Spectrometer Control

INTERFACING & PROGRAMMING MANUAL

SpectrAcq, DataLink, DataScan,
JY232, JY488, SPEX232, & SPEX488
TRIAX Series

Jarad (in service)

X380

800-438-7739
8 ~~4~~ 3 - 1

Part Number 80123
Revised August 12, 1997

JOBIN YVON - SPEX
Instruments S.A., Inc.

About the Manuals

You may have more than one manual, depending on your system configuration. To find the manual that has the information you need, these guidelines may help.

- Each manual generally covers a product and the features and accessories peculiar to and/or contained within that product.
- Accessories that can be applied to various products are usually covered by separate documentation.
- Software that is exclusively used with one instrument or system is normally covered in the manual for that product.
- Software that is also used with other products is covered in its own manual.
- If you are reading about a product that interacts with other products, you will be referred to other documentation as necessary.

Important Firmware Version Notice

This manual has been extensively rewritten to correspond to changes made in the firmware. These changes simplify the use of the hand-held terminal.

Programs written for Flash version 2.4 or earlier of DataScan / DataLink need to be changed. When you update to Flash version 3.0 or later, if you use the intelligent commands "p", "t", or "s".
The "p" command has an additional input; parameter #19.
The "p" command now returns an error code.
The "t" command now returns cycle# in addition to point#.
The "s" command sets scan cycle to read.

For details, refer to Section 10, Command Descriptions.

If you have a KeyLink or HandScan, to determine the firmware version present in the supporting spectrometer controller, install your hand-held terminal according to the procedure outlined in the Getting Started section of the manual that came with it. Then press the <STAT> key. If you have the current version, the display will show the version numbers for BOOT and FLASH. The FLASH number is the important one in this instance. If your hand-held terminal is connected to a DataLink / DataScan Controller- Photometer, the flash version should be 3.0 or higher. If your hand-held terminal is connected to a JY232 or SPEX232 Spectrometer Control Interface, the flash version should be 2.0 or higher. If your KeyLink or HandScan does not respond to the <STAT> key as described above, refer to the Service Policy in Section 12 for information about contacting us to arrange an upgrade of your firmware.

If you are communicating from a computer, and do not have a KeyLink or HandScan, establish communications via either RS-232 (as instructed in Section 4), or IEEE 488 (as instructed in Section 5). Once communicating, send the UTIL GET VERSION commands for boot and main as described in Section 10.1.
If your computer is connected to a DataLink / DataScan Controller- Photometer, the Main version should be 3.0 or higher. If your computer is connected to a JY232 or SPEX232 Spectrometer Control Interface, the main version should be 2.0 or higher. If your spectrometer controller does not respond as described, refer to the Service Policy in Section 12 for information about contacting us to arrange an upgrade of your firmware.

270 Spectrometer Commands:

Boot Mode Command	write Mode	Meaning	Response	Comment
Boot MOde				
O2000<null>	0	Boot	*	Need 1/2 s wait
<sp>	0	Request mode Where am I?	B=>Boot F or f=> Main	No <CR> after *
<248> (ASCII character 248 or hex F8) (no CR)		go from flash to indep mode(??)		200 ms wait
<222> in hex DE	0	Reboot if hung		200 ms wait
E#<null> # is hex		Set IEEE add	<2>	

Notes: Write: Mode 7 No EOI, Mode 0 send EOI; Read: Mode 0 No EOI

HiIQ mode or Main mode or Flash mode

Monochromator #: 0

Slit #	0	Front entrance
	1	side entrance
	2	front exit
	3	side exit

Command	Mode	Meaning	Response
A	Wrt 0	Motor Init	o or b
Wait >15s			
B0,400,800,2000<cr> B Mon, Min, Max, ramp <cr>		Set motor speed	o or b
C0<cr> Cmotor<cr>		Read motor speed	o400,800,2000<cr> or b
E	Wrt 7 Rd 0	motor busy	o or b q or z
F0,1000<cr> Fmon,steps<cr>		move motor + higher steps	o or b
G0,100000<cr> Gmon,steps<cr>		Set motor position (calibrate)	o or b
H0<cr> Hmon<cr>		Read motor position	o100000<cr> or b
K		Limits	b or o0<cr>
L	Wrt 7 Rd 0	Stop motor	o or b
W0<cr>		Open Shutter (100ms)	o or b
X0<cr>		Close Shutter (100 ms)	o or b
a0<cr>		turret Position 1	o or b
b0<cr>		Turret Position 2	o or b

c0<cr>		Set side entrance port More ent mirror to side	o or b
d0<cr>		Set front entrance port Move ent mirror to front	o or b
e0<cr>		Set side exit port Move exit mirror to side	o or b
f0<cr>		Set front exit port Move exit mirror to front	o or b
g0,0,400<cr> gmon,slit,freq<cr>		Set slit speed	o or b
h0,0<cr> hmon,slit<cr>		Read slit speed	o400<cr> or b
i0,0,0<cr> i(mono, slit,position)<cr>		Set slit position	o or b
j0,0<cr>		Read slit position	o100<cr> or b
k0,0,500<cr> kmon,slit,steps<cr>		Move slit relative	o or b
l (lower case L)	Wrt 7 Rd 0	Accessory busy	o(q or z) or b q=>busy
y<cr>	Wrt 0 Rd 1	Boot V 3.5...<cr> (Get Boot Version #)	
z<cr>	Wrt 0 Rd 1	Main V 3.5 ...<cr> (Get main version #)	
u			

CTS/RTS

Dec		hex
<58><rate> (<status><24>)	baud* (return)	<3A><baud> (<status><18>)
<27>	Echo	<1B>
<56><add> (<high><low> <status><24>)	EEPROM Read return	<38><add> [illegible]
<59><add><high><low><checksum> (<status><24>)	EEPROM Write*	<3B><add><high><low><check>
<15><#> (<status><24>)	Goto filter# *	<F><#>
<29> (<pos><status><24>)	Query Pos *	1D <[illegible]>
<255><255>	Reset	<FF><FF>
<1> (<status><24>)	Step dn *	<1>
<7> (<status><24>)	Step up *	<7>
<52> (<status><24>)	zero *	<34>

Table of Contents

1 Overview:

The JY232, SPEX232, JY488, and SPEX488 are a family of products shared by the two-spectrometer product lines of Instruments SA. For the Jobin-Yvon line, the JY- prefix is used for the control interfaces, for Spex systems, the SPEX prefix is used. Although the computer interface and commands are the same in the two product lines, the physical location of the connectors and boards varies.

There are controller-based photometers for both product lines that use an expanded version of the same command set. There are additional commands for data acquisition and control of the optional high voltage power supplies. DataLink is the name of the controller-photometer for J-Y. For the Spex line it is the DataScan.

WARNING: *The stepper motor drive electronics are not compatible* between the Spex and J-Y brands. If a DataLink is connected to a Spex monochromator control input, *damage will result.* If a DataScan is connected to a J-Y monochromator motor, *nothing will happen.*

In terms of the interfacing protocol and commands, there is no difference between the two brands. System configuration details, however, do vary depending on the different spectrometer types within either brand. This manual covers both versions of spectrometer interfaces, and generally refers to any of them as the "spectrometer controller".

Some of the commands are relevant only to particular options or components and, therefore, may or may not be used by all systems. In these cases, the command descriptions in this manual advise you of appropriate use.

The spectrometer controller interface supports the KeyLink / HandScan hand-held terminal, providing the capability to control spectral scanning as well as several optional automated devices in the spectrometer and spectrometer controller (including shutters, slits, mirrors, turrets, and high voltages).

2 Getting Started:

There are several features of this interface that provide unusual flexibility to command your system. The sections 3 through 6 outline the various methods to start communications, depending on what you are interfacing to. Please select the section that is appropriate for the method you will use to command your system.

2.1 Command Set Structure:

The spectrometer controller family supported by this command set has been designed with a multi-purpose interface. This interface will communicate with a simple ASCII "terminal" or an "intelligent" computer program.

The command set's "terminal" communications mode accommodates the limited capabilities of the HandScan / KeyLink hand-held terminals. This mode accepts commands limited to one ASCII character, and sends back ANSI format character strings to be displayed on the terminal. The spectrometer controller's internal program *bootstraps into the terminal communications mode* in order support these terminals.

When you will be sending commands from a computer program, a command can be issued to change to the "intelligent" communications mode. This stops the character strings intended for the terminal display, and enables more useful responses from the spectrometer controller.

The single character commands used by the terminal are also used in the intelligent communications mode. There are additional single and multiple character commands that are available only in the intelligent communications mode.

3 RS-232 Communication with a Hand-held Terminal:

The KeyLink / HandScan hand-held terminals are shipped with a cable (#36406) to connect to the spectrometer controller's RS-232 port or the SPEX232 / JY232 interface connector on your spectrometer or stepper motor driver. The connector on the spectrometer or spectrometer controller is a male 25 pin D-type. Depending on the instrument you have, this connector may be labeled "RS-232," "Control Input," "SPEX232," or "JY232".

Please refer to the Getting Started section of your KeyLink / HandScan manual for further instructions.

4 RS-232 Communication with a Computer:

The SPEX232 / JY232 or DataScan / DataLink spectrometer control interfaces are designed to connect to a computer equipped with an RS-232 serial communications port.

4.1 Cable Connection:

A 25 pin standard RS-232 null modem cable (our part # 97133) should be used to connect the spectrometer controller to the computer. If you are using an IBM compatible computer, for example, you may use any available COM port. In the null modem cable, connections are wired as shown. If the COM port you intend to use on your PC is a 9 pin (IBM PC/AT type), use a 9 to 25 pin adapter (our part 97134).

If you have a HandScan or KeyLink, be careful not to use the #36406 cable provided with it to connect to your computer's COM port. That is not wired as a null modem cable.

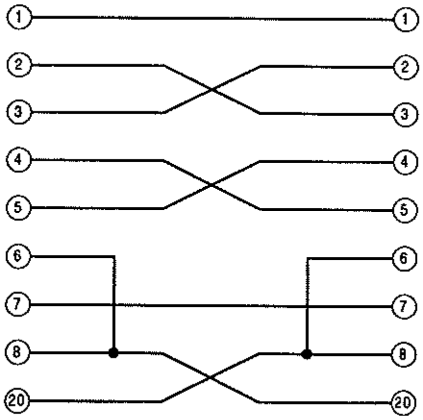


Figure 1: Null modem cable pin connections

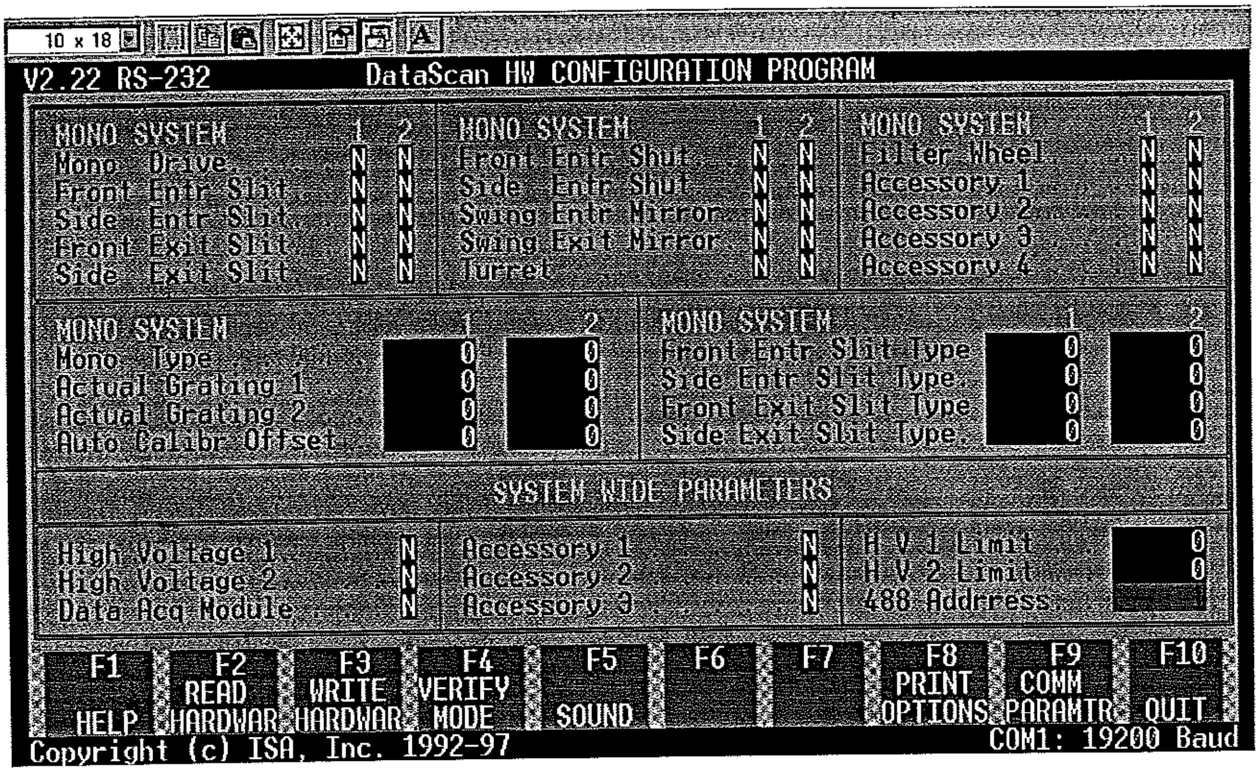
4.2 Data Bit and Handshaking Requirements:

The Spectrometer Controller communicates serially using 8 bit data bytes with 1 stop bit. There is no parity checking. Your PC should assert DTR when unable to accept data from the spectrometer controller. Refer to Appendix 2 for connector pin assignments.

4.3 Testing Communications:

To set up to control your spectrometer over the RS-232 interface, we strongly recommend the following tests be performed with the sample programs provided, using an IBM compatible PC. Especially if you intend to use a different type of computer, we stress it is important to borrow a PC and complete these tests. For peace of mind, it is better to build on a solid foundation. Please assure yourself first that the hardware and example programs are functional. In this way, you can be sure that all is well before starting to translate and modify the sample programs for your own application.

We provide a support diskette with these spectrometer controllers, which includes program examples to communicate with a DOS PC. Insert the support diskette into your floppy drive, and at the DOS prompt, type "A:<Enter>" to access the floppy. (Or use "B:<Enter>" if appropriate for the floppy drive that you are using.) The first program you should run to verify communication to the spectrometer controller is in the UTILITY directory. Type "cd\utility" followed by "<Enter>" to access the directory. The program name is HWCFG232. This is a working program that has been in routine use for configuration at the factory. To run this program: the COM port to be used, and the baud rate should be set. You should use COM 1 or COM 2, as the program requires interrupts. Type "hwcfg232" followed by "<Enter>" to start the configuration program. A screen will appear showing the communications port and baud rate settings. If these are correct, press "<Enter>" otherwise change the settings as needed and then press <Enter>. (When shipped from the factory, the default settings are for COM port 1, at 19200 baud). If the interface is properly installed and functional, the program will establish communications and read back the present configuration from the spectrometer controller. If communication fails, an error message with appropriate advice to correct the problem will be displayed.



```
V2.22 RS-232            DataScan HW CONFIGURATION PROGRAM

MONO SYSTEM          1  2 | MONO SYSTEM          1  2 | MONO SYSTEM        1  2
Mono  Drive.......   N  N | Front Entr Shut...   N  N | Filter Wheel....   N  N
Front Entr Slit...   N  N | Side  Entr Shut...   N  N | Accessory 1.....   N  N
Side  Entr Slit...   N  N | Swing Entr Mirror    N  N | Accessory 2.....   N  N
Front Exit Slit...   N  N | Swing Exit Mirror    N  N | Accessory 3.....   N  N
Side  Exit Slit...   N  N | Turret............   N  N | Accessory 4.....   N  N

MONO SYSTEM                1      2 | MONO SYSTEM                 1      2
Mono  Type.........        0      0 | Front Entr Slit Type        0      0
Actual Grating 1...        0      0 | Side Entr Slit Type.        0      0
Actual Grating 2...        0      0 | Front Exit Slit Type        0      0
Auto Calibr Offset.        0      0 | Side Exit Slit Type.        0      0

                        SYSTEM WIDE PARAMETERS

High Voltage 1....  N | Accessory 1..........  N | H V 1 Limit.....     0
High Voltage 2....  N | Accessory 2..........  N | H V 2 Limit.....     0
Data Acq Module...  N | Accessory 3..........  N | 488 Address.....

  F1      F2       F3      F4      F5     F6    F7     F8       F9      F10
         READ    WRITE   VERIFY                       PRINT    COMM
 HELP  HARDWAR  HARDWAR   MODE    SOUND              OPTIONS  PARAMTR   QUIT
Copyright (c) ISA, Inc. 1992-97                          COM1: 19200 Baud
```

Review the configuration on the screen. At the factory, this was set up to match your system. A record of the configuration was printed by using the <F8> function. This page was included with the documentation shipped with the spectrometer controller. Normally, there will be no further need to use this program, unless you replace gratings or need to change the maximum high voltage limit, if you have that option. You may want to familiarize yourself with some of the parameters and options on the screen, such as grating g/mm, and HV limit. Press <F10> to exit the program

4.4 Automatic Baud Rate Selection:

Serial communications with the spectrometer controller can be started without concern about baud rate in most cases. When the spectrometer controller is powered on, it bootstraps itself into an idle condition, ready to "Autobaud" or match the baud rate that the terminal or host computer is communicating with. This can be accomplished by sending a space character (ASCII char 32 in DECIMAL, 20 in HEX) to the spectrometer controller. (HandScan / KeyLink users please note that the **Decimal Point** key will send a "<Space>".) The spectrometer controller automatically selects the baud rate to match the device or computer that is communicating with it. The compatible baud rates are: 1200, 2400, 4800, 9600, and 19200. Because of this feature, most computers and terminals can be simply plugged in and used. The necessity of setting baud rate switches has been eliminated. The first character received is compared with stored data to determine its baud rate. However, due to allowances in the EIA RS-232C standard, it is not always possible to make a proper baud match on the first attempt. You may have to send the space character more than once. When this is done, and the autobaud has been successful, the spectrometer controller will send the asterisk character "*" back. Under some conditions, the "*" may be followed by more characters. If the spectrometer controller is already running, sending "<Space>" will get an F, B, or string response. If the spectrometer controller is already running and autobauded, it will not re-autobaud. It is important to remember this when switching programs on the host computer. For example: After running HWCFG232 at 9600 baud, you want to run the example program SCAN_232.BAS. But the Basic program runs at 4800 baud. It will not communicate unless the spectrometer controller is powered off, then back on. This is the only way to set the baud rate.

4.5 Preparing to program via RS-232.

To help you get started writing routines with the Programmer's Command Set; we provide example programs on the Support Diskette. Excerpts of these examples can be used as building blocks for your programs. In particular, when opening communications, careful attention should be paid to insure that the procedure used in the example is followed. The example GWBASIC programs SCAN_232.BAS and 232_SCAN.BAS are also provided in the \USERPROG directory of the support diskette. GWBASIC or QBASIC must be available on your computer to run these programs. There is a READ.ME ASCII text file on the support diskette that may contain further, updated information.

Note: These programs will not work with the TRIAX series instruments. See section 4.6.

Run these programs and use them to control your spectrometer. Get a feel for how the system responds to commands by watching the drive move on your spectrometer.

4.6 Preparing to program via RS-232 for TRIAX Series

4.7 RS-232 Communications Startup:

The part of your program that will establish communications must follow the steps outlined in the "RS-232 Start Up Procedure" flow chart. The example BASIC programs provided on the support diskette and listed in Appendix 3 include working examples of the startup procedure.

It is important to understand that the command set's internal program bootstraps into the terminal communications mode to allow the spectrometer controller to be used with a "dumb terminal", such as the HandScan / KeyLink. In this mode, the spectrometer controller accepts single ASCII character commands, and returns strings of ASCII characters preceded by the ANSI <Escape> X, Y sequence for character positioning on the terminal screen.

When starting RS-232 communications with the spectrometer controller, your program should follow the example outlined in the flowchart on the following pages. You must determine if a baud rate has been previously established, and if so which program and communication mode is active. "Flush buffer" instructions are needed to clear characters from a previous run or to clear out the terminal communications mode display prompts. The spectrometer controller will send these prompts until you establish yourself as a computer program by switching to the intelligent communications mode.

Notes on RS-232 Start Up Procedures

A: Send Autobaud character "<space>"; response will be "*" if this is the first time power up or after a re-boot.
OR
WHERE AM I command "<space>", valid if the spectrometer controller has already been started; response will be "B" (for BOOT) or "F" (for MAIN), depending on the previous state of the spectrometer controller.

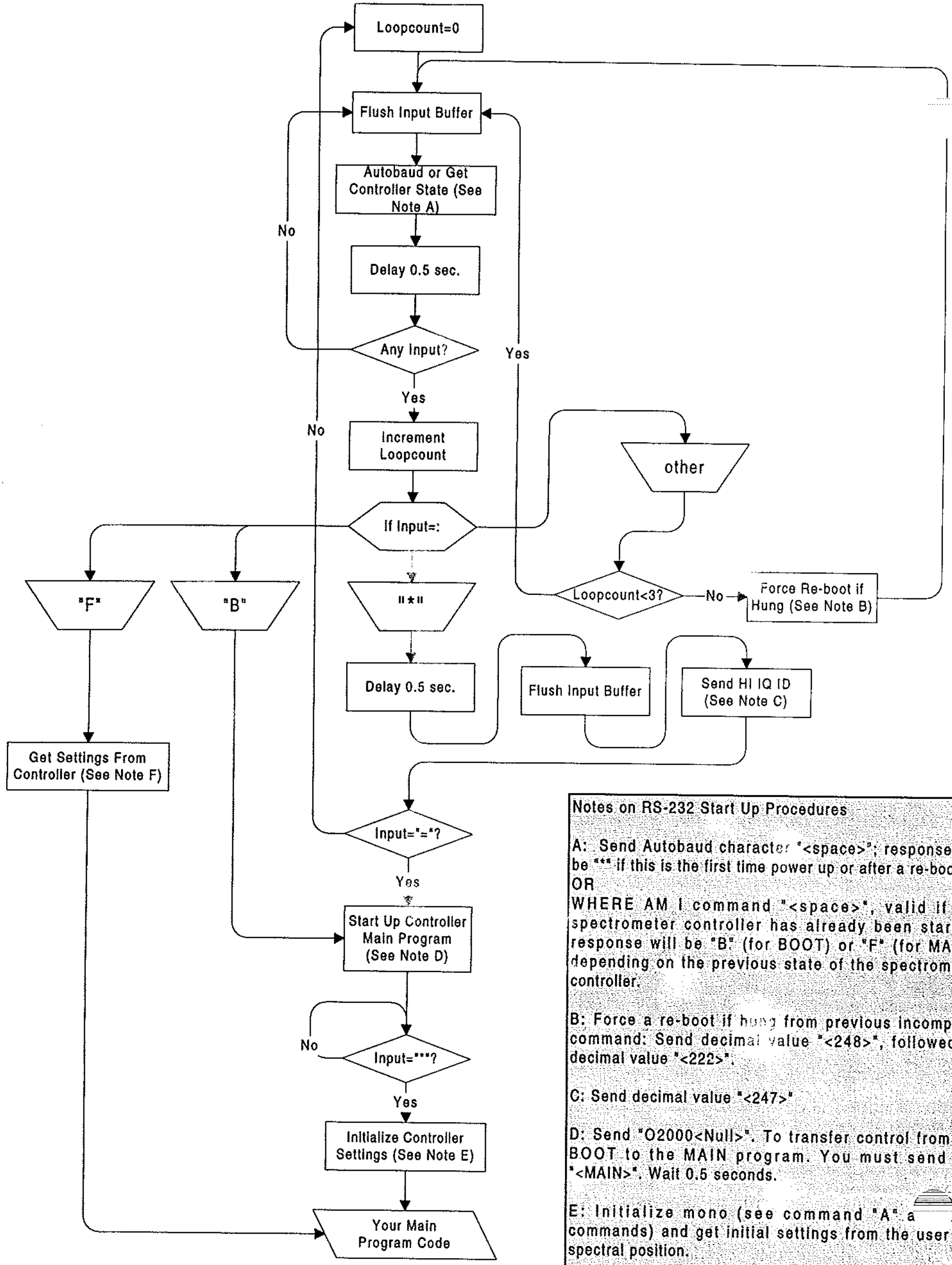
B: Force a re-boot if hung from previous incomplete command: Send decimal value "<248>", followed by decimal value "<222>".

C: Send decimal value "<247>"

D: Send "O2000<Null>". To transfer control from the BOOT to the MAIN program. You must send the "<MAIN>". Wait 0.5 seconds.

E: Initialize mono (see command "A" and SET commands) and get initial settings from the user e.g. spectral position.

F: You can read your last position etc. from the spectrometer controller. You do not have to re-initialize the spectrometer.


Loopcount=0
Flush Input Buffer
Autobaud or Get Controller State (See Note A)
Delay 0.5 sec.
No
Any Input?
Yes
Yes
Increment Loopcount
No
other
If Input=:
"F"
"B"
"*"
Loopcount<3?
No
Force Re-boot if Hung (See Note B)
Delay 0.5 sec.
Flush Input Buffer
Send HI IQ ID (See Note C)
Get Settings From Controller (See Note F)
Input="="?
Yes
Start Up Controller Main Program (See Note D)
No
Input="*"?
Yes
Initialize Controller Settings (See Note E)
Your Main Program Code

Notes on RS-232 Start Up Procedures

A: Send Autobaud character "<space>"; response will be "*" if this is the first time power up or after a re-boot.
OR
WHERE AM I command "<space>", valid if th spectrometer controller has already been started; response will be "B" (for BOOT) or "F" (for MAIN depending on the previous state of the spectromet controller.

B: Force a re-boot if hung from previous incomple command: Send decimal value "<248>", followed t decimal value "<222>".

C: Send decimal value "<247>"

D: Send "O2000<Null>". To transfer control from the BOOT to the MAIN program. You must send th "<MAIN>". Wait 0.5 seconds.

E: Initialize mono (see command "A" a [illegible]ET commands) and get initial settings from the user e spectral position.

F: You can read your last position etc. from the spectrometer controller. You do not have to re-initializ the spectrometer.

To Autobaud (establish the communications baud rate), follow this procedure:

1 Send the autobaud character "<Space>"

2 If you receive "*", go to step 3.
If you receive "B", go to step 5
If you receive "F", go to step 6
If you receive "<Escape>", go to step 8
If you receive any other character or nothing, try again.

3 If this is the first communication after its power-up or bootstrap, the spectrometer controller will respond with "*" when it recognizes your baud rate. Due to allowances in the RS-232 standard, it may be necessary to send the "<Space>" more than once before receiving the "*". A string of characters intended for the hand-held terminal display will also be sent following the "*". Flush these extra characters. If you receive some character other than "*", skip to step 4.

Once the "*" is received, communications is established. Send "<247>" as one byte. This is acknowledged by the character "=". This switches the spectrometer controller to the intelligent communications mode, and turns off the strings of response characters intended for the hand-held terminal. The <247> command is only recognized at this point during the start-up sequence. If any other character is received at this point in the sequence, the spectrometer controller assumes that it is connected to a terminal, and will respond accordingly. Follow this by sending "<Space>". You should receive a "B": If so, go to step 5. If not, go to step 7.

4 If the spectrometer controller has completed the autobaud sequence at some previous time, it will respond to receiving the <Space> differently from the above, sending back a character which tells you which internal program is running in the spectrometer controller. If you receive "B" for boot, go to step 5, if you receive "F" for main, go to step 6. If you receive <Escape> plus a string of characters, go to step 8. If you do not receive any of these responses, go to step 7.

5 If you receive "B", you are talking to the BOOT program. Send "O2ØØØ<Null>" to transfer to the spectrometer controller's MAIN program. Then wait 0.5 seconds to be sure that the main program is ready to accept additional commands before proceeding. Follow this with "<Space>". You should receive an "F". If so, go to step 6. If you receive <Escape> plus a string of characters, go to step 8. If you do not receive either of these responses, go to step 7.

6 If you receive the "F" character, you are talking in intelligent communications mode to the MAIN program. The spectrometer controller is ready to accept commands. Receipt of this character also implies that a program has previously run the spectrometer controller. In this case you may not have to send the INIT and SET commands described later. Instead, you may wish to read back the previous positions from the spectrometer controller into your program.

7 If the spectrometer controller program is not responding for any reason, you may force a re-boot by sending "<248>" followed by "<222>", both as single bytes. The <248> sets the spectrometer controller in Intelligent communications mode, in case it was not in that mode. The <222> re-boots the spectrometer controller program from most, but not all hung conditions. However, if you wish to change baud rates, you must turn the spectrometer controller power off, then on again, and go to step 1.

8 If you receive <Escape> plus a string of characters, communications has been previously established in hand-held terminal communications mode. Send "<248>", pause, and then send "<Space>". You should receive an "F". If so, go to step 6. If not, go to step 7.

If you wish to alternate control between the computer and a KeyLink / HandScan terminal, this can be accommodated. The computer will have to communicate at the same baud rate as the hand-held terminal (19,200 baud). A commercial RS-232 A/B switch box can be connected with the common connection to the spectrometer controller. Connect the hand-held terminal and computer to the A and B connections. Start up the spectrometer controller connected to the hand-held terminal, following the instruction manual for the hand-held terminal. Switch the cable connection to the computer, and then change to the intelligent communications mode by sending the pseudo-command <248> from the computer at any time after you are in the main program. The spectrometer controller sends no response character for a pseudo-command.

To change back to the hand-held terminal, your computer should issue the "Y" command "go to terminal communications mode" and receive the "o" response before you physically switch the connection back to the hand-held terminal.

More detailed information on alternating control between the hand-held terminal and your computer can be found in the UTIL command descriptions in Section 10. Information on switching between IEEE 488 and RS-232 is also in the UTIL commands section.

4.8 Re-booting a Spectrometer Controller

If your program exits before you complete a command, the spectrometer controller will hang. It will wait for the rest of the command input. Re-powering the spectrometer controller can clear this condition.

Assuming that you are communicating in the intelligent mode, you may send the pseudo-command "<222>" as a single byte to re-boot the spectrometer controller. The <222> is ignored if the spectrometer controller is not hung waiting for additional parameters. Pseudo-commands do not cause the spectrometer controller to send back any response character.

If you are in the terminal communications mode, you may force a re-boot by sending the pseudo-commands "<248>" and "<222>", in that sequence, both as single bytes. The "<248>" first insures that the spectrometer controller is in the intelligent communications mode to accept the <222>.

You may want to re-boot the spectrometer controller at some time, without manually turning it off and back on. To do this, intentionally send it an incomplete command. For example, send "G", MOTOR SET POSITION without specifying the position. Follow with the pseudo-command "<222>". This will put you at the beginning of the start-up sequence.

5 IEEE 488 Communication with a Computer:

If you have a JY488 / SPEX488 interface for your spectrometer, or a DL488 / DS488 option in your controller-photometer you may communicate with the spectrometer controller via the General Purpose Interface Bus (GPIB).

Please refer to the documentation provided with your computer's IEEE 488 interface for information about how to send the various characters.

IEEE 488 can take command of the spectrometer controller at any time by asserting the REM line. If IEEE 488 is initializing the system, the entire auto-baud sequence used for serial communication (as described on page 9, 14) is skipped. The system is automatically forced into the intelligent communications mode, where the spectrometer controller expects to receive the commands as outlined later in the Command Descriptions in Section 10. You must also send "O2ØØØ<Null>", to transfer control to the main program that resides in the spectrometer controller. Then wait 0.5 seconds to be sure that the main program is ready to accept additional commands before proceeding.

If you desire to change the IEEE 488 address, the command to do so must be issued instead of the "O2ØØØ<Null>" command. (I.e. the CHANGE IEEE 488 ADDRESS command, described later, is valid when the spectrometer controller is in the system initialization portion of its boot program, not in the main program). The new address will be saved by the spectrometer controller in non-volatile memory. When you change the address, you should POWER OFF the spectrometer controller and make the necessary changes to your program to reflect this address change. Then when you POWER ON the spectrometer controller and restart your program, access the spectrometer controller at the new address.

The optional 488 interface is designed as an IEEE 488 Talker/Listener. The primary IEEE 488 address is set at the factory to 1, however, it can be reconfigured with a command to set the address from 1 to 31 (see the UTIL CHANGE 488 ADDRESS command on page 47).

Note that the IEEE address can also be changed via RS-232, if your spectrometer controller is so equipped. Use the HWCFG232.EXE program on the support diskette if you are using a DOS computer, or, after establishing RS-232 communications, send the UTIL CHANGE IEEE 488 ADDRESS command from your own program.

The support diskette provided with each DataLink / DataScan, JY488 or SPEX488 includes utility programs and BASIC program examples to communicate with a DOS PC.

