
Table 4-10. General Use Functions 4-10
Table 4-11. GPIB-422CT Functions 4-11

Table 5-1. Data Transfer Termination Methods 5-12
Table 5-2. GPIB Status Conditions 5-48
Table 5-3. GPIB Error Conditions 5-49
Table 5-4. Serial Port Error Conditions 5-50
Table 5-5. Wait Mask Values 5-58

Table 6-1. Primary Address Configurations 6-4

Table 7-1. GPIB Characteristics 7-7
Table 7-2. Serial Port Characteristics 7-7
Table 7-3. GPIB Configuration Functions 7-8
Table 7-4. Serial Port Configuration Functions 7-9
Table 7-5. General Use Functions 7-9
Table 7-6. GPIB-422CT Functions 7-10
Table 7-7. Serial Poll Response Byte 7-11

Table 8-1. Data Transfer Termination Methods 8-5
Table 8-2. SRQ Mask Bits 8-12
Table 8-3. GPIB-422CT Status Conditions 8-15
Table 8-4. GPIB Error Conditions 8-16
Table 8-5. Serial Error Conditions 8-17

Table C-1. RS-422 Serial Port Pinouts..... C-2

Chapter 1
Description of the GPIB-422CT

Introduction

The GPIB-422CT, shown in Figure 1-1, is a high-performance serial-to-GPIB interface. It provides a computer with an RS-422 port, a means of controlling, talking, and listening on the GPIB. The GPIB-422CT is also capable of interfacing RS-422 instruments and peripherals to the GPIB.

The GPIB-422CT has all the software and logic required to implement the physical and electrical specifications of the IEEE-488 and the RS-422 standards. It is capable of both interpreting and executing high-level commands that you send to it over the serial port, and performing GPIB-to-RS-422 protocol conversion.

Special Note for Macintosh Users

The Macintosh serial port has been designed to be compatible with both the RS-422 and the RS-232 serial ports. The GPIB-422CT is fully compatible with the Macintosh serial port. For more information about the use of the GPIB-422CT with a Macintosh, refer to the sections entitled *The RS-422 Port*, in this chapter, and *Configure the Operating Parameters* in Chapter 3.



Figure 1-1. The GPIB-422CT

What Your Kit Contains

Your kit should contain the following components:

Component	Part Number
One of the following boxes:	
• GPIB-422CT (64K RAM - 115 VAC)	776176-01
• GPIB-422CT (256K RAM - 115 VAC)	776176-02
• GPIB-422CT (64K RAM - 230 VAC)	776176-31
• GPIB-422CT (256K RAM - 230 VAC)	776176-32
GPIB-422CT User Manual	320115-01
GPIB-CT IBCL Reference Manual	320132-01

Optional Equipment

Component	Part Number
Rack-Mount Kit:	
Single (1 unit)	180480-01
Dual (2 units)	180480-02
RS-422 Shielded Cables - 2 m	
9-pin DB to p-pin DB (use with Mac)	180269-01
9-pin DB to circular mini-8 (use with Mac Plus, Mac SE, Mac II)	181459-04
Double-Shielded GPIB Cables:	
GPIB Type X2 Cable – 1 m	763061-01
GPIB Type X2 Cable – 2 m	763061-02
GPIB Type X2 Cable – 4 m	763061-03

GPIB-422CT Specifications

Tables 1-1 through 1-3 specify the electrical, environmental, and physical characteristics of the GPIB-422CT.

Table 1-1. Electrical Characteristics

Characteristic	Specification
Power Supply Unit	Wall mount type, 115 VAC or 230 VAC, 50/60 Hz input, 9 VDC @ 1A max output
Voltage	9 VDC unregulated
Current	640 mA typical; 750 mA max

Table 1-2. Environmental Characteristics

Characteristic	Specification
Operating Temperature	0° to 50° C
Storage Temperature	0° to 70° C
Relative Humidity	10% to 95% noncondensing conditions
Noise Emissions	FCC Class A Verified

Table 1-3. Physical Characteristics

Characteristic	Specification
Case Size	1.6 in. by 5.7 in. by 8.4 in. (40.6 mm by 144.8 mm by 213.4 mm)
Case Material	All metal enclosure
Rack Mounting	Single or dual kits available
Weight	28 oz. (without power supply unit)

The GPIB-422CT Front Panel

The front panel of the GPIB-422CT is shown in Figure 1-2. The power switch and six LEDs are mounted on the GPIB-422CT front panel.



Figure 1-2. The GPIB-422CT Front Panel

The LEDs show the current status of the GPIB-422CT at all times. Table 1-4 describes each LED.

Table 1-4. LED Descriptions

LED	Indication
POWER	Indicates that power to the unit has been applied and the ON/OFF switch is in the ON position.
READY	Indicates that the power-on self-test has passed successfully and the unit is ready to operate.
TALK	Indicates that the GPIB-422CT is configured as a GPIB Talker.
LISTEN	Indicates that the GPIB-422CT is configured as a GPIB Listener.
SRQ	Indicates that the GPIB signal line SRQ* is asserted (low).
ATN	Indicates that the GPIB signal line ATN* is asserted (low).

Note: * indicates that the signal is active low (negative logic).

The GPIB-422CT Rear Panel

The rear panel of the GPIB-422CT is shown in Figure 1-3. The power cable, RS-422 cable, and GPIB cable are shown connected to the rear panel of the GPIB-422CT.

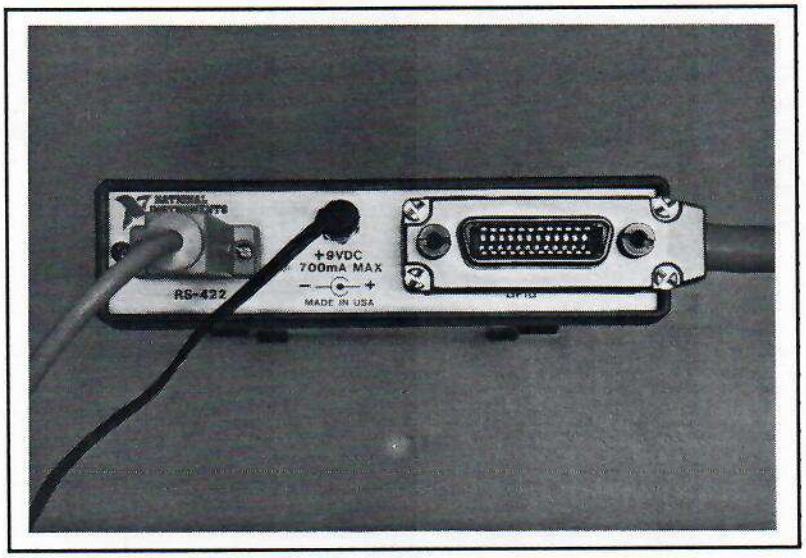


Figure 1-3. The GPIB-422CT Rear Panel

The RS-422 Port

The RS-422 port on the GPIB-422CT is configured to support the Macintosh implementation of the RS-422 as well as the standard RS-422 electrical specification. The positions of jumpers across pins W-4 through W-6 determine whether the RS-422 port is Macintosh-compatible or standard RS-422. For specific jumper settings, see the section entitled *Configure the Operating Parameters* in Chapter 3.

The RS-422 port on the GPIB-422CV uses a standard 9-pin DB-subminiature male connector with screwlock assemblies and will accept a standard 9-pin DB-subminiature female connector. A diagram of the serial connector and the signals supported is shown in Figure 1-4. For more information on the RS-422 signals, refer to Appendix C, *The Serial Connection.*

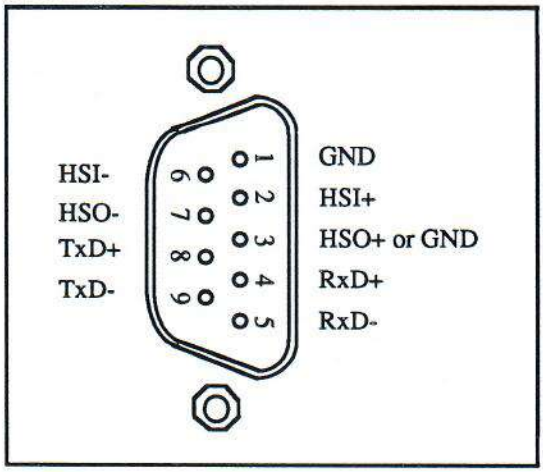


Figure 1-4. The RS-422 Connector and Signal Designations

The GPIB Port

The GPIB connector is a standard 24-pin shielded AMP Champ female connector with metric screwlock hardware. A diagram of the GPIB connector and the signals supported is shown in Figure 1-5 (a * suffix indicates that the signal is active low).

