KEITHLEY

Model 500-SERIAL
RS-232 to IEEE-488 Converter
Instruction Manual

WARRANTY

Keithley Instruments, Inc. warrants this product to be free from defects in material and workmanship for a period of 1 year from date of shipment.

Keithley Instruments, Inc. warrants the following items for 90 days from the date of shipment: probes, cables, rechargeable batteries, diskettes, and documentation.

During the warranty period, we will, at our option, either repair or replace any product that proves to be defective.

To exercise this warranty, write or call your local Keithley representative, or contact Keithley headquarters in Cleveland, Ohio. You will be given prompt assistance and return instructions. Send the product, transportation prepaid, to the indicated service facility. Repairs will be made and the product returned, transportation prepaid. Repaired or replaced products are warranted for the balance of the original warranty period, or at least 90 days.

LIMITATION OF WARRANTY

This warranty does not apply to defects resulting from product modification without Keithley's express written consent, or misuse of any product or part. This warranty also does not apply to fuses, software, non-rechargeable batteries, damage from battery leakage, or problems arising from normal wear or failure to follow instructions.

THIS WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR USE. THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES.

NEITHER KEITHLEY INSTRUMENTS, INC. NOR ANY OF ITS EMPLOYEES SHALL BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF ITS INSTRUMENTS AND SOFTWARE EVEN IF KEITHLEY INSTRUMENTS, INC., HAS BEEN ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH DAMAGES. SUCH EXCLUDED DAMAGES SHALL INCLUDE, BUT ARE NOT LIMITED TO: COSTS OF REMOVAL AND INSTALLATION, LOSSES SUSTAINED AS THE RESULT OF INJURY TO ANY PERSON, OR DAMAGE TO PROPERTY.

KEITHLEY DAC

Data Acquisition and Control Division
Keithley Instruments, Inc. • 28775 Aurora Road • Cleveland, Ohio 44139 • (216) 248-0400 • Fax: 349-4569

WEST GERMANY:	**Keithley Instruments GmbH • Heiglhofstr. 5 • Munchen 70 • 089-71002-0 • Telex: 52-12160 • Fax: 089-7100259**
GREAT BRITAIN:	**Keithley Instruments, Ltd. • The Minster • 58, Portman Road • Reading, Berkshire RG 3 1EA • 011 44 734 575 666 • Fax: 011 44 734 596 469**
FRANCE:	**Keithley Instruments SARL • 3 Allee des Garays • B.P. 60 • 91124 Palaiseau/Z.I. • 1-6-0115 155 • Telex: 600 933 • Fax: 1-6-0117726**
NETHERLANDS:	**Keithley Instruments BV • Avelingen West 49 • 4202 MS Gorinchem • P.O. Box 559 • 4200 AN Gorinchem • 01830-35333 • Telex: 24 684 • Fax: 01830-30821**
SWITZERLAND:	**Keithley Instruments SA • Kriesbachstr. 4 • 8600 Dubendorf • 01-821-9444 • Telex: 828 472 • Fax: 0222-315366**
AUSTRIA:	**Keithley Instruments GesmbH • Rosenhugelstrasse 12 • A-1120 Vienna • (0222) 84 65 48 • Telex: 131677 • Fax: (0222) 8403997**
ITALY:	**Keithley Instruments SRL • Viale S. Gimignano 4/A • 20146 Milano • 02-4120360 or 02-4156540 • Fax: 02-4121249**

Instruction Manual
Model 500-SERIAL
RS-232 to IEEE-488 Converter

Data Aquisition and Control Division
Cleveland, Ohio, U. S. A.
Document Number: 500-SERIAL Rev. A
Publiction Date: September 1990

TABLE OF CONTENTS

SECTION 1
Introduction to the 500-SERIAL

The 500-SERIAL (Figure 1-1) is an RS-232C to IEEE-488 converter which may be used to control GPIB instruments from a terminal or computer having a standard serial port. Unlike other serial-to-GPIB converters, the 500-SERIAL is approximately the same size as a standard serial connector shell. It derives all operating power from the host's serial port, and needs no external power supply. A complete GPIB controller can be assembled using the 500-SERIAL with a personal computer or terminal (see Figure 1-2).

The 500-SERIAL connection is quite simple. The unit can operate using only the data transmit, data receive, signal ground, and DTR lines of the serial port. Hardware handshaking can be implemented by adding the RTS and CTS lines. The 500-SERIAL can control an IEEE-488 system at up to 100 ft (30m) from the computer or terminal, which exceeds normal IEEE-488 distance standards. The 500-SERIAL can also be connected to the controller through a modem for remote control of GPIB systems over even greater distances.

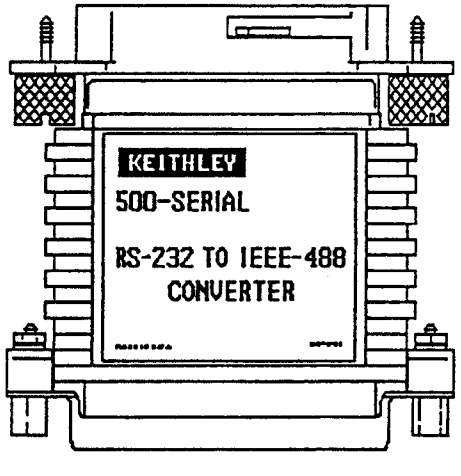

Figure 1-1. 500-SERIAL Adapter (Actual Size)

The following items are included with the 500-SERIAL (see Figure 1-2).

Model 500-SERIAL converter block

15ft (4.5m) 25-pin male to 25-pin female serial cable.

9-to-25 pin serial adapter for AT style serial ports

Demo disk (5-1/4") for use on DOS computers.

Instruction Manual

For applications assistance or service contact your local Keithley office or representative.

Introduction to the RS-232C Standard

The RS-232C standard describes mechanical and electrical specifications for an interface through which many types of devices communicate. These devices include mainframe, mini, and micro computers, data terminals, printers, modems, test and measurement equipment, and some types of laboratory instruments. Computers normally deal with data as "bytes" (8 bits of information processed in parallel). In contrast, RS-232C (also called "serial" or "asynchronous" communication) sends and receives information bit-by-bit. A basic function of the serial interface is to disassemble each byte, transmit the bits, and reassemble the byte at the receiving end. To facilitate this process, RS-232C defines voltage levels, transmission rates (baud rates), handshaking lines, parity, and the number of bits needed to send and frame data.

Most PC-based serial ports use transmit, receive, and signal ground lines plus assorted hand-shaking and status lines. The most common serial signals are shown in Table 1-1. These signals are described from the perspective of a terminal or computer connected to the 500-SERIAL. Consult the documentation for your specific terminal or computer to determine the actual RS-232C pinout.