5.1 Supported IEEE 488 Computer Interface Boards:

The IEEE 488 example programs work with several of the National Instruments PC interface boards.

- GPIB-PCII/PCIIA 488.2 Interface board: The driver supplied by National should be version 1.2 or newer, and their BASIC support disk should be version 2.Ø or newer.

- Older GPIB-PCIIA boards require National's revision C13 or newer software.

- AT-GPIB 488 boards require National's revision E7 or newer software.

- AT-GPIB 488.2 board: Must be version 2.1.1 software, and their BASIC support disk version 2.2 or newer.

- GPIB-PCIII board: this model must be replaced with one of the above. Contact your local National Instruments dealer with regard to their current replacement program.

There are other boards by National and other suppliers for IBM compatible and MacIntosh computers. Many of these boards can function in *SIMILAR* fashion, and these instructions *MAY* be helpful in setting them up. As we cannot support or guarantee reliable communications with other boards and software, we strongly recommend that you use the National Instruments products as described above.

5.2 Establishing GPIB Communications:

The IEEE board and its associated software driver must be installed in your computer as per National Instruments' instructions. The driver must be installed via your computer's CONFIG.SYS file.

This set up assumes that the National Instruments software driver has the PCII board name as GPIBØ and the first device name as DEV1. Run the IBCONF program and set the configuration as follows:

GPIBØ	
Primary GPIB Address	Ø
Secondary GPIB Address	None
Timeout Setting	T3s
EOS byte	ØØH
Terminate read on EOS	no
Set EOI with EOS on write	no
Type of compare on EOS	7-bit
Set EOI w/ last byte of write	yes
System controller	yes
Assert REN when SC	no
Enable auto serial polling	yes
Timing	5ØØ nanoseconds
Enable 488.2 protocols	yes
CIC protocols	no
Handler Type	PC2
Interrupt setting	none
Base I/O address	Ø2B8H
DMA channel	1

DEV1	
Primary GPIB address	1
Secondary GPIB address	None
Timeout Setting	T1Øs
EOS byte	ØDh ← 13 (CR)
Terminate read on EOS	yes
Set EOI with EOS on write	no
Type of compare on EOS	7-bit
Set EOI w/ last byte of write	yes
Repeat addressing	no

The above are default settings. Refer to the example program listings of 488_SCAN.BAS and SCAN_488.BAS for details.

The part of your program that will establish communications must follow the steps outlined in the "IEEE 488 Start Up Procedure" flow chart. The example BASIC programs provided on the support diskette and listed in Appendix 142 include working examples of the startup procedure.
Note that timeout settings vary by command. In most cases 300 milliseconds is sufficient. Longer timeout recommendations are given, for the commands that need them, in Section 10: Command Descriptions.

Note that single byte reads rely on an explicit byte count of 1 to terminate, rather than EOI. Multi-byte reads assume that the EOS character is <CR>. See the 488SCAN.BAS listing in Appendix 3. You will find this near the variable declarations at the beginning of the program.

- ICH$ and ACK$ are 1 byte program variables
- • RBUF$ is set to 132 characters to handle the largest expected string
- The spectrometer controller does not assert EOI

For examples of low level IEEE 488 bus signals to use and expect when issuing commands, refer to Appendix 5 on page 176

5.3 To test GPIB communications:

This test assumes that you have correctly completed the GPIB set-up above.

Connect your GPIB cable between the National Instruments board in your computer and the SPEX488 or DS488. If other devices are to be used on the same bus, it is recommended that they be disconnected temporarily, to reduce the possible sources of problems while establishing communications for the first time.

From the directory where the National Instruments programs reside, Run IBIC, and issue the following commands at the prompt.

CAUTION: ***Do not press the <F3> key, as this will change the configuration in your spectrometer controller.***

IBFIND GPIB0	Finds the PCII board in the PC
IBFIND DEV1	Finds the DS488 or SPEX 488 at address 1
IBWRT " "	Send one space character (must be enclosed in Quotation marks)
IBRD 1	Read 1 character. (Should come back a 'b')
IBWRT "O2000\x0"	Starts main program
IBRD 1	Read 1 character. (Should come back a '*')
IBWRT " "	Send one space character (must be enclosed in Quotation marks)
IBRD 1	Read 1 character. (Should come back an 'F' or 'f')
IBWRT "A"	Begins initialization.

If you have successfully communicated, you should receive a B or a F. If you have an error message displayed, refer to the National Instruments documentation to interpret it.

Spectrometer Control

To gain further assurance of communication to the spectrometer controller, the following test will read the configuration from the nonvolatile memory in the DS488 or SPEX488.

Insert the DataScan \SPEX232\ SPEX488 support diskette into your Floppy Drive, and at the DOS prompt, type "A:<Enter>" to access the floppy. (Use "B:<Enter>", if appropriate for your computer.)

Type "cd\utility\ieee488" followed by "<Enter>". Then type "hwcfg488 1" followed by"<Enter>". You should see the HW CONFIGURATION screen on your computer's monitor.
Press the <F2> key to read the hardware configuration from the non-volatile memory in the DataScan or SPEX232. If the GPIB interface is properly installed and functional, the program will read and display the settings from your system. If communication fails, refer to the National Instruments documentation for further troubleshooting information. It is important to keep in mind that this is a working program that has been in routine use for configuration at the factory, and will access the DS488 or SPEX488 if the National Instruments board is properly installed and functional.

Review the configuration on the screen. At the factory, this was set up to match your system. A record of the configuration was printed by using the <F8> function. This was included with the documentation shipped with the spectrometer controller. Normally, there will be no further need to use this program, unless you replace gratings or need to limit the maximum high voltage if you have a DataScan / DataLink. You may want to familiarize yourself with some of the parameters and options on the screen, such as grating g/mm, and HV limit. Press < F10> to exit the program.

5.4 To change the GPIB address:

The J-Y and Spex spectrometer controllers are each shipped from the factory with a default device address of 1. To change the address, insert the DataLink / JY232 / JY488 or DataScan / SPEX232 / SPEX488 support diskette into your Floppy Drive, and at the DOS prompt, type "A:<Enter>" to access the floppy. (Use "B:<Enter>", if appropriate for your computer.)

Type "cd\utility\ieee488" followed by "<Enter>". Then type "hwcfg488 1" followed by "<Enter>".

You should see the HW CONFIGURATION screen on your computer's monitor. Press <F2> to read the hardware configuration from the address shown in the lower right corner of the screen.

If you wish to change the address, move the cursor to that field and enter the new number. Any address from 1 to 31 is valid. However, depending on the configuration of your PC interface board you may be limited to 1 to 15. To write the new address into non-volatile memory in the SPEX488 or DS488, press <F3>. To verify that it was accepted, press <F2>, and see the address that is returned on the screen. When the spectrometer controller is powered off, then on again, the new address will become active. Subsequent communication must be to the new address.

An alternative method of changing the GPIB address is possible for DataScan and the SPEX232/488 interface. You may write the new address over your computer's serial COM port via RS-232. A null modem cable (Spex P/N 97133) should be used.

To use the serial port, a similar program is provided on the support disk, in the UTILITY directory. The program name is HWCFG232. To run this program, the COM port to be used, and the baud rate should be specified.

On the support diskette, access the subdirectory A:\UTILITY and execute the program by typing "hwcfg232" A screen will appear showing the communications port and baud rate settings. If these are correct, press, otherwise change the settings as needed. and press <Enter>. (When shipped from the factory, the default settings are for COM port 1, at 19200 baud). The program will establish communications and read back the present configuration from the spectrometer controller. This configuration will then be displayed in the HW CONFIGURATION screen on your computer's monitor. You can now change the address as in the above IEEE488 example.

5.5 Preparing to program via IEEE 488:

After success in communicating using the National Instruments hardware and software you may proceed to command the spectrometer controller using the Programmer's Command Set.

Due to National Instruments' licensing restrictions, we cannot distribute ready to run source code example programs for the SPEX488 and DS488 interface. The GWBASIC programs SCAN_488.BAS and 488_SCAN.BAS provided on the support diskette must be merged with DECL.BAS from National Instruments, and National's BIB.M must be in the same directory as the SCAN_488 and 488_SCAN programs. GWBASIC must be available on your computer as well. If you are using a compiler other than GWBASIC, check the National Instruments support disk for the appropriate files.

There is a READ.ME file on the DataScan support diskette that may contain further, updated information.

When these programs are executed, you will be prompted to enter the GPIB address. Unless you have changed the address as described earlier, the address will be 1, as shipped from the factory.

SCAN_488.BAS is a simple program intended to demonstrate proper usage of the various commands by example. It includes usage of DataScan acquisition and high voltage commands in the intelligent communications mode. You will see how to set and read motor speeds, read the firmware version number from the FLASH RAM, as well as other commands.

488_SCAN.BAS is a spectrometer control program that has been used at the factory to test systems like yours. It offers a formatted input screen to enter parameters and execute a scan routine. Conversions from working spectral units to motor steps are performed by the program. Data acquisition commands are not included, but you may easily add them.

Please understand that these are rudimentary example programs provided free of charge. They have been tested within reason. They are intended as references for programmers to excerpt from. We do not guarantee or support these in the same way as our complete software packages. They are intended as examples only. If you construct programs based on these Command Set examples, we strongly recommend that you add prudent error trapping and protection features to your program to protect your system and enhance ease-of-use. Much aggravation can be avoided by such things as limits on stepper motor travel and speed, as well as other variables, and validity checking on commands.

5.6 IEEE 488 Communications Startup:

The part of your program that will establish communications must follow the steps outlined in the IEEE 488 Start Up Procedure flow chart on page 26.

It is important to understand that the command set's internal program bootstraps into the RS-232 terminal communications mode. In this mode, the spectrometer controller accepts single ASCII character commands, and responds sending ASCII characters preceded by the ANSI <Escape> X, Y sequence for character positioning on the terminal screen.

Spectrometer Control

When the spectrometer controller is addressed on the IEEE 488 bus, it automatically sets itself in the intelligent communications mode and you need only establish whether you are talking to the BOOT or the MAIN internal spectrometer controller program.

Send the `where am I' command, "<Space>", and the spectrometer controller will respond with "B" for boot, or "F" for main. This tells you which internal program is running in the spectrometer controller.

If you receive "B", you are talking to the BOOT program. Send "O2000" plus the "<Null>" character to transfer to the spectrometer controller's MAIN program. Wait 0.5 second to be sure that the main program is ready to accept additional commands before proceeding.

If you receive the "F" character, you are talking to the MAIN program. This means that a program has previously run the spectrometer controller. In this case you may not have to send the INIT and SET commands described later. Instead, you may wish to read back the previous positions into your program.

In intelligent communications mode, the terminal prompts are turned off. Commands are acknowledged with single characters.

The flush buffer commands clear characters from a previous run or clear out the ASCII terminal communications mode prompts that are sent by the spectrometer controller until you establish yourself as a computer program by switching to the intelligent communications mode.

5.7 Re-booting a Spectrometer Controller

If your program exits before you complete a command, the spectrometer controller will hang. It will wait for the rest of the command input. Re-powering the spectrometer controller can clear this condition.

Assuming that you are communicating in the intelligent mode, you may send the pseudo-command "<222>" as a single byte to re-boot the spectrometer controller. The <222> is ignored if the spectrometer controller is not hung waiting for additional parameters. Pseudo-commands do not cause the spectrometer controller to send back any response character.

If you are in the terminal communications mode, you may force a re-boot by sending the pseudo-commands "<248>" and "<222>", in that sequence, both as single bytes. This first insures that the spectrometer controller is in the intelligent communications mode to accept the <222>.

You may want to re-boot the spectrometer controller at some time, without manually turning it off and back on. To do this, intentionally send it an incomplete command. For example, send "G", MOTOR SET POSITION without specifying the position. Follow with the pseudo-command "<222>". This will put you at the beginning of the start-up sequence.

Spectrometer Control

Notes on IEEE-488 Start Up Procedure

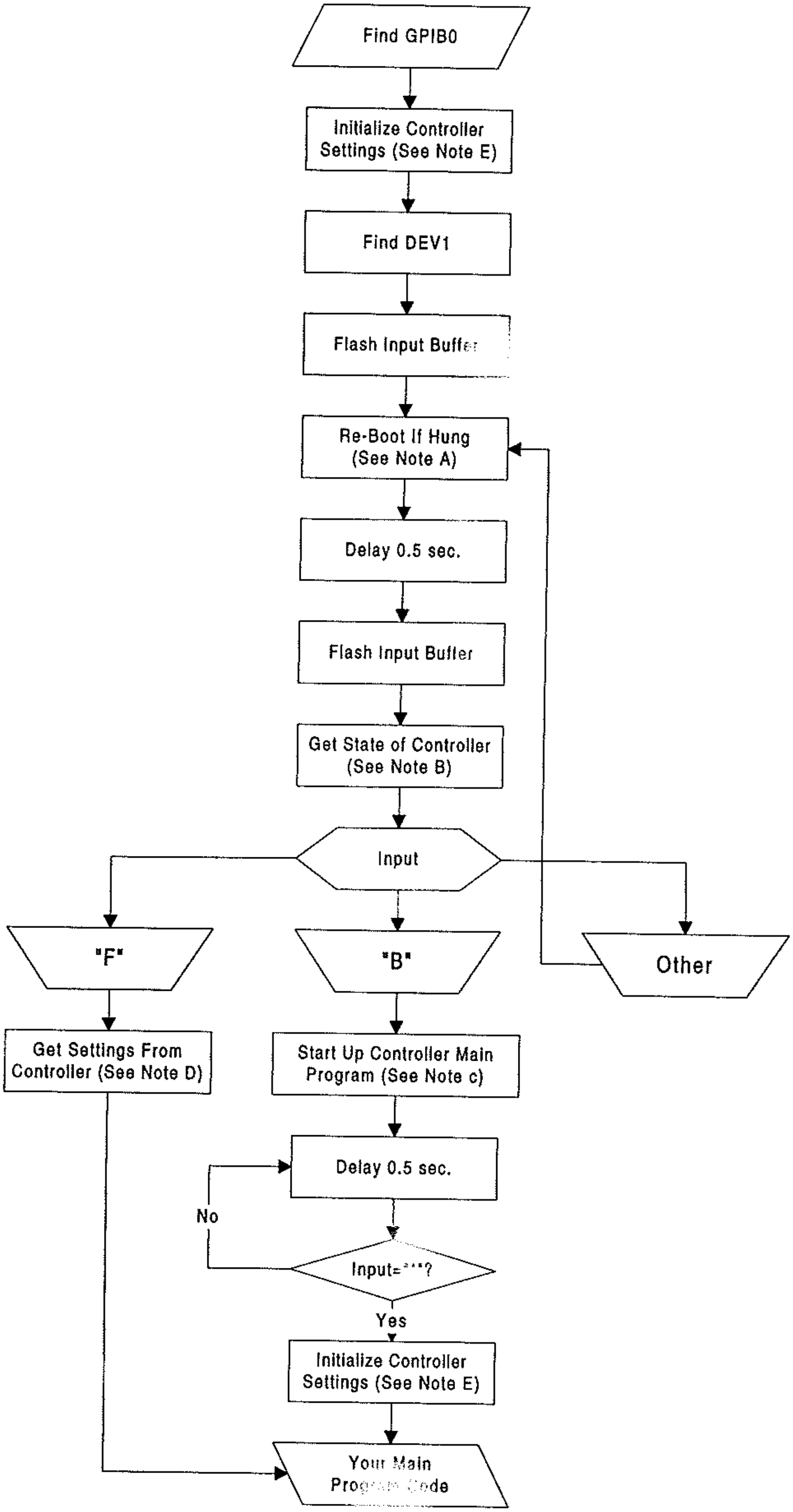
A: Send decimal value "<222>". This will force a re-boot if hung from an incomplete command.

B: Send WHERE AM I command "<Space>"; response will be "B" (for BOOT) or "F" (for MAIN) depending on the previous state of the spectrometer controller.

C: Send "O2000<Null>". To transfer control from the BOOT to the MAIN program. You must send the "<Null>". Wait 0.5 second.

D: You can read your last position etc. from the spectrometer controller. You do not have to re-initialize the spectrometer controller.

E: Initialize mono (see command "A" and SET commands) and get initial settings from the user, e.g. spectral position.


Find GPIB0
Initialize Controller Settings (See Note E)
Find DEV1
Flash Input Buffer
Re-Boot If Hung (See Note A)
Delay 0.5 sec.
Flash Input Buffer
Get State of Controller (See Note B)
Input
"F"
"B"
Other
Get Settings From Controller (See Note D)
Start Up Controller Main Program (See Note c)
Delay 0.5 sec.
No
Input="*"?
Yes
Initialize Controller Settings (See Note E)
Your Main Program Code

6 Using Both RS-232 and IEEE 488:

On a spectrometer controller that is equipped with both the RS-232 and IEEE 488 interfaces, if the IEEE 488 is the first method of communications used, then RS-232 will be disabled since the autobaud process has been bypassed.

However, if RS-232 is the first method of communications used, once autobauding is performed, and communications is established in terminal communications mode (e.g. using HandScan / KeyLink or other "dumb terminal") at initialization time, IEEE 488 may take control by the REN action of the IEEE 488 BUS. If you want to return to RS-232 in terminal communications mode from IEEE 488, you must issue the "UTIL SET TERMINAL COMMUNICATIONS MODE" command (see the command descriptions in Section 10) and wait for confirm before you set the IEEE 488 to LOCAL to allow the RS-232 interface to take control.

7 Configuration for Instruments and Options:

Normally the monochromator type and configuration are loaded in the non-volatile memory of your spectrometer controller. If, for any reason, this configuration does not match your present system, you may use the HWCFG program to modify it. There are two versions of the program. If you are using a GPIB interface, access the subdirectory A:\UTILITY\IEEE488 and execute the program by typing "hwcfg488 1" (where 1 is the current device address). If you are using a serial interface, access the subdirectory A:\UTILITY and execute the program by typing "hwcfg232" A screen will appear showing the communications port and baud rate settings. If these are correct, press, otherwise change the settings as needed. and press <Enter>. (When shipped from the factory, the default settings are for COM port 1, at 19200 baud). The program will establish communications and read back the present configuration from the spectrometer controller. This configuration will then be displayed in the HW CONFIGURATION screen on your computer's monitor.

CAUTION: DO NOT PRESS THE <F3> KEY, AS THIS WILL CHANGE THE CONFIGURATION IN YOUR SPECTROMETER CONTROLLER.

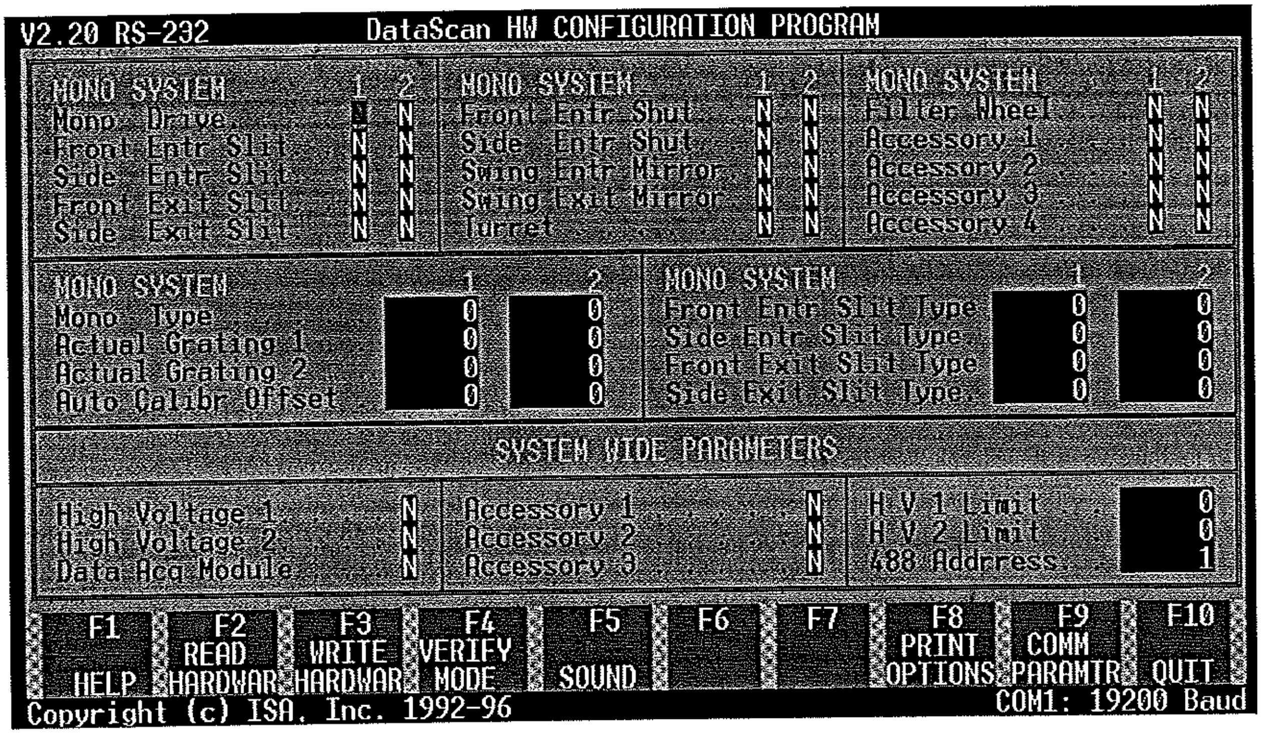
Press the <F2> key to read the hardware configuration from the non-volatile memory in the spectrometer controller. If the interface is properly installed and functional, the program will read and display the settings from your system. If communication fails, an error message will be displayed. If you are using an RS-232 interface, check the possible faults advised by the message. If you are using the GPIB, Refer to National Instruments' documentation to decode the message.

Always print the current configuration using <F8> before making modifications.

Review the configuration on the screen. At the factory, this was set up to match your system. A record of the configuration was printed by using the <F8> function. This was included with the documentation shipped with the spectrometer controller. The most common use of this program is to change settings when you replace gratings or if you need to limit the maximum high voltage if you have a DataScan. You may want to familiarize yourself with some of the parameters and options on the screen, such as grating g/mm, and HV limit.

For example: If you have other monochromators to be operated through this spectrometer controller, put the cursor on MONO TYPE then press <F1>. A help screen will appear showing all valid type numbers. (The Monochromator type for a 270M is "13").

Spectrometer Control



```
V2.20 RS-232            DataScan HW CONFIGURATION PROGRAM

MONO SYSTEM          1  2   MONO SYSTEM          1  2   MONO SYSTEM       1  2
Mono  Drive          N  N   Front Entr Shut      N  N   Filter Wheel      N  N
Front Entr Slit      N  N   Side  Entr Shut      N  N   Accessory 1       N  N
Side  Entr Slit      N  N   Swing Entr Mirror    N  N   Accessory 2       N  N
Front Exit Slit      N  N   Swing Exit Mirror    N  N   Accessory 3       N  N
Side  Exit Slit      N  N   Turret               N  N   Accessory 4       N  N

MONO SYSTEM                 1       2   MONO SYSTEM                1       2
Mono  Type                  0       0   Front Entr Slit Type       0       0
Actual Grating 1            0       0   Side Entr Slit Type        0       0
Actual Grating 2            0       0   Front Exit Slit Type       0       0
Auto Calibr Offset          0       0   Side Exit Slit Type        0       0

                          SYSTEM WIDE PARAMETERS

High Voltage 1         N   Accessory 1            N   H V 1 Limit          0
High Voltage 2         N   Accessory 2            N   H V 2 Limit          0
Data Acq Module        N   Accessory 3            N   488 Address          1

Copyright (c) ISA, Inc. 1992-96                          COM1: 19200 Baud
```

To write a new configuration to be permanently stored in the spectrometer controller, press <F3>. To verify that it was accepted, press <F2> and see the new configuration is returned on the screen. Press <F10> to quit when finished. When the spectrometer controller is powered off, then on again, the new configuration will become active.

8 Communications Conventions:

This is a definition of the communications that should occur when a computer interfaces with the spectrometer controllers.

1. Whenever you see <CR>, it means to use ASCII Carriage Return. (13 in DECIMAL).

2. The parameter "Mono System #" is used to select which of the spectrometer controller's monochromator drive ports you are addressing. This can be either Ø or 1 for a DataScan / DataLink. For a SPEX232 / JY232 or SPEX488 / JY488 it is always Ø, as these interfaces are individually dedicated to spectrometers

3. The parameter "Channel #" selects which data acquisition channel you are addressing. The first channel is Ø, the second is channel 1. (This differs from controller labeling that starts with 1 as the first channel.) Channel # = 2, where appropriate, means select BOTH the first and second channels. An example of this is the command "ACQ START," which can accept 2 for the Channel # to start the data acquisition on both input channels simultaneously.

4. Any string that is placed inside of a pair of double quotes, i.e. "abcd" means that you should send all characters exactly as shown inside the double quotes. You should include spaces, But do not include the double quote characters. Also if you see a character symbolized using <> in a string, you send the character only, not the symbols. For example: if you see"1,Ø,<Null>", you should send only 5 ASCII characters: the 1, the comma, the Ø, another comma, and the Null character (decimal Ø).

5. Anything placed inside of a pair of square brackets, i.e. [Ø..1] indicates the valid range of the parameter associated with it.

9 Command Syntax and Confirmation :

This section outlines the rules of syntax required to successfully send commands and recognize confirmation responses that will inform you that the spectrometer controller has or has not received a valid command string.

9.1 Standard Commands:

As a prerequisite to sending operational commands to your spectrometer controller, you must establish communications first. Refer to Section 4 for RS-232 or Section 5 for IEEE 488. Only after communications have been successfully established can other commands be sent to it.

All commands are ASCII text, unless specifically noted otherwise. The "Standard Commands" are only 1 character and are detailed in the command descriptions in Section 10. For commands requiring additional input, the command is sent, immediately followed by the relevant parameter(s). The command parameters form a string of characters representing the data that is sent or received.

For example, if you wished to send 1000 as a parameter for the command MOTOR SET POSITION, you should send the character string "1000". The general format for sending commands and/or parameters is the following:

1. Send the command (e.g. MOTOR INIT). This is a single character with no termination.

2a. If there is only one parameter, send the parameter, then the PARAMETER BLOCK END character "<CR>" (13 in DECIMAL, 0d in HEX).

2b. If there are multiple parameters, send the PARAMETER DELIMITER character "," between parameters, then send the PARAMETER BLOCK END "<CR>" character after the last parameter.

3. When the spectrometer controller has received all the parameters, it will respond with a single CONFIRMATION character indicating whether or not it has received all the expected parameters. A CONFIRMATION character of "o" indicates that the parameters were OK; "b" indicates that the parameters were BAD. Your program must read the confirmation, if it is ignored, the system will hang.

4a. If the spectrometer controller has been issued a command that would require it to send back information (e.g. MOTOR READ POSITION), then that information will be sent immediately after the CONFIRMATION character "o". If there is only one parameter for the spectrometer controller to send back, then that parameter will be sent followed by the PARAMETER BLOCK END character "<CR>".

4b. If there are multiple parameters to be sent back to the host, the CONFIRMATION character "o" is sent first, followed immediately (without delimiter) by the first parameter. The PARAMETER DELIMITER character "," will be sent between parameters; and the PARAMETER BLOCK END character "<CR>" will be sent after the last parameter. For example, a typical response to MOTOR READ SPEED might be: "o400,800,2000" followed by "<CR>".

9.2 Extended Commands:

The command set is not limited by the number of characters in the English alphabet.

Spectrometer Control

The character "Z" has been assigned for access to extended commands. The first parameter following this command is recognized as the number representing the extended command you wish to execute. Depending on the command number you send, you may be required to send more parameters following the command.

Otherwise, extended commands are handled in the same way as the standard commands above.

For an example, see the description of the command Z11: TTL AUTOMATIC OUTPUTS on page 88.

9.3 Pseudo-Commands:

These commands perform some special utility functions. The syntax is the same as standard single byte commands, except that the bytes sent as commands are not text. They are values such as <222>. In this document, they are expressed as decimal values, between the < and > symbols. The pseudo-command to re-boot the spectrometer controller is expressed as <222>. Another difference is that pseudo commands do not generate a response character from the spectrometer controller.

10 Command Descriptions:

This section details the usage of the Programmer's Command Set.

The commands are sorted into functional groups, each command within a group has a common prefix. The groups are as follows.

- UTIL: the utility commands facilitate startup and communications related functions.
- MOTOR: the motor commands control the spectrometer grating drive.
- SLIT: the slit commands provide control of the automated slits in spectrometers so equipped.
- ACC: the accessory commands control automated grating turrets, entrance / exit port selection mirrors, and blocking shutters.
- ACQ: the acquisition commands control the data collection through spectrometer controllers that include signal channels.
- TTL: the TTL commands read and control the setting of logic lines that are available on a connector on the spectrometer controller. These logic signals can be used to interface to your process.
- THRESHOLD: the threshold commands are used to establish the signal levels at which TTL lines will be asserted to highs or lows.
- SCAN: the scan commands provide semi- autonomous operation of the spectrometer controller. Complete scan definitions can be loaded in advance and then be executed with a simple start scan command.

Individual commands can be located with the Index at the end of this manual. They are listed by their Characters and by their Command names. Certain key words are also listed in the Index to help you locate the command you are looking for.

There are two distinct methods of programming to perform scanning experiments with data acquisition: elementary and independent.

In the elementary command method of running an experiment, your program has absolute control over every step of the experiment. Your program also has absolute responsibility to perform all the required steps according to the protocol outlined in this manual. For instance, a program may issue the following series of commands in order to set up and acquire one data point:

1. HIGH VOLTAGE SET to turn on the high voltage.
2. MOTOR MOVE RELATIVE to set the spectral position
3. MOTOR BUSY (loop) to verify completion of the positioning move.
4. INTEGRATION TIME SET to set the signal integration time
5. GAIN SET to set the gain level for the desired channels.
6. ACQ START to start the acquisition.

7 ACQ BUSY (loop) to verify completion of the signal integration.

8. READ DATA to read back the acquired data and over range status.

Please note that the above routine is far from complete. It is Intended to illustrate the level of elementary steps that would be involved in running an experiment.

Some commands provide independent operation. All the parameters needed to perform a routine are sent at one time. These parameters include the positions, integration time, and gains, comparable to the above example. In this independent command method, a setup command loads a string of operating parameters:

1 SCAN SET PARAMETERS sets all the operating parameters

2 SCAN START begins the routine.

3. SCAN GET DATA reads back the complete scan of data.

With this independent method, the host program is free to perform other tasks while the scan is in process. If desired, data can be read "on the fly" as the scan is progressing, or after the scan is completed. See the other commands having the SCAN- prefix in Section 10.8 for additional capabilities of the independent commands.

10.1 Utility Commands:

The commands in this section provide the means to properly start up communications with the spectrometer controller, as well as other, related commands.

This section includes commands used to switch communications modes between "intelligent" communications mode for computer control and "terminal" communications mode for hand-held terminal control.

Also described here are the commands used to determine the spectrometer controller's internal program version numbers, which is useful to know when contacting us for service. There is also a command to determine out which part of the spectrometer controller internal program you are talking to.

There is a command to allow you to change the spectrometer controller's IEEE address, as well.

Spectrometer Control

UTIL WHERE AM I "<Space>"

When communicating over an RS-232 link, this character is first used for automatic baud rate selection. Refer to the discussion on Autobauding on pages 9, 14 for a complete explanation of this process. If Autobauding has not been established, you will receive no response. (No response can also indicate that the program is hung). It may be necessary to send the "<Space>" character more than once, before the spectrometer controller responds with "<*>". After Autobauding, while the spectrometer controller is still in the terminal communications mode, the response to the "<Space>" character will be "<Escape>" followed by a string of characters intended for the terminal's display. If you are in the intelligent communications mode, the response to the "<Space>" character will identify which part of the spectrometer controller's internal program you are talking to; the BOOT program will respond "<B>", the MAIN program will respond "<F>"

When communicating over the IEEE 488 bus, the intelligent communications mode is set automatically, so the response to the "<Space>" character will identify which part of the spectrometer controller's internal program you are talking to; the BOOT program will respond "<B>", the MAIN program will respond "<F>". No response indicates that the program is hung.

If you receive an "F" in response to sending "<Space>", you also know that the spectrometer controller has been previously accessed.

For more detail on the startup sequence, refer to the flowchart for your communications interface: RS-232 on page 12 or IEEE 488 on page 26.

The spectrometer controller will respond to this command differently than it does to the standard commands. The normal "o" confirmation character will not be sent back. There will be no "<CR>"after the response, either.

Normally, if you receive the "B" response, indicating that you are talking to the BOOT program, you will want to send the string "O2000<Null> to start the spectrometer controller's MAIN program.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Program status
Example: Send "<Space>"	Receive Nothing if not yet successful in autobauding, or the system is hung or; Receive "*" if autobauding is successful for the first time or; Receive "<Escape>" followed by a string of characters if communicating in the Terminal mode or; Receive "B" if communicating in Intelligent mode with the BOOT program or; Receive "F" if communicating in Intelligent mode with the MAIN program

Spectrometer Control

UTIL STARTUP INTELLIGENT MODE "<247>"

This command is used during the startup sequence for RS-232 communications only. It puts the spectrometer controller in intelligent communications mode.