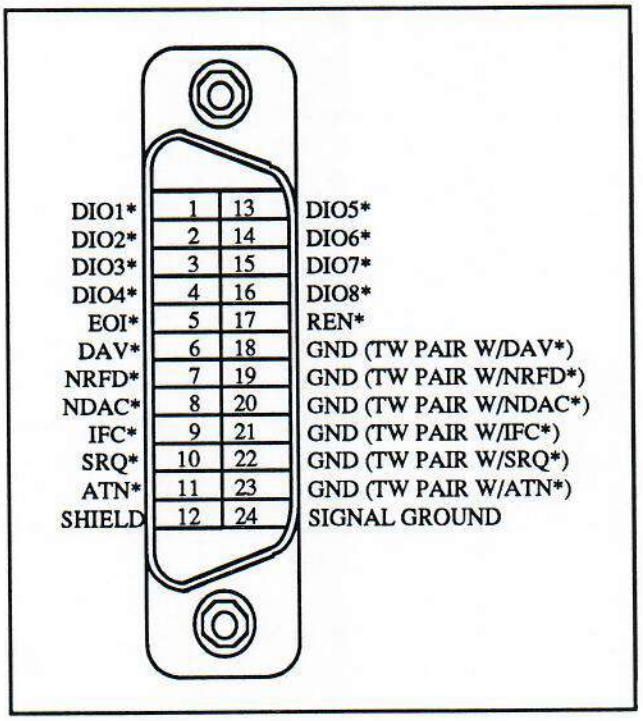


Figure 1-5. The GPIB Connector and Signal Designations

Chapter 2
The GPIB-422CT Modes of Operation

Introduction

The GPIB-422CT is capable of operating in one of two modes: *S mode* or *G mode*. This chapter helps you determine the mode you will use and the chapters of the manual that apply to that mode.

Choosing Between S Mode or G Mode

You will be connecting the GPIB-422CT to both a serial device and a GPIB device. The GPIB-422CT'S mode of operation is determined by the role the serial device plays in your setup.

Is the serial device a Controller? Is it going to manage the GPIB, address devices, and perform other GPIB Controller functions? If so, use the GPIB-422CT in the S mode.

Or, is the serial device going to be only a Talker and/or Listener, where a GPIB device(s) will manage the bus and send and receive data to and from the serial device? If so, use the GPIB-422CT in the G mode.

Figure 2-1 shows the GPIB-422CT operating in the S mode where a personal computer is controlling a GPIB plotter.

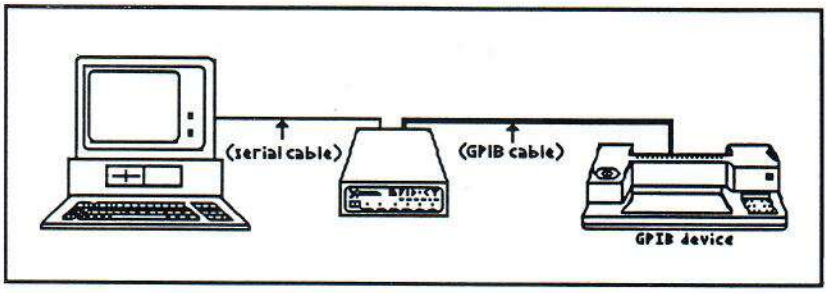


Figure 2-1. Personal Computer Controlling a GPIB Plotter

Figure 2-2 shows the GPIB-422CT operating in the G mode where the GPIB-422CT enables a serial printer to be programmed from a GPIB Controller.

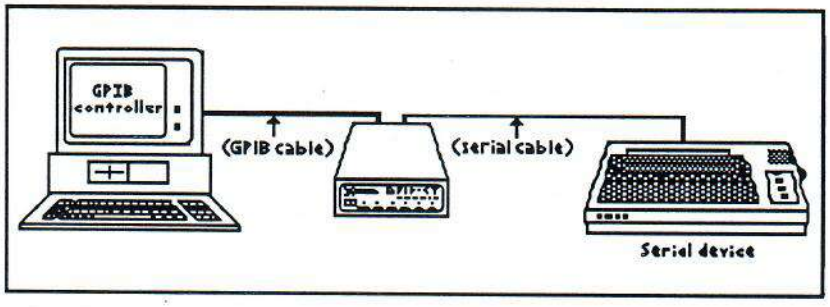


Figure 2-2. A Serial Printer Connected to a GPIB Controller

Now, turn to Chapter 3 for S mode installation and configuration, then to Chapters 4 and 5 to begin programming your GPIB-422CT in S mode.

Or, turn to Chapter 6 for G mode installation and configuration, then to Chapters 7 and 8 to begin programming your GPIB-422CT in G mode.

Chapter 3
Installation and Configuration in S Mode

If you plan to operate in S mode, use this chapter to install and configure the GPIB-422CT. Then read Chapters 4 and 5 to learn about its programming messages.

Installation

There are five basic steps to installing the GPIB-422CT.

1. Inspect the GPIB-422CT for damage that may have been caused in shipment.
2. Verify the voltage requirement.
3. Configure the operating parameters.
4. Connect the cables.
5. Power on the unit.

These steps are described in more detail in the following sections.

Step 1: Inspection

Before you install the GPIB-422CT, inspect the shipping container and its contents for damage. If damage appears to have been caused in shipment, file a claim with the carrier. Retain the packing material for possible inspection and/or for reshipment.

If the equipment appears to be damaged, do not attempt to operate it. Contact National Instruments for instructions.

Step 2: Verify the Voltage Requirement

The GPIB-422CT is shipped from the factory with either a 115V or 230V wall-mount supply. Verify that the voltage on the supply matches the voltage that is supplied in your area.

Caution: Operating the unit at any voltage other than the one specified could damage the unit.

Step 3: Configure the Operating Parameters

The GPIB-422CT is shipped from the factory configured to operate in S mode. The serial port is configured at 9600 baud, 1 stop bit, no parity, and 8 data bits. If you wish to change any of the GPIB-422CT parameters, you must open the unit and set the configuration switches. To change the configuration switches follow these steps:

1. Disconnect power to the unit and disconnect any cables that may be connected to the unit.

2. Unscrew the two screws on the opposite sides of the rear panel.

3. Grab the rear panel bezel and pull it straight away from the rest of the unit. The board should slide out the back of the enclosure.

4. To change the electrical specifications of the RS-422 port so that it is compatible with the Apple Macintosh serial port, locate the three jumpers at location W4 through W6. Move all three jumpers so that they rest across the pins labeled 1 and 2 (see Figures 3-1 and 3-2).

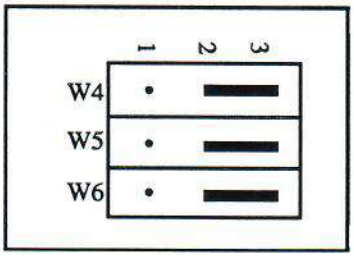


Figure 3-1. Standard RS-422 Configuration (Factory Setting)

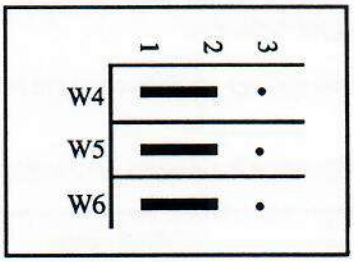


Figure 3-2. Macintosh Serial Port Configuration

5. Locate the configuration DIP switches (U20 and U22) on the printed wire board.

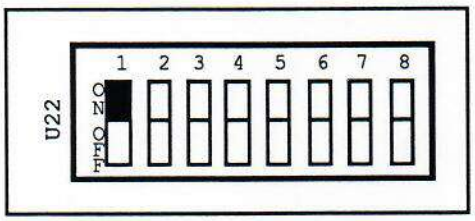
6. Set the switches for the desired mode of operation. Refer to the following section, *Set Configuration Switches.*

   Caution: Most of the circuitry in the GPIB-422CT uses advanced CMOS technology and can be damaged by static electricity. Avoid touching any of the components and take any necessary CMOS handling precautions.