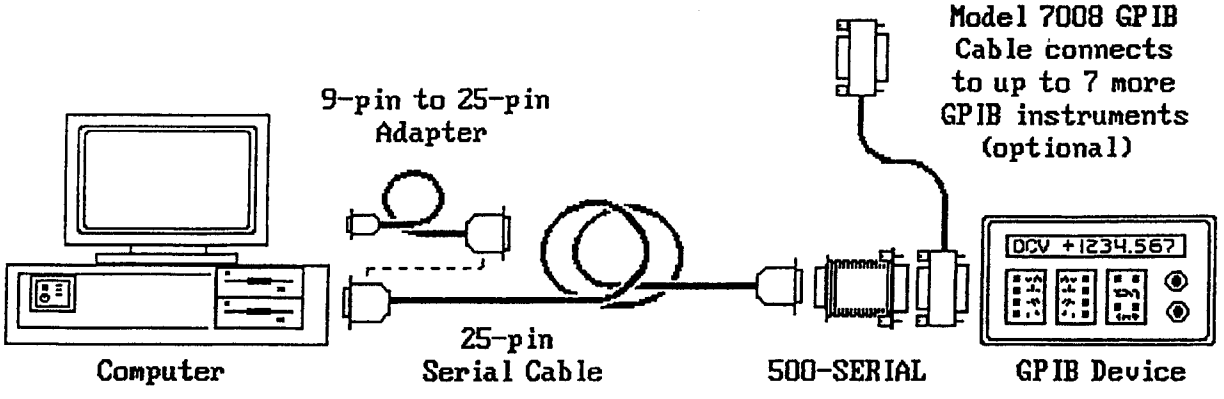

Figure 1-2. Connections with 500-SERIAL Adapter

Table 1-1. Commonly-used Serial Port Signal Lines.

Signal	Description
TRANSMIT	- Data transmit. The DTE's transmit pin is wired to the DCE's transmit pin.
RECEIVE	- Data receive. The DTE's receive pin is wired to the DCE's receive pin.
DTR	- "Data Terminal Ready"; normally used with DSR for hardware handshaking. A DTE device sets DTR ON to indicate it is ready to communicate. Some DTE devices hard-wire the DTR line ON. The 500-SERIAL uses DTR (or DSR) only for power.
DSR	- "Data Set Ready"; normally used with DTR for hardware handshaking. The DCE device must switch DSR ON before communication can proceed. The 500-SERIAL uses DSR (or DTR) only for power.
RTS	- "Request to Send". The DTE device sets RTS ON to request permission to transmit. Used with CTS for hardware handshaking.

Signal	Description
CTS	- "Clear to Send" - The DCE device must turn CTS ON to grant the DTE permission to send. Used with RTS for hardware handshaking.
GROUND	- Pin 1 on a some 25-pin ports is a "protective" ground which is tied to earth ground and the equipment chassis. Pin 7 on a 25-pin port and pin 1 on a 9-pin port are "signal" ground, which establishes ground potential for communication circuits. Signal ground is required.
RLSD or CD	- "Receive Line Signal Detect", also called "Carrier Detect". A DCE device sets RLSD ON when it receives a valid carrier signal. Typically, a modem turns RLSD ON when it is connected to a phone line and receiving a carrier signal from a remote device. The 500-SERIAL does not use RLSD.
RI	- "Ring Indicator". DCE sets RI ON to indicate that a ring signal is being received. Typically, a modem turns RLSD ON when it receives a ring signal via the phone lines. The 500-SERIAL does not use the RI pin.

A system using serial communication between the controller and instruments offers several advantages over similar IEEE-488 based systems. The RS-232C standard supports a maximum transmission distance of 50ft (15m) versus 65ft (20m) for IEEE-488. However, greater serial distances can routinely be achieved using lower baud rates and low-noise cabling. Serial ports are standard equipment on most computers and terminals, making it simple to implement communication links without resorting to add-on interfaces or expensive cabling. The configuration for GPIB control with the 500-SERIAL is also simpler than IEEE-488, consisting of only the data transmit, data receive, signal ground, CTS, RTS, and DTR or DSR lines (refer to Table 1-1).

Despite the advantages of RS-232C, variations in the standard can complicate connection and communication of devices. First, various computers and terminals use 25-pin female (DB-25S), 25-pin male (DB-25P), and 9-pin male (DB-9P) connectors for the serial port. These may necessitate adapters or "gender changers" to connect devices. Second, the serial ports of the sending and receiving devices must be initialized for the same baud rate, parity, data, and start/stop bits, or errors will occur. Third, some serial ports use a current loop rather than voltage levels, and can only be connected to another current-loop port. Fourth, some serial ports user more or fewer lines than others, necessitating special wiring at one or both ends for the devices to communicate. Fifth, equipment vendors sometimes employ unassigned or less-used pins for specialized, non-standard purposes. Such usage may conflict with serial ports on other equipment.

Finally, and perhaps most significantly, two types of serial devices exist which implement slightly different wiring in their respective ports. As the name implies, Data Terminal Equipment ("DTE") includes devices such as terminals and most personal computers operating as terminals. Data Communications Equipment ("DCE") includes peripheral devices such as modems. Any standard function implemented on a DTE connector pin has a corresponding mate of the same name on the DCE connector. For two 9-pin or 25-pin ports, the pin number will be the same, but the direction of data flow will be opposite. For example, pin 2 on 25-pin DTE and DCE are both named "transmit data", but data actually flows from pin 2 on the DTE *to* pin 2 on the DCE device. Table 1-2 shows signal names and pin-out information for the most commonly-used subsets of DTE and DCE, 25-pin and 9-pin connectors.

The different pin-out arrangements in DTE and DCE devices permit them to communicate through a "straight-through" cable where each pin on one end connects to the same pin on the other end of the cable. Connecting two DCE or two DTE devices requires a cable in which the input, output, and handshaking lines are wired to different pins at each end of the cable. Such a cable is often called a "null-modem" cable because it permits two computers having the same type of port configuration to communicate directly without use of modems. While DTE and DCE arrangements can simplify some types of serial connections, they also require the user to know and accommodate the port configurations of the serial equipment.

Table 1-2. DTE/DCE Signals, 25- and 9-pin Connectors

25-PIN	9-PIN	SIGNAL NAME
1	-	Protective Ground
2	3	Transmit
3	2	Receive
4	7	RTS (Request to Send)
5	8	CTS (Clear to Send)
6	6	DSR (Data Set Ready)
7	5	Signal Ground
8	1	RLSD (Receive Line Signal Detect)
20	4	DTR (Data Terminal Ready)
22	9	RI (Ring Indicator)

Table 1-3. Wiring the 500-SERIAL to Another DCE

SIGNAL	500S	25-PIN DCE	9-PIN DCE
Prot. Ground	1	1	-
Data Transmit	3	2	3
Data Receive	2	3	2
CTS	5	4	8
RTS	4	5	7
DTR	20	6	4
Signal Ground	7	7	5
RLSD	8	8	1
DSR	6	20	6
RI	22	22	9

The 500-SERIAL adapter is designed as a DCE device, and must be connected to a DTE device with a straight-through cable. Table 3 shows wiring information to permit the 500-SERIAL to communicate with a DCE device.