If you start communications to the spectrometer controller via IEEE 488, the intelligent communications mode is set automatically. In that case there is no need for this command.

The only valid time to send this command is during the RS-232 startup sequence, immediately following receipt of the "*" that confirms that the autobaud selection was successful. To be sure you use this command at the correct point in the startup sequence, please refer to the startup flowchart for RS-232 communications on page 12.

Also see the UTIL SET INTELLIGENT MODE command which performs a similar function at times other than during the startup sequence.

The spectrometer controller will respond to this command differently than it does to the standard commands. No confirmation character will be sent back. only the "=" will be sent back to advise you that the spectrometer controller has switched to the intelligent communications mode. There will be no "<CR>" termination on this response.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Acknowledgement
Example: Send "<247>"	Receive "="

Spectrometer Control

UTIL SET INTELLIGENT MODE "<248>"

This command is used to put the spectrometer controller into Intelligent Communications mode.

This command's intended use is to take control away from the hand-held controller, and give control to your program. To use the spectrometer controller in this way, the first communication after power on must be from the hand-held terminal. In this way, the baud rate selection is established with the hand-held terminal while the BOOT sequence is in the proper state to autobaud.

Also see the UTIL STARTUP INTELLIGENT MODE command which performs a similar function during the RS-232 startup sequence.

The spectrometer controller will not send any response characters upon receipt of this pseudo command. No confirmation character will be sent back. There will be no "<CR>" termination either.

You should wait 200 milliseconds after sending this command before issuing other commands.

After sending the "<248>" command, you may verify its effect by sending the UTIL WHERE AM I "<Space>" command.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. No response at all.
Example: Send "<248>"	Receive nothing

Spectrometer Control

UTIL SET TERMINAL MODE "<Y>"

This command is used to put the spectrometer controller into Terminal Communications mode.

This command's intended use is to switch control to the hand-held controller, and away from your program. To enable you to use the spectrometer controller in this way, the first communication after power on must have been from the hand-held controller. That way, the autobaud rate selection was established while the BOOT sequence was in the proper state to autobaud.

Also see the UTIL SET INTELLIGENT MODE command, which performs the opposite function.

The spectrometer controller will respond to this command in the normal way, sending back the "o" character to confirm receipt of the command.

Note: After you receive the confirmation, press the <.> key on the hand-held terminal twice to re-establish communications.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends: 1. Confirmation
Example: Send "<Y>"	Receive "o"

Spectrometer Control

UTIL START MAIN PROGRAM "O2000<Null>"

This "command" is actually a string of characters that is sent to the spectrometer controller as part of the start up sequence. The command is valid only when the spectrometer controller's BOOT program is running and the spectrometer controller is in the intelligent communications mode. It is used to jump from the BOOT to the MAIN program in the spectrometer controller.

If you are communicating over an RS-232 link, you may need to send the UTIL SET INTELLIGENT MODE "<248>" command (see page 41) to switch the spectrometer controller from the terminal communications mode to the intelligent communications mode. If you are using IEEE 488, this is not necessary.

The spectrometer controller will respond to this command differently than it does to the standard commands. The "<*>" character will be sent back to confirm that the MAIN program is running.

Wait 500 milliseconds after issuing the "O2000<Null>" command to be sure that the spectrometer controller has had sufficient time to establish itself in the MAIN program and send back the "<*>".

See the UTIL WHERE AM I "<Space>" command on page 38 which offers an easy way to verify that you are in the MAIN program.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "O2000<Null>"	Receive: "<*>"

Spectrometer Control

UTIL RE-BOOT IF HUNG "<222>"

This pseudo command is used to restart the program in the spectrometer controller if it becomes hung. Note that depending on the condition that causes the spectrometer controller to hang, this command may not necessarily always un-hang and re-boot it.

The "<222>" command is helpful if you want to force the spectrometer controller to a known state. In this case, deliberately hang it by sending an incomplete command, such as MOTOR SET POSITION without any position parameter. Then you may issue the "<222>" command to re-bootstrap the spectrometer controller.

The "<222>" will be ignored if the Spectrometer controller is not hung.

The spectrometer controller will respond to this command differently than it does to the standard commands. No confirmation character will be sent back.

Wait 200 milliseconds after issuing this command to be sure that the spectrometer controller has had sufficient time to bootstrap before issuing other commands.

To proceed from this point, see the UTIL WHERE AM I "<Space>" command on page 38 which offers an easy way to determine if the re-boot was successful.

If you are communicating over an RS-232 link, see the UTIL SET INTELLIGENT MODE "<248>" command on page 41 to switch the spectrometer controller from terminal communications to intelligent communications mode. If you are using IEEE 488, this is not necessary.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Nothing
Example: Send "<222>"	Receive Nothing

Spectrometer Control

UTIL READ MAIN VERSION "<z>"

This command is used to read the firmware revision number of the spectrometer controller's MAIN program.

This command is also helpful when contacting us for service assistance. If you can advise us of the firmware version you are using, we can respond more effectively to your needs.

The spectrometer controller will respond to this command in the normal way, by sending back the "o" character to confirm receipt of the command, and following that with a string such as "V3.3<CR>" to advise you of the version of the MAIN program firmware in the spectrometer controller. The length of the string could, conceivably, vary. The "<CR>" denotes the end of the string.

A similar command, UTIL READ BOOT VERSION, provides you the means to read the other firmware revision number.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends: 1. Confirmation 2. MAIN version number
Example: Send "<z>"	Receive "o" Receive "V3.3<CR>" or similar response

Spectrometer Control

UTIL READ BOOT VERSION "<y>"

This command is used to read the firmware revision number of the spectrometer controller's BOOT program.

This command is also helpful when contacting us for service assistance. If you can advise us of the firmware version you are using, we can respond more effectively to your needs.

The spectrometer controller will respond to this command in the normal way, by sending back the "o" character to confirm receipt of the command, and following that with a string such as "V2.3<CR>" to advise you of the version of the BOOT program firmware in the spectrometer controller. The length of the string could, conceivably, vary. The "<CR>" denotes the end of the string.

A similar command, UTIL READ MAIN VERSION, provides you the means to read the other firmware revision number.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends: 1. Confirmation 2. BOOT version number
Example: Send "<y>"	Receive "o" Receive "V2.3<CR>" or similar response

Spectrometer Control

UTIL CHANGE IEEE 488 ADDRESS "<E>"

This command provides a means to change the IEEE 488 address of the spectrometer controller on the GPIB bus.

This command is only valid at one point in the startup sequence. While in the BOOT portion of the spectrometer controller's internal program, instead of sending the "O2000<Null>" string to transfer to the MAIN program, send the "<E>" command.

Please do not confuse this "<E>", which is a BOOT command, with the MAIN command "<E>" that will return the motor busy status.

To be sure you use this command at the correct point in the startup sequence, please refer to the startup flowchart for RS-232 communications on page 12 or the IEEE-488 startup flowchart on page 26.

Note that the value of the new address sent to the spectrometer controller should be sent in ASCII characters to be read directly as a hexadecimal address. For example, to set the IEEE address to (decimal) 11, you send "EB<Null>"

The spectrometer controller will respond to this command differently than it does to the standard commands. The standard "<o>" confirmation character will not be sent back. Instead, the character with the binary value of "<2>" will be sent back to acknowledge that the spectrometer controller has received a new address. The "<2>" response is the same, regardless of the address sent. There will be no "<CR>" termination on this response.

At this point, the new address has only been stored in the spectrometer controller's non-volatile memory.

Unless you use this command again, the new address will remain in effect each time you turn on the spectrometer controller.

HOST sends 1. Command 2. New address in ASCII HEX	SPECTROMETER CONTROLLER sends 1. Acknowledgement
Example: Send "E3<Null>"	Receive "<2>"

To enable GPIB communication using the new address, you must turn the spectrometer controller off, then on again.

10.2 Grating Motor Commands:

These commands control the movement of the grating scan drive to change the spectral position of the instrument. In other words, these groups of commands deal with control of the portion of the spectrum allowed to pass through the spectrometer to the sample or detector.

The movements are expressed in steps. Because of this, various spectrometers, regardless of scanning mechanism, can be controlled. This also provides you the opportunity to work in whatever spectral units you prefer. Simply write your program to convert your units to steps. In this way, your program to step in electron volts, for example can control a monochromator that has a mechanism for scanning directly in Angstrom units.

The base spectral unit and steps per unit factors for all currently supported spectrometers are listed in Appendix 1: Monochromator Setup Parameters.

If you are writing a routine for an autocalibrating monochromator (180D, 220M, 270M, or HR460), please note: If your program that moves the grating or slit drives will also require movement of a accessory, it is best to send the slit, turret, and shutter commands first. These are positioned by DC motors that can complete their movements while you send other commands. Follow these with the slit and monochromator grating drive commands. These are positioned by stepper motors, and must be handled one at a time.

This group of elementary commands provides direct control over each motor movement. If you prefer to download a scan routine to have the controller automatically scan and acquire a series of datapoints, see the Independent Scan Commands starting on page 92.

Spectrometer Control

MOTOR INIT "A"

This command is used to initialize the monochromator, and normally only needs to be called once when all the hardware is powered on and the software is started.

For autocalibrating monochromators like the 180M, 220M, 220M+, 270M, and the HR460, this command does quite a bit. The monochromator wavelength drive motor is initialized, and upon completion, ends up at a particular (phase sensed) step position associated with the upper limit switch. This is the only known position when the monochromator is first powered on and initialized.

The wavelength value that corresponds to this initial position can be found on a label on the back of the instrument. The value needed is the "switch position" or
"DMD.INI" number. Do not use the "offset" number.

Typically the base grating is 1200 g/mm, see Appendix 1 for the actual base grating groove density for your monochromator. At the factory, the corresponding step position value for the base grating of your monochromator is stored in the non-volatile memory of the controller. At initialization, this value is used as the starting position. After initialization, any movements will cause the position value to be updated. The host can then determine the step position by using the MOTOR READ POSITION command.

Note that the "A" command is used to move the drive accurately to the initialization position. This is not the same as movement by any other command that results in hitting the switch. In all other cases the movement is halted in an uncontrolled way. Thereafter, an accurate step position can no longer be assumed.

The autocalibrating monochromators can also have motorized slits, a motorized turret, motorized mirrors, and automatic shutters. These accessories are also initialized to their known positions. Motorized slits are left nominally closed (step position Ø), the turret is initialized to POSITION Ø (the same side that the turret goes to when the mono is powered on), the mirrors are initialized to the FRONT (or AXIAL) position, and the shutters are left closed.

The initialization of a fully optioned autocalibrating monochromator can take 1-2 minutes. The confirmation character is not sent until the entire initialization process is completed. It is important that the host program and interface driver do not time out before this process is completed.

If you are communicating with the spectrometer controller via IEEE-488, set the timeout to 1ØØ seconds for this command. After initialization is complete, reduce the timeout to 3ØØ ms for normal commands.

For all other (non-autocalibrating) monochromators, this command will simply attempt to move the monochromator drive off the limit switch if it detects that it is up against one.

HOST sends: 1. Command	SPECTROMETER CONTROLLER sends: 1. Confirmation
Example: Send "A"	Receive "o"

Spectrometer Control

MOTOR SET SPEED "B"

This command is used to set the speeds (in hertz) that should be used to move the monochromator. The Min Frequency is the starting speed of the mono; the Max Frequency is the fastest speed that the motor can move reproducibly; and the Ramp Time is the time it should take for the motor to ramp up from the Min Frequency to the Max Frequency. The Monochromator Setup Parameters table in Appendix 1 specifies the optimum speeds for each type of monochromator that is supported.

The values for Min Frequency and Max Frequency should be within the range of 100 - 80,000. Ramp Time should be within the range of 100 - 65,535.

HOST sends 1. Command 2. Mono System # [0..1] 3. Min Frequency Steps/Sec. 4. Max Frequency Steps/Sec. 5. Ramp Time Milliseconds (ms)	SPECTROMETER CONTROLLER sends 1.Confirmation
Example: Send "B0,400,800,2000<CR>"	Receive "o"

MOTOR READ SPEED "C"

This command is used to read back the speeds that were set using the previous command (MOTOR SET SPEED). If the MOTOR SET SPEED command was never issued, the default speed parameters for the monochromator will be returned. Using either the HWCFG232 or the HWCFG488 program provided on the support diskette specifies these parameters.

HOST sends 1. Command 2. Mono System # [0..1]	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Min Frequency Steps/Sec 3. Max Frequency Steps/Sec 4. Ramp Time Milliseconds (ms)
Example: Send "C0<CR>"	Receive "o" Receive "400,800,2000<CR>"

Spectrometer Control

MOTOR BUSY CHECK "E"

This command is also used with SLIT MOTORS.

This command is used to poll the Spectrometer controller after the command MOTOR MOVE RELATIVE or SLIT MOVE RELATIVE is issued in order to check to see if the motor has completed its move. The "Busy Status" that is returned is not terminated by a <CR> character. The values that may be returned are "q" and "z" (FUNCTION BUSY and FUNCTION NOT BUSY, respectively).

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Busy Status ["q" or "z"]
Example: Send "E"	Receive "o" Receive "q"

Spectrometer Control

MOTOR MOVE RELATIVE "F"

This command is used to move a monochromator a relative number of steps from its current position. When this command is issued, the monochromator will start moving at the Min Frequency set with the MOTOR SET SPEED command, and ramp up to the Max Frequency.

The host computer regains control as soon as the move is initiated, not after the entire move has completed. Therefore, it is necessary to poll the spectrometer controller, after instructing it to move the spectrometer, to see if the move has completed; this is accomplished using the MOTOR BUSY command.

Only 1 stepper motor can be moved at a time. This means that if a slit or the wavelength drive of a monochromator is currently being moved (even if it is connected to the other monochromator drive port), you cannot send the MOTOR MOVE RELATIVE command. You must test the MOTOR BUSY status or otherwise be sure that no other motors are moving.

If a stepper motor is busy, you may not move an accessory (DC) motor, such as a mirror, shutter, or turret. You may, however, start an accessory (DC) motor moving and then send other commands before it has reached its destination.

Using a positive number of steps as the parameter for "Steps to Move" will change the spectral position to a greater step position (move forward); a negative number will change the spectral position to a lesser step position (move backwards). Please note that the host computer is responsible for performing backlash correction on negative moves; it is not performed automatically with this command. For each of the supported monochromators, the recommended number of steps for wavelength drive backlash correction is listed in the Monochromator Setup Parameters Table in Appendix 1.

The only limit to the value that the "Steps to Move" parameter can have is the total number of possible steps for the installed monochromator.

HOST sends 1. Command 2. Mono System # [Ø..1] 3. Steps to Move	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "FØ,1ØØØ<CR>"	Receive "o"

Spectrometer Control

MOTOR SET POSITION "G"

This command is only used to correct the stored value of the actual current step position of the monochromator. For non-autocalibrating monochromators, a first pass calibration should be done when the instrument is powered on. Take the position reading from the monochromator's mechanical counter and convert it to steps. Use the appropriate steps/unit factor from the Monochromator Setup Parameters table in Appendix 1. Enter the result using this command.

Afterwards, if higher precision is needed, search to a known calibration position in your region of spectral interest, and update the value. Such a fine calibration is typically made using a known spectral emission peak or a transition edge from absorption to transmission of a filter. Keep backlash error to a minimum by making your final pass to the calibration position from the same direction you will use for your measurements. The required backlash overshoot is given in Appendix 1: Monochromator Setup Parameters.

After the positioning precisely on a known spectral feature, and having taken backlash into account, use this command to enter the step position that corresponds to the spectral line you use.

With autocalibrating spectrometers (220M, 220M+, 270M, HR460); the instrument self-calibrates to a precise phase sensitive step position associated with the upper limit switch after the MOTOR INIT command has been executed. This factory calibrated switch position is included with your calibration documentation shipped with the instrument from the factory. It is also marked on a label on the rear of the spectrometer. Don't confuse this position with the offset value, also labeled on the spectrometer. This autocalibration is significantly more accurate than the first pass manual calibration described above. If your work requires the ultimate in calibration accuracy at your particular region of spectral interest, you may prefer to do the fine calibration mentioned above.

Keep in mind also that the monochromators give most reproducible results when scanning in the direction they were designed for:

- For wavelength drives; scan in the direction of increasing wavelength
- For wavenumber drives; scan in the direction of decreasing wavenumbers

HOST sends 1. Command 2. Mono System # [0..1] 3. Mono Step Position	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "G0,1000000<CR>"	Receive "o"

Spectrometer Control

MOTOR READ POSITION "H"

This command is used to read back the current monochromator step position. Whenever the monochromator is moved, the step position continuously updates itself to reflect the current position. This command can be issued after the MOTOR MOVE RELATIVE command, while polling the Busy Status. The sequence of commands might be as follows:

1. Issue the MOTOR MOVE RELATIVE command to initiate a spectral position move.

2. Issue the MOTOR BUSY command to test the Busy Status.

3. If the motor is still busy, issue the MOTOR READ POSITION command to retrieve its current position.

4. Repeat 2 and 3 until the MOTOR BUSY command returns a status of not busy.

5. Issue the MOTOR READ POSITION command to verify the new step position.

HOST sends 1. Command 2. Mono System # [Ø..1]	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Motor Step Position
Example: Send "HØ<CR>"	Receive "o" Receive "1ØØØØØØ<CR>"

Spectrometer Control

MOTOR LIMIT STATUS "K"

This command is used to check the limit status of all installed monochromators. The grating scan motor of each spectrometer has an allowable operating range, if this range is exceeded in either direction, the drive will mechanically trip a limit switch. The value returned in response to the limit status command is the ASCII representation of the Binary limit status of each monochromator. If there are no monochromators connected (but there are connectors for them), then a limit condition may be set for each missing monochromator. Therefore, when checking limit conditions, you should only check those valid for the monochromator you are interested in. Do not automatically check for limits on both monochromators if only one is really connected.

This command does not return limit status for autocalibrating monochromators.

The ASCII string returned must be converted into a 1 Byte (8 bit) Binary number. To do this in C, you could first read the ASCII string (into variable ascii_stat), then use the following code segment:

```
unsigned char    binary_stat;
binary_stat = (unsigned char)atoi( ascii_stat );
```

Once the string is converted properly, each bit can be decoded in the following manner to determine if there is a limit condition:

In this diagram, bit #0 is the Least Significant Bit, and bit #7 is the Most Significant Bit.

MSB							LSB
7	6	5	4	3	2	1	0

Example 1
If bit #0 has the value of 1, while all the rest of the bits have the value of 0, the binary value of the entire byte is 1.

Example 2
If bit #7 has the value of 1 while all the rest of the bits have the value of 0, the binary value of the entire byte is 128.

If you are programming in BASIC, to decode the limit bits, make successive tests comparing and subtracting binary place values from the number.

- First determine if the returned byte is greater or equal to 128. If so, bit 7 is set; subtract 128. If not, bit 7 is not set, do not subtract 128.
- Next, repeat the procedure using 64 in place of 128. This will test for the state of bit 6.
- Next, repeat the procedure using 32 in place of 128. This will test for the state of bit 5.

Continue decreasing the comparison value by powers of 2 until all bits have been tested.

Limit Condition Assignments to Bits in Motor Limit Status Byte:

Spectrometer Control

For MONO TYPES #1 through #13:

BIT#	Limit Condition
Ø	Lower limit on first monochromator.
1	Upper limit on first monochromator.
2	Lower limit on second monochromator.
3	Upper limit on second monochromator.
4	UNUSED.
5	UNUSED.
6	UNUSED.
7	UNUSED.

For MONO TYPES #14 through #19:

BIT#	Limit Condition
Ø	UNUSED.
1	UNUSED.
2	UNUSED.
3	UNUSED.
4	Either limit on first monochromator.
5	Either limit on second monochromator.
6	UNUSED.
7	UNUSED.

When bit #4 or bit #5 has a value of 1, the determination of whether the LOWER or UPPER limit has been reached must be made by checking the direction of the last monochromator drive movement.

For ALL MONO TYPES:
A binary value of zero (Ø) means that no limits are hit.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Motor Limit Status
Example: Send "K"	Receive "o" Receive "Ø<CR>"

MOTOR STOP "L"

This command is also used with SLIT MOTORS.

This command is used to stop the monochromator or slits if they are moving. If the monochromator has ramped up to a frequency higher than its Min Frequency, this command will initiate a ramp down. Since the motor may not stop immediately, the MOTOR BUSY command should subsequently be called until it returns a Busy Status of not busy.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "L"	Receive "o"

10.3 Slit Commands

This series of commands controls automated slits. These commands only apply if you have motorized slits installed on your monochromator system.

These values are to be used to designate the slit to be controlled:

Slit #0	Front (Axial) Entrance Slit
Slit #1	Side (Lateral) Entrance Slit
Slit #2	Front (Axial) Exit Slit
Slit #3	Side (Lateral) Exit Slit

SLIT SET SPEED "g"

This command is used to set the speed that should be used to move slits. The Frequency parameter is used to specify the speed in hertz. Note that although slits are motors, they do not ramp as the monochromator motor does.

The Frequency parameter should be within the range of 10 - 10,000.

Refer to Appendix 1 for recommended settings to for your spectrometer's slits.

HOST sends 1. Command 2. Mono System # [0..1] 3. Slit # [0..3] 4. Frequency Steps/Sec.	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "g0,0,400<CR>"	Receive "o"

SLIT READ SPEED "h"

This command is used to read back the speed that was set using the previous command (SLIT SET SPEED). If the SLIT SET SPEED command was never issued, the default speed parameter for the specified slit will be returned.

HOST sends 1. Command 2. Mono System # [0..1] 3. Slit # [0..3]	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Frequency Steps/Sec.
Example: Send "h0,0<CR>"	Receive "o" Receive "400<CR>"

Spectrometer Control

SLIT SET POSITION "i"

This command is used to set (calibrate) the current step position of the specified slit. Motorized slits are normally initialized during the execution of the MOTOR INIT command, and they are initialized to the Ø step position.

The SLIT SET POSITION command should only be called with Ø as the Slit Step Position parameter, and it should only be called after the MOTOR INIT command has been executed.

Attempting to calibrate a slit by manually measuring the width of the opening will not yield accurate results.

HOST sends 1. Command 2. Mono System # [Ø..1] 3. Slit # [Ø..3] 4. Slit Step Position	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "iØ,Ø,Ø<CR>"	Receive "o"

SLIT READ POSITION "j"

This command is used to read back the current position of the specified slit. Whenever the slit is moved, the step position continuously updates itself to reflect the current position. This command can be issued after the SLIT MOVE RELATIVE command, while polling the Busy Status with the MOTOR BUSY command to get a continuously updated reading of the slit's current position.

HOST sends 1. Command 2. Mono System # [Ø..1] 3. Slit # [Ø..3]	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Slit Step Position
Example: Send "jØ,Ø<CR>"	Receive "o" Receive "1ØØ<CR>"

Spectrometer Control

SLIT MOVE RELATIVE "k"

This command is nearly identical to the MOTOR MOVE RELATIVE command.

This command is used to move the specified slit a relative number of steps from its current position. When this command is issued, the slit will start moving immediately.

The host computer regains control as soon as the move is initiated, not after the entire move has completed. Therefore, it is necessary to poll the spectrometer controller, after instructing it to move the slit, to see if the move has completed; this is accomplished using the MOTOR BUSY command (described earlier in this document).

Using a positive number of steps as the parameter for "Steps to Move" will move the slit motor to a greater step position (wider opening); a negative number will move the slit motor to a lesser step position (narrower opening). Please note that backlash correction is not performed automatically with this command. The host computer program is responsible for performing backlash on negative moves. The recommended number of steps for backlash for slits is given in the Monochromator setup parameters table in Appendix 1.

Only 1 slit can be moved at a time. You cannot move a slit if the monochromator motor or any other slit is moving (even if it is connected to a different monochromator drive port if the spectrometer controller is so equipped). You must test the MOTOR BUSY status or be sure that no other motors are moving before you send a SLIT MOVE RELATIVE command.

The only limit to the value that the "Steps to Move" parameter can have is the total number of possible steps for the installed slit.

HOST sends 1. Command 2. Mono System # [Ø..1] 3. Slit # [Ø..3] 4. Steps to Move	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "kØ,Ø,5ØØ<CR>"	Receive "o"

10.4 Monochromator Accessory Commands:

These commands actuate the automated mechanical accessories that are interfaced to the spectrometer controller.

Each controllable "accessory" is operated by a DC motor or solenoid. As such, each has only two positions. The shutters reach their new positions in a matter of milliseconds, but turrets and side or lateral mirror assemblies may need several seconds to come to their new positions.

When writing your program, it is important to allow for these delays. Once an accessory command is accepted by the spectrometer controller, you are free to send other commands immediately. However, some commands depend on the positioning of accessories. Before using those commands, you should first test using the ACC BUSY CHECK command to be sure that all accessories have reached their destinations.

If you are writing a routine for an autocalibrating monochromator (180D, 220M, 220M+, 270M, or HR460), please note: If your program that moves an accessory will also require movement of the grating or slit drives, it is best to send the slit, turret, and shutter commands first. These are positioned by DC motors that can complete their movements while you send other commands. Follow these with the slit and monochromator grating drive commands. These are positioned by stepper motors, and must be handled one at a time.

ACC SHUTTER OPEN "W"

This command is used to open the "Active" shutter. It is possible to have 2 entrance shutters installed on a monochromator (Front/Axial and Side/Lateral), but only one can be "Active" at any given time. The "Active" shutter is the one that is in the light path.

If the Entrance Mirror is in the Front (Axial) position, or if there is no Entrance Mirror, then the Front (Axial) Entrance Shutter is "Active".

If the Entrance Mirror is in the Side (Lateral) position, then the Side (Lateral) Entrance Shutter is "Active".

If this command is used with the ACC BUSY CHECK command, it has a Delay of 1ØØ ms.

Note that the shutter commands described here do not control the exposure shutters associated with multichannel detectors that are operated by the electronics of the multichannel detection system.

HOST sends 1. Command 2. Mono System # [Ø..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "WØ<CR>"	Receive "o"

Spectrometer Control

ACC SHUTTER CLOSE "X"

This command is used to close the "Active" shutter. Refer to the SHUTTER OPEN command above for more information.

If this command is used with the ACC BUSY CHECK command, it has a Delay of 100 ms.

HOST sends 1. Command 2. Mono System # [0..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "X0<CR>"	Receive "o"

ACC TURRET POSITION 1 "a"

This command is used to rotate the turret to the "Other" grating. In this context, "Other" refers to the grating that is not in use after the system is initialized using the MONO INIT command.

If this command is used with the ACC BUSY CHECK command, it has a Delay of 10000 ms.

HOST sends 1. Command 2. Mono System # [0..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "a0<CR>"	Receive "o"

ACC TURRET POSITION 0 "b"

This command is used to rotate the turret to the "Default" grating. In this context, "Default" refers to the grating that is in use after the system is initialized using the MONO INIT command.

If this command is used with the ACC BUSY CHECK command, it has a Delay of 10000 ms.

HOST sends 1. Command 2. Mono System # [0..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "b0<CR>"	Receive "o"

Spectrometer Control

ACC ENTR MIRROR SIDE "c"

This command is used to move the entrance mirror to the Side (Lateral) position.

If this command is used with the ACC BUSY CHECK command, it has a Delay of 15ØØØ ms.

HOST sends 1. Command 2. Mono System # [Ø..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "cØ<CR>"	Receive "o"

ACC ENTR MIRROR FRONT "d"

This command is used to move the entrance mirror to the Front (Axial) position.

If this command is used with the ACC BUSY CHECK command, it has a Delay of 15ØØØ ms.

HOST sends 1. Command 2. Mono System # [Ø..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "dØ<CR>"	Receive "o"

ACC EXIT MIRROR SIDE "e"

This command is used to move the exit mirror to the Side (Lateral) position.

If this command is used with the ACC BUSY CHECK command, it has a Delay of 15ØØØ m

HOST sends 1. Command 2. Mono System # [Ø..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "eØ<CR>"	Receive "o"

Spectrometer Control

ACC EXIT MIRROR FRONT "f"

This command is used to move the exit mirror to the Front (Axial) position.

If this command is used with the ACC BUSY CHECK command, it has a Delay of 15ØØ ms.

HOST sends 1. Command 2. Mono System # [Ø..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "fØ<CR>"	Receive "o"

ACC BUSY CHECK "l" ℓ

This command is used to test if one of the monochromator accessories described above is still moving. Be careful not to confuse this l (lowercase L) with a number 1 or an uppercase i). This test simply counts down the delay, and returns not busy when the timer runs down. This command has the same format as the MOTOR BUSY command, and it returns the same Busy Status characters ("q" for FUNCTION BUSY and "z" for FUNCTION NOT BUSY). A <CR> character does not terminate the Busy Status.

It is very useful to test for accessory busy before beginning an acquisition. This will ensure that all of the accessories are in their proper positions before integration begins.

It is not necessary, however, to make this test each time an accessory is moved. Unlike the monochromator and slit motors, it is perfectly fine to initiate multiple accessory moves in succession, without waiting for each to complete. For example, if you wanted to rotate the turret and move the entrance mirror, you could send the appropriate commands without waiting in between; then, before you start your data acquisition, you should make the ACC BUSY CHECK. When multiple accessories are moved in succession, only the longest Delay will be counted down; Delays are not added together.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Busy Status ["q" or "z"]
Example: Send "l"	Receive "o" Receive "q"

q — busy

z — not busy

10.5 Data Acquisition Commands:

The data acquisition commands relate only to the DataScan / DataLink controller- photometers, and to SpectrAcq spectrometer controllers that include SAQ-CTI signal input channels.

These commands facilitate the setup and integration of signals from photomultiplier or solid state single-channel detectors through the signal input channels of the above controllers.

Generally, these commands apply to all three controller types. When a command is specific to one or two controller types, this is noted in the description.

This group of elementary commands provides direct control over the acquisition of each datapoint. If you prefer to download a scan routine to have the controller automatically acquire a series of datapoints, see the Independent Scan Commands starting on page 92.

Data Acquisition Guidelines (for DataLink / DataScan Only):

Before performing any type of data acquisition using the AUTOGAIN option, the amplifier offsets must first be computed or set to obtain accurate measurements. In any of the FIXED GAINS, offsets are not subtracted. This applies to BOTH operating methods described below, with version 3.0 or higher of Flash.

This is accomplished by issuing the command MEASURE ACQUISITION OFFSETS or SET ACQUISITION OFFSETS. These offsets are automatically computed into the final datapoint result that is read by the host while acquiring with AUTOGAIN.

If you are running in any of the fixed gains, the offset is NOT automatically factored into the resulting data. If desired, offset subtraction must be performed after your controlling program reads the data back.

The command set is designed to allow maximum flexibility for the controlling program. Therefore, whenever possible, the data that is returned is in its "rawest" form; this is why, in fixed gains, there is no offset processing performed. However, with AUTOGAINing, any or all of the different gains can be used during a given integration period (depending on the fluctuation of the signal being measured); and since each gain can have a different offset value associated with it, it is most convenient to handle that processing in the firmware.

A few operating parameters must be set before the data acquisition can take place:

1. The integration time must be set.
2. The gain must be set for the channel being used.
3. If using a photomultiplier, the high voltage must be turned on.

Spectrometer Control

ACQ MEASURE OFFSETS "w"

This command relates only to the DataScan / DataLink controller- photometers. It is used to compute the offsets associated with each gain setting. The offsets are to be factored into the final data as it is acquired in AUTOGAIN. Since there are 4 gains in this data acquisition system, 4 separate offsets will be computed.

Before issuing this command, you should ensure that all hardware are in their desired states. This depends largely on the type of detector you are using, the type of experiment you are performing, and your personal preference. For example, in some cases it may be necessary to turn off the high voltage and open the shutters before measuring the offsets. In other cases, you may want to close the shutters and leave the high voltage on (to do a background subtraction, for example). But whatever method you choose, you must have all devices in their proper states before sending this command.

The spectrometer controller will take an average of 100 reads per channel to allow an accurate computation of the offsets. As a result, the response to this command will be delayed longer than most.

If you are communicating with the spectrometer controller via IEEE 488, set the timeout to 2 seconds for this command.

After offset computation is complete, reduce the timeout to 300 ms for normal commands.

The 4 values that are returned are the actual offsets that were measured (in counts per ADC read, where 1 ADC read takes 2 ms). Each value will be within the range of a signed integer (2 bytes).

We recommend that you save the measured offset values to disk storage. See ACQ SET OFFSETS to save time on subsequent power up sequences for the spectrometer controller. For many applications where environmental conditions are stable, weekly or monthly calculation of offsets may be sufficient.