7. Close the unit and re-insert the screws removed in Step 2.

Set Configuration Switches

The DIP switch at location U22 on the printed wire board is used to configure the serial port parameters of the GPIB-422CT while in S mode. The DIP switch, shown in Figure 3-3, has eight configuration switches.



ON indicates that the switch is pressed on the "ON" side.

OFF indicates that the switch is pressed on the "OFF" side.

Figure 3-3. S Mode Switch Settings

Tables 3-1 and 3-2 show the possible configurations of the eight switches and what each configuration indicates.

Note: The factory default settings are shown in bold italic.

Table 3-1. Configuration Parameters for Switches 1 through 5

Switch	Position	Indication
1	***OFF***	***Configures the GPIB-422CT to operate in S mode.***
	ON	Configures the GPIB-422CT to operate in G mode.
2	OFF	7 bits/character.
	ON	***8 bits/character.***
3	***OFF***	***1 stop bit/character.***
	ON	2 stop bits/character.
4	***OFF***	***Parity generation/checking disabled.***
	ON	Parity generation/checking enabled.
5	***OFF***	***Odd parity.***
	ON	Even parity.

Table 3-2. Configuration Parameters for Switches 6 through 8

Switches 6	7	8	Indication
OFF	OFF	OFF	300 baud
OFF	OFF	ON	600 baud
OFF	ON	OFF	1200 baud
OFF	ON	ON	2400 baud
ON	OFF	OFF	4800 baud
ON	***OFF***	***ON***	***9600 baud***
ON	ON	OFF	19200 baud
ON	ON	ON	38400 baud

Default Settings for Configuration Switch

Figure 3-4 shows the GPIB-422CT's factory default setting for switch U22. Switch 1 is OFF; this indicates that the GPIB-422CT will be operating in S mode. Switch 2 is ON; this indicates that the GPIB-422CT will be using 8 bits/character for its serial data transfers. Switch 3 is OFF; this indicates 1 stop bit per character. Switches 4 and 5 are both OFF; this indicates that parity is disabled. Switches 6 through 8 are in the ON OFF ON positions, respectively, indicating that the serial port will be operating at 9600 baud.

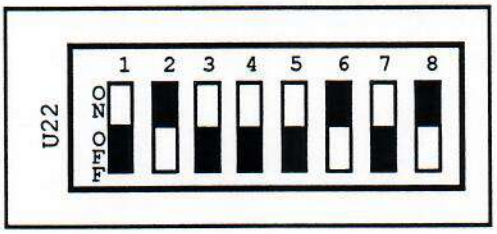


Figure 3-4. Factory Default Settings for Switch U22

Note: Switch U20 is not used by S or G mode, but can be defined and used by user applications in IBCL mode.

Step 4: Connect the Cables

Connect the cables as follows:

1. Connect the serial cable to the GPIB-422CT and securely fasten the holding screws. Connect the other end of the cable to your serial device. Be sure to use only shielded serial cable, and obey all RS-422 cabling restrictions.

2. Connect the GPIB cable to the GPIB-422CT and tighten the thumb screws on the connector. Connect the other end to your GPIB system. Be sure to obey all IEEE-488 cabling restrictions, and use only double-shielded GPIB cable.

3. Connect the power jack of the wall-mount power supply to the power receptacle on the rear panel of the GPIB-422CT, then plug the supply into an AC outlet of the correct voltage.

Step 5: Power on the Unit

Power on your GPIB-422CT by using the front panel rocker switch. The POWER LED should come on immediately. The READY indicator should come on after the GPIB-422CT has passed its power-on self-test, indicating the unit is ready for operation.

If the READY indicator does not come on within seven seconds after the unit is powered on, recheck all connections and switch settings and retry the power-on sequence. If the READY light still fails to come on, contact National Instruments for further instructions.

Chapter 4
Programming in S Mode

This chapter describes how to program the GPIB-422CT in S mode using programming messages and data strings. It describes programming messages, their format, and how they are processed, along with the functions and function arguments that make up the programming messages.

Programming Messages

You program the GPIB-422CT by sending it programming messages (which are ASCII strings) and data strings by way of its serial port.

Programming Message Format

A programming message consists of a function name, one or more arguments (optional), followed by a carriage return (<CR>), a linefeed (<LF>), or a carriage return followed by a linefeed (<CR><LF>).

You can enter programming messages in any combination of uppercase and lowercase letters.

Example of a Programming Message

The following line of BASIC code contains the function name `clr` and the arguments `3` and `4`:

```
PRINT #1,"clr 3,4"
```

This programming message tells the GPIB-422CT to clear the devices at GPIB addresses 3 and 4. `PRINT #1` is the BASIC command to send characters to the serial port after the serial port has been opened with the `"OPEN COM..."` statement. In this example, BASIC automatically sends a <CR>, so it is not necessary to include it here.

Example of a Programming Message with Data String

The following lines of BASIC code contain the function name `wrt`, the argument `2`, and the data string `"IN;CI;"`:

```
PRINT #1, "wrt 2"
PRINT #1, "IN;CI;"
```

This programming message is telling the GPIB-422CT to write to the device at primary address 2.

`"IN;CI;"` is the data string that contains the data `wrt` will send out on the GPIB. In this case, a <CR> is automatically sent by BASIC following each print string, so, again, it is not necessary to include it here.

The `cmd` and `wrt` programming messages are followed by a data string that can contain 7- or 8-bit data.

How Messages are Processed

The GPIB-422CT processes a programming message on a line-by-line basis. The GPIB-422CT buffers the entire message, interprets the function name and arguments, then executes the message.

The data portions of the `wrt` and `cmd` functions are not processed on a line-by-line basis. The data immediately following a wrt and a cmd are sent directly to the GPIB.

The GPIB-422CT recognizes <CTRL> h (hex 8) in a programming message as a backspace and erases the previous character. The GPIB-422CT recognizes <CTRL> h in a data string as a data byte and does not erase the previous character.

Function Names

The function names have been selected to indicate each function's purpose, thereby making your programs easy to understand. However, if you wish to reduce some overhead in your program and do not mind giving up these advantages, you may use only as much of the function name as is necessary to distinguish it from other functions. This abbreviated form of the function

name is shown in **boldface** in the function tables and in the syntax portions of the function descriptions.

Function Argument Delimiters

When you type in a function, separate the first argument from the function name with at least one space. Separate each additional argument with at least one space or a comma.

In the syntax portions of the function descriptions in Chapter 5, the information within the square brackets ([]) are optional. If you want to include optional information, do not type the brackets, only the information inside the brackets.

Abbreviations for Arguments

The function descriptions in Chapter 5 use abbreviations for some arguments. They are as follows:

`addr`	a GPIB address	
`alist`	one or more `addrs`	
`bool`	a Boolean value:	1 = true, on, or enable
		0 = false, off, or disable

GPIB Address

Each device on the GPIB has a GPIB address. The GPIB-422CT's address is 0 at power on and can be changed using the `caddr` function. Refer to the manuals of your GPIB devices to learn their addresses. You will need to know these when you begin to program the GPIB-422CT.

Only the lower five bits of each GPIB address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary address. For example, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. A plus sign (+) separates the primary address from the secondary address. The listen address is 32 (primary address plus 32), the

talk address is 64 (primary address plus 64), and the secondary address is 2 or 98, which are equivalent. The next paragraph explains the \x notation.

```
0+2 or 0+98 or 32+98 or 0+\x62
```

Numeric String Argument

Another type of argument is a numeric string. A numeric string represents an integer, which you can express using decimal, octal, or hexadecimal digits. To specify an octal integer, precede it with a backslash (\). To specify a hexadecimal integer, precede it with a backslash x (\x) or backslash X (\X).

Each of the following numeric strings represents the decimal integer value 112:

```
 112
\160
\x70
```

The GPIB address argument described in the preceding discussion, *GPIB Address*, consisted of one or two numeric strings.

Status Information

The function descriptions in Chapter 5 explain that the GPIB-422CT *records* specific status and error information. This means that it stores that information in its memory so that the status information is available to you when you request it.

The function descriptions also explain that the GPIB-422CT *returns* to you certain information. This means that the GPIB-422CT sends information to you over the serial port. You then read this information from your serial port.

Serial Port Error Handling

The GPIB-422CT continuously monitors the serial port for transmission errors. If it encounters an error in the serial data, the GPIB-422CT records the error. You can program the GPIB-422CT to ignore serial port errors using the `spign` function.