Serial Hardware Handshaking

The serial sending and receiving ports must coordinate the flow of data so that the sender does not exceed the receiver's input capacity. This situation may arise if the transmission rate exceeds the receiver's ability process and clear data from its input buffer.

The 500-SERIAL implements both Xon/Xoff and hardware handshaking. Either method can be used to assure reliable serial communication. Xon/Xoff handshaking is handled through specialized programming, and requires a detailed knowledge of the fundamentals of serial communication. Conversely, software support for hardware handshaking is handled automatically by some programming languages, including BASICA and QuickBASIC, and requires only that the feature be switched on within the 500-SERIAL. The 500-SERIAL command set includes an "H" instruction which enables hardware handshaking. This feature is implemented in the initialization programs listen in the Appendix section.

Hardware handshaking uses the CTS and RTS lines, and necessitates a minimum of six lines in the serial cable: data transmit, data receive, signal ground, RTS, CTS, and either DSR or DTR for power.

Introduction to the IEEE-488 (GPIB) Bus

The following information provides a basic introduction to the IEEE-488 bus, also known as the General Purpose Interface Bus, (or "GPIB"). For more detail, refer to other publications which offer further information on IEEE-488.

The IEEE-488 bus consists of a set of 24 lines with a well-defined electrical specification for connection schemes, signal levels, transmission speeds, and related factors. The IEEE specification also defines the mechanical features of the bus, such as connector type, shape, and thread type for the connector fastening screws.

Table 1-4. IEEE-488 Signal Lines

DIO0 - DIO7	8 lines used to pass data and bus commands between GPIB devices. Data is usually ASCII.
DAV, NDAC, NRFD	Used for "handshaking" between devices.
ATN (ATtentioN)	True means a bus command is being sent. False means data is being sent.
EOI (End Or Identify)	True means last byte of data being sent.
REN (Remote ENable)	True means remote control is enabled (devices will be controlled from the bus).
IFC (InterFace Clear)	True asserts an "Interface Clear". This resets all devices on the IEEE-488 bus.
SRQ (Service ReQuest)	True means a device has requested service.

Another part of the IEEE specification defines names, functions, and electrical specs for the signal lines. Sixteen of the lines carry data or control signals, while the rest are ground or shield lines. The signals are listed in Table 1-4. The IEEE-488 bus may have up to 14 (or in special cases, 30) devices connected. One of the devices is usually a system controller. The controller may be a dedicated GPIB controller, or a personal computer with GPIB interface card installed. Some computers use external interfaces which connect to an available port on the computer. The 500-SERIAL RS-232C to IEEE-488 converter is similar to this type of interface, but does have a significant difference. The 500-SERIAL actually connects to the GPIB instrument, effectively converting the instrument to a serial device. The 500-SERIAL provides a means of sending the instrument's usual GPIB commands through the controller's serial port. Since the RS-232C port is generally the most widely-available port on computers, the 500-SERIAL makes GPIB instruments compatible with a very wide range of computers and terminals.

Addressing on the IEEE-488 Bus

To assure reliable communication, each device on the IEEE-488 bus is assigned a unique bus address. There are 30 possible bus addresses. You will normally set the address of each device with a dip switch on the device, or by programming the address from the front panel of the instrument. Check the manual for your instruments for more information, and suggested default addresses.

System Controller

The system controller (your PC, GPIB interface, and software) coordinates which instruments send data and which receive it. To facilitate this process, each device may be put into several states. These are, "TALKER" state, "LISTENER" state, "IDLE", and "CONTROLLER".

Bus commands have been pre-defined for managing the states of instruments on the bus. They are sent by the current controller with the "ATN" line true. ATN true means a command is being sent, false means data is being sent.

The TALKER and LISTENER states are exactly what they appear to be. A TALKER sends data over the bus while a LISTENER receives it and handshakes with the talker.

More than one device may listen (or receive the data) at the same time. The IEEE-488 handshake lines operate in an open-collector mode so the handshake will not be completed until all listeners have received the data. The slowest device listening at any given time determines the data rate.

Notes:

1. All devices on the bus listen and handshake bus commands.

2. Most commands apply to all devices on the bus, but some are only acted upon by devices which are in the listener state.

3. Only listeners respond to bus data.

4. Only talker is permitted at any time.

5. Only the system controller may assert IFC and REN. This allows it to interrupt data transfers in progress.

Using the 500-SERIAL with GPIB Controllers

The 500-SERIAL is designed to be an interface in a GPIB control system where the main controlling device is a serial terminal or personal computer with serial port. While physically connected to a GPIB instrument, the 500-SERIAL is still a part of the controller. In this context, there are some modes of connection and operation which are not permitted. Specifically, the 500-SERIAL cannot be attached to serial instruments or peripherals so that they may be controlled by a true GPIB controller (such as a PC with GPIB card). Similarly, the 500-SERIAL will not permit a GPIB instrument, such as a voltmeter, to act as a controller for the purpose of sending data to a serial printer or other serial peripheral. These are examples of using the 500-SERIAL in the wrong direction. Proper connection and use are shown in Figure 1-2 and Section 2.

Notes:

SECTION 2
500-SERIAL Installation

500-SERIAL Connection

The 500-SERIAL requires an available RS-232 serial port on the computer. The actual port connector will most likely be a 25-pin or 9-pin male DB-style connector. The 500-SERIAL packages includes a 15-foot serial cable with 25-pin male and 25-pin female connectors, and a 25- to 9-pin adapter. This cable is a "straight-through" cable which permits connecting the 500-SERIAL (a DCE device) to a serial port wired as DTE equipment. If your computer or terminal has a serial port wired as DCE, you will need a special adapter. Refer to Section 1 and Appendix A of this manual for DTE/DCE pin-out information.

500-SERIAL Hardware Installation

1. Connect the male end of the supplied serial cable to the mating 25-pin connector on the 500-SERIAL block and tighten the retaining screws.

2. Connect the female end of the supplied serial cable to the serial port on an IBM or compatible personal computer (a 25-pin, male, D-shell connector is commonly used). A 25-pin to 9-pin adapter is included for connecting the cable to the 9-pin serial port found on some personal computers. Tighten the retaining screws.

3. Plug the GPIB end of the 500-SERIAL into the GPIB port of the instrument you wish to control. Alternately, you can connect the 500-SERIAL through one end of a GPIB cable, and connect up to 7 more GPIB instruments by daisy-chaining off the other end of the cable. Tighten the retaining screws.

CAUTION

The 500-SERIAL uses metric threads which can be identified by their black color. Do not attempt to use the 500-SERIAL with non-metric (silver) threads or you will strip the threads.