HOST sends 1. Command 2. Channel # [0..1]	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Offset Gain 1 3. Offset Gain 10 4. Offset Gain 100 5. Offset Gain 1000
Example: Send "w0<CR>"	Receive "o" Receive "2,21,-19,22<CR>"

Spectrometer Control

ACQ SET OFFSETS "x"

This command relates only to the DataScan / DataLink controller- photometers. Because offset measurement is normally not required each time the spectrometer controller is powered on, the ACQ SET OFFSETS command is provided to load offsets that were measured and computed previously. If you intend to use the acquisition channels in more than one gain range the offsets are important. Therefore you should at least SET the offsets, if not MEASURE them, each time the spectrometer controller is powered on.

This command is used to explicitly set the acquisition offsets to some desired values. This command has the same effect and implications as the end result of the previous command ACQ MEASURE OFFSETS. The difference is that ACQ SET OFFSETS is user-determined while ACQ MEASURE OFFSETS is instrument-determined.

HOST sends 1. Command 2. Channel # [Ø..1] 3. Offset Gain 1 4. Offset Gain 1Ø 5. Offset Gain 1ØØ 6. Offset Gain 1ØØØ	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "xØ,2,21,-19,22<CR>"	Receive "o"

Spectrometer Control

ACQ CHANNEL GAIN SET "R"

This command is used to set the gain to use with the specified data acquisition channel. The valid values for the Gain parameter are from 0 to 4, with 0 to 3 specifying different amplifications, and 4 specifying AUTOGAIN. The following table shows the meaning of each number:

Parameter	Amplification
0	x 1
1	x 10
2	x 100
3	x 1000
4	AUTOGAIN

If AUTOGAIN is specified, then the gain with the highest amplification that is not over-ranged will be selected for each reading of the Analog to Digital Converter (ADC) for the duration of the integration time. Data is scaled to the lowest gain that was actually used during the integration; the gain that the data was scaled to is returned. For example, if you use an integration of 100 ms and are measuring a source with a 60 Hz intensity fluctuation, gains 0 and 1 may be used during the integration. The data that was read on gain 1 will be divided by 10 and added to the data that was read on gain 0. Hence, the data that is read back using ACQ READ DATA may have been acquired over multiple gains.

HOST sends 1. Command 2. Channel # [0..1] 3. Gain Level [0..4]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "R0,4<CR>"	Receive "o"

ACQ GAIN READ "S"

This command is used to read back the gain level for the specified channel that was set using the previous command ACQ GAIN SET.

HOST sends 1. Command 2. Channel # [0..1]	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Gain Level [0..4]
Example: Send "S0<CR>"	Receive "o" Receive "4<CR>"

Spectrometer Control

ACQ INTEGRATION TIME SET "O"

This command is used to set the integration time that will be used when the command ACQ START is sent. The integration time is in milliseconds and should be a factor of 2, with 2 ms being the shortest time. If the time is not set to a factor of two, the time will be rounded up by 1 ms (e.g. 5 ms will become 6 ms). The value returned when you issue the command ACQ INTEGRATION TIME READ will be the 6 ms (not 5). The maximum integration on any gain level (including AUTOGAIN) is 300,000 ms (5 minutes).

HOST sends 1. Command 2. Channel # [0..1] 3. Time Milliseconds (ms)	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "O0,50<CR>"	Receive "o"

ACQ INTEGRATION TIME READ "P"

This command is used to read back the integration that was set using the previous command ACQ INTEGRATION TIME SET. DataLink / DataScan systems take an ADC reading once every 2 ms. The value read back for these will always be a multiple of 2 .

HOST sends 1. Command 2. Channel # [0..1]	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Time Milliseconds (ms)
Example: Send "P0<CR>"	Receive "o" Receive "50<CR>"

Spectrometer Control

ACQ START "M"

The integration time and gain must be set prior to sending this command.

This command is used to start integrating data on the chosen channel. When data acquisition is started, the spectrometer controller will read data from the Analog to Digital Converter (ADC) once every 2 milliseconds (ms). Therefore, the integration time that is used by the system will always be a factor of two, with 2 ms being the shortest time. If the integration time is set to an odd number (e.g. 5 ms), it will be rounded up by 1 ms (e.g. 6 ms). The data that is returned by the ACQ DATA READ command will always be normalized to represent 1 ADC reading every 1 ms.

The channel parameter sets the input channel. Channel Ø is the firs input, channel 1 is the second. This differs from the labeling on the spectrometer controller that is marked 1 and 2. The command allows the number 2 as a parameter for Channels 1 and 2 to integrate signal on both channels simultaneously. The 2 ms rule still applies. In this case, you must send the command ACQ DATA READ twice in order to get the back the data from each channel.

For a 2-channel simultaneous data acquisition, the integration times for both channels must be the same. If the times are not the same, the integration time for the first channel will be used.

HOST sends 1. Command 2. Channel # [Ø..2]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "MØ<CR>"	Receive "o"

ACQ STOP "N"

This command is used to stop the integration of data that was initiated with the ACQ START command. Integrations in process on all channels will be stopped.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "N"	Receive "o"

Spectrometer Control

ACQ BUSY "Q"

This command is used to poll the spectrometer controller after the command ACQ START is issued in order to check to see if the data acquisition has completed. The Busy Status that is returned is not terminated with a <CR> character (as is the case with all busy tests). The values that may be returned are "q" for FUNCTION BUSY and "z" for FUNCTION NOT BUSY.

HOST sends	SPECTROMETER CONTROLLER sends
1. Command	1. Confirmation 2. Busy Status ["q" or "z"]
Example: Send "Q"	Receive "o" Receive "q"

ACQ READ DATA "T"

This command is used to read back the acquired data. Before issuing this command, the ACQ BUSY command should be sent to make sure that the integration is complete. If the ACQ START command was issued as a 2-channel data acquisition, the READ ACQ DATA command must be issued twice to retrieve data from both channels.

Along with the acquisition data, an over range status and gain level are also returned. If running AUTOGAIN, the acquisition data will already have offsets accounted for, and the data is scaled according to the amplification of the gain level. The over range status can be either a Ø or a 1, with Ø indicating that the data is not over ranged. A status of 1 means that the signal is too strong for the gain that was used. (In AUTOGAIN, it means that the least sensitive gain was over-ranged). The gain level that is returned for fixed gains is the same value that was set with the GAIN SET command. When using AUTOGAIN the returned gain level is the one that the acquired data was scaled to.

The value for ACQ DATA will be within the range of • 2 X10^9.

HOST sends		SPECTROMETER CONTROLLER sends	
1. Command		1. Confirmation	
2. Channel #	[Ø..1]	2. Acq Data	
		3. Over range Status	[Ø..1]
		4. Gain Level	[Ø..4]
Example: Send "TØ<CR>"		Receive "o" Receive "135ØØØ,Ø,3<CR>"	

10.6 High Voltage Commands:

HIGH VOLTAGE SET "U"

This command is used to set the specified high voltage module to a specified voltage. The maximum voltage for the installed module must not be exceeded in the HV Level parameter of this command.

For safety, we strongly recommend that you set HV to Ø when exiting your program.

Note also that when HV is turned on, you should allow some settling time for your detector to become stable before acquiring data. Four seconds is sufficient for many detectors. However, you should be aware that some photomultiplier tubes may take several minutes to settle when high voltage is applied. Cooled photomultipliers may take hours to stabilize to the extent necessary to measure the low light levels they are intended for. You may wish to experiment with your system to arrive at an optimal delay.

HOST sends 1. Command 2. HV Module # [Ø..1] 3. HV Level Volts	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "UØ,8ØØ<CR>"	Receive "o"

HIGH VOLTAGE READ "V"

This command is used to read back the present high voltage setting for the specified module that was established using HIGH VOLTAGE SET.

HOST sends 1. Command 2. HV Module # [Ø..1]	SPECTROMETER CONTROLLER sends 1. Confirmation 2. HV Level Volts
Example: Send "VØ<CR>"	Receive "o" Receive "8ØØ<CR>"

10.7 TTL I/O Port Commands:

These commands assert and read the TTL I/O lines on the SPECTROMETER CONTROLLER. Note that the SPEX232, SPEX488, JY232, and JY488 do not support these commands. The TTL lines generally are used to interface to ancillary devices that relate to the process that the spectrometer controller is associated with.

TTL WRITE OUTPUT "m"

This command is used to "toggle" the lines of the output port of the spectrometer controller. Each line of the output port is represented as 1 bit out of a total of 8 (or 1 byte).

The Binary value that results from setting the desired bits to 1 should be converted into an ASCII string.

In this diagram, bit #Ø is the Least Significant Bit, and bit #7 is the Most Significant Bit.

7	6	5	4	3	2	1	Ø

Example 1
To raise only the first line of the output port high, we would set bit #Ø to the value of 1, while all the rest of the bits would have the value of Ø. The binary value of the entire byte would be 1, therefore, the string to send would be "1".

Example 2
If we wanted to raise all of the lines of the output port high, we would set all 8 bits to have the value of 1. The binary value of the entire byte would be 255, therefore, the resulting string would be "255".

HOST sends 1. Command 2. Input Port Value [Ø..255]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "m255<CR>"	Receive "o"

TTL READ OUTPUT "n"

This command is used to read back the current value of the output port that was set either by issuing the previous command TTL WRITE OUTPUT or by enabling TTL AUTOMATIC OUTPUTS. The value is read back as a number which should be converted to its binary equivalent, then examined 1 bit at a time to determine which lines of the output port are raised high.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Input Port Value [Ø..255]
Example: Send "n"	Receive "o" Receive "255<CR>"

TTL READ INPUT "o"

This command is used to read back the current value of the input port. The value is read back as a number which should be converted to its binary equivalent, then examined 1 bit at a time to determine which lines of the input port are raised high.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Input Port Value [Ø..255]
Example: Send "o"	Receive "o" Receive "255<CR>"

TTL AUTOMATIC OUTPUTS "Z11"

This is an "extended" command. The Z character serves as a prefix for all extended commands. The spectrometer controller as part of the command designation reads the first parameter (11, in this case).

This command is used to enable or disable the automatic updating of the lines of the output port on the spectrometer controller when using the independent method scan commands. These lines can be used to provide a TTL signal to an external device that can then monitor some of the activities in the instrument or trigger some external event based on the states of the output port lines. If you are using the command TTL WRITE OUTPUT to control some of these port lines, you may want to disable TTL AUTOMATIC OUTPUTS for those lines.

For this command, 8 parameters corresponding to the 8 output port lines are necessary. Each parameter should either be a 1 (Enable) or a Ø (Disable).

7	6	5	4	3	2	1	Ø

Line	States	Description
Ø	Hi--Start of scan / Lo--End of scan.	This line is raised high when a scan is started using the SCAN START command, and it is lowered when the scan completes.
1	Hi--Start of integ. Lo--End of integ.	This line is raised high when each data integration begins, and it is lowered when each data integration completes.
2	Hi--Ch #1 data >= THRESHOLD.	A threshold is a user-defined value (see next command, THRESHOLD WRITE VALUES). This line goes high when the data acquired on channel 1 (after full integration) equals or exceeds the threshold High Value set for channel 1.
3	Hi--Ch #1 data <= THRESHOLD.	This line goes high when the data acquired on channel 1 (after full integration) equals or falls below the threshold Low Value set for channel 1.

4 Hi--Ch #2 data >= THRESHOLD.

This line goes high when the data acquired on channel 2 (after full integration) equals or exceeds the threshold High Value set for channel 2.

5 Hi--Ch #2 data <= THRESHOLD.

This line goes high when the data acquired on channel 2 (after full integration) equals or falls below the threshold Low Value set for channel 2.

6 Hi--Wait for input trigger.

This line is raised high when the INDEPENDENT SCAN is waiting for a trigger input.

7 Not used

Reserved for future use.

HOST sends 1. Command 2. Line Ø [Ø..1] 3. Line 1 [Ø..1] 4. Line 2 [Ø..1] 5. Line 3 [Ø..1] 6. Line 4 [Ø..1] 7. Line 5 [Ø..1] 8. Line 6 [Ø..1] 9. Line 7 [Ø..1]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "Z11,1,1,Ø,Ø,Ø,Ø,1,Ø<CR>"	Receive "o"

Spectrometer Control

THRESHOLD WRITE VALUES "I"

Be careful not to confuse this I (uppercase i) with a number 1or a lower case L. This command is used to write the User Defined Threshold values for the acquisition data.

This command is used with the "Independent" Scan Commands to specify the High and Low boundaries for data acquired on both channels, so that when data that falls outside that range is acquired, a line on the TTL output port can be set high or low.

To benefit from this feature, you must enable the output port lines that correspond to the data you want to monitor using the command TTL AUTOMATIC OUTPUTS.

For the High Value of channels 1 and 2, output port lines 2 and 4 (respectively) will be raised high if the acquired data on those channels equal or exceed those values. The output port lines will be lowered again when the data falls below the specified high values.

For the Low Value of channels 1 and 2, output port lines 3 and 5 (respectively) will be raised high if the acquired data on those channels equal or fall below those values. The output port lines will be lowered again when the data rises above the specified low values.

All 4 parameters must be within the range of • 2 X10^{9}.

HOST sends 1. Command 2. Channel 1 High Value 3. Channel 1 Low Value 4. Channel 2 High Value 5. Channel 2 Low Value	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "I000,200,1000,200<CR>"	Receive "o"

Spectrometer Control

THRESHOLD READ VALUES "J"

This command is used to read back the User Defined Threshold values that were set using the WRITE THRESHOLD VALUES command.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Channel 1 High Value 3. Channel 1 Low Value 4. Channel 2 High Value 5. Channel 2 Low Value
Example: Send "J"	Receive "o" Receive "1000,200,1000,200<CR>"

10.8 Independent Scan Commands:

These commands refer to a complete scan and acquisition routine definition for systems with SpectrAcq, or DataScan / DataLink spectrometer controllers. Once started, the spectrometer controller can perform the routine unattended. The data will be stored in memory and can be accessed at the end of the scan or "on the fly."

Data Memory Allocation:

The maximum number of total data points that can be stored is 5001. Total, in this case, means the amount of memory required to store all data points, taking into account the number of cycles and channels in the scan experiment, as well as the Data Storage mode.

For Example, if your routine will acquire 500 datapoints in each of two channels, and it will run for three cycles, the total memory needed is 500 X 2 X 3 = 3000 datapoints.

Data Acquisition Guidelines (for DataLink / DataScan Only):

Before performing any type of data acquisition using the AUTOGAIN option, the amplifier offsets must first be computed or set to obtain accurate measurements. In any of the FIXED GAINS, offsets are not subtracted. This applies to BOTH operating methods described below, with version 3.0 or higher of Flash.

This is accomplished by issuing the command MEASURE ACQUISITION OFFSETS or SET ACQUISITION OFFSETS. These offsets are automatically computed into the final datapoint result that is read by the host while acquiring with AUTOGAIN.

If you are running in any of the fixed gains, the offset is NOT automatically factored into the resulting data. If desired, offset subtraction must be performed after your controlling program reads the data back.

The command set is designed to allow maximum flexibility for the controlling program. Therefore, whenever possible, the data that is returned is in its "rawest" form; this is why, in fixed gains, there is no offset processing performed. However, with AUTOGAINing, any or all of the different gains can be used during a given integration period (depending on the fluctuation of the signal being measured). Since each gain can have a different offset value associated with it, it is most convenient to handle that processing in the firmware.

A few operating parameters must be set before the data acquisition can take place:

1. The integration time must be set.
2. The gain must be set for the channel being used.
3. If using a photomultiplier, the high voltage must be turned on.

SCAN SET PARAMETERS "p"

This command is used to specify all of the parameters that the SpectrAcq or DataScan / DataLink spectrometer controller will need in order to perform a complete scan independently.

Keep in mind also that the monochromators give the most reproducible results when scanning in the direction they were designed for:

- For wavelength drives; scan in the direction of increasing wavelength

- For wavenumber drives; scan in the direction of decreasing wavenumbers

The following list shows all the parameters that must sent with this command.

1. Scan Type - This is the first parameter, it will determine the usage of the other parameters. A value from 0 to 3 sets the scan type.
   0 Mono 1 Scan - monochromator connected to Mono Port 1 is scanned.
   1 Mono 2 Scan - monochromator connected to Mono Port 2 is scanned.
   2 Synchronous Scan - both monochromators scan synchronized, with the second separated by an offset.
   3 Time Base Scan - data acquired over time, with the monochromator parked at one position.

2. Start Position - a value in steps[1] that defines the starting position for all types of scans except Time Base Scan. If running Time Base Scan, this parameter is ignored. For Mono 1 or Synchronous scans, this parameter sets the start position for mono 1. For a Mono 2 scan, it sets the start position for mono 2.

3. End Position - a value in steps[1] that defines the ending position of a scan for all types except Time Base Scan. If running Time Base SCAN, this parameter is ignored. For a Mono 1 or Synchronous scan this is the end position for Mono 1. For a Mono 2 scan, it sets the end position for Mono 2.

4. Increment - a value in steps[1] that defines the scan drive increment between acquisition points of a scan. This number must be positive and less than or equal to the total travel of the monochromator you are scanning. If running a Time Base scan, this parameter is ignored.

5. Integration Time - period of time in milliseconds over which the data will be acquired at each point. For all scan types except Time Base Scan, the effective limit for this parameter is 300,000 ms (5 minutes), and this number should be an even number. See the ACQ INTEGRATION TIME SET command on page 81 for a further explanation.

   Because the Time Base Scans use a different scaling technique, in this case the limit on this parameter depends on the intensity of the signal as well as the gain that is being used. If we always read the maximum signal possible, then the following table indicates the maximum integration times:

Spectrometer Control

GAIN	INTEGRATION TIME
1 / AUTOGAIN	520 ms
10	5200 ms
100	52000 ms
1000	520000 ms

6. Scan Cycles - the number of times (255 max) you wish to repeat the same scan over the same defined region.

7. Dwell Time - the period of time that the system waits after moving the monochromator, before starting the integration, in milliseconds. This number must be positive and can be as large as 2,147,483,648 ms. of (the maximum value for a signed long integer). If no Dwell Time is desired, send "<0>" for this parameter.

8. Delay Time - the period of time that the system waits after completing each cycle, in milliseconds. Same as above, this number must be positive and can be as big as the maximum value for a signed long integer. If no Delay Time is desired a value of 0 may be used.

9. Mono 2 Start Position - for a Mono 2 scan or a Synchronous scan, this is the starting position in steps[1] for monochromator 2. For a Mono 1 Scan or Time Base Scan, the position in steps[1] where you wish to park your second monochromator. If a second monochromator does not exist in the spectrometer controller's configuration, this parameter will be ignored.

10. There are two different meanings for this parameter, depending on the scan type used:

    - Mono 1 Park Position - for a Mono 2 Scan or Time Base Scan, the position in steps[1] where you wish to park your first monochromator.

    - Sync Direction Indicator - for a Synchronous Scan, this parameter should be a position in steps[1] for monochromator 2 that indicates the direction that mono 2 will move from its start position. If the position entered is greater than the start position, the second monochromator will scan towards higher step positions. If a smaller position value is used, the scan direction is reversed.

11. Mono 2 Increment - for a Synchronous Scan only, the increment in steps[1] between acquisition points for the second monochromator. This number must be positive and less than or equal to the total travel of monochromator 2. If a second monochromator does not exist in the spectrometer controller's configuration, this parameter will be ignored.

12. Time Increment - for a Time Base Scan, the total amount of time in milliseconds between integration starts. If Time Increment is zero, then the Integration Time becomes the effective Time Increment. This number must be positive and can be as big as 2,147,483,648 ms. It must also be less than or equal to the Total Time. If not running a Time Base Scan this parameter will be ignored.

13. Total Time - for a Time Base Scan, the total time length of the scan in milliseconds. This number must be positive and can be as big as 2,147,483,648 ms. If not running a Time Base Scan this parameter will be ignored.

14. Channel Select - this indicates the data acquisition channel to be used for this experiment. 0 for channel 1; 1 for channel 2; and 2 to use both channels simultaneously.

[1] When calculating the value for offset, the Steps / Base unit and ratio of actual grating grooves per mm to base grating grooves per mm must be taken into account. Refer to Appendix 1 for the values needed to calculate the step positions and offsets for your monochromators.

15. Gain Channel 1 - if running the experiment with 0 or 2 in the above parameter (Channel), the gain set here will be active. 0 for gain 1; 1 for gain 10; 2 for gain 100; 3 for gain 1000; 4 for AUTOGAIN.

16. Gain Channel 2 - if running the experiment with 1 as the Channel parameter this gain will be used.

17. Shutter Mode - use 0 for AUTO shutter mode; 1 for MANUAL shutter mode. In AUTO mode, the active shutter will automatically be opened (if controllable) at the start of each cycle and closed at the end of each cycle.

18. Trigger Mode - use 0 for NO TRIGGER; 1 to wait for a trigger at the start of the EXPERIMENT; 2 to wait for a trigger at the start of each CYCLE; 3 to wait for a trigger at the start of each DATA POINT. The trigger can be in form of a hardware trigger on the I/O port or a software trigger (character "@").

19. Data Mode - use 0 to STACK the acquired scans of data separately in memory. Use 1 to SUM the acquired scans of data in memory. The Data mode is used when there is more than 1 cycle in the experiment. STACKed mode will put each data point for each cycle into a unique memory location; SUMmed mode will put each data point for the first cycle into a unique memory location, then sum the data of each subsequent cycle into the same memory location.

If the spectrometer controller receives faulty parameters, it will return one of the following error codes.

0 NO ERROR - This is good.

1 SCAN TYPE ERROR - The Scan Type parameter was not set within its valid range.

2 LOW INTEGRATION TIME ERROR - The Integration Time parameter was set to less than 1 ms.

3 LOW CYCLES ERROR - The Cycles parameter was set to 0. There must be at least 1 cycle.

4 CHANNEL NUMBER ERROR - The Channel parameter was not set within its valid range.

5 GAIN ERROR - The Gain Channel 1 or Gain Channel 2 parameter was not set within its valid range.

6 SHUTTER MODE ERROR - The Shutter Mode parameter was not set within its valid range.

7 TRIGGER MODE ERROR - The Trigger Mode parameter was not set within its valid range.

8 DATA ERROR - The Data Mode parameter was not set within its valid range.

9 LOW TOTAL TIME ERROR - The Total Time parameter was set to less than 1 ms for the TIME BASE SCAN. This error will not appear if running a scan type other than TIME BASE.

10 LOW STEP INCREMENT ERROR - The Step Position Increment parameter was set to 0 for the MONO 1 SCAN, MONO 2 SCAN, or SYNCHRONOUS SCAN. This error will not appear if running a TIME BASE SCAN.

11 SIZE ERROR - The defined experiment will produce more than the allowable 5001 data points.

Spectrometer Control

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Error Code
Example: Send "pØ,96ØØ,176ØØ,32,5Ø,2,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø ,4,Ø,Ø,Ø,Ø<CR>"	Receive "o" Receive "Ø<CR>"

SCAN START "q"

This command is used to start the experiment defined using the SCAN SET PARAMETERS command. The previous command must have been executed with NO ERROR (Ø) returned; otherwise the experiment will not start.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "q"	Receive "o"

SCAN STOP "v"

This command is used to stop the running routine initiated by SCAN START.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "v"	Receive "o"

Spectrometer Control

SCAN CURRENT STATUS "r"

This command is used to inquire the hardware status from the SpectrAcq or DataScan / DataLink spectrometer controller. It will return the following statuses:

0	The scan is idle and waiting to be started.
1	A monochromator is currently moving.
2	The spectrometer controller is performing the data acquisition.
3	The spectrometer controller is in the "DWELL" phase; or for Time Base Scan, it is pausing in between data acquisitions.
4	The spectrometer controller is in the "DELAY" phase.
5	NOT USED.
6	The spectrometer controller is waiting for a trigger.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2. Scan Status
Example: Send "r"	Receive "o" Receive "1<CR>"

SCAN CYCLE TO READ "s"

This command is used to set the cycle number to read stored data from the SpectrAcq or DataScan / DataLink spectrometer controller. The cycle number to read always defaults to 1 at the start of a new scan. When issuing the SCAN GET DATA command, the data value will be read from the cycles number that has been set with this command. If you are running a multi-cycle experiment and the Data Mode parameter is set to STACKED, you will need to use this command.

HOST sends 1. Command 2. Cycle #[1 ...]	SPECTROMETER CONTROLLER sends 1. Confirmation
Example: Send "s2<CR>"	Receive "o"

Spectrometer Control

SCAN GET DATA POINT NUMBER "t"

This command is used to get the number of the last data point that was acquired by the SpectrAcq or DataScan / DataLink spectrometer controller. The cycle number will also be returned.

HOST sends 1. Command	SPECTROMETER CONTROLLER sends 1. Confirmation 2 Data Pt. # 3. Cycle #
Example: Send "t"	Receive "o" Receive "3Ø,1<CR>"

Spectrometer Control

SCAN GET DATA "u"

This command is used to read back the acquired data from the SpectrAcq or DataScan / DataLink spectrometer controller. The cycle number must first be set using the SCAN SET CYCLE TO READ and the Data Index must be specified, for the spectrometer controller to send back the desired data point. Note that the first data point is numbered 1, not 0. If only 1 channel is specified in the Channel parameter when setting up the experiment, then the data for that channel along with it's over range and gain information will be returned. If 2 channels were specified, the data, over range, and gain will be returned for channel 1; then for channel 2.

If you may possibly be reading data before a scan is completed, be sure to check the SCAN STATUS to see if the scan is completed. If you want to read the data during a scan in process, use SCAN GET DATA POINT NUMBER to determine the range of valid data points to be read.

In all scan types other than Time Base Scan, if AUTOGAIN is used, the data is automatically scaled to the lowest gain that was used during the acquisition (see ACQ READ DATA for further explanation).

In Time Base Scans the data is scaled to the highest gain (1000) then returned.

For TIME BASE scans, the results will be more accurate if the data is read only after completion of the scan. This removes the uncertainty associated with host communication time from the interval between datapoints. For less time critical experiments, however, data can easily be read as soon as it is acquired in hardware.

The data is normalized to reflect 1 read of the ADC per 1 ms. In other than Time Base Scan, the ADC is actually read once every 2 ms; in Time Base Scan the ADC is read every 5 ms.

The second of the two values that are returned, "Over range & Gain," is a number that has two items of information encoded in it. If any of the ADC reads reached maximum, over range will be indicated by adding 8 to the gain level.

If you set any of the fixed gains, the gain level number will be as it was set.

If you set AUTOGAIN, the data will be scaled to one of the 4 gain levels. To determine the gain that was used for scaling, simply subtract 8 from the value returned. If the value is 8 or greater, an over range occurred. If the value is less than 8, then it is not over-ranged, and the remaining value is the gain used for scaling.

HOST sends	SPECTROMETER CONTROLLER sends
1. Command 2. Data Index # [1...]	1. Confirmation 2. Data 3. Over range & Gain 4. *Data 5. *Over range & Gain
Example: Send "u30<CR>"	Receive "o" Receive "1025,0<CR>"

* If 2 channel acquisition.

10.9 TRIAX Specific Commands

Spectrometer Control

The following commands will work only with the TRIAX series monochromators.

MW_X_SET_WORKING_ABS_POSITION "Z60"

Sets wavelength of TRIAX drive according to BASE grating of 1200 grooves/mm independent of current position

HOST sends 1. Command 2. Mono Number 3. Wavelength<CR>	SPECTROMETER CONTROLLER sends Move in nm
Example: Send "Z60,1,0.00<CR>"	Receive "o"

MW_X_MOVE_WORKING_ABS_POSITION "Z61"

Moves wavelength of TRIAX drive according to BASE grating.

HOST sends 1. Command 2. Mono Number 3. Wavelength<CR>	SPECTROMETER CONTROLLER sends Move in nm
Example: Send "Z61,1,546.074<CR>"	Receive "o"

MW_X_READ_WORKING_ABS_POSITION "Z62"

Reads wavelength of TRIAX drive according to BASE grating.

HOST sends 1. Command 2. Mono Number 3. Wavelength<CR>	SPECTROMETER CONTROLLER sends Move in nm 1. Wavelength
Example: Send "Z62,1<CR>"	Receive "o" Receive "546.074<CR>"

Spectrometer Control

MW_X_SET_INDEX_DEVICE_POS "Z451"

Move Index Device (Turret) to new position. TRIAX turret is Ø based, i.e. Ø,1,2 are the valid positions.

HOST sends 1. Command 2. Mono Number 3. Device Type 4. Device Number 5. New Position	SPECTROMETER CONTROLLER sends
Example: Send "Z451,Ø,Ø,Ø,1<CR>"	Receive "o"

MW_X_READ_INDEX_DEVICE_POS "Z452"

Reads Index Device (Turret) position.

HOST sends 1. Command 2. Mono Number 3. Device Type 4. Device Number	SPECTROMETER CONTROLLER sends 1. Device Position
Example: Send "Z452,Ø,Ø,Ø<CR>"	Receive "o" Receive "1<CR>"

MW_X_INDEX_DEVICE_STATUS "Z453"

Gets Index Device (Turret) status.

HOST sends 1. Command 2. Mono Number 3. Device Type 4. Device Number	SPECTROMETER CONTROLLER sends 1. Status 'z' if NOT BUSY 'q' if BUSY
Example: Send "Z453,Ø,Ø,Ø<CR>"	Receive "o" Receive "z<CR>"

11 In Case of Difficulty ...

Your instrument is designed to provide years of reliable service. If you are experiencing a problem, reviewing this section before contacting us will save time and help you eliminate some simple errors that can be easily corrected.

11.1 Troubleshooting:

Some of the more common difficulties that may be encountered are listed below. With each, some suggestions are given that will help correct the problem for most cases.

The HandScan / KeyLink terminal does not perform the self-test of its display upon power-on of the supporting spectrometer controller or spectrometer.

- Check cabling according to the Getting Started section.
- Check the supporting spectrometer controller's power connections and fuses.
- Check the supporting spectrometer controller's internal cable connections.

No communications with the HandScan / KeyLink. (No version number displayed after pressing **Decimal Point.**)

- Check cable connections for partially dislodged connectors.
- Try pressing **Decimal Point** a few more times.
- Refer to The HandScan / KeyLink manual Appendix to check and/or correct the terminal configuration parameters.
- Refer to the Service Policy section to contact us for further information, or possible exchange of the terminal.

System not responding to any commands:

- Check external cable connections. See the Getting Started section for proper connection.
- Review the instructions pertaining to the interface type (IEEE 488 or RS-232) referred to in the Getting Started section to check for proper set-up and communications. Be sure to read the footnotes.
- Refer to the manuals provided with the instrument(s) in your system, for further troubleshooting relating to the spectrometer controller.
- Check system configuration with the HWCFG program on the support diskette to see that it matches the actual hardware.
- If you are running your own software along with the HandScan / KeyLink, stop your program and try starting over and accessing the system with the hand-held terminal only. See Getting Started to check for proper set-up and communications. Be sure to read the footnotes. Next load the example programs on the support software diskette provided with the system to see if the system can be accessed from an IBM compatible PC.
- Try using the re-boot command "<222>"

System responds to some commands, but not all.

- Check to be sure that the failing command is valid under the present conditions.
- Check all required parameters and make certain that they are within software and hardware limits for that function.
- Check to see that the configuration matches the actual hardware by running the HWCFG program on the support diskette. In particular, check that the device you are having difficulty with is configured properly.
- If you are running your own software along with the HandScan / KeyLink, stop your program and try starting over and accessing the problematic functions with the hand-held terminal only. See Getting Started to check for proper set-up and communications. Be sure to read the footnotes. Next load the example programs on the support software diskette provided with the system to see if the device in question can be accessed from an IBM compatible PC.

12 Service Policy:

If you need assistance in resolving a problem with your instrument, contact our Customer Service Department directly, or through our representative or affiliate in your area, if outside the United States.

Often it is possible to correct, reduce, or localize the problem to a replaceable component through discussion with our Customer Service Engineers.

All instruments are covered by a warranty. The warranty statement is printed on the inside cover of this manual. Service for out-of-warranty instruments is also available, for a fee. Contact us for details and cost estimates.

If an instrument or component must be returned, The method described in this section should be followed to expedite servicing and reduce your downtime.

If your problem relates to software, please verify your computer's operation by running any diagnostic routines that were provided with it. If there is a support diskette provided, refer to the manual for that diskette, and follow the troubleshooting procedures. Be ready to provide version numbers for the DOS that you are using, as well as the software version and firmware version of any controller or interface options in your system. Also knowing the memory type and allocation, and other computer hardware configuration data from the PC's CMOS Setup utility may be useful.

12.1 Return Authorization:

A Return Authorization Number issued by our Customer Service Department must accompany all Instruments and Components returned to the factory.