GPIB Read and Write Termination Method (END and EOS)

You program the GPIB-422CT to Talk in order to send data messages over the GPIB, and to Listen in order to receive data messages from the GPIB.

The IEEE-488 specification defines two ways that GPIB Talkers and Listeners can identify the last byte of data messages: `END` and `EOS`. The two methods permit a Talker to send data messages of any length without the Listener(s) knowing in advance the number of bytes in the transmission.

`END` message	The Talker asserts the EOI (End Or Identify) signal while the last data byte is being transmitted. The Listener stops reading when it detects a data byte accompanied by EOI.
`EOS` character	The Talker sends an EOS (end-of-string) character at the end of its data string. The Listener stops receiving data when it detects the EOS character. Either a 7-bit ASCII character or a full 8-bit binary byte can be used.

The two methods can be used individually or in combination. It is important that the Listener be configured to detect the end of a transmission.

The GPIB-422CT always terminates GPIB `rd` operations on the `END` message. Using the `eos` and `eot` functions, you can change the other default GPIB read and write termination methods.

S Mode - Default Settings

Tables 4-1 and 4-2 list power-on characteristics of the GPIB-422CT and the functions you can use to change those characteristics.

Table 4-1. Serial Port Characteristics

Characteristic	Power-on Value	Function
Echo bytes to serial port	no	`echo`
Ignore serial port errors	yes	`spign`
Send XON/XOFF	no	`xon`
Recognize XON/XOFF	no	`xon`

Table 4-2. GPIB Characteristics

Characteristic	Power-on Value	Function
Primary/secondary address	pad=0, sad=none	`caddr`
End-of-string modes	none	`eos`
Send END on writes	yes	`eot`
IST bit setting	0	`ist`
GPIB-422CT is System Controller	yes	`rsc`
I/O timeout	10 sec	`tmo`
Serial poll timeout	.1 sec	`tmo`

List of S Mode Functions by Group

The GPIB-422CT functions are divided into three main groups: GPIB functions, Serial Port functions, and General Use functions.

GPIB Functions

The GPIB functions are divided into subgroups as shown in the following tables. The subgroups are listed with the most frequently used groups first. Often, the I/O and bus management functions are the only ones you need.

Table 4-3. I/O Functions

Function	Purpose
`rd` count, address	Read data
`wrt` count, address list data	Write data

Table 4-4. Bus Management Functions

Function	Purpose
`clr` address list	Clear specified device(s)
`loc` address list	Go to Local
`trg` address list	Trigger selected device(s)

Table 4-5. GPIB Initialization Functions

Function	Purpose
caddr address	Change the IEEE-488 address of the GPIB-422CT
eos modes, eoschar	Change or disable GPIB end-of-string termination mode
eot on/off	Enable or disable END termination message on GPIB write operations
onl on/off	Place the GPIB-422CT online/offline
rsc on/off	Request System Control
tmo values	Change or disable time limits

Table 4-6. Serial Poll Functions

Function	Purpose
rsp address list	Conduct (request) a serial poll
rsv status byte	Request service and/or set or change the serial poll status byte

Table 4-7. Low-Level Controller Functions

Function	Purpose
`cac` mode	Become Active Controller
`cmd` count commands	Send IEEE-488 commands
`gts` mode	Go from Active Controller to Standby
`pct` address	Pass Control
`sic` time	Send interface clear
`sre` on/off	Set/clear remote enable

Table 4-8. Parallel Poll Functions

Function	Purpose
`ist` on/off	Set or clear individual status bit for use in GPIB-422CT response to Parallel Polls
`ppc` values	Parallel Poll Configure
`ppu` address list	Parallel Poll Unconfigure
`rpp`	Conduct (request) a Parallel Poll

Serial Port Functions

Table 4-9. Serial Port Initialization Functions

Function	Purpose
echo on/off	Echo characters received from serial port
spign on/off	Ignore serial port errors
xon modes	Change serial port XON/XOFF protocol

General Use Functions

Table 4-10. General Use Functions

Function	Purpose
ibcl	Switch to IBCL operating system
id	Identify system
stat modes	Return GPIB-422CT status
wait mask	Wait for selected event(s)

Alphabetical List of S Mode Functions

The following is an alphabetical list of all functions.

Table 4-11. GPIB-422CT Functions

Function	Purpose
cac mode	Become Active Controller
caddr address	Change the IEEE-488 address of the GPIB-422CT
clr address list	Clear specified device(s)
cmd count commands	Send IEEE-488 commands
echo on/off	Echo characters received from serial port
eos modes, eoschar	Change or disable GPIB end-of-string termination mode
eot on/off	Enable or disable END termination message on GPIB write operations
gts mode	Go from Active Controller to Standby
ibcl	Switch to IBCL operating system
id	Identify system
ist set/clear	Set or clear individual status bit for use in GPIB-422CT response to Parallel Polls
loc address list	Go to Local
onl on/off	Place the GPIB-422CT online/offline

(continues)

Table 4-11. GPIB-422CT Functions (continued)

Function	Purpose
`pct` address	Pass Control
`ppc` values	Parallel Poll Configure
`ppu` address list	Parallel Poll Unconfigure
`rd` count, address	Read data
`rpp`	Conduct (request) a Parallel Poll
`rsc` on/off	Request System Control
`rsp` address list	Conduct (request) a serial poll
`rsv` status byte	Request service and/or set or change the serial poll status byte
`siv` time	Send interface clear
`spign` on/off	Ignore serial port errors
`sre` on/off	Set remote enable
`stat` modes	Return GPIB-422CT status
`tmo` values	Change or disable time limits
`trg` address list	Trigger selected device(s)
`wait` mask	Wait for selected event(s)
`wrt` count, address list data	Write data
`xon` modes	Change serial port XON/XOFF protocol

Chapter 5
S Mode Functions

This chapter contains descriptions of S mode functions that you use to program the GPIB-422CT. These functions are in alphabetical order and are formatted to provide you an easily usable reference.

Points to Remember

- The programming examples for each function description are in Microsoft BASIC Version 3.0.
- In the syntax portion of the function descriptions, arguments enclosed in square brackets (`[]`) are optional. Do not enter the brackets as part of your argument.
- Terminate each programming message with a carriage return (<CR>), a linefeed (<LF>), or a carriage return followed by a linefeed (<CR><LF>). The terminator is denoted by a <CR> in the syntax portions of the function descriptions. In the programming examples, the BASIC `PRINT #` statement automatically sends a carriage return at the end of the string, so a carriage return is not placed there explicitly.
- To send more than one programming message per `PRINT` statement, embed a <CR> (denoted by `CHR$(13)`) or a <LF> (denoted by `CHR$(10)`) in the statement. For example, to send the two programming messages "send interface clear" (`SIC`) and "send remote enable" (`SRE`), you could use either of these two sequences:

```
PRINT #1,"sic"
PRINT #1,"sre 1"
```

or

```
PRINT #1,"sic"+CHR$(13)+"sre 1"
```

- For all examples, the communications port has been assigned to file number 1 (`#1`) by the BASIC `OPEN "COM..."` statement.

- The I/O and bus management functions should meet most of your needs. In the descriptions that follow, these functions are marked with an asterisk (*). These are the most frequently used functions.
- It is only necessary for you to send enough characters of the function name to distinguish it from other functions. Those characters are shown in **boldface** in the syntax portion of each function description.

cac - Become Active Controller

Type: Low-level Controller function

Syntax: **cac** [bool]<CR>

Purpose: You use cac to change the GPIB-422CT from Standby Controller to Active Controller when the I/O and bus management functions do not meet the needs of your device. cac gives you more precise control over the GPIB than the I/O and bus management functions.

Remarks: If the argument bool is 0, the GPIB-422CT takes control immediately; that is, it takes control asynchronously. If the argument bool is 1, the GPIB-422CT takes control after any handshake that is in progress completes; that is, it takes control synchronously.

If you call cac without an argument, the GPIB-422CT returns to you the current Controller status, which is 0 if the GPIB-422CT is not the Active Controller and 1 if the GPIB-422CT is the Active Controller.

If you call cac with an argument and the GPIB-422CT is not CIC, the GPIB-422CT records the ECIC error.

The power-on Controller status of the GPIB-422CT is Idle Controller.

See Also: gts and sic.