NOTE

The DTR and/or DSR lines in the computer's serial port supply power to the 500-SERIAL, and must be in a true (high) state for the 500-SERIAL to function. A serial cable used with the 500-SERIAL must, at minimum, contain lines for the signal ground, data send, data receive, and DTR or DSR pins. RTS and CTS are required if hardware hand-shaking will be used.

500-SERIAL Software Initialization

As a first step, any program using the 500-SERIAL must initialize the computer's serial port and the 500-SERIAL. These steps establish the serial port to be used, as well as the baud rate, start and stop bits, data bits, etc. Initialization also resets the 500-SERIAL and enables it to automatically set its own baud rate to match the rate programmed for the computer's serial port. The following steps outline the initialization process. See the example disk or the Appendix section of this manual for complete initialization routines written in Interpreter BASICA and Microsoft QuickBASIC 4.5.

INITIALIZATION STEPS FOR 500-SERIAL

The following steps should be included in any program which uses the 500-SERIAL to properly initialize the unit.

1. Set up an error handling routine which simply resumes the next program line when an error occurs.

2. Reset the DTR and DSR lines on the COM port. This can be done through programming by closing and opening the serial port, or by specifically resetting the serial port lines. See the Appendix section for programming examples.

3. Initialize the serial port. The 500-SERIAL can communicate at up to 19,200 baud. The selected baud rate also depends on what the programming language can support, and on the desired communication distance. Interpreter BASIC supports up to 9600 baud. Low-noise cable and 9600 baud (or slower) are required for distances over 50 feet.

4. Send five carriage returns separated by a 0.1 second delay to allow the 500-SERIAL to adjust its baud rate.

5. Send the following commands to the 500-SERIAL:

```
"I"       ' INIT command
"EC;0"    ' Turn echo off
"H;1"     ' Turn on hardware handshake
"X;0"     ' Turn off XON/XOFF
            handshake
"TC;2"    ' Set serial terminator to CR
"TB;4"    ' Set bus terminator to CRLF
```

6. Provide 0.1 second delay for COM port receive buffer to fill.

7. Empty the buffer of garbage and echoed characters by using the BASIC "INPUT$(LOC$)" command or its equivalent.

8. Send a "C" (Device Clear) command to clear the GPIB bus.

Note that an error handling routine (step 1) must be implemented. In interpreter BASIC, this routine consists of the statement

ON ERROR GOTO <line number>

with the statement at <line number> consisting of "RESUME NEXT". Typically, the line number for the error trap will be higher than the "END" statement which marks the end of the main program, e.g. if the END statement is line 1000, the error trap might begin at line 1100.

Error trapping is required during initialization because the 500-SERIAL may generate erroneous signals or data as it internally adjusts to the system baud rate. These errors are a normal part of initialization, and can simply be trapped and discarded. After the 500-SERIAL is initialized, the error trapping routine may be replaced with a new routine which provides useful service in response to specific error conditions.

Using the 500-SERIAL with GPIB Device Commands.

The general method of using the 500-SERIAL adapter is to open the desired serial port on the computer for input and output, and then write strings to or read strings from the port. In BASIC-type languages, the commands used for I/O include the "PRINT #" statement for output to the 500-SERIAL, and "INPUT #" or "LINE INPUT #" statements for input. The general format is as follows:

PRINT #1, "<output string>"

INPUT #1, "<input data>"

The 500-SERIAL device includes an internal processor and firmware which enables the unit to interpret and process commands and data. Most output strings sent to the 500-SERIAL will combine instructions specific to the 500-SERIAL with instructions specific to the GPIB device, although some functions will involve only commands for the 500-SERIAL. The GPIB device-specific commands can take many forms, depending on the instrument. Some commands sent through the 500-SERIAL will be general commands which affect all instruments on the GPIB bus, while others include the address of a single GPIB device to be affected.

As an example, assume that a 500-SERIAL is connected to a Keithley GPIB voltmeter at bus address 17, and that the system is being programmed in BASIC:

The command

PRINT #1, "I"

will initialize the 500-SERIAL hardware.

The command

PRINT #1, "OA;17;R3;X"

will cause the 500-SERIAL to output the command "R3" to the instrument at address 17. In this case, "OA" enables the output function of the 500-SERIAL. "R3" is the device dependent command string which sets the voltmeter range, and "X" is a general GPIB "Execute" command.

The command

PRINT #1, "EN;17"

causes any data held in the meter's output queue to be sent back to the terminal/computer.

The command

LINE INPUT #1, A$

will return a complete data string from the bus to the controller, including any embedded punctuation.

This sequence is enough to control most instruments and collect data from them after the 500-SERIAL has been initialized. Consult documentation for your chosen programming language for details on read and writing to serial ports. Example programs for Interpreter BASIC and Microsoft QuickBASIC 4.5 are included on the 500-SERIAL example diskette.

Error Handling Within the 500-SERIAL

The 500-SERIAL is designed to handle errors with a minimum effect on the controlling computer and associated GPIB instruments. Any illegal or erroneous commands sent to the 500-SERIAL will be ignored by the 500-SERIAL. Illegal addresses are trapped by the 500-SERIAL and not sent over the bus.

Notes:

SECTION 3
500-SERIAL Commands and Programming

The following ASCII commands are recognized by the 500-SERIAL processor and executed immediately upon receipt (unless embedded in a string to be sent to a GPIB instrument). Sequence Commands permit low level access to the 500-SERIAL.

COMMAND	DESCRIPTION
A	- Abort IO
C	- Device Clear
C;<addr>	- Selective Device Clear
EC;n	- Echo ON/OFF (1/0)
EO;n	- EOI ON/OFF (1/0)
EN or EN;<addr>	- ENter
H;n	- Turn hardware handshake ON/OFF
I	- Init or Reset
L or L;<addr>	- Local
LL	- Local Lockout
O;cmd$	- Output (unaddressed form)
OA;<addr>;cmd$	- Output (Addressed form)
RE	- REmote
RS	- ReSume
SP;<addr>	- Serial Poll
SQ	- SRQ or Service reQuest check
TB;n	- Bus Terminator
TC;n	- Serial Terminator
TR or TR;nn	- TRigger
X;n	- Turn Xon/Xoff protocol ON/OFF
<CTRL> Q	- XON
<CTRL> S	- XOFF
<CTRL> A	- Escape

SEQUENCE COMMANDS	DESCRIPTION
/A	- Attn
/T;<addr>	- Talk
/UT	- UnTalk
/L;<addr>	- Listen
/UL	- UnListen
/ML	- My Listen address
/MT	- My Talk address

Introduction

The command set for the 500-SERIAL can be divided into two categories: those which control or communicate only with the 500-SERIAL, and those which pass GPIB commands to or read data from the GPIB device(s) connected to the 500-SERIAL. Refer to the command descriptions in the following pages for details on each command.