If in the United States, SPEX and Jobin-Yvon customers may contact the Customer Service department directly:

- By phone at (908) 549-7144.
- The fax number for Service is (908) 549-2571.
- For Spex service you may write to:
  SPEX Industries,
  Spectrometer Customer Service
  3880 Park Avenue, Edison, NJ 08820.
- The correspondence address for Jobin-Yvon service in the U.S. is:
  Instruments SA, Inc.
  J-Y Optical Systems Division,
  Spectrometer Customer Service,
  3880 Park Avenue, Edison, NJ 08820.

From other locations worldwide, contact the representative or affiliate for your area.

To issue a Return Authorization number, we require:

- The model and serial number of the instrument
- A list of items and/or components to be returned
- A description of the problem, including operating settings

- The instrument user's name, mailing address, telephone, and telex numbers
- The shipping address for shipment of the instrument to you after service
- Your Purchase Order number and billing information for non-warranty services
- Our original Sales Order number is helpful to know
- Your Customer Account Number, if known, is also helpful
- Any special instructions

Appendix 1: Monochromator Setup Parameters:

Mono Type	Model	Base Grating	Steps / Base Unit	Min Limit	Max Limit	Min Freq., Hz	Max Freq., Hz	Ramp Time, ms	Back-lash Steps	Spectrograph Ent/Exit Focal Length, mm	Included Angle, °	Incline, °	Trig Cal 1	Trig Cal 2	Model
Triax 1	Triax 180,190	1200	N/A	0	1400	1000	4000	250	400	190	30	-10			Triax 180,190
Triax 2	Triax 320	1200	N/A	0	1400	1000	4000	250	400	320	21.26	12.5			Triax 320
Triax 3	Triax 320 microstep	1200	N/A	0	1400	1000	4000	250	400	320	21.26	12.5			Triax 320 microstep
2	500M, X	1200	400 / A	0	15000	1000	36000	3000	20000	500	22.54	0	-1	-1	500M, X
2	750M	1200	400 / A	0	15000	1000	36000	3000	20000	750	13.738	0	-1	-1	750M
2	1000M	1200	400 / A	0	15000	1000	36000	3000	20000	1000	11.251	0	-1	-1	1000M
2	1250M	1200	400 / A	0	15000	1000	36000	3000	20000	1260	11.0158	0	-1	-1	1250M
2	1702	1200	400 / A	0	15000	1000	36000	3000	20000	750	13.6	0	-1	-1	1702
2	1704	1200	400 / A	0	15000	1000	36000	3000	20000	1000	9.8	0	-1	-1	1704
3	1269	1200	500 / A	0	15000	1000	36000	2000	25000	1260	9.434	0	-1	-1	1269
4	1403	1800	400 / cm^{-1}	11000	31000	1000	28000	2000	20000	1695	10	0	-1	-1	1403
5	1404	1200	400 / A	0	15000	1000	28000	2000	20000	1695	10	0	-1	-1	1404
6	1680	1200	50 / nm	0	1000	400	400	1000	500	225.55	30	NA	-1	-1	1680
7	1681	1200	50 / nm	0	1000	400	400	1000	500	225.55	30	0	-1	-1	1681
8	1870B	1200	50 / A	0	13000	400	400	1000	5000	500 / 480	24.33	2.87	-1	-1	1870B
9	1870C	1200	400 / A	0	13000	1000	32000	2000	20000	500 / 480	24.33	2.87	-1	-1	1870C
10	1877A	1200	50 / nm	0	1000	400	400	1000	1000	340 / 594.15	30	0	-1	-1	1877A

Spectrometer Control

Mono Type	Model	Base Grating	Steps / Base Unit	Min Limit	Max Limit	Min Freq., Hz	Max Freq., Hz	Ramp Time, ms	Back-lash Steps	Spectrograph Ent/Exit Focal Length, mm	Included Angle, °	Incline, °	Trig Cal 1	Trig Cal 2	Model
	1877C														1877C
11	1877B 1877D 1877E[2]	1200	4000/nm	0	1000	1000	40000	2000	40000	340 / 594.15	30	0	-1	-1	1877B 1877D 1877E
12	340S	1200	50 / nm	0	1000	400	400	1000	500	227.3/342	30	0	-1	-1	340S
12	340E	1200	50 / nm	0	1000	400	400	1000	500	342	30	0	-1	-1	340E
12	1877E[3]	1200	50 / nm	0	1000	400	400	1000	500	340 / 594.15	30	0	-1	-1	1877E
NA	220/270 slit	NA	157.48 /mm	0	7	300	300	1000	8	NA	NA	NA	0	-1	220/270 slit
13	270M	1200	32 / nm	0	1100[3]	2560	2560	1000	320	227.3/267.84	18	11	0	-1	270M
14	H10	1200	20 / nm	0	850	300	450	2000	200	100/94	61.4	N.A	-1	-1	H10
15	DH10	1200	20 / nm	0	860	300	450	2000	200	200/188	61.4	N.A	-1	-1	DH10
15	H20	1200	20 / nm	0	860	300	450	2000	200	200/188	61.4	N.A	-1	-1	H20
18	HR320	1200	20 / A	0	13000	300	450	2000	200	320	24	0	-1	-1	HR320
19	HR460	1200	160 / nm	0	1300	2560	5500	2000	320						HR460

[2] Spectrograph Stage

[3] Prefilter stage.

Spectrometer Control

Mono Type	Model	Base Grating	Steps / Base Unit	Min Limit	Max Limit	Min Freq., Hz	Max Freq., Hz	Ramp Time, ms	Back-lash Steps	Spectrograph Ent/Exit Focal Length, mm	Included Angle, °	Incline, °	Trig Cal 1	Trig Cal 2	Model
	CP200	Varies	NA	See grating specifications		NA	NA	NA	N.A	210.95/187.7	13.67	15.27	-1	-1	CP200

Appendix 2: Connector Pin Assignments

SPEX232 / JY232 Interface and DataScan / DataLink RS-232 connector on the spectrometer controller:

Pin#	Name	Function
2	TXD	Transmits data from the SPEX-232
3	RXD	Receives data
6	DTR	Data Terminal Ready (to receive a byte)
7	Ground	Reference / return for all other lines
22	+5V	To assert other lines high as required for additional handshaking

IEEE 488 Interface connector

Contact#	Name	Function
1	DIO 1	Data input / output line
2	DIO 2	Data input / output line
3	DIO 3	Data input / output line
4	DIO 4	Data input / output line
5	EOI (24)	End Or Identify
6	DAV	Data Valid
7	NRFD	Not Ready For Data
8	NDAC	Not Data Accepted
9	IFC	Interface Clear
1Ø	SRQ	Service Request
11	ATN	Attention
12	SHIELD	Protective Shield
13	DIO 5	Data input / output line
14	DIO 6	Data input / output line
15	DIO 7	Data input / output line
16	DIO 8	Data input / output line
17	REN (24)	Remote Enable
18	GND (6)	Signal ground for DAV
19	GND (7)	Signal ground for NRFD
2Ø	GND (8)	Signal ground for NDAC
21	GND (9)	Signal ground for IFC
22	GND (1Ø)	Signal ground for SRQ
23	GND (11)	Signal ground for ATN
24	GND LOGIC	Signal ground for EOI, REN

Appendix 3: Sample RS-232 Programs

The support Diskette provided with your system contains sample programs written in GWBASIC. Please understand that these example programs are provided free of charge and they have been tested within reason. We do not guarantee or support these programs. They are intended as examples only.

If you construct programs based on these Command Set examples, we strongly recommend that you add prudent error trapping and protection features to your program to protect your system and enhance ease-of-use. Much aggravation can be avoided by such things as checking limits on stepper motor travel and speed, as well as other variables.

232_SCAN.BAS is a spectrometer control program that has been used at the factory to test systems like yours. Conversions from working units to motor steps are performed by the program. Data acquisition capabilities are not provided. It offers a formatted input screen to enter parameters and execute a scan routine. A copy of this screen was printed at the factory during the final testing of your system. It is included in the documentation shipped with the spectrometer controller.

```
1 REM
9 REM -----------------------------------
10 REM
11    RBUF$ = SPACE$(132)  : PBUF$ = SPACE$(132)
12    OBUF$ = SPACE$(132)
13    JUNK$ = SPACE$(132)
14    ICH$ = SPACE$(1)
15    ACK$ = SPACE$(1)
16    CR$ = CHR$(13)
17    comment$ = space$(132)
18    MINHARDWAREFREQ# = 1#
20    debug =0
30    ON ERROR GOTO 16900
100 REM ----------------------------------
110  REM GW-BASIC Example Program
120  REM
130  REM This is a sample program intended for reference only.
140 REM ----------------------------------
150 REM
160 REM Test Program for Maple Wood / Leaf Scan Commands
170 REM RS-232 version.
180 REM
190 REM ----------------------------------
200 REM ----------------------------------
210 REM ----------------------------------
220    PRINT "Begin RS-232 Communications Setup"
230 REM ----------------------------------
240 REM     Open Com Port 2
250 REM 4800 baud, no parity, 8 bits, 1 stop, suppress Request to Send
260 REM
270    OPEN "com2:4800,N,8,1,RS,DS0" FOR RANDOM AS #2
272    LOOPit = 0
274    DELAY = .5: GOSUB 6900        'Wait a little'
280 REM ..
290 REM     Flush anything in input buffer
310    WHILE (LOC(2) > 0)
320      JUNK$ = INPUT$(1, #2): SOUND 2000, 1
330    WEND
```

```
430 REM
440 REM     Begin Auto Baud Sequence
450 REM ..                                    'Send <SP>'
460    PRINT #2, " "; : SOUND 400, 1
470    DELAY = .5: GOSUB 6900        'Wait a little'
480    IF (LOC(2) > 0) THEN 510 ELSE 490
490 REM ..
500    GOTO 450                        'No Response. Try Again'
510    ICH$ = INPUT$(1, #2): SOUND 1000, 1
515    LOOPit = LOOPit + 1
520    PRINT "Received ", ICH$
530    IF (ICH$ = "*") GOTO 590        'Autobaud ACK O.K.'
540 REM
550 REM  If already auto bauded, take appropriate action.
560    IF (ICH$ = "B") GOTO 690        'Init Program ACK O.K.'
562    IF (ICH$ = "F") GOTO 760        'Main Program ACK O.K.'
564 REM
565    WHILE (LOC(2) > 0)               'Flush input'
566      JUNK$ = INPUT$(1, #2): SOUND 2000, 1
567    WEND
568 REM
569    IF ( LOOPit < 3 ) GOTO 280        'Bad Response. Try Again'
570 REM
575 REM  Maybe we have already auto bauded; these commands will put us in
576 REM   a known state without restarting the instrument.
577    PRINT #2, CHR$(248);          'Send Set Intelligent Mode Pseudo Command'
578    PRINT #2, CHR$(222);             'Send Software Reboot if Hung'
580    GOTO 280                         'Keep on Trying
590 REM ..
600    DELAY = .5: GOSUB 6900           'Wait a little'
610    INPUT #2, BOOTMSG$               'Flush out Boot Message'
620    PRINT BOOTMSG$
630 REM
640     PRINT #2, CHR$(247);              'Send Util Startup Intelligent Mode
command'
650 REM
660    ICH$ = INPUT$(1, #2)             'ACK of Util Startup Intelligent Mode'
670    IF (ICH$ <> "=") GOTO 272        'Bad ACK. Try Again'
680 REM
690 REM ..    Jump to Main Program
700    PRINT "Jump to Main Program"
710    PRINT #2, "O";                   'Send Jump Sequence'
720    PRINT #2, "2000"; : PRINT #2, CHR$(0);
730 REM
740    WHILE (INPUT$(1, #2) <> "*")     'Wait for Main Program Response'
750    WEND
760 REM ..
770    SOUND 500, 4: SOUND 1000, 4: SOUND 500, 4
780    PRINT "RS-232 Communications Established"
790   PRINT #2, "y";                   'NO comma or <CR> after command'
800   GOSUB 6200      'Wait For Confirmation'
810 REM
820   INPUT #2, VERNO$
830   PRINT "Boot Version:", VERNO$
840 REM
850   PRINT #2, "z";
860   GOSUB 6200      'Wait For Confirmation'
```

```
870   INPUT #2, VERNO$
880   PRINT "Flash Version:", VERNO$
5700 REM
5750 REM ----- Send Mono Init Command
5800  ICH$ = "A"
5900  GOSUB 14500      'Put ICH$ to RS-232 and wait for confirm
5950 PRINT " " : PRINT " " :  PRINT "Hit any key:"
6000 WHILE (INKEY$ = "") : WEND
6050 REM
6100 GOTO 16600  '*** Jump To User Interface
6150 REM_____________________________________________________________________
6200 REM ***** Input ACK ****  and test for valid
6205  if (debug = 1) then print "wait for ACK"
6210   ACK$ = INPUT$(1, #2): IF (SAYOK = 1) THEN SOUND 1000, 2
6220   IF (ACK$ = "o") THEN 6230 ELSE 6260
6230   REM ..
6240   IF (SAYOK = 1) THEN PRINT "Receive was O.K."
6250   GOTO 6300
6260 REM ..
6270   IF (ACK$ = "b") THEN PRINT "Receive was bad"
6280   PRINT "Character Received was: "; ASC(ACK$), ACK$
6290   STOP
6300 REM ..
6310   return
6900 REM_____________________________________________________________________
6950 REM ***** Time Delay of delay *****
7000   T0 = TIMER
7050   WHILE ((TIMER - T0) < DELAY)
7100   WEND
7150   RETURN
7200 REM_____________________________________________________________________
9700 REM_____________________________________________________________________
9750 REM ***** Short Move to target and display position until done *****
9800  GOSUB 11650            'Move Relative
9850 WHILE ( MOTORBUSY = 1 )
9900    GOSUB 17100         'Get and Display Position
9950    GOSUB 12250         'Test Motor Busy
10000  WEND
10050  GOSUB 17100         'Get and Display Position
10100  RETURN
10150 REM_____________________________________________________________________
10200 REM ***** Backlash Move to target and display position until done *****
10250  IF ( TARGET# > STEPS# ) THEN GOTO 10450
10300  TMP# = TARGET# : TARGET# = TMP# - BACKLASH# : GOSUB 10600
10350  IF (HALTSCAN = 1) THEN RETURN
10400  TARGET# = TMP#
10450 REM ...
10500  GOSUB 10600
10550  RETURN
10600 REM_____________________________________________________________________
10650 REM ***** Move to target and display position until done *****
10700  GOSUB 11650           'Move Relative
10750  HALTSCAN = 0
10800  WHILE ( MOTORBUSY = 1 )
10850    GOSUB 26050         'GetKey
10900    IF (KEYVALUE = 0) THEN GOTO 11350                 'no key
10950    IF (KEYVALUE = F6KV) THEN GOSUB 30650 : GOTO 11350  'shutter
```

```
11000     IF (KEYVALUE = F8KV) THEN 11050 ELSE 11250      'halt
11050 REM ... 'STOP SCAN
11100      GOSUB 12000    'stop motor
11150      HALTSCAN = 1  'flag to others than scan was halted
11200      GOTO 11350
11250  REM ... invalid key during scan
11300    SOUND 500,1
11350  REM ...
11400    GOSUB 17100         'Get and Display Position
11450    GOSUB 12250         'Test Motor Busy
11500  WEND
11550  GOSUB 17100         'Get and Display Position
11600  RETURN
11650 REM_____________________________________________________________________
11700 REM ***** Move Mono Relative in Steps *****
11750   RELSTEPS# = TARGET# - STEPS#
11800   OBUF$ = "F" + "0" + "," + STR$(RELSTEPS#) + CR$
11850   GOSUB 13950      'Put OBUF$ to RS-232 and wait for confirm
11900   MOTORBUSY = 1
11950   RETURN
12000 REM_____________________________________________________________________
12050 REM ***** Motor Stop *****
12100   ICH$ = "L"
12150   GOSUB 14500      'Put ICH$ to RS-232 and wait for confirm
12200   RETURN
12250 REM_____________________________________________________________________
12300 REM ***** Motor Busy *****
12350   ICH$ = "E"
12400   GOSUB 14500      'Put ICH$ to RS-232 and wait for confirm
12450 REM
12460   ACK$ = INPUT$(1, #2)
12650   IF (ACK$ = "q") THEN 12700 ELSE 12750
12700   MOTORBUSY = 1 : GOTO 12800
12750   MOTORBUSY = 0 : GOTO 12800
12800 REM ...
12850   RETURN
12860 REM_____________________________________________________________________
12861 REM ***** Motor Status *****
12862   ICH$ = "K"
12863   GOSUB 14500      'Put ICH$ to RS-232 and wait for confirm
12864   REM
12865   GOSUB 14250      'Get RBUF$ from RS-232
12866   limit = val(rbuf$)
12870   return
12900 REM ________________________________________________________
12950 REM ***** Set Mono Position *****
13000   STEPS# = WAVE# * STEPSPERUNIT#
13050   OBUF$ = "G0," + STR$(STEPS#) + CR$
13100   GOSUB 13950      'Put OBUF$ to RS-232 and wait for confirm
13150   RETURN
13200 REM ________________________________________________________
13250 REM ***** Read Mono Position
13300   OBUF$ = "H0" + CR$
13350   GOSUB 13950      'Put OBUF$ to RS-232 and wait for confirm
13400   REM
13450   GOSUB 14250      'Get RBUF$ from RS-232
13500   PBUF$ = RBUF$
```

```
13550   STEPS# = VAL(PBUF$) : WAVE# = STEPS# / STEPSPERUNIT#
13600   RETURN
13650 REM______________________________________________________________
13700 REM ***** Set Motor Speed
13750   FREQMIN$ = STR$(F1#) : FREQMAX$ = STR$(F2#) : RAMPTIME$ = STR$(RT#)
13800   OBUF$ = "B"+"0,"+ FREQMIN$ +","+ FREQMAX$ +","+ RAMPTIME$ + CR$
13850   GOSUB 13950     'Put OBUF$ to RS-232 and wait for confirm
13900   RETURN
13950 REM______________________________________________________________
14000 REM ***** Put OBUF$ to RS-232 ***** and wait for confirm
14010  if (debug = 1) then print "Send OBUF$=",obuf$
14050  print #2,OBUF$;
14150  GOSUB 6200     'Wait For Confirmation'
14200  RETURN
14250 REM______________________________________________________________
14300 REM ***** Get RBUF$ from RS-232 *****
14310  if (debug = 1) then print "Wait for RBUF$"
14350  input #2,RBUF$
14360  if (debug = 1) then print "Received RBUF$=",obuf$
14450  RETURN
14500 REM______________________________________________________________
14550 REM ***** Put ICH$ to RS-232 ***** and wait for confirm
14560  if (debug = 1) then print "Send ICH$=",ich$
14600  print #2,ICH$;
14700  GOSUB 6200     'Wait For Confirmation'
14750  RETURN
14800 REM______________________________________________________________
14850 REM ***** Accessory Busy *****
14900  ICH$ = "1" : GOSUB 14500 'Put ICH$ to RS-232 and confirm
14950  REM
15000   ACK$ = INPUT$(1, #2)
15100  IF (ACK$ = "q") THEN ACCESSORYBUSY = 1 ELSE ACCESSORYBUSY = 0
15150  RETURN
15200 REM______________________________________________________________
15250 REM______________________________________________________________
15300 REM *** SHUTTER_OPEN  else SHUTTER_CLOSE ***
15350  IF ( SHUTTER = 1) THEN OBUF$ = "W0" + CR$ ELSE OBUF$ = "X0" + CR$
15400  GOSUB 13950     'Put OBUF$ to RS-232 and wait for confirm
15450  GOSUB 19050     'Display Shutter Position
15452  ACCESSORYBUSY = 1
15500  RETURN
15550 REM______________________________________________________________
15600 REM *** TURRET_POSITION_1  else  TURRET_POSITION_0 ***
15650  IF ( TURRET = 1) THEN OBUF$ = "a0" + CR$ ELSE OBUF$ = "b0" + CR$
15700  GOSUB 13950     'Put OBUF$ to RS-232 and wait for confirm
15750  GOSUB 18600      'Display Accessories
15752  ACCESSORYBUSY = 1
15800  RETURN
15850 REM______________________________________________________________
15900 REM *** ENTR_MIRROR_SIDE_POSITION  else  ENTR_MIRROR_FRONT_POSITION ***
15950  IF ( ENTMIRROR = 1) THEN OBUF$ = "c0" + CR$ ELSE OBUF$ = "d0" + CR$
16000  GOSUB 13950     'Put OBUF$ to RS-232 and wait for confirm
16050  GOSUB 18600      'Display Accessories
16052  ACCESSORYBUSY = 1
16100  RETURN
16150 REM______________________________________________________________
16200 REM *** EXIT_MIRROR_SIDE_POSITION  else  EXIT_MIRROR_FRONT_POSITION ***
```

```
16250  IF ( EXTMIRROR = 1) THEN OBUF$ = "e0" + CR$ ELSE OBUF$ = "f0" + CR$
16300  GOSUB 13950      'Put OBUF$ to RS-232 and wait for confirm
16350  GOSUB 18600       'Display Accessories
16352  ACCESSORYBUSY = 1
16400  RETURN
16450 REM____________________________________________________________________________
16500 REM
16550 REM --------------------------------
16600    ON ERROR GOTO 16900
16650 REM --------------------------------
16700 REM ...
16750   GOSUB 36350         'Input Last Set of User Parameters
16800   GOTO 19350           'Jump To Main
16850 REM____________________________________________________________________________
16900 REM .....Error Handler
16950   PRINT "***** Error =", ERR, " on line number", ERL
17000   LASTERROR = ERR
17050   REM IF (LASTERROR = 62) THEN RESUME NEXT ELSE STOP
17060   resume next
17100 REM____________________________________________________________________________
17150 REM ***** Get and Display Position *****
17200  GOSUB 13200       'Read Mono Position
17210  gosub 12860       'Check limits
17250  LOCATE 1, 40
17300  PRINT USING ": #####.###  ########  ##"; WAVE#, STEPS#, limit
17350  RETURN
17400 REM____________________________________________________________________________
17450 REM ***** Display Scan *****
17500  LOCATE 3, 40
17550  PRINT USING ": #####.###"; STARTSCAN#
17600  LOCATE 4, 40
17650  PRINT USING ": #####.###"; ENDSCAN#
17700  LOCATE 5, 40
17750  PRINT USING ": #####.###    ######"; SCANSPEED#, SCANFREQ#
17800  RETURN
17850 REM____________________________________________________________________________
17900 REM ***** Display Tweak Speed *****
17950  LOCATE 13, 40
18000  IF (TWEEKSPEED <> 1) THEN PRINT ":  Fast Increment"
18050  IF (TWEEKSPEED = 1) THEN PRINT ":  Slow Increment"
18100  RETURN
18150 REM____________________________________________________________________________
18200 REM ***** Markers *****
18250  LOCATE 16, 10
18300  PRINT USING ": #####.###"; MARKERA#
18350  LOCATE 17, 10
18400  PRINT USING ": #####.###"; MARKERB#
18450  LOCATE 18, 10
18500  PRINT USING ": #####.###"; MARKERC#
18550  RETURN
18600 REM____________________________________________________________________________
18650 REM ***** Display Accessories *****
18700  LOCATE 8, 40
18750  PRINT ": ", TURRET
18800  LOCATE 9, 40
18850  PRINT ": ", ENTMIRROR
18900  LOCATE 10, 40
```

```
18950  PRINT ": ", EXTMIRROR
19000  RETURN
19050 REM________________________________________________________________
19100 REM ***** Display Shutter *****
19150  LOCATE 7, 40
19200  IF (SHUTTER = 0) THEN PRINT ":  CLOSED" ELSE PRINT ":  OPEN  "
19250  RETURN
19300 REM________________________________________________________________
19350 REM________________________________________________________________
19400 REM ***** Main *****
19450  F1KV = 187
19500  F2KV = 188
19550  F3KV = 189
19600  F4KV = 190
19650  F5KV = 191
19700  F6KV = 192
19750  F7KV = 193
19800  F8KV = 194
19850  F9KV = 195
19900  F10KV = 196
19950  LEFTKV = 203
20000  RIGHTKV =205
20050  TKV =  84
20100  EKV =  69
20150  XKV =  88
20200  CTRLA = 1
20250  CTRLB = 2
20300  CTRLC = 3
20350 REM
20400  ON KEY (1) GOSUB 21050
20450  ON KEY (2) GOSUB 21100
20500  ON KEY (3) GOSUB 21150
20550  ON KEY (4) GOSUB 21200
20600  ON KEY (5) GOSUB 21250
20650  ON KEY (6) GOSUB 21300
20700  ON KEY (7) GOSUB 21350
20750  ON KEY (8) GOSUB 21400
20800  ON KEY (9) GOSUB 21450
20850  ON KEY (10) GOSUB 21500
20900  ON KEY (12) GOSUB 21550
20950  ON KEY (13) GOSUB 21600
21000  GOTO 21700
21050  HOTKEY = F1KV : RETURN
21100  HOTKEY = F2KV : RETURN
21150  HOTKEY = F3KV : RETURN
21200  HOTKEY = F4KV : RETURN
21250  HOTKEY = F5KV : RETURN
21300  HOTKEY = F6KV : RETURN
21350  HOTKEY = F7KV : RETURN
21400  HOTKEY = F8KV : RETURN
21450  HOTKEY = F9KV : RETURN
21500  HOTKEY = F10KV : RETURN
21550  HOTKEY = LEFTKV : RETURN
21600  HOTKEY = RIGHTKV : RETURN
21650 REM
21700  KEY(1) ON: KEY(2) ON: KEY(3) ON: KEY(4) ON: KEY(5) ON
21750  KEY(6) ON: KEY(7) ON: KEY(8) ON: KEY(9) ON: KEY(10) ON
```

```
21800  KEY(12) ON: KEY(13) ON
21850 REM
21900  GOTO 22100
21950  CLS
22000  LOCATE 15, 20
22050  INPUT "Mono Type: ", MONOWANTED
22100  GOSUB 26700              'Get Mono Type and Parameters
22150  PRINT " ": PRINT "Hit any key:"
22200  WHILE (INKEY$ = "") : WEND
22250 REM
22300  CLS
22350 REM
22400  SAYOK = 0       'TURN OFF RS-232 ACK sound and OK print out flag
22450  MOTORBUSY = 0 : MOVEBUSY = 0 : HALTSCAN = 0
22452  ACCESSORYBUSY = 0 : tupdate = timer
22500  WAVE# = 0# : TARGET# = 0#
22550  TWEEKINC = 1
22600  TWEEKSPEED = 1 : TWEEKMAX = STEPSPERUNIT# + 1
22650  STEPS# = 0#
22700  SHUTTER = 0
22750  TURRET = 0
22800  ENTMIRROR = 0
22850  EXTMIRROR = 0
22900  SCANSPEED# = SCANFREQ# / STEPSPERUNIT#
22950 REM
23000  MENUROW = 1 : MENUCOL = 15 : GOSUB 25000       'Put up Main Menu
23050  ROW = 20 : COL = 20 : MESSAGE$ = "Command: " : GOSUB 28100
23100  ROW = 20 : COL = 20+9
23150 REM
23200  GOSUB 30150       'Calibrate
23250  GOSUB 17100        'Display Position
23300  GOSUB 17400        'Display Scan
23350  GOSUB 17850        'Display Tweak Speed
23400  GOSUB 18600        'Display Accessories
23450  GOSUB 15250        'Do and Display Shutter
23500  GOSUB 18150        'Display Markers
23550  F1# = MINFREQ# : F2# = MAXFREQ# : RT# = RAMPTIME#
23600  GOSUB 13650        'Set Motor Speed
23650 REM
23700 WHILE (1)
23750   GOSUB 26050          'GetKey
23800   IF (KEYVALUE = 0) THEN GOTO 24850
23850   IF (KEYVALUE = 19) THEN GOTO 21950   '(Ctrl-S) Change Mono Type
23900   IF (KEYVALUE = F1KV) THEN GOSUB 28350 : GOTO 24850
23950   IF (KEYVALUE = F2KV) THEN GOSUB 28800 : GOTO 24850
24000   IF (KEYVALUE = F3KV) THEN GOSUB 29200 : GOTO 24850
24050   IF (KEYVALUE = F4KV) THEN GOSUB 29600 : GOTO 24850
24100   IF (KEYVALUE = F5KV) THEN GOSUB 30150 : GOTO 24850
24150   IF (KEYVALUE = F6KV) THEN GOSUB 30650 : GOTO 24850
24200   IF (KEYVALUE = F7KV) THEN GOSUB 30950 : GOTO 24850
24250   IF (KEYVALUE = F8KV) THEN GOSUB 32550 : GOTO 24850
24300   IF (KEYVALUE = F9KV) THEN GOSUB 32850 : GOTO 24850
24350   IF (KEYVALUE = F10KV) THEN GOSUB 33150 : GOTO 24850
24400   IF (KEYVALUE = LEFTKV) THEN GOSUB 33400 : GOTO 24850
24450   IF (KEYVALUE = RIGHTKV) THEN GOSUB 34100 : GOTO 24850
24500   IF ( (AKEY$ = "T") OR (AKEY$ = "t") ) THEN GOSUB 34750 : GOTO 24850
24550   IF ( (AKEY$ = "E") OR (AKEY$ = "e") ) THEN GOSUB 35050 : GOTO 24850
```

```
24600   IF ( (AKEY$ = "X") OR (AKEY$ = "x") ) THEN GOSUB 35350 : GOTO 24850
24650   IF ( (AKEY$ = "A") OR (AKEY$ = "a") ) THEN GOSUB 35650 : GOTO 24850
24700   IF (KEYVALUE = CTRLA) THEN GOSUB 35950 : GOTO 24850
24701   IF ( (AKEY$ = "B") OR (AKEY$ = "b") ) THEN GOSUB 36310 : GOTO 24850
24702   IF (KEYVALUE = CTRLB) THEN GOSUB 36316 : GOTO 24850
24703   IF ( (AKEY$ = "C") OR (AKEY$ = "c") ) THEN GOSUB 36330 : GOTO 24850
24704   IF (KEYVALUE = CTRLC) THEN GOSUB 36336 : GOTO 24850
24750   LOCATE 23,1 : PRINT "Key Value: ",KEYVALUE : SOUND 1000,1
24800   IF (KEYVALUE = 26 ) THEN STOP
24850   REM ...
24852   if ( timer - tupdate < 0.2 ) then goto 24890
24853    tupdate = timer
24854     gosub 17100     'Update position and limits
24856     gosub 14800     'Test accessory busy
24858     if (ACCESSORYBUSY = 1) then LOCATE 23,20 :  Print "Accessory Busy"
24860     if (ACCESSORYBUSY = 0) then  LOCATE 23,1 : PRINT SPACE$(70)
24890   REM ...
24900 WEND
24950 STOP
25000 REM_____________________________________________________________________
25050 REM_____________________________________________________________________
25100 REM ***** Main Menu *****
25150  LOCATE MENUROW+0,MENUCOL: PRINT "<- Arrow -> to Tweak Mono"
25200  LOCATE MENUROW+1,MENUCOL: PRINT "F1  - Set"
25250  LOCATE MENUROW+2,MENUCOL: PRINT "F2  - Start Position"
25300  LOCATE MENUROW+3,MENUCOL: PRINT "F3  - End Position"
25350  LOCATE MENUROW+4,MENUCOL: PRINT "F4  - Scan Speed"
25400  LOCATE MENUROW+5,MENUCOL: PRINT "F5  - Calibrate"
25450  LOCATE MENUROW+6,MENUCOL: PRINT "F6  - Shutter"
25500  LOCATE MENUROW+7,MENUCOL: PRINT " T   - Turret"
25550  LOCATE MENUROW+8,MENUCOL: PRINT " E   - Entrance Mirror"
25600  LOCATE MENUROW+9,MENUCOL: PRINT " X   - Exit Mirror"
25650  LOCATE MENUROW+10,MENUCOL: PRINT "F7 - Scan"
25700  LOCATE MENUROW+11,MENUCOL: PRINT "F8 - HALT"
25750  LOCATE MENUROW+12,MENUCOL: PRINT "F9 - Tweak Speed"
25800  LOCATE MENUROW+13,MENUCOL: PRINT "F10 - EXIT"
25850  LOCATE MENUROW+15,1 : PRINT "Marker A"
25900  LOCATE MENUROW+16,1 : PRINT "Marker B"
25950  LOCATE MENUROW+17,1 : PRINT "Marker C"
26000  RETURN
26050 REM_____________________________________________________________________
26100 REM ***** GetKey *****
26150  IF ( HOTKEY = 0 ) THEN GOTO 26350
26200  KEYVALUE = HOTKEY : HOTKEY = 0
26250  RETURN
26300 REM
26350  AKEY$ = INKEY$
26400  L = LEN(AKEY$)+1
26450  ON L GOTO 26500,26550,26600
26500  KEYVALUE = 0 : GOTO 26650
26550  KEYVALUE = ASC(AKEY$) : GOTO 26650
26600  KEYVALUE = ASC(MID$(AKEY$, 2)) + 128 : GOTO 26650
26650  RETURN
26700 REM_____________________________________________________________________
26750 REM ***** Read Mono Parameter File *****
26800  OPEN "232_scan.ini" FOR INPUT AS #3
26850  NRECSKIP = 7 : LASTERROR = 0
```

```
26900  REM ...
26950  INPUT #3, MONOTYPE, COMMENT$: PRINT MONOTYPE, COMMENT$
27000  IF (LASTERROR <> 62) THEN 27200
27050    CLOSE #3
27100    PRINT "Parameters for Mono Type ",MONOWANTED," NOT found."
27150    GOTO 22000
27200  REM ...
27250  IF (MONOTYPE = MONOWANTED) THEN 27300 ELSE 27700
27300     INPUT #3, STEPSPERUNIT#, COMMENT$: PRINT STEPSPERUNIT#, COMMENT$
27350     INPUT #3, MINFREQ#, COMMENT$: PRINT MINFREQ#, COMMENT$
27400     INPUT #3, MAXFREQ#, COMMENT$: PRINT MAXFREQ#, COMMENT$
27450     INPUT #3, RAMPTIME#, COMMENT$: PRINT RAMPTIME#, COMMENT$
27500     INPUT #3, BACKLASH#, COMMENT$: PRINT BACKLASH#, COMMENT$
27550     INPUT #3, BASEGRATE#, COMMENT$: PRINT BASEGRATE#, COMMENT$
27600     INPUT #3, COMMENT$: PRINT COMMENT$
27650   GOTO 27950
27700      FOR I = 1 TO NRECSKIP
27750        INPUT #3, COMMENT$
27800      NEXT I
27850      PRINT COMMENT$
27900   GOTO 26900
27950 REM ...
28000  CLOSE #3
28050  RETURN
28100 REM________________________________________________________________________
28150 REM ***** Print Message *****
28200  LOCATE ROW,COL : PRINT SPACE$(50) : LOCATE ROW,COL
28250  PRINT MESSAGE$
28300  RETURN
28350 REM________________________________________________________________________
28400 REM *****  F1  *****
28450  MESSAGE$ = "F1" : GOSUB 28100
28500  LOCATE 22,20
28550  INPUT "Move to: ", W#
28600  TARGET# = W# * STEPSPERUNIT#
28650  LOCATE 22,1 : PRINT SPACE$(70)
28700  GOSUB 10150        'Move to target, display and wait till done
28750  RETURN
28800 REM________________________________________________________________________
28850 REM *****  F2  *****
28900  MESSAGE$ = "F2" : GOSUB 28100
28950  LOCATE 22,20
29000  INPUT "Start of Scan: ", STARTSCAN#
29050  LOCATE 22,1 : PRINT SPACE$(70)
29100  GOSUB 17400                     'DisplayScan
29150  RETURN
29200 REM________________________________________________________________________
29250 REM *****  F3  *****
29300  MESSAGE$ = "F3" : GOSUB 28100
29350  LOCATE 22,20
29400  INPUT "End of Scan: ", ENDSCAN#
29450  LOCATE 22,1 : PRINT SPACE$(70)
29500  GOSUB 17400                     'DisplayScan
29550  RETURN
29600 REM________________________________________________________________________
29650 REM *****  F4  *****
29700  MESSAGE$ = "F4" : GOSUB 28100
```

```
29750  LOCATE 22,20
29800  INPUT "Scan Speed: ", SCANSPEED#
29850  LOCATE 22,1 : PRINT SPACE$(70)
29900  SCANFREQ# = SCANSPEED# * STEPSPERUNIT#
29950  IF ( SCANFREQ# <= MAXFREQ# ) THEN GOTO 29960
29952  SCANFREQ# = MAXFREQ# : SCANSPEED# = SCANFREQ# / STEPSPERUNIT#
29954  GOSUB 17400 :  SOUND 500,1 : GOTO 29600
29956 REM
29960  IF ( SCANFREQ# >= MINHARDWAREFREQ# ) THEN GOTO 30000
29962  SCANFREQ# = MINHARDWAREFREQ# : SCANSPEED# = SCANFREQ# / STEPSPERUNIT#
29964  GOSUB 17400 : SOUND 500,1 : GOTO 29600
30000 REM ...
30050  GOSUB 17400                   'DisplayScan
30100  RETURN
30150 REM___________________________________________________________________
30200 REM *****  F5  *****
30250  MESSAGE$ = "F5" : GOSUB 28100
30300  LOCATE 22,20
30350  INPUT "Where are you: ", WAVE#
30400  LOCATE 22,1 : PRINT SPACE$(70)
30450 REM
30500  GOSUB 12900       'Set Mono Position
30550  GOSUB 17100       'Display Position
30600  RETURN
30650 REM___________________________________________________________________
30700 REM *****  F6  *****
30750  MESSAGE$ = "F6" : GOSUB 28100
30800  SHUTTER = 1 - SHUTTER
30850  GOSUB 15250
30900  RETURN
30950 REM___________________________________________________________________
31000 REM *****  F7  *****
31050  MESSAGE$ = "F7" : GOSUB 28100
31100 REM
31150  LOCATE 11, 40
31200  PRINT ":  Positioning"
31250  HALTSCAN = 0
31300  TARGET# = STARTSCAN# * STEPSPERUNIT#
31350  GOSUB 10150         'Move to target, display and wait till done
31400 REM
31450  IF ( HALTSCAN = 1 ) THEN GOTO 32150      'User halt
31500 REM
31550  DELAY = 1! : GOSUB 6950       'Delay before Start
31600  LOCATE 11, 40
31650  PRINT ":  STARTED     "
31700  TARGET# = ENDSCAN# * STEPSPERUNIT#
31750 REM Load Scan Speed
31800  F1# = MINFREQ# :  F2# = SCANFREQ# : RT# = RAMPTIME#
31850  IF ( SCANFREQ# < MINFREQ# ) THEN F1# = SCANFREQ#
31900  GOSUB 13650        'Set Motor Speed
31950 REM
32000  GOSUB 10150        'Move to target, display and wait till done
32050 REM
32100  GOSUB 32550        'Stop Scan and re-enable Hot Keys
32150 REM ...
32200  HALTSCAN = 0
32250  LOCATE 11, 40
```

```
32300  PRINT ":  STOPPED     "
32350 REM Load Normal Speed
32400  F1# = MINFREQ# : F2# = MAXFREQ# : RT# = RAMPTIME#
32450  GOSUB 13650        'Set Motor Speed
32500  RETURN
32550 REM
32600 REM *****  F8  *****
32650  MESSAGE$ = "F8" : GOSUB 28100
32700  HALTSCAN = 1
32750  WHILE ( INKEY$ <> "" ) : WEND  'Flush input buffer
32800  RETURN
32850 REM
32900 REM *****  F9  *****
32950  MESSAGE$ = "F9" : GOSUB 28100
33000  TWEEKSPEED = TWEEKMAX - TWEEKSPEED
33050  GOSUB 17850             'Display Tweek Speed
33100  RETURN
33150 REM
33200 REM *****  F10  *****
33250  MESSAGE$ = "F10" : GOSUB 28100
33300  GOSUB 36950       'Shut down
33350  RETURN
33400 REM
33450 REM *****  Left Arrow  *****
33500  MESSAGE$ = "Left" : GOSUB 28100
33550  IF ( TWEEKSPEED <> 1 ) THEN 33650
33600  IF ( (TIMER-T1) > .2 ) THEN 33650 ELSE 33700
33650  F = 1 : GOTO 33750
33700  F = F + 3 : IF ( F > 5*TWEEKMAX ) THEN F = 5*TWEEKMAX
33750  TARGET# = (STEPS# - TWEEKINC * TWEEKSPEED * F)
33800  IF (TARGET# < 0#) THEN TARGET# = 0# : SOUND 1000,1
33850 REM
33900  GOSUB 9700        'Short Move to target, display and wait till done
33950  T1 = TIMER
34050  RETURN
34100 REM
34150 REM *****  Right Arrow  *****
34200  MESSAGE$ = "Right" : GOSUB 28100
34250  IF ( TWEEKSPEED <> 1 ) THEN 34350
34300  IF ( (TIMER-T2) > .2 ) THEN 34350 ELSE 34400
34350  F = 1 : GOTO 34450
34400  F = F + 3 : IF ( F > 5*TWEEKMAX ) THEN F = 5*TWEEKMAX
34450  TARGET# = (STEPS# + TWEEKINC * TWEEKSPEED * F)
34500 REM
34550  GOSUB 9700        'Short Move to target, display and wait till done
34600  T2 = TIMER
34700  RETURN
34750 REM
34800 REM ***** "T" Turret  *****
34850  MESSAGE$ = "T" : GOSUB 28100
34900  TURRET = 1 - TURRET
34950  GOSUB 15550       'Do Turret and Display Accessories
35000  RETURN
35050 REM
35100 REM ***** "E" ENTmirror  *****
35150  MESSAGE$ = "E" : GOSUB 28100
35200  ENTMIRROR = 1 - ENTMIRROR
```

```
35250  GOSUB 15850       'Do Entrance Mirror and Display Accessories
35300  RETURN
35350 REM_____________________________________________________________
35400 REM ***** "X" EXTmirror  *****
35450  MESSAGE$ = "X" : GOSUB 28100
35500  EXTMIRROR = 1 - EXTMIRROR
35550  GOSUB 16150       'Do Exit Mirror and Display Accessories
35600  RETURN
35650 REM_____________________________________________________________
35700 REM ***** "A" go to marker A *****
35750  MESSAGE$ = "Go to A" : GOSUB 28100
35800  TARGET# = MARKERA# * STEPSPERUNIT#
35850  GOSUB 10150        'Move to target, display and wait till done
35900  RETURN
35950 REM_____________________________________________________________
36000 REM ***** "Ctrl-A" Set marker A *****
36050 MESSAGE$ = "Set A" : GOSUB 28100
36100  LOCATE 22,20
36150  INPUT "Marker A: ", MARKERA#
36200  LOCATE 22,1 : PRINT SPACE$(70)
36250  GOSUB 18150        'Display Markers
36300  RETURN
36310 REM_____________________________________________________________
36311 REM ***** "B" go to marker B *****
36312  MESSAGE$ = "Go to B" : GOSUB 28100
36313  TARGET# = MARKERB# * STEPSPERUNIT#
36314  GOSUB 10150        'Move to target, display and wait till done
36315  RETURN
36316 REM_____________________________________________________________
36317 REM ***** "Ctrl-B" Set marker B *****
36318 MESSAGE$ = "Set B" : GOSUB 28100
36319  LOCATE 22,20
36320  INPUT "Marker B: ", MARKERB#
36321  LOCATE 22,1 : PRINT SPACE$(70)
36322  GOSUB 18150        'Display Markers
36323  RETURN
36330 REM_____________________________________________________________
36331 REM ***** "C" go to marker C *****
36332  MESSAGE$ = "Go to C" : GOSUB 28100
36333  TARGET# = MARKERC# * STEPSPERUNIT#
36334  GOSUB 10150        'Move to target, display and wait till done
36335  RETURN
36336 REM_____________________________________________________________
36337 REM ***** "Ctrl-C" Set marker C *****
36338 MESSAGE$ = "Set C" : GOSUB 28100
36339  LOCATE 22,20
36340  INPUT "Marker C: ", MARKERC#
36341  LOCATE 22,1 : PRINT SPACE$(70)
36342  GOSUB 18150        'Display Markers
36343  RETURN
36350 REM_____________________________________________________________
36400 REM ***** Input User Parameters  *****
36450  OPEN "232_scan.usr" FOR INPUT AS #3
36500  INPUT #3, MONOWANTED, COMMENT$
36550  INPUT #3, STARTSCAN#, COMMENT$
36600  INPUT #3, ENDSCAN#, COMMENT$
36650  INPUT #3, SCANFREQ#, COMMENT$
```

```
36700  INPUT #3,MARKERA#, COMMENT$
36750  INPUT #3,MARKERB#, COMMENT$
36800  INPUT #3,MARKERC#, COMMENT$
36850  CLOSE #3
36900  RETURN
36950 REM_____________________________________________________________________
37000 REM ***** EXIT  *****
37050 REM Save User Parameters on exit
37100  OPEN "232_scan.usr" FOR OUTPUT AS #3
37150  PRINT #3, MONOTYPE,      " ;Mono Type"
37200  PRINT #3, STARTSCAN#,    " ;Start of Scan"
37250  PRINT #3, ENDSCAN#,      " ;End of Scan"
37300  PRINT #3, SCANFREQ#,     " ;Scan Freq"
37350  PRINT #3,MARKERA#,       " ;Marker A"
37400  PRINT #3,MARKERB#,       " ;Marker B"
37450  PRINT #3,MARKERC#,       " ;Marker C"
37500  CLOSE #3
37550 REM
37600  Close #2                    'Close RS-232 port
37650  SYSTEM                      'Exit BASIC
37700  END
```