Examples:

```
1. PRINT #1,"cac 0"   'Take control immediately.

2. PRINT #1,"cac 1"   'Take control synchronously.

3. PRINT #1,"CAC"     'Are we the Active Controller?

   response: 1<CR><LF>                (...yes, we're CAC)
```

caddr - Change the GPIB Address of the GPIB-422CT

Type: Initialization function

Syntax: `caddr [addr]<CR>`

Purpose: You use `caddr` at the beginning of your program to change the GPIB address of the GPIB-422CT.

Remarks: The argument `addr` is a device address that specifies the new GPIB address for the GPIB-422CT. `addr` consists of a primary address and an optional secondary address. The secondary address is separated from the primary address by a plus sign (+). Both addresses are expressed as numeric strings.

Only the lower five bits of each address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary address. Therefore, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. The listen address is 32, the talk address is 64, and the secondary address is 2 or 98, which are equivalent.

`0+2` or `0+98` or `32+98` or `0+\x62`

If you specify a primary address without a secondary address, secondary addressing is disabled.

If you call `caddr` without an argument, the GPIB-422CT returns to you its current GPIB address.

The address assigned by this function remains in effect until you call `caddr` again, call `onl`, or you turn off the GPIB-422CT.

The power-on default is 0 with secondary addressing disabled.

Examples:

```
1. PRINT #1,"caddr 0+22"      'Give GPIB-422CT a
                              'primary address of 0
                              'and a secondary
                              'address of 22.

2. PRINT #1,"CADDR 1"         'Change GPIB-422CT
                              'primary address to 1
                              'and disable secondary
                              'addressing.

3. PRINT #1,"CADDR"           'Return current GPIB-
                              '422CT address.
```

response: **1<CR><LF>**

clr - Clear Specified Device *

Type: Bus Management function

Syntax: `clr [alist]<CR>`

Purpose: You use `clr` to reset the internal or device functions of the specified devices. For example, a multimeter might require that you send it either the GPIB Device Clear or Selected Device Clear command to change its function, range, and trigger mode back to its default setting. Use `clr` to do this.

Remarks: The argument `alist` is a list of `addrs` separated by commas or spaces. `addrs` are device addresses that specify the GPIB addresses you wish to clear.

A device address consists of a primary address and an optional secondary address. The secondary address is separated from the primary address by a plus sign (+).

Only the lower five bits of each address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary address. Therefore, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. The listen address is 32, the talk address is 64, and the secondary address is 2 or 98, which are equivalent.

`0+2` or `0+98` or `32+98` or `0+\x62`

If you call `clr` with `alist`, the GPIB-422CT clears only the devices specified in `alist` (Selected Device Clear).

If you call `clr` without `alist`, the GPIB-422CT clears all devices (Device Clear).

If this is the first function you call that requires GPIB Controller capability, and you have not disabled System Controller capability with `rsc`, the GPIB-422CT sends Interface Clear (IFC) to make itself CIC. It also asserts Remote Enable.

If you passed control to some other GPIB device, control must be passed back to you or you must send IFC to make yourself CIC before making this call. Otherwise, the ECIC error will be posted.

See Also: Appendix D for more information on clearing devices, and Appendix B for more error information.

Examples:

```
1. PRINT #1,"clr 14+30,16+12,18,3+26,6"
                        'Selectively clear 5
                        'devices.

2. PRINT #1,"CLR"       'Issue Device Clear to all
                        'devices.
```

cmd - Send GPIB Commands

Type: Specialized Controller function

Syntax: `cmd [#count]<CR>`
`commands<CR>`

Purpose: You use `cmd` when the I/O and bus management functions do not meet the needs of your device. `cmd` gives you precise control over the GPIB. For example, in applications that require command sequences not sent by other functions, you can use `cmd` to transmit any sequence of interface messages (commands) over the GPIB.

Remarks: The argument `count` is a numeric string preceded by a number sign (#). `#count` specifies the number of GPIB command bytes (interface messages) to send, which is a number between 1 and 255. The number of command bytes must not include the carriage return (<CR>) or linefeed (<LF>) that you include to indicate the end of the programming message.

The argument `commands` is a list of GPIB commands. These commands are represented by their ASCII character equivalents. For example, the GPIB Untalk (UNT) command is the ASCII character underscore (_).

If you call `cmd` without `#count`, the GPIB-422CT recognizes the end of the command string when it sees a <CR> or an <LF>. `#count` is required only if the command string contains a <CR> or an <LF> character. However, a <CR> or an <LF> in the command string would be unusual since neither of these is a defined GPIB command.

The GPIB commands, or interface messages, are listed in Appendix A. They include device talk and listen addresses, secondary addresses, messages, device clear and trigger instructions, and other management messages.

Do not use `cmd` to send programming instructions to devices. Use `rd` and `wrt` to send or receive device programming instructions and other device-dependent information.

The cmd operation terminates when:

- The GPIB-422CT successfully transfers all commands.
- The GPIB-422CT detects an error (GPIB-422CT is not CIC).
- The I/O time limit is exceeded.
- The Take Control (TCT) command is in your command string and is sent to the GPIB.
- The Interface Clear (IFC) message is received from the System Controller (not the GPIB-422CT).

After cmd terminates, the GPIB-422CT records the number of command bytes it actually sent. If one of the events described above occurs, the count may be less than expected.

If you specify `#count` and enter more than `#count` command bytes, the excess command bytes up to the <CR><LF> are discarded.

If you call cmd and the GPIB-422CT is not CIC, the GPIB-422CT records the ECIC error.

If the GPIB-422CT is CIC but not Active Controller, it takes control and asserts ATN before sending the command bytes. It remains Active Controller afterward.

See Also: Appendix A to convert hex values to ASCII characters.

Examples:

```
1. PRINT #1,"CMD"          'Program device at address
                           '11 to listen and GPIB-
                           '422CT at address 0 to
                           'talk.
   PRINT #1,"+@"           'Device listen address is
                           '43 or ASCII + and GPIB-
                           '422CT talk address is 64
                           'or ASCII @.
   PRINT #1,"WRT"          'Write the string "ABCDE"
   PRINT #1,"ABCDE"        'to device at address 11.

2. PRINT #1,"cmd"+CHR$(13)+"_?W"+CHR$(9)
                           'Pass control to device 23
                           '(CHR$(9)=TCT command).
```

echo - Echo Characters Received from Serial Port

Type: Serial Port function

Syntax: **echo** [bool]<CR>

Purpose: You use echo when a terminal is connected to the GPIB-422CT and you wish to display what you type on the screen of the terminal.

Remarks: If the argument bool is 1, characters received from the serial port are echoed back to the serial port. If the argument bool is 0, characters are not echoed. If the argument bool is 1 and echoing was previously disabled, characters will not be echoed until this command has been completely processed, that is, the *next* programming message will be echoed.

If you call echo without an argument, the GPIB-422CT returns the current setting.

Examples:

The following examples show commands as you would enter them at a terminal.

1. echo 1<CR> 'Turn on character echoing.

2. ECHO 0<CR> 'Disable character echoing.

3. echo<CR> 'What is the echo status?

   response: **0<CR><LF>** (character echo is disabled)

eos - Change/Disable GPIB EOS Termination Mode

Type: Initialization function

Syntax: **eos** [[R][X][B] eoschar]<CR>

or

eos D<CR>

Purpose: You use eos at the beginning of your program if you wish to use an eos mode when you transfer data to and from the GPIB. eos tells the GPIB-422CT when to stop reading information from the GPIB. eos also enables the GPIB-422CT to tell other devices that it is finished writing information to the GPIB. eos defines a specific character, end-of-string (EOS), to be recognized as a string terminator.

Remarks: The arguments R, X, B, and D specify GPIB termination methods. They enable or disable the corresponding eos mode. If a particular letter is specified, the corresponding eos mode is enabled. If it is not specified, the corresponding eos mode is disabled.

eoschar is a numeric string that represents a single ASCII character. For example, 10 represents the ASCII linefeed character.

Table 5-1. Data Transfer Termination Methods

Description	Letter
REOS - terminate read when EOS is detected.	R
XEOS - set EOI with EOS on write functions.	X
BIN - compare all 8 bits of EOS byte rather than low 7 bits (all read and write functions).	B
DISABLE - disable all eos modes.	D

Methods R and B determine how GPIB read operations performed by the GPIB-422CT terminate. If Method R alone is chosen, reads terminate when the low seven bits of the byte that is read match the low seven bits of the EOS character. If Methods R and B are chosen, a full 8-bit comparison is used.