Delimiters

Several of the 500-SERIAL commands consist of multiple parts. For instance, the Enter (EN) and Echo (EC) commands include an address and numeric parameter, respectively. The Addressed Output command (OA) includes a command string to be passed to the GPIB device at an address which is also specified in the command. In each case, a semi-colon (;) is used as a delimiter between the individual parts of the complete command string. Note that any device-specific command sent to the GPIB instrument is considered as one part, even though it may itself include punctuation and consist of several parameters.

The complete command is sent to the 500-SERIAL as a string. For example, in BASIC or QuickBASIC, a "PRINT" statement might assume the form:

```
PRINT #1, "OA;03;BW4,0,0,0,100"
```

In this case, #1 is the DOS device number of the serial port to which the 500-SERIAL is connected. The output string consists of everything between the quotes (" "). Semi-colons follow only OA and 03, which are instructions specific to the 500-SERIAL.

The first portion, "OA", is the addressed form of the 500-SERIAL output command. The number "03" is the address of the GPIB instrument to receive the GPIB command. The string "BW4,0,0,0,100" is the actual device-specific command intended for the instrument.

Any commands sent to any GPIB device must conform to the device's requirements for delimiters. In this case, the command "BW4" includes four additional parameters (0, 0, 0, 100) which are separated by commas.

A - Abort I/O

Format A

Purpose Generates an interface clear (IFC), and is used to reset the interface on all instruments on the bus. The 500-SERIAL takes active control (ATN true) with remote enabled (REN). This command resets the interface but not the configuration of the instruments connected to the bus. The C "DEVICE CLEAR" command must be used to reset the devices to a known state. Also see the "I" command.

Bus Sequence /REN, IFC, delay, /IFC, ATN, REN leaves ATN true

C - Device Clear

Format C

Purpose The 500-SERIAL takes control of the bus (asserts ATN) and issues the device clear command (DCL). The 500-SERIAL left in the controller active state and all instruments connected to the bus are reset to their power up or default state.

Bus Sequence ATN, DCL leaves ATN true

C;<addr> - Selective Device Clear

Format C;<addr>

Where:

<addr> = bus address. Two digits must be specified. i.e. address 1 must be entered "01".

Purpose The 500-SERIAL takes active control and "addresses to listen" the device whose address is specified. The selective device clear (SDC) is issued. This resets the addressed device to its power up or default state. The 500-SERIAL remains in the controller active state.

Bus Sequence ATN, UNL, UNT, LAG, SDC leaves ATN true

EC - Echo

Format EC;n

Where n is specified as:

1 - Turn on echo
0 - Turn off echo

Purpose This command is used when the 500-SERIAL is to be controlled by a computer program. "EC" turns off the echo of each character and disables the prompt.

EOI - End or Identify

Format

EO;n

Where n is specified as:

1 - Turn on EOI
0 - Turn off EOI

Purpose

The EOI line has two uses. EOI is "End Or Identify". The "Identify" function is for parallel polling. The "End" message is the one discussed here.

When the EOI function is enabled, the "ENTER" routine will terminate if an EOI is received with a character. On output, an EOI ("End") is sent when a terminator character is sent.

If the EOI function is disabled, then an EOI is not sent with the terminator and an EOI is ignored on input.

EN - Enter

Format

EN
EN;<addr>

Where:

<addr> = optional bus address for the addressed form of the command. Two digits must be specified. i.e. address 1 must be entered "01".

Purpose

The 500-SERIAL inputs the string from the bus without first generating any bus commands. It is assumed that there is some device on the bus to perform the handshake. If no talker is addressed or no device sends data then it will wait.

For both the addressed and unaddressed enter command, the 500-SERIAL inputs data from the bus and sends it out the RS-232 port until:

a. A terminator character exit is enabled and the terminator character is received.

b. An EOI exit is enabled and an EOI is received.

c. A <CTRL>A (01H) is received from the RS-232 port. (Send CHR$(1) from BASICA).

If an address is specified along with a string to output, the device whose address is specified is addressed to talk. The 500-SERIAL becomes an active listener and reads data into the string. The 500-SERIAL is left in the listener state.

Subsequent data may be received without specifing the address, (via the unaddressed version of the command), thus avoiding unnecessary bus sequences.

Bus Sequence

ATN, UNL, TAG, /ATN, data. Leaves ATN false, PC in listener active state, and device in talker active state.

H - Hardware Handshake

Format H;n

Where n is entered as

0 - Turn OFF hardware handshake
1 - Turn ON hardware handshake

Purpose RTS and CTS are used to stop the flow of data between the 500-SERIAL to the computer until they can catch up with the data stream. This is particularly valuable when a block of data is being received faster than it can be processed or stored on the disk.

I - Init

Format I

Purpose Generates an interface clear (IFC) and is used to reset the interface on all instruments on the bus. The 500-SERIAL takes active control (ATN true) with remote enabled (REN). This command resets the interface but not the configuration of the instruments connected to the bus. The C "DEVICE CLEAR" command must be used to reset the devices to a known state. Also see the "I" command.

Bus Sequence IFC, REN, delay, /IFC, ATN, /REN, REN leaves ATN, REN true

L - Local

Format

L
L;<addr>

Where

<addr> = optional address for the addressed form of the command. Two digits must be specified. i.e. address 1 is entered "01".

Purpose

If the address is not specified the 500-SERIAL drops (un-asserts) the remote line but otherwise generates no bus sequence. If the bus address is specified the 500-SERIAL takes active control and "addresses to listen" the device whose address is specified. The "go to local" command (GTL) is issued. This puts the addressed device in local control. The 500-SERIAL remains in the controller active state.

Bus Sequence

ATN, UNL, UNT, LAG, GTL leaves ATN true

LL - Local Lockout

Format

LL

Purpose

The 500-SERIAL takes control of the bus and issues the locallockout (LLO) command over the bus. This puts any devices which are listeners in remote only operation. (Front panel operation is disabled.)

Bus Sequence

ATN, LLO

O - Output

Format

O;cmd$ <terminator>
OA;<addr>;cmd$<terminator>

Where:

<addr> = optional address for the addressed form of the command. Two digits must be specified, i.e. address 1 is entered "01".

cmd$ = An ASCII command string.

<terminator> = the terminator sequence defined by the TB and TC commands.

Purpose

In the unaddressed mode the 500-SERIAL outputs the string over the bus without first generating any bus commands. It is asssumed that there is some device on the bus to perform the handshake. if the 500-SERIAL is in the controller active state, then the output string is a series of bus commands.

If an address is specified along with a string to output then the device whose address is specified is addressed to listen. The 500-SERIAL then becomes an active talker and sends the string as data. The 500-SERIAL is left in the talker state.

Bus Sequence

ATN, UNT, UNL, LAG, /ATN, data

RE - Remote

Format

RE
RE;<addr>

Where:

<addr> = optional address for the addressed form of the command. Two digits must be entered, i.e. address 1 must be entered "01".