Spectrometer Control

SCAN_232.BAS is a simple program intended to teach the usage of the various commands by example. It includes usage of DataScan / DataLink acquisition and high voltage commands in the intelligent communications mode. You will see how to set and read motor speeds, read the firmware version number from the FLASH RAM in the DataScan / DataLink, as well as other commands.

```
100 REM ------------------------------
200  REM GW-BASIC Example Program
300  REM
400  REM This is a sample program intended for reference only.
500 REM ------------------------------
600 REM
700 REM Test Program for Maple Wood Scan and Data Acq Commands
800 REM RS-232 version.
900 REM
1000 REM -------------------------
1100 REM ----------------------------
1200     ON ERROR GOTO 29000
1300 REM -----------------------
1400 REM Some Set Up Constants for your mono.
1500     MONOSTEPSPERUNIT# = 50#
1600 REM ---------------------------
1700    PRINT "Begin RS-232 Communications Setup"
1800 REM --------------------------
1900 REM     Open Com Port 2
2000 REM 4800 baud, no parity, 8 bits, 1 stop, suppress Request to Send
2100 REM
2200    OPEN "com2:4800,N,8,1,RS,DS0" FOR RANDOM AS #2
2300 REM ..
2400 REM     Flush anything in input buffer
2500    DELAY = .5: GOSUB 28200        'Wait a little'
2600    WHILE (LOC(2) > 0)
2700      JUNK$ = INPUT$(1, #2): SOUND 2000, 5
2800    WEND
2900 REM
3000 REM  If we have already auto bauded these commands will put us in
3100 REM   a known state without restarting the instrument.
3200    PRINT #2, CHR$(248);             'Send Util Set Intelligent Mode Pseudo
Command'
3300    PRINT #2, CHR$(222);             'Send Software Reboot if Hung'
3400    DELAY = .5: GOSUB 28200          'Wait a little'
3500    WHILE (LOC(2) > 0)               'Flush input'
3600      JUNK$ = INPUT$(1, #2): SOUND 2000, 5
3700    WEND
3800 REM
3900 REM     Begin Auto Baud Sequence
4000 REM ..                              'Send <SP>'
4100    PRINT #2, " "; : SOUND 400, 2
4200    DELAY = .5: GOSUB 28200       'Wait a little'
4300    IF (LOC(2) > 0) THEN 4600 ELSE 4400
4400 REM ..
4500    GOTO 4000                        'No Response. Try Again'
4600    ICH$ = INPUT$(1, #2): SOUND 1000, 2
4700    PRINT "Received ", ICH$
4800    IF (ICH$ = "*") GOTO 5400        'Autobaud ACK O.K.'
4900 REM
5000 REM  If already auto bauded, take appropriate action.
5100    IF (ICH$ = "B") GOTO 6400        'Init Program ACK O.K.'
```

```
5200     IF (ICH$ = "F") GOTO 7100        'Main Program ACK O.K.'
5300     GOTO 2300                        'Bad Response. Try Again'
5400 REM ..
5500     DELAY = .5: GOSUB 28200          'Wait a little'
5600     INPUT #2, BOOTMSG$               'Flush out Boot Message'
5700     PRINT BOOTMSG$
5800 REM
5900      PRINT #2, CHR$(247);              'Send Util Startup Intelligent Mode
command'
6000 REM
6100     ICH$ = INPUT$(1, #2)             'ACK of Util Startup Intelligent Mode'
6200     IF (ICH$ <> "=") GOTO 2300       'Bad ACK. Try Again'
6300 REM
6400 REM ..     Jump to Main Program
6500     PRINT "Jump to Main Program"
6600     PRINT #2, "O";                   'Send Jump Sequence'
6700     PRINT #2, "2000"; : PRINT #2, CHR$(0);
6800 REM
6900     WHILE (INPUT$(1, #2) <> "*")     'Wait for Main Program Response'
7000     WEND
7100 REM ..
7200     SOUND 500, 5: SOUND 1000, 5: SOUND 500, 5
7300     PRINT "RS-232 Communications Established"
7400     PRINT "Begin RS-232 Command Demo"
7500     SAYOK = 1         'ACK sound and OK print out flag'
7600     GOSUB 29700       'Put up the main menu'
7700 REM
7800 REM
7900 REM ----------------------------
8000 REM ..     Input Command
8100     PRINT
8200 REM ..
8300     INPUT "Command No:(99 for menu)"; CMDNO
8400     IF (CMDNO = 99) THEN GOSUB 29700 ELSE 8600
8500     GOTO 9200
8600 REM ..
8700     IF (CMDNO = 998) GOTO 32000      'Stop'
8800     IF (CMDNO = 999) GOTO 32000      'Exit'
8900     IF (CMDNO >= MINCMD) AND (CMDNO <= MAXCMD) GOTO 9400
9000     SOUND 700, 5
9100     PRINT "Command Range is"; MINCMD; " to "; MAXCMD
9200 REM
9300     GOTO 8200
9400 REM .. Got a Valid Command
9500     PRINT "Command selected is:"; CMDNO
9600 REM ____________________________________________
9700     IF (CMDNO = 1) THEN 9800 ELSE 10700
9800 REM ..
9900   INPUT "Move Relative, How much ?", WMONOPOS#
10000   MONOPOS# = WMONOPOS# * MONOSTEPSPERUNIT#
10100   PRINT #2, "F";
10200   PRINT #2, "0,";
10300   PRINT #2, MONOPOS#
10400   GOSUB 26700               'Wait For Confirmation'
10500    GOTO 8000
10600 REM
10700 REM ____________________________________________
```

```
10800    IF (CMDNO = 2) THEN 10900 ELSE 11800
10900 REM ..
11000    PRINT #2, "C";                   'NO comma or <CR> after command'
11100    PRINT #2, "0"                    'Parameter followed by <CR>'
11200    GOSUB 26700     'Wait For Confirmation'
11300 REM
11400    INPUT #2, FMIN#, FMAX#, RAMPT%: SOUND 1000, 2
11500    PRINT "fmin, fmax, ramp_time": PRINT FMIN#, FMAX#, RAMPT%
11600    GOTO 8000
11700 REM
11800 REM _______________________________________________________
11900   IF (CMDNO = 3) THEN 12000 ELSE 12600
12000 REM ..
12100    PRINT #2, "A";                   'NO comma or <CR> after command'
12200    GOSUB 26700     'Wait For Confirmation'
12300 REM
12400   GOTO 8000
12500 REM
12600 REM _______________________________________________________
12700   IF (CMDNO = 4) THEN 12800 ELSE 14100
12800 REM ..
12900   PRINT #2, "y";                    'NO comma or <CR> after command'
13000   GOSUB 26700     'Wait For Confirmation'
13100 REM
13200   INPUT #2, VERNO$
13300   PRINT "Boot Version:", VERNO$
13400 REM
13500   PRINT #2, "z";
13600   GOSUB 26700     'Wait For Confirmation'
13700   INPUT #2, VERNO$
13800   PRINT "Flash Version:", VERNO$
13900   GOTO 8000
14000 REM
14100 REM _______________________________________________________
14200   IF (CMDNO = 5) THEN 14300 ELSE 15600
14300 REM ..
14400   INPUT "fmin, fmax, ramp_time"; FREQMIN$, FREQMAX$, RAMPTIME$
14500   PRINT #2, "B";                    'NO comma or <CR> after command'
14600   PRINT #2, "0,";                   'Build Parameter String'
14700   PRINT #2, FREQMIN$;               ' separated by commas and NO <CR>'
14800   PRINT #2, ",";
14900   PRINT #2, FREQMAX$;
15000   PRINT #2, ",";
15100   PRINT #2, RAMPTIME$               'Last parameter followed by <CR>'
15200   GOSUB 26700     'Wait For Confirmation'
15300 REM
15400   GOTO 8000
15500 REM
15600 REM _______________________________________________________
15700   IF (CMDNO = 6) THEN 15800 ELSE 16700
15800 REM ..
15900   PRINT #2, "p";
16000   PRINT #2, "0,"; 'Scan Type'
16100   PRINT #2, "10000,25000,250,500,3,1500,5000,0,0,0,0,0,0,4,0,0,0"
16200   GOSUB 26700     'Wait For Confirmation'
16300   MAXDATA# = (25000# - 10000#) / 250# + 1#
16400   PRINT "This Scan will produce", MAXDATA#, " points"
```

```
16500   GOTO 8000
16600 REM
16700 REM ____________________________________________________
16800   IF (CMDNO = 7) THEN 16900 ELSE 17500
16900 REM ..
17000   PRINT "Start of Scan"
17100   PRINT #2, "q";
17200   GOSUB 26700      'Wait For Confirmation'
17300   GOTO 8000
17400 REM
17500 REM ____________________________________________________
17600   IF (CMDNO = 8) THEN 17700 ELSE 20900
17700 REM ..
17800   PRINT "Start Data Loging"
17900   OPEN "O", #3, "SCANDATA.PRN"
18000   PTNUMBER# = 0#
18100   SAYOK = 0                  'Turn of ACK OK during tight loop'
18200 REM ..
18300   PRINT #2, "t";             'Get point number of last data taken'
18400   GOSUB 26700                'Wait For Confirmation'
18500   INPUT #2, INDEX#
18600   IF (INDEX# > PTNUMBER#) THEN 18700 ELSE 18200
18700   REM ..
18800   PRINT "index:", INDEX#
18900   WHILE (INDEX# > PTNUMBER#)
19000     PTNUMBER# = PTNUMBER# + 1#
19100     PRINT #2, "u";
19200 REM     DELAY = .05: GOSUB 20900      'Wait a little'
19300 REM     PRINT #2, PTNUMBER#;
19400 REM     DELAY = .05: GOSUB 20900      'Wait a little'
19500 REM     PRINT #2, ""
19600     PRINT #2, PTNUMBER#
19700     GOSUB 26700     'Wait For Confirmation'
19800     INPUT #2, DDATA#, RANGE%
19900     PRINT #3, PTNUMBER#, DDATA#, RANGE%
20000 REM      PRINT "pt#  data  range", PTNUMBER#, DDATA#, RANGE%
20100   WEND
20200   IF (PTNUMBER# = MAXDATA#) THEN 20300 ELSE 18200
20300   REM ..
20400   CLOSE #3
20500   PRINT "End Data Loging"
20600   SAYOK = 1
20700   GOTO 8000
20800 REM
20900 REM ____________________________________________________
21000   IF (CMDNO = 9) THEN 21100 ELSE 22000
21100 REM ..
21200   INPUT "Where are you ?", WMONOPOS#
21300   MONOPOS# = WMONOPOS# * MONOSTEPSPERUNIT#
21400   PRINT #2, "G";
21500   PRINT #2, "0,";
21600   PRINT #2, MONOPOS#
21700   GOSUB 26700                'Wait For Confirmation'
21800   GOTO 8000
21900 REM
22000 REM ____________________________________________________
22100   IF (CMDNO = 10) THEN 22200 ELSE 23200
```

```
22200 REM ..
22300   PRINT #2, "H";
22400   PRINT #2, "0"
22500   GOSUB 26700             'Wait For Confirmation'
22600   REM
22700   INPUT #2, MONOPOS#
22800    PRINT  "Mono  Position  (steps,  wave  length)=",  MONOPOS#,  MONOPOS#  /
MONOSTEPSPERUNIT#
22900   GOTO 8000
23000 REM
23100 REM _______________________________________________________
23200   IF (CMDNO = 11) THEN 23300 ELSE 24000
23300 REM ..
23400   PRINT #2, "r";
23500   GOSUB 26700     'Wait For Confirmation'
23600   STATUS$ = INPUT$(1, #2)
23700   PRINT "Hardware Status:", ASC(STATUS$)
23800   GOTO 8000
23900 REM
24000 REM _______________________________________________________
24100   IF (CMDNO = 12) THEN 24200 ELSE 24900
24200   PRINT "Change to Low IQ Mode"
24300   PRINT #2, "Y";
24400   GOSUB 26700     'Wait For Confirmation'
24500   PRINT : PRINT "Change Switch to Hand Held Position"
24600   PRINT "  then Hit <.> on Hand Held twice"
24700   GOTO 8000
24800 REM _______________________________________________________
24900   IF (CMDNO = 13) THEN 25000 ELSE 26100
25000   PRINT "Going to Intelligent Communications Mode"
25100   PRINT "Change Switch to Computer Position"
25200   INPUT " and hit any key"; JUNK$
25300   PRINT #2, CHR$(248);    'Util Set Intelligent Mode Pseudo Command'
25400 REM NO CONFIRMATION FOR pseudo Command
25500 REM
25600 REM     Flush anything in input buffer
25700    WHILE (LOC(2) > 0)
25800      JUNK$ = INPUT$(1, #2): SOUND 2000, 1: PRINT JUNK$;
25900    WEND
26000   GOTO 8000
26100 REM _______________________________________________________
26200   PRINT "***Command Not Implimented***"
26300 REM _______________________________________________________
26400   GOTO 8000
26500 REM
26600 REM -------------------------------
26700 REM .....Input ACK character and test for valid
26800 REM ----------------------------------
26900   ACK$ = INPUT$(1, #2): IF (SAYOK = 1) THEN SOUND 1000, 2
27000   IF (ACK$ = "o") THEN 27100 ELSE 27400
27100   REM ..
27200   IF (SAYOK = 1) THEN PRINT "Receive was O.K."
27300   GOTO 27800
27400 REM ..
27500   IF (ACK$ = "b") THEN PRINT "Receive was bad"
27600   PRINT "Character Received was: "; ASC(ACK$), ACK$
27700   STOP
```

```
27800 REM ..
27900   RETURN
28000 REM
28100 REM ----------------------------------
28200 REM .....Time Delay of delay
28300 REM -----------------------------
28400   T0 = TIMER
28500   WHILE ((TIMER - T0) < DELAY)
28600   WEND
28700   RETURN
28800 REM
28900 REM -----------------------------
29000 REM .....Error Handler
29100 REM ---------------------------------
29200   PRINT "***** Error =", ERR, " on line number", ERL
29300   RESUME
29400   RETURN
29500 REM
29600 REM ---------------------------------
29700 REM .....Main Menu
29800 REM ---------------------------------
29900   MINCMD = 1: MAXCMD = 13
30000   PRINT : PRINT "Command:"
30100   PRINT "Move Mono Relative        1"
30200   PRINT "Read Motor Speed          2"
30300   PRINT "Motor Init                3"
30400   PRINT "Flash Version #           4"
30500   PRINT "Set Motor Speed           5"
30600   PRINT "Set Scan Parameters       6"
30700   PRINT "Start Scan                7"
30800   PRINT "Log Data                  8"
30900   PRINT "Set Mono Position         9"
31000   PRINT "Read Mono Position       10"
31100   PRINT "Hardware Status          11"
31200   PRINT "Go to Low IQ             12"
31300   PRINT "Go to Hi  IQ             13"
31400   PRINT "----------------------------"
31500   PRINT "Command Menu             99"
31600   PRINT "Stop                    998"
31700   PRINT "Exit                    999"
31800   RETURN
31900 REM --------------------------------
32000 REM .....EXIT
32100 REM --------------------------------
32200   PRINT "End RS-232 Command Demo"
32300   CLOSE #2
32400   IF (CMDNO = 998) THEN STOP ELSE SYSTEM
32500   END
```

Appendix 4: Sample IEEE 488 Programs

The support Diskette provided with your system contains sample programs written in GWBASIC. Please understand that these example programs are provided free of charge and they have been tested within reason. We do not guarantee or support these programs. They are intended as examples only.

If you construct programs based on these Command Set examples, we strongly recommend that you add prudent error trapping and protection features to your program to protect your system and enhance ease-of-use. Much aggravation can be avoided by such things as checking limits on stepper motor travel and speed, as well as other variables.