Methods X and B together determine when GPIB write operations performed by the GPIB-422CT send the END message. If Method X alone is chosen, the END message is sent automatically with the EOS byte when the low seven bits of that byte match the low seven bits of the EOS character. If Methods X and B are chosen, a full 8-bit comparison is used.

Note: Defining an EOS byte for the GPIB-422CT does not cause the GPIB-422CT to insert that byte into the data string when performing GPIB writes. To send the EOS byte, you must include it in the data string that you send following the `wrt` programming message.

By default, no `eos` modes are enabled.

If you call `eos` with B alone as an argument, the GPIB-422CT records the EARG error.

If you call `eos` without an argument, the GPIB-422CT returns to you the current `eos` settings.

The assignment made by this function remains in effect until you call `eos` again, call `onl`, or you turn off the GPIB-422CT.

See Also: Chapter 4, the section entitled *GPIB Read and Write Termination Method.*

Examples:

```
1. PRINT #1,"eos R,B,10" 'Terminate read when <LF>
                         'is detected; compare all
                         '8 bits; do not send EOI
                         'with <LF>.
   PRINT #1,"rd 10 5"    'Read 10 bytes from device
                         '5 into serial port
                         'buffer.
   RESP$=INPUT$(10,#1)   'Input 10 bytes from
                         'serial port buffer.
   LINE INPUT #1,COUNT$  'Input string that
                         'indicates number of bytes
                         'actually read from GPIB.
   PRINT COUNT$;" bytes were read from GPIB"
                         'Print number of bytes
                         'that were read from
                         'the GPIB.

2. PRINT #1,"EOS X,13"   'Send EOI with <CR> on
                         'wrt; do not terminate
                         'when <CR> is detected
                         'on rd; compare 7 bits.
   PRINT #1,"wrt #10 5"  'GPIB-422CT sends EOI with
                         '<CR> (CHR$(13)) to tell
                         'Listeners that this is
                         'the last byte of data.
   PRINT #1,"012345678"

3. PRINT #1,"eos"        'What are the current EOS
                         'settings?
```

response: **X,13<CR><LF>**

eot - Enable/Disable END Message on GPIB Writes

Type: Initialization function

Syntax: `eot [bool]<CR>`

Purpose: You use `eot` at the beginning of your program if you wish to change how the GPIB-422CT terminates GPIB writes. Using `eot`, you tell the GPIB-422CT to automatically send or not send the GPIB END message with the last byte that it writes to the GPIB.

Remarks: If the argument `bool` is 1, the GPIB-422CT automatically sends the END message with the last byte of each `wrt`. If the argument `bool` is 0, END is not sent. The power-on default is 1.

If you call `eot` without an argument, the GPIB-422CT returns to you a 1 to indicate END termination is currently enabled, or a 0 to indicate END termination is currently disabled.

The assignment made by `eot` remains in effect until you call `eot` again, call `onl`, or you turn off the GPIB-422CT.

The GPIB-422CT sends the END message by asserting the GPIB EOI signal during the last byte of a data transfer. `eot` is used primarily to send variable length data.

See Also: Chapter 4, the section entitled *GPIB Read and Write Termination Method.*

Examples:

```
1. PRINT #1,"eot 0"          'Disable END termination.

2. PRINT #1,"EOT 1"          'Send END with last byte.
   PRINT #1,"WRT 3"          'Write data to device at
                             'address 3.
   PRINT #1,"ABCDE"          'The EOI line is
                             'automatically asserted
                             'when the last byte (the
                             'letter E) is sent to tell
                             'the Listeners it is the
                             'last byte of data.

3. PRINT #1,"eot"            'What is the current EOT
                             'setting?

   response: 1<CR><LF>  (END termination is currently
                           enabled)
```

gts - Go from Active Controller to Standby

Type: Specialized Controller function

Syntax: `gts [bool]<CR>`

Purpose: You use `gts` to change the GPIB-422CT from Active Controller to Standby Controller. You use `gts` when the I/O and bus management functions do not meet the needs of your device. For example, you use `gts` if you wish to let two external devices to talk to each other directly. The GPIB-422CT can selectively participate in the handshake of the data transfer and hold off the handshake when it detects the END message. The GPIB-422CT can then take control synchronously without possibly corrupting the transfer.

Remarks: If the argument `bool` is 1, shadow handshaking is enabled. If the argument `bool` is 0, shadow handshaking is not performed.

If you call `gts` without an argument, the GPIB-422CT returns to you the current Controller status: `CSB,0` if the GPIB-422CT is in Standby without shadow handshaking; `CSB,1` if the GPIB-422CT is in Standby with shadow handshaking; `CAC` if the GPIB-422CT is CIC but is not in Standby, that is, it is the Active Controller; and `CIDLE` if the GPIB-422CT is not the CIC, that is, is an IDLE Controller.

`gts` causes the GPIB-422CT to go to the Controller Standby state and to unassert the ATN signal if it is initially the Active Controller. `gts` permits GPIB devices to transfer data without the GPIB-422CT participating in the transfer.

If you enable shadow handshaking, the GPIB-422CT participates in the data handshake as an Acceptor without actually reading the data. It monitors the transfers for the END (EOI or end-of-string character) message and holds off subsequent transfers. This mechanism allows the GPIB-422CT to take control synchronously on a subsequent operation such as `cmd` or `rpp`.

Before performing a `gts` with a shadow handshake, you should call `eos` to establish the proper end-of-string character

or to disable the EOS detection if the end-of-string character used by the Talker is not known.

If you call `gts` with an argument and the GPIB-422CT is not CIC, the GPIB-422CT records the ECIC error.

See Also: `cac`.

Examples:

```
1. PRINT #1,"gts 0"        'GTS without shadow
                           'handshaking.

2. PRINT #1,"GTS 1"        'GTS with shadow
                           'handshaking.

3. PRINT #1,"gts"          'What is standby status?

   response: CSB,1<CR><LF> (GPIB-422CT is in
                              standby status
                              with shadow
                              handshaking)
```

ibcl - Enter ibcl Operating System

Type: General Use function

Syntax: `ibcl<CR>`

Purpose: You use `ibcl` if you wish to enter the interface bus control language operating system.

Remarks: After entering `ibcl`, the user has access to most of the same GPIB words (commands) as in S mode. In addition to the GPIB words, there are other predefined words that the user could use to write complex programs and custom applications.

See Also: The *GPIB-CT IBCL Reference Manual* included with your package.

Examples:

```
1. PRINT #1,"ibcl"          'Enter the ibcl operating
                            'system.
```

response: **<CR><LF>**
ok<CR><LF>

```
2. PRINT #1,": init 5 "+CHR$(34)+" IN;"+CHR$(34)+
          " wrt,"           'Define a word
                            'called 'init' to initialize
                            'an IBM 7320 Color Plotter.
```

response: **ok<CR><LF>**

```
3. PRINT #1,"init"          'Executed the word
                            'defined in Example 2.
```

response: **ok<CR><LF>**

```
4. PRINT #1,"bye"          'Exit ibcl and return to
                           'the NI610 operating
                           'system.
```

response:* **END,CMPL,REM,CIC,LACS**
NGER
NSER
165

* Assuming you had continuous status reporting enabled when you called `ibcl`.

id - Identify System

Type: General Use function

Syntax: `id<CR>`

Purpose: You use `id` if you wish to know the revision level of your software, or if you wish to know how much RAM is installed in your GPIB-422CT.

Remarks: The identification is returned in three strings. The first two strings identify the company product model, the software revision level, and a copyright notice. The third string identifies the number of bytes of RAM in the GPIB-422CT.

Example:

```
1. PRINT #1,"id"          'Get system
                          'identification.
```

typical response:

```
GPIB-422CT Rev. A.2<CR><LF>
(c)1990 National Instruments<CR><LF>
64K bytes RAM<CR><LF>
```

ist - Set or Clear Individual Status Bit

ist: Parallel Poll function

Syntax: `ist [bool]<CR>`

Purpose: You use `ist` when the GPIB-422CT participates in a parallel poll that is conducted by another device that is Active Controller.

Remarks: If the argument `bool` is 1, the GPIB-422CT's individual status bit is set to 1. If the argument `bool` is 0, the GPIB-422CT's individual status bit is cleared. The power-on default is 0.