Purpose

In the unaddressed form the 500-SERIAL asserts the remote line but otherwise generates no sequence. There is no change in the controller state.

If the address is form is used the 500-SERIAL takes active control and "addresses to listen" the device whose address is specified. The 500-SERIAL remains in the controller active state.

Bus Sequence

REN, ATN, UNL, UNT, LAG leaves ATN true

RS - Resume

Format

RS

Purpose

This command removes the 500-SERIAL from the active controller state. If it was addressed to talk (MTA) or listen (MLA) the it will become either a talker or listener respectively when the "RS" command is issued.

To put the 500-SERIAL into a completely idle state as far as the bus is concerned, an untalk (UNT) and unlisten (UNL) should be issued to take the it out of the talker or listener functions. The LOCAL command should be issued to un-assert the remote line. Then the RESUME command can then be issued to un-assert ATN.

Bus Sequence

/ATN leaves ATN false

SP - Serial Poll

Format SP;<addr>

Where:

<addr> = bus address. Two digits must be specified. i.e. address 1 is entered "01". The serial poll byte is returned as a two ASCII characters representing one byte in hex.

Purpose Causes the addressed instrument to return its status via the serial poll byte.

Bus Sequence ATN, UNL, TAG, SPE, /ATN, data, ATN, SPD, UNT

SQ - SRQ Check

Format SQ

Purpose This routine checks to see if a service request has been received. If a request has been received, SQ returns a "Y"; otherwise it returns an "N".

TB - Bus Termination

Format TB;n

Where n is entered as:

0 - Not used (ignored)
1 - Line Feed
2 - Carriage return
3 - Line Feed + Carriage Return
4 - Carriage Return + Line Feed

Purpose Sets the bus terminator character. The default terminator selection is 1 (LF)

The terminator is checked for in data read from the bus and if active it causes the ENTER command to quit looking for data and return when a terminator character is encountered.

TC - Serial Terminator

Format TC;n

Where n is specified as:

0 - Not used (ignored)
1 - Line Feed
2 - Carriage return
3 - Line Feed + Carriage Return
4 - Carriage Return + Line Feed

Purpose Sets the serial terminator character. The default terminator selection is 2 (CR). The terminator is checked for in data received from the serial port and terminates string data.

TR - Trigger

Format

TR
TR;<addr>

Where:

<addr> = optional address for the addressed form of the command. Two digits must be specified. i.e. address 1 is entered "01".

Purpose

In the unaddressed form of the command the 500-SERIAL takes control of the bus (asserts ATN) and issues the group execute trigger (GET) command. It is left in the controller active state and all instruments addressed as listeners will receive a trigger.

Bus Sequence

ATN, GET

In the addressed mode the 500-SERIAL takes active control and "addresses to listen" the device whose address is specified. The group execute trigger (GET) is issued. This puts the addressed device in remote control and triggers it. The 500-SERIAL remains in the controller active state.

Bus Sequence

ATN, UNL, UNT, LAG, GET leaves ATN true

X - XON/XOFF

Format

X;n

Where n is specified as:

1 - Turn on XON/XOFF feature
0 - Turn off XON/XOFF feature

Purpose

XON and XOFF are used to stop the flow of data from the 500-SERIAL to the computer until it can catch up with the data stream. This is particularly valuable when a block of data is being received faster than it can be processed or stored on the disk. An XOFF (control S) stops the data and XON (control Q) starts data again.

XON

Format **<Ctrl> Q**
17 / CHR$(17)

Purpose Restarts the transmission of data from the 500-SERIAL when it has been stopped with a <CTRL> S.

XOFF

Format **<Ctrl> S**
19 / CHR$(19)

Purpose Temporarily stops the transmission of data from the 500-SERIAL.

Escape

Format **<Ctrl> A**
CHR$(1) (from BASICA)

Purpose This character causes the 500-SERIAL to break out of whatever it was doing, flush the input buffer, and wait for a new command. This is especially useful if the bus gets "hung" waiting for a non-existant device to complete a handshake.

Attention

Format /A

Purpose The 500-SERIAL takes control of the bus (asserts ATN) It is left in the controller active state.

Bus Sequence ATN leaves ATN true

/L - Listen

Format /L;<addr>

Where:

<addr> = bus address. Two digits must be specified. i.e. addres 1 is entered as "01".

Purpose The 500-SERIAL takes control of the bus (asserts ATN) and issues the listen command (LAG) to the device whose address specified.

The 500-SERIAL is left in the controller active state. The MLA command should be used by the 500-SERIAL to address itself to listen rather than having it attempt to issue a "LISTEN" to itself.

Bus Sequence ATN, LAG leaves ATN true

/ML - My Listen Address

Format /ML

Purpose The 500-SERIAL takes control of the bus (asserts ATN). It then sets itself up to be a listener when control of the bus is released (RESUME or "RS"). No bus sequence is generated aside from asserting ATN.

Bus Sequence ATN leaves ATN true

/MT - My Talk Address

Format /MT

Purpose The 500-SERIAL takes control of the bus (asserts ATN). It then sets itself up to be a talker when control of the bus is released (RESUME or "RS").

Bus Sequence ATN, UNT leaves ATN true

/T - Talk

Format /T;<addr>

Where:

<addr> = bus address. Two digits must be specified, i.e. address 1 is entered as "01".

Purpose The 500-SERIAL takes control of the bus (asserts ATN) and issues the talk command (TAG) to the device whose address specified.

The 500-SERIAL is left in the controller active state. The MTA call should be used to address the SERIAL-500 to talk rather than having it attempt to issue a "TALK" to itself.

Bus Sequence ATN, TAG leaves ATN true

/UL - UnListen

Format /UL

Purpose The 500-SERIAL takes control of the bus (asserts ATN) and issues the unlisten (UNL) command over the bus. This instructs all listeners to get off the bus. The 500-SERIAL is left in the controller active state.

Bus Sequence ATN, UNL leaves ATN true

/UT - Untalk

Format /UT

Purpose The 500-SERIAL takes control of the bus (asserts ATN) and issues the untalk (UNT) command over the bus. This instructs all talkers to get off the bus. The PC is left in the controller active state.

Bus Sequence ATN, UNT leaves ATN true

Programming Considerations

When the 500-SERIAL first powers up it comes up with prompt. This is useful for interactive operation, but can be a nuisance when using a computer program to talk to it. It also echoes every character typed. To eliminate this and the prompt you can turn the echo off with "EC;0". The "EC" command will be echoed back and must be cleared out of the input buffer. After this however, only the expected data will be returned and nothing is echoed.

The following example applies to a PC running BASICA or QUICKBASIC and shows how to handle the autobaud and turn off the echo and communicate with the 500-SERIAL.