SCAN_488.BAS is a simple program intended to teach the usage of the various commands by example. It includes usage of DataScan / DataLink acquisition and high voltage commands in the intelligent communications mode. You will see how to set and read motor speeds, read the firmware version number from the FLASH RAM in the DataScan / DataLink, as well as other commands.

```
100 REM --------------------------------------
200  REM GW-BASIC Example Program
300  REM
400  REM This is a sample program intended for reference only. It assumes
500  REM  you have a National Instruments IEEE-488 card and driver loaded.
600  REM The driver device addresses [1..31] are referenced as
700  REM  "DEV1".."DEV31". see the National Instruments Software Ref. Manual
800  REM  for further details.
900  REM
1000  REM Factory default IEEE-488 address for this instrument is 1.
1100  REM
1200  REM You MUST merge this code with DECL.BAS and BIB.M MUST be in this
1300  REM  directory. These files are supplied by National Instruments.
1400 REM ------------------------------
1500 REM
1600 REM Test Program for Maple Wood Scan and Data Acq Commands
1700 REM IEEE-488 version
1800 REM
1900 REM -----------
2000 REM --------------------------------
2100     ON ERROR GOTO 35400
2200 REM ---------------------------------
2300 REM Some Set Up Constants
2400     MONOSTEPSPERUNIT# = 50#
2500 REM ----------------------------
2600    PRINT "Begin IEEE-488 Communications Setup"
2700 REM --------------------------------------
2800 REM     OPEN
2900 REM
3000    BOARD$ = "GPIB0"
3100    DEVICE$ = "DEV"
3200    RBUF$ = SPACE$(132)  : PBUF$ = SPACE$(132)
3300    OBUF$ = SPACE$(132)
3400    JUNK$ = SPACE$(132)
3500    ICH$ = SPACE$(1)
3600    ACK$ = SPACE$(1)
3700    CR$ = CHR$(13)
3800 REM ..
3900    INPUT "Device Address [1..31]: ",DA$
```

```
4000     DA = VAL(DA$)
4100     IF (DA > 0) AND (DA < 32) GOTO 4300
4200     SOUND 350,3 : GOTO 3800
4300 REM ..
4400     DEVICE$ = DEVICE$ + DA$
4500 REM
4600     CALL IBFIND(BOARD$, BD%)
4700     MSG$ = "IBFIND ERROR on " + BOARD$
4800     IF (BD% < 0) THEN GOSUB 39200 : STOP
4900 REM ..
5000     CALL IBFIND(DEVICE$, DV%)
5100     MSG$ = "IBFIND ERROR on " + DEVICE$
5200     IF (DV% < 0) THEN GOSUB 39200 : STOP
5300     EOSV% = &HD
5400     V% = EOSV% + &H1400
5500     CALL IBEOS ( DV%, V% )
5600     CALL IBTMO( DV%, T300MS% )
5700 REM ..
5800 REM      Flush anything in input buffer
5900     CALL IBRD(DV%,JUNK$) : SOUND 2000,1 : PRINT JUNK$
6000     REM      May time out
6100 REM
6200 REM  If we have already auto bauded these commands will put us in
6300 REM   a known state without restarting the instrument.
6400 REM
6500     ICH$ = CHR$(222) : CALL IBWRT(DV%,ICH$)
6600     DELAY = .5 : GOSUB 34600       'Wait for possible reset
6700 REM      Flush anything in input buffer
6800     CALL IBRD(DV%,JUNK$) : SOUND 2000,1 : PRINT JUNK$
6900     REM      May time out
7000 REM
7100 REM ..                                 'Send <SP>'
7200     ICH$ = " ": CALL IBWRT(DV%, ICH$): SOUND 400, 2
7300     MSG$ = "IBWRT ERROR"
7400     IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
7500 REM
7600     CALL IBRD(DV%, ICH$): SOUND 1000, 2
7700     MSG$ = "IBRD ERROR"
7800     IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
7900 REM
8000     PRINT "Received ", ICH$
8100 REM
8200     IF (ICH$ = "B") GOTO 8700          'Init Program ACK O.K.'
8300     IF (ICH$ = "F") GOTO 9900          'Main Program ACK O.K.'
8400     GOTO 5700                          'Bad Response. Try Again'
8500 REM ..
8600 REM
8700 REM ..    Jump to Main Program
8800     PRINT "Jump to Main Program"
8900     OBUF$ = "O2000" + CHR$(0)
9000     CALL IBWRT(DV%, OBUF$) : MSG$ = "IBWRT ERROR"
9100     IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
9200 REM
9300 REM ..
9400     DELAY = .5 : GOSUB 34600        'Wait for main to start up
9500     CALL IBRD(DV%, ICH$) : MSG$ = "IBRD ERROR"
9600     IF (IBSTA% AND EERR) THEN GOSUB 39200 : RESUME
```

```
9700 REM May time out
9800    IF (ICH$ <> "*") GOTO 9300    'Wait for Main Program Response'
9900 REM ..
10000    SOUND 500, 5: SOUND 1000, 5: SOUND 500, 5
10100    PRINT "IEEE-488 Communications Established"
10200    PRINT "Begin IEEE-488 Command Demo"
10300    SAYOK = 1         'ACK sound and OK print out flag'
10400    GOSUB 36100       'Put up the main menu'
10500 REM
10600 REM
10700 REM ---------------------
10800 REM ..      Input Command
10900    PRINT
11000 REM ..
11100    INPUT "Command No:(99 for menu)"; CMDNO
11200    IF (CMDNO = 99) THEN GOSUB 36100 ELSE 11400
11300    GOTO 12000
11400 REM ..
11500    IF (CMDNO = 998) GOTO 38500      'Stop'
11600    IF (CMDNO = 999) GOTO 38500      'Exit'
11700    IF (CMDNO >= MINCMD) AND (CMDNO <= MAXCMD) GOTO 12200
11800    SOUND 700, 5
11900    PRINT "Command Range is"; MINCMD; " to "; MAXCMD
12000 REM
12100    GOTO 11000
12200 REM .. Got a Valid Command
12300    PRINT "Command selected is:"; CMDNO
12400 REM ______________________________________________________
12500    IF (CMDNO = 1) THEN 12600 ELSE 13600
12600 REM ..
12700   INPUT "Move Relative, How much ?", WMONOPOS#
12800   MONOPOS# = WMONOPOS# * MONOSTEPSPERUNIT#
12900   OBUF$ = "F" + "0" + "," + STR$(MONOPOS#) + CR$
13000   CALL IBWRT(DV%, OBUF$) : MSG$ = "IBWRT ERROR"
13100   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
13200 REM
13300   GOSUB 32900              'Wait For Confirmation'
13400    GOTO 10800
13500 REM
13600 REM ______________________________________________________
13700    IF (CMDNO = 2) THEN 13800 ELSE 14900
13800 REM ..
13900    OBUF$ = "C" + "0" + CR$
14000    CALL IBWRT( DV%, OBUF$ ) : MSG$ = "IBWRT ERROR"
14100    IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
14200    GOSUB 32900      'Wait For Confirmation'
14300 REM
14400    CALL IBRD( DV%, RBUF$) : SOUND 1000, 2
14500    PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
14600    PRINT "fmin, fmax, ramp_time": PRINT PBUF$
14700    GOTO 10800
14800 REM
14900 REM ______________________________________________________
15000   IF (CMDNO = 3) THEN 15100 ELSE 15900
15100 REM ..
15200    ICH$ = "A" : CALL IBWRT(DV%,ICH$)    'NO comma or <CR> after command'
15300    MSG$ = "IBWRT ERROR"
```

```
15400    IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
15500    GOSUB 32900     'Wait For Confirmation'
15600 REM
15700    GOTO 10800
15800 REM
15900 REM ____________________________________________________________
16000    IF (CMDNO = 4) THEN 16100 ELSE 18400
16100 REM ..
16200    ICH$ = "y" : CALL IBWRT(DV%, ICH$)    'NO comma or <CR> after command'
16300    MSG$ = "IBWRT ERROR"
16400    IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
16500    GOSUB 32900     'Wait For Confirmation'
16600 REM
16700    CALL IBRD(DV%, RBUF$) : MSG$ = "IBRD ERROR"
16800    IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
16900 REM
17000    PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
17100    PRINT "Boot Version:", PBUF$
17200 REM
17300    ICH$ = "z" : CALL IBWRT(DV%, ICH$) : MSG$ = "IBWRT ERROR"
17400    IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
17500 REM
17600    GOSUB 32900     'Wait For Confirmation'
17700    CALL IBRD(DV%, RBUF$) : MSG$ = "IBRD ERROR"
17800    IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
17900 REM
18000    PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
18100    PRINT "Flash Version:", PBUF$
18200    GOTO 10800
18300 REM
18400 REM ____________________________________________________________
18500    IF (CMDNO = 5) THEN 18600 ELSE 19600
18600 REM ..
18700    INPUT "fmin, fmax, ramp_time"; FREQMIN$, FREQMAX$, RAMPTIME$
18800    OBUF$ = "B"+"0"+"," + FREQMIN$ + "," + FREQMAX$ + "," + RAMPTIME$+CR$
18900    CALL IBWRT(DV%, OBUF$) : MSG$ = "IBWRT ERROR"
19000    IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
19100 REM
19200    GOSUB 32900     'Wait For Confirmation'
19300 REM
19400    GOTO 10800
19500 REM
19600 REM ____________________________________________________________
19700    IF (CMDNO = 6) THEN 19800 ELSE 21000
19800 REM ..
19900    OBUF$ = "p0,"
20000    OBUF$ = OBUF$ + "10000,25000,250,500,3,1500,5000,0,0,0,0,0,0,4,0,0,0"
20100    OBUF$ = OBUF$ + CR$
20200    CALL IBWRT(DV%, OBUF$) : MSG$ = "IBWRT ERROR"
20300    IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
20400 REM
20500    GOSUB 32900     'Wait For Confirmation'
20600    MAXDATA# = (25000# - 10000#) / 250# + 1#
20700    PRINT "This Scan will produce", MAXDATA#, " points"
20800    GOTO 10800
20900 REM
21000 REM ____________________________________________________________
```

```
21100   IF (CMDNO = 7) THEN 21200 ELSE 22100
21200 REM ..
21300   PRINT "Start of Scan"
21400   ICH$ = "q" : CALL IBWRT(DV%, ICH$)   'NO comma or <CR> after command'
21500   MSG$ = "IBWRT ERROR"
21600   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
21700 REM
21800   GOSUB 32900      'Wait For Confirmation'
21900   GOTO 10800
22000 REM
22100 REM _______________________________________________________
22200   IF (CMDNO = 8) THEN 22300 ELSE 26300
22300 REM ..
22400   PRINT "Start Data Loging"
22500   OPEN "O", #3, "SCANDATA.PRN"
22600   PTNUMBER# = 0#
22700   SAYOK = 0                'Turn of ACK OK during tight loop'
22800 REM ..
22900   ICH$ = "t" : CALL IBWRT(DV%, ICH$)   'NO comma or <CR> after command'
23000   MSG$ = "IBWRT ERROR"
23100   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
23200 REM
23300   GOSUB 32900              'Wait For Confirmation'
23400   CALL IBRD(DV%,RBUF$) : MSG$ = "IBRD ERROR"
23500   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
23600 REM
23700   PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
23800   INDEX# = VAL(PBUF$)
23900   IF (INDEX# > PTNUMBER#) THEN 24000 ELSE 22800
24000   REM ..
24100   PRINT "index:", INDEX#
24200   WHILE (INDEX# > PTNUMBER#)
24300     PTNUMBER# = PTNUMBER# + 1#
24400     OBUF$ = "u" + STR$(PTNUMBER#) + CR$
24500     CALL IBWRT(DV%, OBUF$) : MSG$ = "IBWRT ERROR"
24600     IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
24700     REM
24800     GOSUB 32900     'Wait For Confirmation'
24900     CALL IBRD(DV%,RBUF$) : MSG$ = "IBRD ERROR"
25000     IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
25100     REM
25200     PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
25300     PRINT #3, PTNUMBER#, PBUF$
25400 REM      PRINT "pt#  data  range", PTNUMBER#, pbuf$
25500   WEND
25600   IF (PTNUMBER# = MAXDATA#) THEN 25700 ELSE 22800
25700   REM ..
25800   CLOSE #3
25900   PRINT "End Data Loging"
26000   SAYOK = 1
26100   GOTO 10800
26200 REM
26300 REM _______________________________________________________
26400   IF (CMDNO = 9) THEN 26500 ELSE 27400
26500 REM ..
26600   INPUT "Where are you ?", WMONOPOS#
26700   MONOPOS# = WMONOPOS# * MONOSTEPSPERUNIT#
```

```
26800   OBUF$ = "G0," + STR$(MONOPOS#) + CR$
26900   CALL IBWRT(DV%,OBUF$) : MSG$ = "IBWRT ERROR"
27000   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
27100   GOSUB 32900              'Wait For Confirmation'
27200   GOTO 10800
27300 REM
27400 REM _______________________________________________________
27500   IF (CMDNO = 10) THEN 27600 ELSE 29000
27600 REM ..
27700   OBUF$ = "H0" +CR$
27800   CALL IBWRT(DV%,OBUF$) : MSG$ = "IBWRT ERROR"
27900   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
28000   GOSUB 32900              'Wait For Confirmation'
28100   REM
28200   CALL IBRD(DV%,RBUF$) : MSG$ = "IBRD ERROR"
28300   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
28400   PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
28500   MONOPOS# = VAL(PBUF$)
28600    PRINT "Mono Position (steps, wave length)=", MONOPOS#, MONOPOS# /
MONOSTEPSPERUNIT#
28700   GOTO 10800
28800 REM
28900 REM _______________________________________________________
29000   IF (CMDNO = 11) THEN 29100 ELSE 30200
29100 REM ..
29200   ICH$ = "r" : CALL IBWRT(DV%, ICH$)   'NO comma or <CR> after command'
29300   MSG$ = "IBWRT ERROR"
29400   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
29500   GOSUB 32900      'Wait For Confirmation'
29600   CALL IBRD(DV%,RBUF$) : MSG$ = "IBRD ERROR"
29700   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
29800   PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
29900   PRINT "Hardware Status: ", PBUF$
30000   GOTO 10800
30100 REM
30200 REM _______________________________________________________
30300   IF (CMDNO = 12) THEN 30400 ELSE 31500
30400   PRINT "Change to Low IQ Mode"
30500   ICH$ = "Y" : CALL IBWRT(DV%, ICH$)   'NO comma or <CR> after command'
30600   MSG$ = "IBWRT ERROR"
30700   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
30800   GOSUB 32900      'Wait For Confirmation'
30900 REM
31000   CALL IBLOC(DV%)
31100   PRINT : PRINT "Change Switch to Hand Held Position"
31200   PRINT "  then Hit <.> on Hand Held twice"
31300   GOTO 10800
31400 REM _______________________________________________________
31500   IF (CMDNO = 13) THEN 31600 ELSE 32300
31600   PRINT "IEEE-488 coming On-line"
31700   PRINT "RS-232 I/O should be SILENT"
31800   INPUT " and hit any key"; XXX$
31900   GOTO 4900
32000 REM
32100 REM     Flush anything in input buffer
32200   GOTO 10800
32300 REM _______________________________________________________
```

```
32400   PRINT "***Command Not Implimented***"
32500 REM _______________________________________________________
32600   GOTO 10800
32700 REM
32800 REM --------------------------------
32900 REM .....Input ACK character and test for valid
33000 REM --------------------------------
33100   CALL IBRD(DV%, ACK$) : IF (SAYOK = 1) THEN SOUND 1000, 2
33200   MSG$ = "IBRD ERROR"
33300   IF (IBSTA% AND EERR) THEN GOSUB 39200 : STOP
33400   IF (ACK$ = "o") THEN 33500 ELSE 33800
33500   REM ..
33600   IF (SAYOK = 1) THEN PRINT "Receive was O.K."
33700   GOTO 34200
33800 REM ..
33900   IF (ACK$ = "b") THEN PRINT "Receive was bad"
34000   PRINT "Character Received was: "; ASC(ACK$), ACK$
34100   STOP
34200 REM ..
34300   RETURN
34400 REM
34500 REM ------------------------------------
34600 REM .....Time Delay of delay
34700 REM ---------------------------------
34800   T0 = TIMER
34900   WHILE ((TIMER - T0) < DELAY)
35000   WEND
35100   RETURN
35200 REM
35300 REM -------------------------------------
35400 REM .....Error Handler
35500 REM ------------------------------
35600   PRINT "***** Error =", ERR, " on line number", ERL
35700   RESUME
35800   RETURN
35900 REM
36000 REM ---------------------------
36100 REM .....Main Menu
36200 REM -----------------------------
36300   MINCMD = 1: MAXCMD = 13
36400   PRINT : PRINT "Command:"
36500   PRINT "Move Mono Relative        1"
36600   PRINT "Read Motor Speed          2"
36700   PRINT "Motor Init                3"
36800   PRINT "Flash Version #           4"
36900   PRINT "Set Motor Speed           5"
37000   PRINT "Set Scan Parameters       6"
37100   PRINT "Start Scan                7"
37200   PRINT "Log Data                  8"
37300   PRINT "Set Mono Position         9"
37400   PRINT "Read Mono Position       10"
37500   PRINT "Hardware Status          11"
37600   PRINT "Go to RS-232             12"
37700   PRINT " (Startup must have been RS-232)"
37800   PRINT "Come back to IEEE-488    13"
37900   PRINT "---------------------------"
38000   PRINT "Command Menu             99"
```

```
38100   PRINT "Stop                    998"
38200   PRINT "Exit                    999"
38300   RETURN
38400 REM --------------------------------------
38500 REM .....EXIT
38600 REM ----------------------------
38700   PRINT "End IEEE-488 Command Demo"
38800   CALL IBLOC(DV%)
38900   IF (CMDNO = 998) THEN STOP ELSE SYSTEM
39000 REM --------------------------
39100 REM -----------------------------------
39200 REM .....IEEE-488 ERROR
39300 REM -----------------------------
39400 REM This routine will notify you that an IB call failed and print
39500 REM the status variables.
39600 REM -----------------------------------
39700 REM
39800     PRINT MSG$
39900 REM
40000     PRINT "ibsta= &H"; HEX$(IBSTA%); " <";
40100     IF IBSTA% AND EERR  THEN PRINT " ERR";
40200     IF IBSTA% AND TIMO THEN PRINT " TIMO";
40300     IF IBSTA% AND EEND THEN PRINT " END";
40400     IF IBSTA% AND SRQI THEN PRINT " SRQI";
40500     IF IBSTA% AND RQS  THEN PRINT " RQS";
40600     IF IBSTA% AND CMPL THEN PRINT " CMPL";
40700     IF IBSTA% AND LOK  THEN PRINT " LOK";
40800     IF IBSTA% AND RREM THEN PRINT " REM";
40900     IF IBSTA% AND CIC  THEN PRINT " CIC";
41000     IF IBSTA% AND AATN THEN PRINT " ATN";
41100     IF IBSTA% AND TACS THEN PRINT " TACS";
41200     IF IBSTA% AND LACS THEN PRINT " LACS";
41300     IF IBSTA% AND DTAS THEN PRINT " DTAS";
41400     IF IBSTA% AND DCAS THEN PRINT " DCAS";
41500     PRINT ">"
41600 REM
41700     PRINT "iberr= "; IBERR%;
41800     IF IBERR% = EDVR THEN PRINT " EDVR <DOS Error>"
41900     IF IBERR% = ECIC THEN PRINT " ECIC <Not CIC>"
42000     IF IBERR% = ENOL THEN PRINT " ENOL <No Listener>"
42100     IF IBERR% = EADR THEN PRINT " EADR <Address error>"
42200     IF IBERR% = EARG THEN PRINT " EARG <Invalid argument>"
42300     IF IBERR% = ESAC THEN PRINT " ESAC <Not Sys Ctrlr>"
42400     IF IBERR% = EABO THEN PRINT " EABO <Op. aborted>"
42500     IF IBERR% = ENEB THEN PRINT " ENEB <No GPIB board>"
42600     IF IBERR% = EOIP THEN PRINT " EOIP <Async I/O in prg>"
42700     IF IBERR% = ECAP THEN PRINT " ECAP <No capability>"
42800     IF IBERR% = EFSO THEN PRINT " EFSO <File sys. error>"
42900     IF IBERR% = EBUS THEN PRINT " EBUS <Command error>"
43000     IF IBERR% = ESTB THEN PRINT " ESTB <Status byte lost>"
43100     IF IBERR% = ESRQ THEN PRINT " ESRQ <SRQ stuck on>"
43200     IF IBERR% = ETAB THEN PRINT " ETAB <Table Overflow>"
43300 REM
43400     PRINT "ibcnt = "; IBCNT%
43500 REM
43600 REM Call the IBONL function to disable the hardware and software
43700 REM  unless error was a time out.
```

```
43800      IF IBSTA% AND TIMO THEN RETURN
43900      V% = 0 : CALL IBONL(BD%, V%) : RETURN
44000 REM
44100      END
```

Spectrometer Control

488_SCAN.BAS is a spectrometer control program that has been used at the factory to test systems like yours. Conversions from working units to motor steps are performed by the program. Data acquisition capabilities are not provided. It offers a formatted input screen to enter parameters and execute a scan routine. A copy of this screen was printed at the factory during the final testing of your system. It is included in the documentation shipped with the spectrometer controller.

```
100 REM ----------------------------
150  REM GW-BASIC Example Program
200  REM
250  REM This is a sample program intended for reference only. It assumes
300  REM  you have a National Instruments IEEE-488 card and driver loaded.
350  REM The driver device addresses [1..31] are referenced as
400  REM  "DEV1".."DEV31". see the National Instruments Software Ref. Manual
450  REM  for further details.
500  REM
550  REM Factory default IEEE-488 address for this instrument is 1.
600  REM
650  REM You MUST merge this code with DECL.BAS and BIB.M MUST be in this
700  REM  directory. These files are supplied by National Instruments.
750 REM ---------------------------------
800 REM --------------------------------
850    PRINT "Begin IEEE-488 Communications Setup"
900 REM ---------------------------
950 REM
1000    BOARD$ = "GPIB0"
1050    DEVICE$ = "DEV"
1100    RBUF$ = SPACE$(132)  : PBUF$ = SPACE$(132)
1150    OBUF$ = SPACE$(132)
1200    JUNK$ = SPACE$(132)
1250    ICH$ = SPACE$(1)
1300    ACK$ = SPACE$(1)
1350    CR$ = CHR$(13)
1360    MINHARDWAREFREQ# = 1#
1400 REM ..
1450    INPUT "Device Address [1..31]: ",DA$
1500    DA = VAL(DA$)
1550    IF (DA > 0) AND (DA < 32) GOTO 1650
1600    SOUND 350,3 : GOTO 1400
1650 REM ..
1700    DEVICE$ = DEVICE$ + DA$
1750 REM
1800    CALL IBFIND(BOARD$, BD%)
1850    MSG$ = "IBFIND ERROR on " + BOARD$
1900    IF (BD% < 0) THEN GOSUB 7250 : STOP
1950 REM ..
2000    CALL IBFIND(DEVICE$, DV%)
2050    MSG$ = "IBFIND ERROR on " + DEVICE$
2100    IF (DV% < 0) THEN GOSUB 7250 : STOP
2150    EOSV% = &HD
2200    V% = EOSV% + &H1400
2250    CALL IBEOS ( DV%, V% )
2300    CALL IBTMO( DV%, T1S% )
2350 REM ..
2400 REM     Flush anything in input buffer
2450    CALL IBRD(DV%,JUNK$) : SOUND 2000,1 : PRINT JUNK$
2500    REM     May time out
2550 REM
```

```
2600 REM  If we have already auto bauded these commands will put us in
2650 REM   a known state without restarting the instrument.
2700 REM
2750    ICH$ = CHR$(222) : CALL IBWRT(DV%,ICH$)
2800    DELAY = .5 : GOSUB 6950        'Wait for possible reset
2850 REM     Flush anything in input buffer
2900    CALL IBRD(DV%,JUNK$) : SOUND 2000,1 : PRINT JUNK$
2950    REM     May time out
3000 REM
3050 REM ..                              'Send <SP>'
3100    ICH$ = " ": CALL IBWRT(DV%, ICH$): SOUND 400, 2
3150    MSG$ = "IBWRT ERROR"
3200    IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
3250 REM
3300    CALL IBRD(DV%, ICH$): SOUND 1000, 2
3350    MSG$ = "IBRD ERROR"
3400    IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
3450 REM
3500    PRINT "Received ", ICH$
3550 REM
3600    IF (ICH$ = "B") GOTO 3850        'Init Program ACK O.K.
3650    IF (ICH$ = "F") GOTO 4450        'Main Program ACK O.K.
3700    GOTO 2350                        'Bad Response. Try Again
3750 REM ..
3800 REM
3850 REM ..    Jump to Main Program
3900    PRINT "Jump to Main Program"
3950    OBUF$ = "O2000" + CHR$(0)
4000    CALL IBWRT(DV%, OBUF$) : MSG$ = "IBWRT ERROR"
4050    IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
4100 REM
4150 REM ..
4200    DELAY = .5 : GOSUB 6950        'Wait for main to start up
4250    CALL IBRD(DV%, ICH$) : MSG$ = "IBRD ERROR"
4300    IF (IBSTA% AND EERR) THEN GOSUB 7250 : RESUME
4350 REM May time out
4400    IF (ICH$ <> "*") GOTO 4150     'Wait for Main Program Response
4450 REM ..
4500    SOUND 500, 5: SOUND 1000, 5: SOUND 500, 5
4550    PRINT "IEEE-488 Communications Established" : PRINT " "
4600    PRINT "Getting EPROM and FLASH Version Numbers"
4650    SAYOK = 0       'Turn OFF ACK sound and OK print out flag
4700 REM  'Put up version number of BOOT and Flash
4750 REM ----- Boot Version
4800   ICH$ = "y" : CALL IBWRT(DV%, ICH$)    'NO comma or <CR> after command
4850   MSG$ = "IBWRT ERROR"
4900   IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
4950   GOSUB 6200     'Wait For Confirmation'
5000 REM
5050   GOSUB 14250     'Get RBUF$ from 488
5100 REM
5150   PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
5200   PRINT "Boot Version:", PBUF$
5250 REM ----- Flash Version
5300   ICH$ = "z" : CALL IBWRT(DV%, ICH$) : MSG$ = "IBWRT ERROR"
5350   IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
5400 REM
```

```
5450   GOSUB 6200     'Wait For Confirmation'
5500   GOSUB 14250     'Get RBUF$ from 488
5550 REM
5600   PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
5650   PRINT "Flash Version:", PBUF$
5700 REM
5750 REM ----- Send Mono Init Command
5755   print "Init Mono, Please Wait..."
5760    CALL IBTMO( DV%, T100S% )      'Allow for long init of 220M, 270M
5800  ICH$ = "A" : CALL IBWRT(DV%,ICH$) : MSG$ = "IBWRT ERROR"
5850  IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
5900  GOSUB 6200     'Wait For Confirmation'
5910  print "Mono Init Done"
5950 PRINT " " : PRINT " " :  PRINT "Hit any key:"
6000 WHILE (INKEY$ = "") : WEND
6050    CALL IBTMO( DV%, T10S% )  'reset time out to something reasonable
6100 GOTO 16600  '*** Jump To User Interface
6150 REM__________________________________________________________________
6200 REM ***** Input ACK ****  and test for valid
6250   CALL IBRD(DV%, ACK$) : IF (SAYOK = 1) THEN SOUND 1000, 2
6300   MSG$ = "IBRD ERROR"
6350   IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
6400   IF (ACK$ = "o") THEN 6450 ELSE 6600
6450   REM ..
6500   IF (SAYOK = 1) THEN PRINT "Receive was O.K."
6550   GOTO 6800
6600 REM ..
6650   IF (ACK$ = "b") THEN PRINT "Receive was bad"
6700   PRINT "Character Received was: "; ASC(ACK$), ACK$
6750   STOP
6800 REM ..
6850   RETURN
6900 REM__________________________________________________________________
6950 REM ***** Time Delay of delay *****
7000   T0 = TIMER
7050   WHILE ((TIMER - T0) < DELAY)
7100   WEND
7150   RETURN
7200 REM__________________________________________________________________
7250 REM ***** IEEE-488 ERROR *****
7300 REM This routine will notify you that an IB call failed and print
7350 REM the status variables.
7400 REM ----------------------
7450 REM
7500     PRINT MSG$
7550 REM
7600     PRINT "ibsta= &H"; HEX$(IBSTA%); " <";
7650     IF IBSTA% AND EERR  THEN PRINT " ERR";
7700     IF IBSTA% AND TIMO THEN PRINT " TIMO";
7750     IF IBSTA% AND EEND THEN PRINT " END";
7800     IF IBSTA% AND SRQI THEN PRINT " SRQI";
7850     IF IBSTA% AND RQS  THEN PRINT " RQS";
7900     IF IBSTA% AND CMPL THEN PRINT " CMPL";
7950     IF IBSTA% AND LOK  THEN PRINT " LOK";
8000     IF IBSTA% AND RREM THEN PRINT " REM";
8050     IF IBSTA% AND CIC  THEN PRINT " CIC";
8100     IF IBSTA% AND AATN THEN PRINT " ATN";
```

```
8150     IF IBSTA% AND TACS THEN PRINT " TACS";
8200     IF IBSTA% AND LACS THEN PRINT " LACS";
8250     IF IBSTA% AND DTAS THEN PRINT " DTAS";
8300     IF IBSTA% AND DCAS THEN PRINT " DCAS";
8350     PRINT ">"
8400 REM
8450     PRINT "iberr= "; IBERR%;
8500     IF IBERR% = EDVR THEN PRINT " EDVR <DOS Error>"
8550     IF IBERR% = ECIC THEN PRINT " ECIC <Not CIC>"
8600     IF IBERR% = ENOL THEN PRINT " ENOL <No Listener>"
8650     IF IBERR% = EADR THEN PRINT " EADR <Address error>"
8700     IF IBERR% = EARG THEN PRINT " EARG <Invalid argument>"
8750     IF IBERR% = ESAC THEN PRINT " ESAC <Not Sys Ctrlr>"
8800     IF IBERR% = EABO THEN PRINT " EABO <Op. aborted>"
8850     IF IBERR% = ENEB THEN PRINT " ENEB <No GPIB board>"
8900     IF IBERR% = EOIP THEN PRINT " EOIP <Async I/O in prg>"
8950     IF IBERR% = ECAP THEN PRINT " ECAP <No capability>"
9000     IF IBERR% = EFSO THEN PRINT " EFSO <File sys. error>"
9050     IF IBERR% = EBUS THEN PRINT " EBUS <Command error>"
9100     IF IBERR% = ESTB THEN PRINT " ESTB <Status byte lost>"
9150     IF IBERR% = ESRQ THEN PRINT " ESRQ <SRQ stuck on>"
9200     IF IBERR% = ETAB THEN PRINT " ETAB <Table Overflow>"
9250 REM
9300     PRINT "ibcnt = "; IBCNT%
9350 REM
9400 REM Call the IBONL function to disable the hardware and software
9450 REM  unless error was a time out.
9500     IF (IBSTA% AND TIMO) THEN RETURN
9550 REM
9600    V% = 0 : CALL IBONL(BD%, V%)
9650    RETURN
9700 REM____________________________________________________________________
9750 REM ***** Short Move to target and display position until done *****
9800  GOSUB 11650           'Move Relative
9850 WHILE ( MOTORBUSY = 1 )
9900    GOSUB 17100         'Get and Display Position
9950    GOSUB 12250         'Test Motor Busy
10000  WEND
10050  GOSUB 17100        'Get and Display Position
10100  RETURN
10150 REM___________________________________________________________________
10200 REM ***** Backlash Move to target and display position until done *****
10250  IF ( TARGET# > STEPS# ) THEN GOTO 10450
10300  TMP# = TARGET# : TARGET# = TMP# - BACKLASH# : GOSUB 10600
10350  IF (HALTSCAN = 1) THEN RETURN
10400  TARGET# = TMP#
10450 REM ...
10500  GOSUB 10600
10550  RETURN
10600 REM___________________________________________________________________
10650 REM ***** Move to target and display position until done *****
10700  GOSUB 11650          'Move Relative
10750  HALTSCAN = 0
10800  WHILE ( MOTORBUSY = 1 )
10850    GOSUB 26050          'GetKey
10900    IF (KEYVALUE = 0) THEN GOTO 11350                     'no key
10950    IF (KEYVALUE = F6KV) THEN GOSUB 30650 : GOTO 11350  'shutter
```

```
11000    IF (KEYVALUE = F8KV) THEN 11050 ELSE 11250      'halt
11050 REM ... 'STOP SCAN
11100     GOSUB 12000    'stop motor
11150     HALTSCAN = 1  'flag to others than scan was halted
11200     GOTO 11350
11250  REM ... invalid key during scan
11300    SOUND 500,1
11350  REM ...
11400    GOSUB 17100         'Get and Display Position
11450    GOSUB 12250         'Test Motor Busy
11500  WEND
11550  GOSUB 17100        'Get and Display Position
11600  RETURN
11650 REM_____________________________________________________________________
11700 REM ***** Move Mono Relative in Steps *****
11750   RELSTEPS# = TARGET# - STEPS#
11800   OBUF$ = "F" + "0" + "," + STR$(RELSTEPS#) + CR$
11850   GOSUB 13950      'Put OBUF$ to 488 and wait for confirm
11900   MOTORBUSY = 1
11950   RETURN
12000 REM_____________________________________________________________________
12050 REM ***** Motor Stop *****
12100   ICH$ = "L"
12150   GOSUB 14500      'Put ICH$ to 488 and wait for confirm
12200   RETURN
12250 REM_____________________________________________________________________
12300 REM ***** Motor Busy *****
12350   ICH$ = "E"
12400   GOSUB 14500      'Put ICH$ to 488 and wait for confirm
12450 REM
12500   CALL IBRD(DV%, ACK$)
12550   MSG$ = "IBRD ERROR"
12600   IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
12650   IF (ACK$ = "q") THEN 12700 ELSE 12750
12700   MOTORBUSY = 1 : GOTO 12800
12750   MOTORBUSY = 0 : GOTO 12800
12800 REM ...
12850   RETURN
12860 REM_____________________________________________________________________
12861 REM ***** Motor Status *****
12862   ICH$ = "K"
12863   GOSUB 14500      'Put ICH$ to RS-232 and wait for confirm
12864   REM
12865   GOSUB 14250      'Get RBUF$ from RS-232
12866   limit = val(rbuf$)
12870   return
12900 REM ____________________________________________________
12950 REM ***** Set Mono Position *****
13000   STEPS# = WAVE# * STEPSPERUNIT#
13050   OBUF$ = "G0," + STR$(STEPS#) + CR$
13100   GOSUB 13950      'Put OBUF$ to 488 and wait for confirm
13150   RETURN
13200 REM ____________________________________________________
13250 REM ***** Read Mono Position
13300   OBUF$ = "H0" + CR$
13350   GOSUB 13950      'Put OBUF$ to 488 and wait for confirm
13400   REM
```

```
13450   GOSUB 14250     'Get RBUF$ from 488
13500   PBUF$ = LEFT$(RBUF$, INSTR(RBUF$,CR$) - 1)
13550   STEPS# = VAL(PBUF$) : WAVE# = STEPS# / STEPSPERUNIT#
13600   RETURN
13650 REM_________________________________________________________________
13700 REM ***** Set Motor Speed
13750   FREQMIN$ = STR$(F1#) : FREQMAX$ = STR$(F2#) : RAMPTIME$ = STR$(RT#)
13800   OBUF$ = "B"+"0,"+ FREQMIN$ +","+ FREQMAX$ +","+ RAMPTIME$ + CR$
13850   GOSUB 13950     'Put OBUF$ to 488 and wait for confirm
13900   RETURN
13950 REM_________________________________________________________________
14000 REM ***** Put OBUF$ to 488 ***** and wait for confirm
14050  CALL IBWRT(DV%,OBUF$) : MSG$ = "IBWRT ERROR"
14100  IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
14150  GOSUB 6200     'Wait For Confirmation'
14200  RETURN
14250 REM_________________________________________________________________
14300 REM ***** Get RBUF$ from 488 *****
14350  CALL IBRD(DV%,RBUF$) : MSG$ = "IBRD ERROR"
14400  IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
14450  RETURN
14500 REM_________________________________________________________________
14550 REM ***** Put ICH$ to 488 ***** and wait for confirm
14600  CALL IBWRT(DV%,ICH$) : MSG$ = "IBWRT ERROR"
14650  IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
14700  GOSUB 6200     'Wait For Confirmation'
14750  RETURN
14800 REM_________________________________________________________________
14850 REM ***** Accessory Busy *****
14900  ICH$ = "1" : GOSUB 14500 'Put ICH$ to 488 and confirm
14950  REM
15000  CALL IBRD(DV%, ACK$) : MSG$ = "IBRD ERROR"
15050  IF (IBSTA% AND EERR) THEN GOSUB 7250 : STOP
15100  IF (ACK$ = "q") THEN ACCESSORYBUSY = 1 ELSE ACCESSORYBUSY = 0
15150  RETURN
15200 REM_________________________________________________________________
15250 REM_________________________________________________________________
15300 REM *** SHUTTER_OPEN  else SHUTTER_CLOSE ***
15350  IF ( SHUTTER = 1) THEN OBUF$ = "W0" + CR$ ELSE OBUF$ = "X0" + CR$
15400  GOSUB 13950     'Put OBUF$ to 488 and wait for confirm
15450  GOSUB 19050     'Display Shutter Position
15452  ACCESSORYBUSY = 1
15500  RETURN
15550 REM_________________________________________________________________
15600 REM *** TURRET_POSITION_1  else  TURRET_POSITION_0 ***
15650  IF ( TURRET = 1) THEN OBUF$ = "a0" + CR$ ELSE OBUF$ = "b0" + CR$
15700  GOSUB 13950     'Put OBUF$ to 488 and wait for confirm
15750  GOSUB 18600      'Display Accessories
15752  ACCESSORYBUSY = 1
15800  RETURN
15850 REM_________________________________________________________________
15900 REM *** ENTR_MIRROR_SIDE_POSITION  else  ENTR_MIRROR_FRONT_POSITION ***
15950  IF ( ENTMIRROR = 1) THEN OBUF$ = "c0" + CR$ ELSE OBUF$ = "d0" + CR$
16000  GOSUB 13950     'Put OBUF$ to 488 and wait for confirm
16050  GOSUB 18600      'Display Accessories
16052  ACCESSORYBUSY = 1
16100  RETURN
```

```
16150 REM_____________________________________________________________________
16200 REM *** EXIT_MIRROR_SIDE_POSITION  else  EXIT_MIRROR_FRONT_POSITION ***
16250  IF ( EXTMIRROR = 1) THEN OBUF$ = "e0" + CR$ ELSE OBUF$ = "f0" + CR$
16300  GOSUB 13950      'Put OBUF$ to 488 and wait for confirm
16350  GOSUB 18600       'Display Accessories
16352  ACCESSORYBUSY = 1
16400  RETURN
16450 REM_____________________________________________________________________
16500 REM
16550 REM -------------------------------
16600    ON ERROR GOTO 16900
16650 REM -------------------------------
16700 REM ...
16750   GOSUB 36350         'Input Last Set of User Parameters
16800   GOTO 19350           'Jump To Main
16850 REM_____________________________________________________________________
16900 REM .....Error Handler
16950   PRINT "***** Error =", ERR, " on line number", ERL
17000   LASTERROR = ERR
17050   IF (LASTERROR = 62) THEN RESUME NEXT ELSE STOP
17100 REM_____________________________________________________________________
17150 REM ***** Get and Display Position *****
17200  GOSUB 13200      'Read Mono Position
17210  gosub 12860      'Check limits
17250  LOCATE 1, 40
17300  PRINT USING ": #####.###  ########  ##"; WAVE#, STEPS#, limit
17350  RETURN
17400 REM_____________________________________________________________________
17450 REM ***** Display Scan *****
17500  LOCATE 3, 40
17550  PRINT USING ": #####.###"; STARTSCAN#
17600  LOCATE 4, 40
17650  PRINT USING ": #####.###"; ENDSCAN#
17700  LOCATE 5, 40
17750  PRINT USING ": #####.#      ######"; SCANSPEED#, SCANFREQ#
17800  RETURN
17850 REM_____________________________________________________________________
17900 REM ***** Display Tweak Speed *****
17950  LOCATE 13, 40
18000  IF (TWEEKSPEED <> 1) THEN PRINT ":  Fast Increment"
18050  IF (TWEEKSPEED = 1) THEN PRINT ":  Slow Increment"
18100  RETURN
18150 REM_____________________________________________________________________
18200 REM ***** Markers *****
18250  LOCATE 16, 10
18300  PRINT USING ": #####.###"; MARKERA#
18350  LOCATE 17, 10
18400  PRINT USING ": #####.###"; MARKERB#
18450  LOCATE 18, 10
18500  PRINT USING ": #####.###"; MARKERC#
18550  RETURN
18600 REM_____________________________________________________________________
18650 REM ***** Display Accessories *****
18700  LOCATE 8, 40
18750  PRINT ": ", TURRET
18800  LOCATE 9, 40
18850  PRINT ": ", ENTMIRROR
```

```
18900  LOCATE 10, 40
18950  PRINT ": ", EXTMIRROR
19000  RETURN
19050 REM_________________________________________________________________
19100 REM ***** Display Shutter *****
19150  LOCATE 7, 40
19200  IF (SHUTTER = 0) THEN PRINT ":  CLOSED" ELSE PRINT ":  OPEN  "
19250  RETURN
19300 REM_________________________________________________________________
19350 REM_________________________________________________________________
19400 REM ***** Main *****
19450  F1KV = 187
19500  F2KV = 188
19550  F3KV = 189
19600  F4KV = 190
19650  F5KV = 191
19700  F6KV = 192
19750  F7KV = 193
19800  F8KV = 194
19850  F9KV = 195
19900  F10KV = 196
19950  LEFTKV = 203
20000  RIGHTKV =205
20050  TKV =  84
20100  EKV =  69
20150  XKV =  88
20200  CTRLA = 1
20250  CTRLB = 2
20300  CTRLC = 3
20350 REM
20400  ON KEY (1) GOSUB 21050
20450  ON KEY (2) GOSUB 21100
20500  ON KEY (3) GOSUB 21150
20550  ON KEY (4) GOSUB 21200
20600  ON KEY (5) GOSUB 21250
20650  ON KEY (6) GOSUB 21300
20700  ON KEY (7) GOSUB 21350
20750  ON KEY (8) GOSUB 21400
20800  ON KEY (9) GOSUB 21450
20850  ON KEY (10) GOSUB 21500
20900  ON KEY (12) GOSUB 21550
20950  ON KEY (13) GOSUB 21600
21000  GOTO 21700
21050  HOTKEY = F1KV : RETURN
21100  HOTKEY = F2KV : RETURN
21150  HOTKEY = F3KV : RETURN
21200  HOTKEY = F4KV : RETURN
21250  HOTKEY = F5KV : RETURN
21300  HOTKEY = F6KV : RETURN
21350  HOTKEY = F7KV : RETURN
21400  HOTKEY = F8KV : RETURN
21450  HOTKEY = F9KV : RETURN
21500  HOTKEY = F10KV : RETURN
21550  HOTKEY = LEFTKV : RETURN
21600  HOTKEY = RIGHTKV : RETURN
21650 REM
21700  KEY(1) ON: KEY(2) ON: KEY(3) ON: KEY(4) ON: KEY(5) ON
```

```
21750  KEY(6) ON: KEY(7) ON: KEY(8) ON: KEY(9) ON: KEY(10) ON
21800  KEY(12) ON: KEY(13) ON
21850 REM
21900  GOTO 22100
21950  CLS
22000  LOCATE 15, 20
22050  INPUT "Mono Type: ", MONOWANTED
22100  GOSUB 26700            'Get Mono Type and Parameters
22150  PRINT " ": PRINT "Hit any key:"
22200  WHILE (INKEY$ = "") : WEND
22250 REM
22300  CLS
22350 REM
22400  SAYOK = 0                    'TURN OFF 488 ACK sound and OK print out
flag
22450  MOTORBUSY = 0 : MOVEBUSY = 0 : HALTSCAN = 0
22452  ACCESSORYBUSY = 0 : tupdate = timer
22500  WAVE# = 0# : TARGET# = 0#
22550  TWEEKINC = 1
22600  TWEEKSPEED = 1 : TWEEKMAX = STEPSPERUNIT# + 1
22650  STEPS# = 0#
22700  SHUTTER = 0
22750  TURRET = 0
22800  ENTMIRROR = 0
22850  EXTMIRROR = 0
22900  SCANSPEED# = SCANFREQ# / STEPSPERUNIT#
22950 REM
23000  MENUROW = 1 : MENUCOL = 15 : GOSUB 25000      'Put up Main Menu
23050  ROW = 20 : COL = 20 : MESSAGE$ = "Command: " : GOSUB 28100
23100  ROW = 20 : COL = 20+9
23150 REM
23200  GOSUB 30150      'Calibrate
23250  GOSUB 17100       'Display Position
23300  GOSUB 17400       'Display Scan
23350  GOSUB 17850       'Display Tweak Speed
23400  GOSUB 18600       'Display Accessories
23450  GOSUB 15250       'Do and Display Shutter
23500  GOSUB 18150       'Display Markers
23550  F1# = MINFREQ# : F2# = MAXFREQ# : RT# = RAMPTIME#
23600  GOSUB 13650       'Set Motor Speed
23650 REM
23700 WHILE (1)
23750   GOSUB 26050         'GetKey
23800   IF (KEYVALUE = 0) THEN GOTO 24850
23850   IF (KEYVALUE = 19) THEN GOTO 21950   '(Ctrl-S) Change Mono Type
23900   IF (KEYVALUE = F1KV) THEN GOSUB 28350 : GOTO 24850
23950   IF (KEYVALUE = F2KV) THEN GOSUB 28800 : GOTO 24850
24000   IF (KEYVALUE = F3KV) THEN GOSUB 29200 : GOTO 24850
24050   IF (KEYVALUE = F4KV) THEN GOSUB 29600 : GOTO 24850
24100   IF (KEYVALUE = F5KV) THEN GOSUB 30150 : GOTO 24850
24150   IF (KEYVALUE = F6KV) THEN GOSUB 30650 : GOTO 24850
24200   IF (KEYVALUE = F7KV) THEN GOSUB 30950 : GOTO 24850
24250   IF (KEYVALUE = F8KV) THEN GOSUB 32550 : GOTO 24850
24300   IF (KEYVALUE = F9KV) THEN GOSUB 32850 : GOTO 24850
24350   IF (KEYVALUE = F10KV) THEN GOSUB 33150 : GOTO 24850
24400   IF (KEYVALUE = LEFTKV) THEN GOSUB 33400 : GOTO 24850
24450   IF (KEYVALUE = RIGHTKV) THEN GOSUB 34100 : GOTO 24850
```

```
24500   IF ( (AKEY$ = "T") OR (AKEY$ = "t") ) THEN GOSUB 34750 : GOTO 24850
24550   IF ( (AKEY$ = "E") OR (AKEY$ = "e") ) THEN GOSUB 35050 : GOTO 24850
24600   IF ( (AKEY$ = "X") OR (AKEY$ = "x") ) THEN GOSUB 35350 : GOTO 24850
24650   IF ( (AKEY$ = "A") OR (AKEY$ = "a") ) THEN GOSUB 35650 : GOTO 24850
24700   IF (KEYVALUE = CTRLA) THEN GOSUB 35950 : GOTO 24850
24701   IF ( (AKEY$ = "B") OR (AKEY$ = "b") ) THEN GOSUB 36310 : GOTO 24850
24702   IF (KEYVALUE = CTRLB) THEN GOSUB 36316 : GOTO 24850
24703   IF ( (AKEY$ = "C") OR (AKEY$ = "c") ) THEN GOSUB 36330 : GOTO 24850
24704   IF (KEYVALUE = CTRLC) THEN GOSUB 36336 : GOTO 24850
24750   LOCATE 23,1 : PRINT "Key Value: ",KEYVALUE : SOUND 1000,1
24800   IF (KEYVALUE = 26 ) THEN STOP
24850   REM ...
24852   if ( timer - tupdate < 0.2 ) then goto 24890
24853    tupdate = timer
24854     gosub 17100     'Update position and limits
24856     gosub 14800     'Test accessory busy
24858     if (ACCESSORYBUSY = 1) then LOCATE 23,20 :  Print "Accessory Busy"
24860     if (ACCESSORYBUSY = 0) then  LOCATE 23,1 : PRINT SPACE$(70)
24890   REM ...
24900 WEND
24950 STOP
25000 REM____________________________________________________________________
25050 REM____________________________________________________________________
25100 REM ***** Main Menu *****
25150  LOCATE MENUROW+0,MENUCOL: PRINT "<- Arrow -> to Tweak Mono"
25200  LOCATE MENUROW+1,MENUCOL: PRINT "F1  - Set"
25250  LOCATE MENUROW+2,MENUCOL: PRINT "F2  - Start Position"
25300  LOCATE MENUROW+3,MENUCOL: PRINT "F3  - End Position"
25350  LOCATE MENUROW+4,MENUCOL: PRINT "F4  - Scan Speed"
25400  LOCATE MENUROW+5,MENUCOL: PRINT "F5  - Calibrate"
25450  LOCATE MENUROW+6,MENUCOL: PRINT "F6  - Shutter"
25500  LOCATE MENUROW+7,MENUCOL: PRINT " T   - Turret"
25550  LOCATE MENUROW+8,MENUCOL: PRINT " E   - Entrance Mirror"
25600  LOCATE MENUROW+9,MENUCOL: PRINT " X   - Exit Mirror"
25650  LOCATE MENUROW+10,MENUCOL: PRINT "F7 - Scan"
25700  LOCATE MENUROW+11,MENUCOL: PRINT "F8 - HALT"
25750  LOCATE MENUROW+12,MENUCOL: PRINT "F9 - Tweak Speed"
25800  LOCATE MENUROW+13,MENUCOL: PRINT "F10 - EXIT"
25850  LOCATE MENUROW+15,1 : PRINT "Marker A"
25900  LOCATE MENUROW+16,1 : PRINT "Marker B"
25950  LOCATE MENUROW+17,1 : PRINT "Marker C"
26000  RETURN
26050 REM____________________________________________________________________
26100 REM ***** GetKey *****
26150  IF ( HOTKEY = 0 ) THEN GOTO 26350
26200  KEYVALUE = HOTKEY : HOTKEY = 0
26250  RETURN
26300 REM
26350  AKEY$ = INKEY$
26400  L = LEN(AKEY$)+1
26450  ON L GOTO 26500,26550,26600
26500  KEYVALUE = 0 : GOTO 26650
26550  KEYVALUE = ASC(AKEY$) : GOTO 26650
26600  KEYVALUE = ASC(MID$(AKEY$, 2)) + 128 : GOTO 26650
26650  RETURN
26700 REM____________________________________________________________________
26750 REM ***** Read Mono Parameter File *****
```

```
26800  OPEN "488_scan.ini" FOR INPUT AS #2
26850  NRECSKIP = 7 : LASTERROR = 0
26900  REM ...
26950  INPUT #2, MONOTYPE, COMMENT$: PRINT MONOTYPE, COMMENT$
27000  IF (LASTERROR <> 62) THEN 27200
27050    CLOSE #2
27100    PRINT "Parameters for Mono Type ",MONOWANTED," NOT found."
27150    GOTO 22000
27200  REM ...
27250  IF (MONOTYPE = MONOWANTED) THEN 27300 ELSE 27700
27300     INPUT #2, STEPSPERUNIT#, COMMENT$: PRINT STEPSPERUNIT#, COMMENT$
27350     INPUT #2, MINFREQ#, COMMENT$: PRINT MINFREQ#, COMMENT$
27400     INPUT #2, MAXFREQ#, COMMENT$: PRINT MAXFREQ#, COMMENT$
27450     INPUT #2, RAMPTIME#, COMMENT$: PRINT RAMPTIME#, COMMENT$
27500     INPUT #2, BACKLASH#, COMMENT$: PRINT BACKLASH#, COMMENT$
27550     INPUT #2, JUNK#, COMMENT$: PRINT SCANFREQ#, COMMENT$
27600     INPUT #2, COMMENT$: PRINT COMMENT$
27650   GOTO 27950
27700      FOR I = 1 TO NRECSKIP
27750        INPUT #2, COMMENT$
27800      NEXT I
27850      PRINT COMMENT$
27900   GOTO 26900
27950 REM ...
28000  CLOSE #2
28050  RETURN
28100 REM
28150 REM ***** Print Message *****
28200  LOCATE ROW,COL : PRINT SPACE$(50) : LOCATE ROW,COL
28250  PRINT MESSAGE$
28300  RETURN
28350 REM
28400 REM *****  F1  *****
28450  MESSAGE$ = "F1" : GOSUB 28100
28500  LOCATE 22,20
28550  INPUT "Move to: ", W#
28600  TARGET# = W# * STEPSPERUNIT#
28650  LOCATE 22,1 : PRINT SPACE$(70)
28700  GOSUB 10150         'Move to target, display and wait till done
28750  RETURN
28800 REM
28850 REM *****  F2  *****
28900  MESSAGE$ = "F2" : GOSUB 28100
28950  LOCATE 22,20
29000  INPUT "Start of Scan: ", STARTSCAN#
29050  LOCATE 22,1 : PRINT SPACE$(70)
29100  GOSUB 17400                    'DisplayScan
29150  RETURN
29200 REM
29250 REM *****  F3  *****
29300  MESSAGE$ = "F3" : GOSUB 28100
29350  LOCATE 22,20
29400  INPUT "End of Scan: ", ENDSCAN#
29450  LOCATE 22,1 : PRINT SPACE$(70)
29500  GOSUB 17400                    'DisplayScan
29550  RETURN
29600 REM
```

```
29650 REM *****  F4  *****
29700  MESSAGE$ = "F4" : GOSUB 28100
29750  LOCATE 22,20
29800  INPUT "Scan Speed: ", SCANSPEED#
29850  LOCATE 22,1 : PRINT SPACE$(70)
29900  SCANFREQ# = SCANSPEED# * STEPSPERUNIT#
29950  IF ( SCANFREQ# <= MAXFREQ# ) THEN GOTO 29960
29952  SCANFREQ# = MAXFREQ# : SCANSPEED# = SCANFREQ# / STEPSPERUNIT#
29954  GOSUB 17400 :  SOUND 500,1 : GOTO 29600
29956 REM
29960  IF ( SCANFREQ# >= MINHARDWAREFREQ# ) THEN GOTO 30000
29962  SCANFREQ# = MINHARDWAREFREQ# : SCANSPEED# = SCANFREQ# / STEPSPERUNIT#
29964  GOSUB 17400 : SOUND 500,1 : GOTO 29600
30000 REM ...
30050  GOSUB 17400             'DisplayScan
30100  RETURN
30150 REM_______________________________________________________________________
30200 REM *****  F5  *****
30250  MESSAGE$ = "F5" : GOSUB 28100
30300  LOCATE 22,20
30350  INPUT "Where are you: ", WAVE#
30400  LOCATE 22,1 : PRINT SPACE$(70)
30450 REM
30500  GOSUB 12900       'Set Mono Position
30550  GOSUB 17100       'Display Position
30600  RETURN
30650 REM_______________________________________________________________________
30700 REM *****  F6  *****
30750  MESSAGE$ = "F6" : GOSUB 28100
30800  SHUTTER = 1 - SHUTTER
30850  GOSUB 15250
30900  RETURN
30950 REM_______________________________________________________________________
31000 REM *****  F7  *****
31050  MESSAGE$ = "F7" : GOSUB 28100
31100 REM
31150  LOCATE 11, 40
31200  PRINT ":  Positioning"
31250  HALTSCAN = 0
31300  TARGET# = STARTSCAN# * STEPSPERUNIT#
31350  GOSUB 10150         'Move to target, display and wait till done
31400 REM
31450  IF ( HALTSCAN = 1 ) THEN GOTO 32150      'User halt
31500 REM
31550  DELAY = 1! : GOSUB 6950       'Delay before Start
31600  LOCATE 11, 40
31650  PRINT ":  STARTED     "
31700  TARGET# = ENDSCAN# * STEPSPERUNIT#
31750 REM Load Scan Speed
31800  F1# = MINFREQ# :  F2# = SCANFREQ# : RT# = RAMPTIME#
31850  IF ( SCANFREQ# < MINFREQ# ) THEN F1# = SCANFREQ#
31900  GOSUB 13650       'Set Motor Speed
31950 REM
32000  GOSUB 10150        'Move to target, display and wait till done
32050 REM
32100  GOSUB 32550       'Stop Scan and re-enable Hot Keys
32150 REM ...
```

```
32200  HALTSCAN = 0
32250  LOCATE 11, 40
32300  PRINT ":  STOPPED     "
32350 REM Load Normal Speed
32400  F1# = MINFREQ# : F2# = MAXFREQ# : RT# = RAMPTIME#
32450  GOSUB 13650        'Set Motor Speed
32500  RETURN
32550 REM____________________________________________________________
32600 REM *****  F8  *****
32650  MESSAGE$ = "F8" : GOSUB 28100
32700  HALTSCAN = 1
32750  WHILE ( INKEY$ <> "" ) : WEND  'Flush input buffer
32800  RETURN
32850 REM____________________________________________________________
32900 REM *****  F9  *****
32950  MESSAGE$ = "F9" : GOSUB 28100
33000  TWEEKSPEED = TWEEKMAX - TWEEKSPEED
33050  GOSUB 17850            'DisplayTweekSpeed
33100  RETURN
33150 REM____________________________________________________________
33200 REM *****  F10  *****
33250  MESSAGE$ = "F10" : GOSUB 28100
33300  GOSUB 36950       'Shut down
33350  RETURN
33400 REM____________________________________________________________
33450 REM *****  Left Arrow  *****
33500  MESSAGE$ = "Left" : GOSUB 28100
33550  IF ( TWEEKSPEED <> 1 ) THEN 33650
33600  IF ( (TIMER-T1) > .1 ) THEN 33650 ELSE 33700
33650  F = 1 : GOTO 33750
33700  F = F + 3 : IF ( F > 5*TWEEKMAX ) THEN F = 5*TWEEKMAX
33750  TARGET# = (STEPS# - TWEEKINC * TWEEKSPEED * F)
33800  IF (TARGET# < 0#) THEN TARGET# = 0# : SOUND 1000,1
33850 REM
33900  GOSUB 9700        'Short Move to target, display and wait till done
33950  T1 = TIMER
34050  RETURN
34100 REM____________________________________________________________
34150 REM *****  Right Arrow  *****
34200  MESSAGE$ = "Right" : GOSUB 28100
34250  IF ( TWEEKSPEED <> 1 ) THEN 34350
34300  IF ( (TIMER-T2) > .1 ) THEN 34350 ELSE 34400
34350  F = 1 : GOTO 34450
34400  F = F + 3 : IF ( F > 5*TWEEKMAX ) THEN F = 5*TWEEKMAX
34450  TARGET# = (STEPS# + TWEEKINC * TWEEKSPEED * F)
34500 REM
34550  GOSUB 9700        'Short Move to target, display and wait till done
34600  T2 = TIMER
34700  RETURN
34750 REM____________________________________________________________
34800 REM ***** "T" Turret  *****
34850  MESSAGE$ = "T" : GOSUB 28100
34900  TURRET = 1 - TURRET
34950  GOSUB 15550       'Do Turret and Display Accessories
35000  RETURN
35050 REM____________________________________________________________
35100 REM ***** "E" ENTmirror  *****
```

```
35150  MESSAGE$ = "E" : GOSUB 28100
35200  ENTMIRROR = 1 - ENTMIRROR
35250  GOSUB 15850       'Do Entrance Mirror and Display Accessories
35300  RETURN
35350 REM__________________________________________________________________
35400 REM ***** "X" EXTmirror  *****
35450  MESSAGE$ = "X" : GOSUB 28100
35500  EXTMIRROR = 1 - EXTMIRROR
35550  GOSUB 16150       'Do Exit Mirror and Display Accessories
35600  RETURN
35650 REM__________________________________________________________________
35700 REM ***** "A" go to marker A *****
35750  MESSAGE$ = "Go to A" : GOSUB 28100
35800  TARGET# = MARKERA# * STEPSPERUNIT#
35850  GOSUB 10150        'Move to target, display and wait till done
35900  RETURN
35950 REM__________________________________________________________________
36000 REM ***** "Ctrl-A" Set marker A *****
36050 MESSAGE$ = "Set A" : GOSUB 28100
36100  LOCATE 22,20
36150  INPUT "Marker A: ", MARKERA#
36200  LOCATE 22,1 : PRINT SPACE$(70)
36250  GOSUB 18150        'Display Markers
36300  RETURN
36310 REM__________________________________________________________________
36311 REM ***** "B" go to marker B *****
36312  MESSAGE$ = "Go to B" : GOSUB 28100
36313  TARGET# = MARKERB# * STEPSPERUNIT#
36314  GOSUB 10150        'Move to target, display and wait till done
36315  RETURN
36316 REM__________________________________________________________________
36317 REM ***** "Ctrl-B" Set marker B *****
36318 MESSAGE$ = "Set B" : GOSUB 28100
36319  LOCATE 22,20
36320  INPUT "Marker B: ", MARKERB#
36321  LOCATE 22,1 : PRINT SPACE$(70)
36322  GOSUB 18150        'Display Markers
36323  RETURN
36330 REM__________________________________________________________________
36331 REM ***** "C" go to marker C *****
36332  MESSAGE$ = "Go to C" : GOSUB 28100
36333  TARGET# = MARKERC# * STEPSPERUNIT#
36334  GOSUB 10150        'Move to target, display and wait till done
36335  RETURN
36336 REM__________________________________________________________________
36337 REM ***** "Ctrl-C" Set marker C *****
36338 MESSAGE$ = "Set C" : GOSUB 28100
36339  LOCATE 22,20
36340  INPUT "Marker C: ", MARKERC#
36341  LOCATE 22,1 : PRINT SPACE$(70)
36342  GOSUB 18150        'Display Markers
36343  RETURN
36350 REM__________________________________________________________________
36400 REM ***** Input User Parameters  *****
36450  OPEN "488_scan.usr" FOR INPUT AS #2
36500  INPUT #2, MONOWANTED, COMMENT$
36550  INPUT #2, STARTSCAN#, COMMENT$
```

```
36600  INPUT #2, ENDSCAN#, COMMENT$
36650  INPUT #2, SCANFREQ#, COMMENT$
36700  INPUT #2,MARKERA#, COMMENT$
36750  INPUT #2,MARKERB#, COMMENT$
36800  INPUT #2,MARKERC#, COMMENT$
36850  CLOSE #2
36900  RETURN
36950 REM_______________________________________________________________________
37000 REM ***** EXIT  *****
37050 REM Save User Parameters on exit
37100  OPEN "488_scan.usr" FOR OUTPUT AS #2
37150  PRINT #2, MONOTYPE,      " ;Mono Type"
37200  PRINT #2, STARTSCAN#,    " ;Start of Scan"
37250  PRINT #2, ENDSCAN#,      " ;End of Scan"
37300  PRINT #2, SCANFREQ#,     " ;Scan Freq"
37350  PRINT #2,MARKERA#,       " ;Marker A"
37400  PRINT #2,MARKERB#,       " ;Marker B"
37450  PRINT #2,MARKERC#,       " ;Marker C"
37500  CLOSE #2
37550 REM
37600  CALL IBLOC(DV%)             'Put 488 off line
37650  SYSTEM                      'Exit BASIC
37700  END
```

Appendix 5: IEEE 488 Bus Signal Examples

The IEEE 488 instrument controller does not set or look for EOI.

1 For a confirm of a single character response you must read exactly 1 byte .

2a For a multi byte response you must read characters until you see "<CR>" (Ød hex), then terminate your read.

2b If you are using a National Instruments board you can configure the device driver to terminate read on EOS and define the EOS character to be ØD HEX.

Following are the 488 BUS signals for a few of our commands using the National Instruments interactive program IBIC.

Assuming that the IEEE 488 HOST Controller is at address Ø, and
the Spectrometer Controller is at IEEE 488 address 1;

REN is always asserted during a conversation.

UNL	=	3F HEX
MTAØ	=	4Ø HEX
MTA1	=	41 HEX
MLAØ	=	2Ø HEX
MLA1	=	21 HEX

WHERE AM I command:

IBWRT "<space>" Sends a space character.

MTAØ	ATN during 488 command cycle
UNL	ATN
MLA1	ATN
2Ø HEX	NOT ATN for data (no EOI)

IBRD 1 Reads 1 character.

UNL	ATN
MLAØ	ATN
MTA1	ATN
42 HEX (="B")	NOT ATN (no EOI)

Spectrometer Control

TRANSFER TO MAIN PROGRAM command:

IBWRT "O2ØØØ\xØ"   Sends "O2ØØØ<NULL>"

MTAØ	ATN
UNL	ATN
MLA1	ATN
4f HEX (="O")	NOT ATN
32 HEX (="2")	NOT ATN
3Ø HEX (="Ø")	NOT ATN
3Ø HEX (="Ø")	NOT ATN
3Ø HEX (="Ø")	NOT ATN
ØØ HEX (=<NULL>)	NOT ATN (no EOI)

MOTOR READ POSITION command:

IBWRT "HØ\xØD"   Sends "HØ<CR>"

MTAØ	ATN
UNL	ATN
MLA1	ATN
48 HEX (="H")	NOT ATN
3Ø HEX (="Ø")	NOT ATN
ØD HEX (="<CR>")	NOT ATN (no EOI)

IBRD 132   Driver is configured to terminate read on EOS of ØD HEX (=<CR>)

UNL	ATN	
MLAØ	ATN	
MTA1	ATN	
6f HEX (="o")	NOT ATN	Command Confirm Character (lower case o)
31 HEX (="1")	NOT ATN	ASCII 1
32 HEX (="2")	NOT ATN	ASCII 2
3Ø HEX (="Ø")	NOT ATN	ASCII Ø
ØD HEX (="<CR>")	NOT ATN	Command Terminator <Carriage Return> (no EOI)

data count receiver = 5 bytes

Appendix 6: Terminology Changes

Readers who have started using earlier versions of this document should note that a few terms have been changed for the sake of clarity.

For new users, these changes are transparent, and avoid some earlier confusing terms. For new users there is no particular benefit in reading this section, unless you are trying to translate earlier terminology used by someone else.

Wherever possible, prior existing terms have been retained and defined more explicitly than before.

The following terms have been changed in this manual:

"Elementary Mode" was misleading, because no change of "Modes" is necessary to program with the Elementary Commands, as opposed to the Independent Commands

"Hi IQ" referred to either the Intelligent Communications Mode, or to the Independent Command Method, which are not the same.

"Independent Mode" was misleading, because no change of "Modes" is necessary to program with the Independent Commands, as opposed to the Elementary Commands

"Lo IQ" referred to either the Terminal Communications Mode or to the Elementary Command Method, which are not the same.

"Programmer's Instruction Set" has been replaced by Programmers Command Set, which is both more accurate and, at times, less confusing.

"Spectrometer Controller" is generally used in place of SPEX232, JY232, SPEX488, JY488 Spectrometer control interfaces, DataScan and DataLink controller-photometers and the SpectrAcq controller.

Index

270 Spectrometer Commands:

Boot Mode **Command**	write **Mode**	**Meaning**	**Response**	**Comment**
Boot MOde		Boot	*	Need 1/2 s wait
O2000<null>	0	Request mode	B=>Boot	No <CR> after *
<sp>	0	Where am I?	F or f=> Main	
<248> (ASCII character 248 or hex F8) (no CR)		go from flash to indep mode(??)		200 ms wait
<222> in hex DE	0	Reboot if hung		200 ms wait
E#<null> # is hex		Set IEEE add	<2>	

Notes: Write: Mode 7 No EOI, Mode 0 send EOI; Read: Mode 0 No EOI

HiIQ mode or Main mode or Flash mode

Monochromator #: 0

Slit #	0	Front entrance
	1	side entrance
	2	front exit
	3	side exit

Command	**Mode**	**Meaning**	**Response**
A	Wrt 0	Motor Init	o or b
Wait >15s			
B0,400,800,2000<cr>		Set motor speed	o or b
B Mon, Min, Max, ramp <cr>			
C0<cr>		Read motor speed	o400,800,2000<cr> or b
Cmotor<cr>			
E	Wrt 7 Rd 0	motor busy	o or b q or z
F0,1000<cr>		move motor + higher steps	o or b
Fmon,steps<cr>			
G0,100000<cr>		Set motor position (calibrate)	o or b
Gmon,steps<cr>			
H0<cr>		Read motor position	o100000<cr> or b
Hmon<cr>			
K	Wrt 7 Rd 0	Limits	b or o0<cr>
L		Stop motor	o or b
W0<cr>		Open Shutter (100ms)	o or b
X0<cr>		Close Shutter (100 ms)	o or b
a0<cr>		turret Position 1	o or b
b0<cr>		Turret Position 2	o or b

c0<cr>		Set side entrance port More ent mirror to side	o or b
d0<cr>		Set front entrance port Move ent mirror to front	o or b
e0<cr>		Set side exit port Move exit mirror to side	o or b
f0<cr>		Set front exit port Move exit mirror to front	o or b
g0,0,400<cr> gmon,slit,freq<cr>		Set slit speed	o or b
h0,0<cr> hmon,slit<cr>		Read slit speed	o400<cr> or b
i0,0,0<cr> i(mono, slit,position)<cr>		Set slit position	o or b
j0,0<cr>		Read slit position	o100<cr> or b
k0,0,500<cr> kmon,slit,steps<cr>		Move slit relative	o or b
l (lower case L)	Wrt 7 Rd 0	Accessory busy	o(q or z) or b q=>busy
y<cr>	Wrt 0 Rd 1	Boot V 3.5…<cr> (Get Boot Version #)	
z<cr>	Wrt 0 Rd 1	Main V 3.5 …<cr> (Get main version #)	
u			