If you call `ist` without an argument, the GPIB-422CT returns the value of its individual status bit.

See Also: `ppc`, and Appendix F, *Parallel Polling*.

Examples:

```
1. PRINT #1,"ist 1"        'Set ist to 1.

2. PRINT #1,"IST 0"        'Clear ist to 0.

3. PRINT #1,"ist"          'What is ist set to?

   response: 0<CR><LF> (ist is currently 0)
```

loc - Go to Local *

Type: Bus Management function

Syntax: `loc [alist]<CR>`

Purpose: You use `loc` to put a device in local program mode. In this mode you can program the device from its front panel. Since a device must usually be placed in remote program mode before it can be programmed from the GPIB, the GPIB-422CT automatically puts the device in remote program mode. You then use `loc` to return devices to local program mode.

Remarks: The argument `alist` is a list of `addrs` separated by commas or spaces. `addrs` are device addresses that specify the GPIB addresses of the devices you wish to return to local mode.

A device address consists of a primary address and an optional secondary address. The secondary address is separated from the primary address by a plus sign (+).

Only the lower five bits of each address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary address. Therefore, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. The listen address is 32, the talk address is 64, and the secondary address is 2 or 98, which are equivalent.

`0+2` or `0+98` or `32+98` or `0+\x62`

If you call `loc` with `alist`, the GPIB-422CT places the specified device(s) in local mode using the Go To Local (GTL) command.

If this is the first function you call that requires GPIB Controller capability, and you have not disabled System Controller capability with `rsc`, the GPIB-422CT sends Interface Clear (IFC) to make itself CIC. It also asserts Remote Enable.

If you passed control to some other GPIB device, control must be passed back to you or you must send IFC to make yourself CIC before making this call. Otherwise, the ECIC error will be posted.

If you call `loc` without `alist`, and the GPIB-422CT is System Controller, the GPIB-422CT returns all devices to local mode by unasserting REN and asserting it again. If you call `loc` without `alist` and the GPIB-422CT is not System Controller, the GPIB-422CT records the ESAC error.

See Also: Appendix B for more error information.

Examples:

```
1. PRINT #1,"loc 6+22,4+23,7"
                          'Put 3 devices in local
                          'mode.

2. PRINT #1,"LOC"         'Put all devices in local
                          'mode.
```

onl - Place the GPIB-422CT Online/Offline

Type: Initialization function

Syntax: `onl [bool]<CR>`

Purpose: You use `onl` to disable communications between the GPIB-422CT and the GPIB, or to reinitialize the GPIB-422CT characteristics to their default values.

Remarks: If the argument `bool` is 1, the GPIB-422CT places itself online. If the argument `bool` is 0, the GPIB-422CT places itself offline. By default, the GPIB-422CT powers up online, is in the Idle Controller state, and configures itself to be the System Controller.

If you call `onl` without an argument, the GPIB-422CT returns the current status of the GPIB-422CT, which is 0 if the GPIB-422CT is offline and 1 if the GPIB-422CT is online.

Placing the GPIB-422CT offline can be thought of as disconnecting its GPIB cable from the other GPIB devices.

Placing the GPIB-422CT online allows the GPIB-422CT to communicate over the GPIB, and also restores all GPIB-422CT settings to their power-on values.

See Also: Tables 4-1 and 4-2 for the GPIB-422CT power-on settings.

Examples:

```
1. PRINT #1,"onl 1"        'Put the GPIB-422CT online
                           'and restore its power-on
                           'settings.

2. PRINT #1,"ONL 0"        'Put the GPIB-422CT
                           'offline to prevent
                           'it from communicating
                           'with the 'GPIB.
```

pct - Pass Control

Type: Specialized Controller function

Syntax: `pct addr<CR>`

Purpose: You use `pct` to pass Controller-In-Charge (CIC) authority from the GPIB-422CT to some other device.

Remarks: The argument `addr` is the address of the device you wish to pass control to. `addr` consists of a primary address and an optional secondary address. The secondary address is separated from the primary address by a plus sign (+). Both addresses are expressed as numeric strings.

Only the lower five bits of each address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary addresses. Therefore, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. The listen address is 32, the talk address is 64, and the secondary address is 2 or 98, which are equivalent.

`0+2` or `0+98` or `32+98` or `0+\x62`

`pct` passes CIC authority from the GPIB-422CT to the device specified by `addr`. The GPIB-422CT automatically goes to Controller Idle State. It is assumed that the target device has Controller capability.

If you call `pct` with an argument and the GPIB-422CT is not CIC, it records the ECIC error.

If you call `pct` without an argument, the GPIB-422CT records the EARG error.

Example:

```
1. PRINT #1,"pct 7+18"  'Pass control to device
                        'with primary address 7
                        'and secondary address 18.
```

ppc - Parallel Poll Configure

Type: Parallel Poll function

Syntax: `ppc addr,ppr,s [addr,ppr,s]...<CR>`

Purpose: You use `ppc` to configure specified devices to respond to parallel polls in a certain manner.

Remarks: `addr` specifies the GPIB address of the device to be enabled or disabled for parallel polls. `addr` consists of a primary address and an optional secondary address. The secondary address is separated from the primary address by a plus sign (+). Both addresses are expressed as numeric strings.

Only the lower five bits of each address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary addresses. Therefore, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. The listen address is 32, the talk address is 64, and the secondary address is 2 or 98, which are equivalent.

`0+2` or `0+98` or `32+98` or `0+\x62`

The argument `ppr` is an integer string between 1 and 8 specifying the data line on which to respond.

The argument `s` is either 0 or 1 and is interpreted along with the value of the device's individual status bit to determine whether to drive the line true or false.

Each group of `addr,ppr,s` can be separated by either a comma or space, just as any list of arguments.

If you call `ppc` without an argument, the GPIB-422CT records the EARG error.

If this is the first function you call that requires GPIB Controller capability, and you have not disabled System Controller capability with `rsc`, the GPIB-422CT sends

Interface Clear (IFC) to make itself CIC. It also asserts Remote Enable.

If you passed control to some other GPIB device, control must be passed back to you or you must send IFC to make yourself CIC before making this call. Otherwise, the ECIC error will be posted.

The GPIB-422CT takes the arguments `ppr` and `s` and constructs the appropriate parallel poll enable (PPE) message for each `addr` specified.

When `addr` is the address of the GPIB-422CT, the GPIB-422CT programs itself to respond to a parallel poll by setting its local poll enable (LPE) message to the value specified.

See Also: `ist`, `ppu`, `rpp`, and Appendix F, *Parallel Polling*.

Example:

```
1. PRINT #1,"PPC 18+23,8,0 23+10,7,1"
                          'Configure 2 devices for
                          'parallel poll.
   PRINT #1,"RPP"         'Conduct a parallel poll
                          'of 2 devices configured
                          'above.
   response:  192<CR><LF> (both devices
                           responded positively)
   INPUT #1,PPR%          'Assign parallel poll
                          'response to integer
                          'variable.
```

ppu - Parallel Poll Unconfigure

Type: Parallel Poll function

Syntax: **ppu** `[alist]<CR>`

Purpose: You use `ppu` if you are performing parallel polls and you wish to prevent certain devices from responding.

Remarks: The argument `alist` is a list of `addrs` that are separated by commas or spaces. `addrs` are device addresses that specify the GPIB addresses of the device or devices to be disabled from parallel polls.

A device address consists of a primary address and an optional secondary address. The secondary address is separated from the primary address by a plus sign (+).

Only the lower five bits of each address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary address. Therefore, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. The listen address is 32, the talk address is 64, and the secondary address is 2 or 98, which are equivalent.

`0+2` or `0+98` or `32+98` or `0+\x62`

If you call `ppu` with `alist`, the GPIB-422CT unconfigures from parallel polls only those devices specified in `alist`.

If you call `ppu` without `alist`, the GPIB-422CT unconfigures all devices from parallel polls.

If this is the first function you call that requires GPIB Controller capability, and you have not disabled System Controller capability with `rsc`, the GPIB-422CT sends Interface Clear (IFC) to make itself CIC. It also asserts Remote Enable.

If you passed control to some other GPIB device, control must be passed back to you or you must send IFC to make yourself CIC before making this call. Otherwise, the ECIC error will be posted.

If the address of the GPIB-422CT is included in `alist`, the GPIB-422CT disables itself from responding to parallel polls.