If the 488 bus hangs up it can be released by sending a <CTRL>A to the 500-SERIAL. This would be performed as follows:

```
PRINT #1,CHR$(1)
```

While the <CTRL>A will release the bus if it hangs, it will not completely reset the 500-SERIAL.

The 500-SERIAL can be completely reset by dropping the power lines (DTR or DSR), waiting for several seconds and raising them again. This turns power to the unit off and then on again. This can be accomplished by closing the "COM1" file, waiting and then opening it again.

The 500-SERIAL has a 120-character input buffer. Command strings sent to the 500-SERIAL should not exceed this length, especially at higher baud rates, or the string may overrun the 500-SERIAL input buffer. The 500-SERIAL supports XON and XOFF and/or hardware handshaking. These features are disabled when the unit is first powered up, and can be turned on with the H and X commands. Handshaking enables the 500-SERIAL and/or controlling PC to suspend and resume transmission in order to avoid buffer overrun.

SECTION 4
500-SERIAL Specifications and Reference

RS-232 Interface

Configuration:	DCE with full duplex, echo on or off.
Format:	8-bit ASCII data, no parity, one or two stop bits.
Baud Rates:	300, 1200, 2400, 4800, 9600, 19,200. Autobaud feature sets baud rate on carriage return. Default is 1200 baud if another speed is not detected.
Terminator:	500-SERIAL commands terminate on carriage return (CR).
Control:	Supports CTS/RTS and Xon/Xoff.
Buffer:	120-character input buffer.
Connector:	Converter accepts 25-pin Sub-D male (DB25P). Supplied cable/adapter mates to computer 25-pin or 9-pin male serial port. Low-noise RS-232 cable and 9600 baud or lower required for transmission distances over 50 feet. See EIA RS-232-C standard.
Handshake:	Hardware handshaking implemented using RTS and CTS lines.

GPIB Interface

Controller Subsets:	C1, C2, C3, C4, and C28.
Expansion:	Controls up to eight GPIB devices.
Terminator:	LF, CR, LFCR, or CRLF.
Byte Rate:	0.1 x baud rate.
Connector:	Plugs directly into GPIB port or cable.

General

Power:	Draws less than 5mA from DTR or DSR serial line, which must be set high (12V).
Environment:	0-50°C, <90% RH (non-condensing).
Dimensions:	50mm x 60mm x 25mm (2" x 2.3" x 0.9").

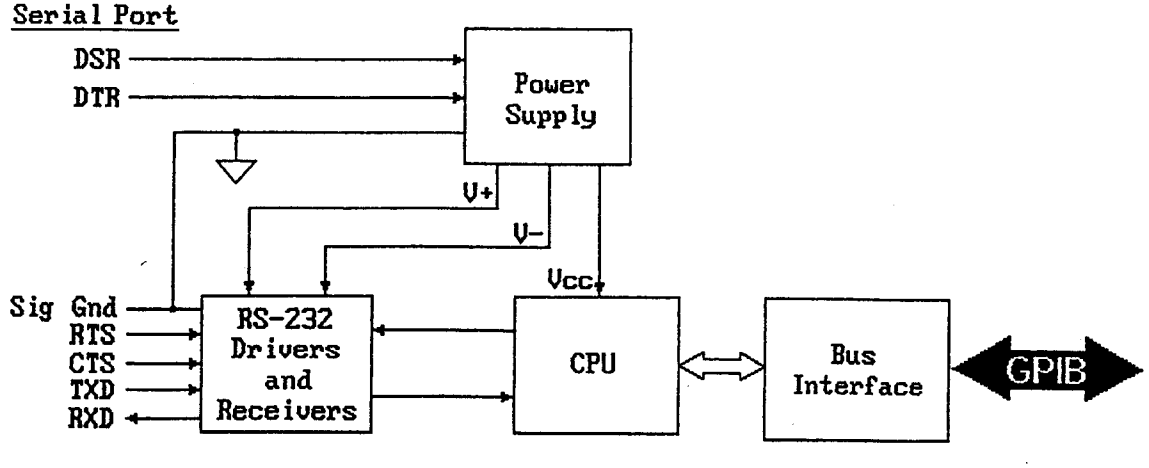

Figure 4-1. 500-SERIAL Block Diagram

Notes:

APPENDIX A
The RS-232 Connection

There are two types of RS-232 connections, DCE and DTE. DCE stand for "Data Communication Equipment", while "DTE stands for Data Terminal Equipment". The RS-232 connection on the 500-SERIAL is set up as DCE. It will plug directly into equipment with a DTE connector and configuration such as is used on most IBM-compatible personal computers.

Connecting the 500-SERIAL to another DCE requires a null modem adapter. Some null modems, whether constructed by the user or purchased commercially, may terminate or loop back the handshaking lines within each connector on the cable. While perfectly functional as a null modem cable, this type of arrangement will not work with the 500-SERIAL. Any null modem cable must pass the DSR or DTR handshaking lines through instead of terminating them. The 500-SERIAL draws its operating power from these lines. See the wiring diagram at the end of this section.

Although the new RS-232 specification calls out the type of connector to use for a DCE or DTE, there are many non-standard serial port configurations.

A DCE connector should have 9 or 25 individual pin sockets. A DTE connector should have 9 or 25 male pins. An example of a non-standard connection is a DCE wiring configuration with a DTE connector. There may also be some confusion as to the names of some pins, e.g. whether pin 2 or 3 is actually receive or transmit. This is chiefly a matter of terminology.

The RS-232C standard names pins on the DCE device according to their function on the DTE device. Thus, convention labels pin 2 as the transmit line for both DTE and DCE, but functionally, the DCE transmit pin actually receives data from the DTE. The signal direction is shown on the RS-232 pin list which follows.

As stated above the 500-SERIAL is a DCE and will connect directly to a standard DTE connector. The 500-SERIAL device gets its power from the handshaking lines. Either Data Terminal Ready (DTR) or Data Set Ready (DSR) must be in the true state (greater than +5 volts; up to +12 volts) for the 500-SERIAL to function.

RS-232 Pin-out

This is the most commonly used subset of the RS-232 lines and their names and pin numbers.

PIN	Function	DATA FLOW To DCE	DATA FLOW From DCE
1	Protective Ground	X	X
2	Transmit	X	
3	Receive		X
4	Request to Send	X	
5	Clear to Send		X
6	Data Set Ready		X
7	Signal Ground	X	X
8	Received Line Signal Detect		X
9-19	(NOT USED)		
20	Data Terminal Ready	X	
21-25	(NOT USED)		

Adapter Pin-out Information

If you need to interface the 500-SERIAL to another DCE, use the following wiring (25-pin connector assumed):

DCE		500-SERIAL
pin 2	connects to pin	3
3		2
4		5
5		4
6		20
7		7
20		6

APPENDIX B
500-SERIAL Disk and Initialization Programs

The 500-SERIAL includes a DOS-readable diskette containing example programs and other utility software. Consult the information file(s) on the disk for a detailed description of the disk's contents. The following program segments demonstrate how to initialize the serial port and 500-SERIAL device in interpreter BASIC and QuickBASIC 4.5.

```
10 '*************************************************************************
20 '*                                                                       *
30 '*  EXAMPLE INITIALIZATION FOR 500-SERIAL ADAPTER AND GWBASIC OR BASICA *
40 '*  Presumes the use of COM1 or COM2.                                    *
50 '*                                                                       *
60 '*************************************************************************
70 '
80 ' Clear screen, establish address, and set up for desired COM port
90 '
100      ON ERROR GOTO 1000
110 '
120        CLS
130 '
140        ADDR$ = "03"                    ' GPIB device primary address
150 '
160 '       OFFS% = 0                      ' COM1
170        OFFS% = 2                       ' COM2
180 '
190 ' Reset the DTR line on COM port to insure reset of the 500-serial
200 '
210         DEF SEG = &H40
220         COMPORT% = PEEK(OFFS% + 1) * 256 + PEEK(OFFS%)
230         OUT COMPORT% + 4, 0
240         DEF SEG
250 '
260 ' Open the COM port and power up the 500-serial. Wait for it to reset.
270 '
280         OPEN "COM2:9600,N,8,2,CS,DS,CD" AS #1
290 '
300         T! = TIMER
310         WHILE T! + .1 > TIMER: WEND
320 '
330 ' Send five carriage returns separated by a short delay to allow
340 ' 500-Serial to set its baud rate.
350 '
360     FOR I = 1 TO 5
370             PRINT #1, CHR$(13);
380             T = TIMER
390             WHILE T + .1 > TIMER: WEND
400     NEXT I
410 '
420 ' Turn the echo off to avoid reading echoed characters as readings.
430 '
440         PRINT #1, "I"               ' Send 'Init' command
450         PRINT #1, "EC;0"            ' turn echo off
```

```
460         PRINT #1, "H;1"          ' turn on hardware handshake
470         PRINT #1, "X;0"          ' XON/XOFF turned off
480         PRINT #1, "TC;2"         ' terminator set to CR
490         PRINT #1, "TB;4"         ' terminator set to CRLF
500 '
510 ' Wait for the receive buffer for COM port to fill
520 '
530         T = TIMER
540         WHILE T + .5 > TIMER: WEND
550 '
560 ' Now empty the buffer of garbage and echoed characters
570 '
580         A$ = INPUT$(LOC(1), #1)          ' Now empty the input buffer
590         A$ = INPUT$(LOC(1), #1)          ' of garbage and echoed characters.
600 '
610 ' Send Device Clear
620 '
630         PRINT #1, "C"
640 '

700   ' < Remainder of program here >

750 END
1000 ' Error Handler for Initialization
1010                    RESUME NEXT
```

```
'*************************************************************************
'*                                                                       *
'*  EXAMPLE INITIALIZATION FOR 500-SERIAL ADAPTER W/QUICKBASIC 4.5       *
'*  Presumes the use of COM1 or COM2.                                    *
'*                                                                       *
'*************************************************************************

' Clear screen, establish address, and set up for desired COM port

        CLS

        addr$ = "03"                    ' GPIB device primary address

'       offs% = 0                      ' COM1
        offs% = 2                       ' COM2
        com$ = "com" + LTRIM$(STR$(offs%)) + ":19200,N,8,1,BIN"

        IeeeOut = FREEFILE              ' Use next available device number
        IeeeIn = FREEFILE

        ON ERROR GOTO ErrHandler

' Reset the DTR line on COM port to insure reset of the 500-serial

        DEF SEG = &H40
        COMPORT% = PEEK(offs% + 1) * 256 + PEEK(offs%)
        OUT COMPORT% + 4, 0
        DEF SEG

' Open the COM port and power up the 500-serial. Wait for it to reset.

        OPEN com$ FOR RANDOM AS #IeeeOut

        t = TIMER
        DO WHILE t + .1 > TIMER: LOOP

' Send five carriage returns sepatated by a short delay to allow 500-Serial
' to set its baud rate.

        FOR i = 1 TO 5
                PRINT #IeeeOut, CHR$(13);
                t = TIMER
                DO WHILE t + .05 > TIMER: LOOP
        NEXT i

' Turn the echo off to avoid reading echoed characters as readings.

        PRINT #IeeeOut, "I"             ' Send 'Init' command
        PRINT #IeeeOut, "EC;0"          ' turn echo off
        PRINT #IeeeOut, "H;1"           ' turn on hardware handshake
        PRINT #IeeeOut, "X;0"           ' XON/XOFF turned off
        PRINT #IeeeOut, "TC;2"          ' terminator set to CR
        PRINT #IeeeOut, "TB;4"          ' terminator set to CRLF

' Wait for the receive buffer for COM port to fill

        t = TIMER
        DO WHILE t + .5 > TIMER: LOOP

' Now empty the buffer of garbage and echoed characters

        a$ = INPUT$(LOC(1), #IeeeIn)

' Send Device Clear
```

```
        PRINT #IeeeOut, "C"
' < Remainder of Program Here >
END
ErrHandler:
        RESUME NEXT
```

KEITHLEY METRABYTE/ASYST/DAC

Data Acquisition and Control Division
Keithley Instruments, Inc. • 28775 Aurora Road • Cleveland, Ohio 44139 • (216) 248-0400 • Fax: 349-4569

WEST GERMANY:	**Keithley Instruments GmbH • Heiglhofstr. 5 • Munchen 70 • 089-71002-0 • Telex: 52-12160 • Fax: 089-7100259**
GREAT BRITAIN:	**Keithley Instruments, Ltd. • The Minster • 58, Portman Road • Reading, Berkshire RG 3 1EA • 011 44 734 575 666 • Fax: 011 44 734 596 469**
FRANCE:	**Keithley Instruments SARL • 3 Allee des Garays • B.P. 60 • 91124 Palaiseau/Z.I. • 1-6-0115 155 • Telex: 600 933 • Fax: 1-6-0117726**
NETHERLANDS:	**Keithley Instruments BV • Avelingen West 49 • 4202 MS Gorinchem • P.O. Box 559 • 4200 AN Gorinchem • 01830-35333 • Telex: 24 684 • Fax: 01830-30821**
SWITZERLAND:	**Keithley Instruments SA • Kriesbachstr. 4 • 8600 Dubendorf • 01-821-9444 • Telex: 828 472 • Fax: 0222-315366**
AUSTRIA:	**Keithley Instruments GesmbH • Rosenhugelstrasse 12 • A-1120 Vienna • (0222) 84 65 48 • Telex: 131677 • Fax: (0222) 8403597**
ITALY:	**Keithley Instruments SRL • Viale S. Gimignano 4/A • 20146 Milano • 02-4120360 or 02-4156540 • Fax: 02-4121249**