See Also: `ist`, `ppc`, `rpp`, and Appendix F, *Parallel Polling*.

Examples:

```
1. PRINT #1,"ppu 14"      'Send the PPU command to
                          'device 14.

2. PRINT #1,"PPU"         'Send the PPU command to
                          'all devices.
```

rd - Read Data *

Type: I/O function

Syntax: **rd** `#count [addr]`<CR>

Purpose: You use `rd` to read data from the GPIB.

Remarks: The argument `#count` is a numeric string preceded by a number sign (#). `#count` specifies the number of bytes to read. `count` must not contain a comma. It can specify a number between 1 and 65535.

The argument `addr` is a device address that specifies the address of the device to be addressed as a Talker. `addr` consists of a primary address and a secondary address. The secondary address is separated from the primary address by a plus sign (+).

Only the lower five bits of each address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary address. Therefore, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. The listen address is 32, the talk address is 64, and the secondary address is 2 or 98, which are equivalent.

`0+2` or `0+98` or `32+98` or `0+\x62`

The GPIB-422CT reads data from the GPIB until the specified byte count is reached, the GPIB END message is received with a data byte, the EOS byte is received, or a timeout occurs.

Because you may not know for certain the number of bytes actually read from the GPIB, the GPIB-422CT returns the received GPIB data to you in the following manner. First, the GPIB-422CT returns to you all bytes it read from the GPIB. Next, it sends null bytes until the total number of bytes returned to you matches your requested count. Finally, it returns a numeric string representing the number of bytes that it actually read from the GPIB.

For example, if you send the GPIB-422CT the programming message "`rd 10`"<CR>, it reads data from the GPIB until it receives 10 bytes of data, the END message, or an `eos` byte. Let's say the GPIB-422CT receives END with the fourth data byte. The GPIB-422CT then returns to you the 4 data bytes, followed by 6 null bytes, followed by an ASCII 4 and <CR><LF>. A null byte is decimal 0. You should always read back count bytes of data from the serial port, then look at the remaining bytes to determine how many of the count bytes were read from the GPIB. Refer to the example at the end of this description.

The GPIB-422CT aborts the GPIB read and records the EABO error if, at any time during the GPIB read, the time limit set for I/O functions expires. This limit is 10 seconds unless you use `tmo` to change it.

If the GPIB-422CT is CIC, `rd` will cause the GPIB-422CT to address itself to listen if it is not already addressed. If you specify the address of the Talker, the GPIB-422CT will also address that device to talk. If you do not specify the address of the Talker, the GPIB-422CT will assume that the Talker has already been addressed.

The GPIB-422CT then places itself in Standby Controller state with ATN off and remains there after the read operation is completed.

If you specify an address, the GPIB-422CT must be CIC to perform the addressing.

If this is the first function you call that requires GPIB Controller capability, and you have not disabled System Controller capability with `rsc`, the GPIB-422CT sends Interface Clear (IFC) to make itself CIC. It also asserts Remote Enable.

If you call `rd` with the address argument, and you previously passed control to some other GPIB device, control must be passed back to you or you must send IFC to make yourself CIC before making this call. Otherwise, the ECIC error will be posted.

If the GPIB-422CT is not CIC and you do not specify the Talker address, the GPIB-422CT assumes it will be addressed by the Controller, then proceeds.

If you call `rd` without an argument, the GPIB-422CT records the EARG error.

See Also: `eos`, `eot`, and `tmo`.

Example:

```
1. PRINT #1,"rd #10 3"     'Read up to 10 bytes from
                           'the GPIB device at
                           'address 3.
   RESP$=INPUT$(10,#1)     'Input 10 bytes from
                           'serial port buffer.
   INPUT #1,COUNT%         'Input ASCII string
                           'representing number of
                           'bytes read from the GPIB.
                           'COUNT% is number of bytes
                           'read from GPIB; remaining
                           'bytes in RESP$ can be
                           'ignored.
```

rpp - Request (Conduct) a Parallel Poll

Type: Parallel Poll function

Syntax: `rpp<CR>`

Purpose: You use `rpp` if you wish to conduct a parallel poll to obtain information from several devices at the same time.

Remarks: `rpp` causes the GPIB-422CT to conduct a parallel poll of previously configured devices by sending the IDY message (ATN and EOI both asserted) and reading the response from the GPIB data lines. The GPIB-422CT pulses the IDY message for greater than or equal to 2 microseconds and expects valid responses within that time. It remains Active Controller afterward.

The GPIB-422CT returns the Parallel Poll Response (PPR) following the poll in the form of a numeric string representing the decimal value of the response.

If this is the first function you call that requires GPIB Controller capability, and you have not disabled System Controller capability with `rsc`, the GPIB-422CT sends Interface Clear (IFC) to make itself CIC. It also asserts Remote Enable.

If you passed control to some other GPIB device, control must be passed back to you or you must send IFC to make yourself CIC before making this call. Otherwise, the ECIC error will be posted.

See Also: `ist`, `ppc`, `ppu`, and Appendix F, *Parallel Polling*.

Example:

```
1. PRINT #1,"ppc 13,1,0 15,3,0"+CHR$(13)+"rpp"
                        'Configure 2 devices for
                        'parallel polls and poll
                        'them.

   response: 5<CR><LF>  (both devices
                         responded positively)

   INPUT #1,PPR%        'Get parallel poll
                        'response from serial
                        'port buffer and assign
                        'it to integer variable
                        'PPR%.
   PRINT #1,"ppu"       'Unconfigure all devices
                        'from parallel polls.
```

rsc - Request System Control

Type: Initialization function

Syntax: **rsc** [bool]<CR>

Purpose: You use rsc if some other device in your GPIB system should be System Controller.

Remarks: If the argument bool is 1, the GPIB-422CT configures itself to be the GPIB System Controller. If the argument bool is 0, the GPIB-422CT configures itself as not System Controller.

If you call rsc without an argument, the GPIB-422CT returns to you its System Controller status, which is 0 if the GPIB-422CT is not currently System Controller or 1 if the GPIB-422CT is System Controller.

As System Controller the GPIB-422CT can send the Interface Clear (IFC) and Remote Enable (REN) messages to GPIB devices. If some other Controller asserts Interface Clear, the GPIB-422CT cannot respond unless it is configured as not System Controller.

In most applications, the GPIB-422CT will be System Controller. In some applications, the GPIB-422CT will never be System Controller. In either case, rsc is used only if the computer is not going to be System Controller while the program executes. The IEEE-488 standard does not specifically allow schemes in which System Control can be passed from one device to another; however, rsc could be used in such a scheme.

The GPIB-422CT configures itself to be System Controller at power-on.

See Also: sic and sre.

Examples:

```
1. PRINT #1,"rsc 1"        'Enable GPIB-422CT to be
                           'System Controller.

2. PRINT #1,"rsc 0"        'Disable system control.

3. PRINT #1,"rsc"          'What is the current
                           'System Controller
                           'status?

   response: 0<CR><LF>     (GPIB-422CT is not the
                            System Controller)
```

rsp - Request (Conduct) a Serial Poll

Type: Serial Poll function

Syntax: **rsp** `alist<CR>`

Purpose: You use `rsp` if you wish to conduct a serial poll to obtain device-specific status information from one or more devices.

Remarks: The argument `alist` is a list of `addrs` that are separated by commas or spaces. `addrs` are device addresses that specify the GPIB addresses you wish to poll.

A device address consists of a primary address and an optional secondary address. The secondary address is separated from the primary address by a plus sign (+).

Only the lower five bits of each address are significant. These bits can be in the range from 0 through 30 for both the primary and the secondary addresses. Therefore, the binary value 01100010 (decimal 98) is interpreted as decimal 2.

The following examples all specify a primary address of 0 and a secondary address of 2. The listen address is 32, the talk address is 64, and the secondary address is 2 or 98, which are equivalent.

`0+2` or `0+98` or `32+98` or `0+\x62`

`rsp` serially polls the specified devices to obtain their status bytes. If bit 6 (the hex 40 or RQS bit) of a device's response is set, its status response is positive; that is, that device is requesting service. Before `rsp` completes, all devices are unaddressed.

The interpretation of each device's response, other than the RQS bit, is device specific. For example, the polled device might set a particular bit in the response byte to indicate that it has data to transfer, and another bit to indicate a need for reprogramming. Consult the device documentation for interpretation of the response byte.

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP