

TEK PROGRAMMERS
MANUAL

Part No. 070-6023-00
Product Group 2F

2710 GPIB Programmers Manual


Please Check the Back of Manual for CHANGE INFORMATION

First Printing DEC 1989

Tektronix[®]
COMMITTED TO EXCELLENCE

Copyright © 1989 Tektronix Inc. All rights reserved. Contents of this publication may not be reproduced in any form without the written permission of Tektronix Inc.

Products of Tektronix Inc. and its subsidiaries are covered by U.S. and foreign patents and/or pending patents.

TEKTRONIX, TEK, SCOPE-MOBILE, and  are registered trademarks of Tektronix Inc. TELEQUIPMENT is a registered trademark of Tektronix U.K. Limited.

Printed in U.S.A. Specification and price change privileges are reserved.

Table of Contents

Table of Contents	i
List of Illustrations	iv
List of Tables	v
Safety Summary	vii
Section 1 - Introduction To GPIB Operation	1-1
What is the GPIB and How Does It Work?	1-1
What You Need To Operate the 2710 Over the GPIB	1-3
System Controller	1-5
Software Device Driver	1-5
2710 Equipped With the GPIB Option	1-5
Interconnecting Cable	1-6
Application Software	1-6
(Optional) Printer or Plotter	1-6
Setting Up For GPIB Operation	1-7
Connecting the Equipment	1-7
Configuring the 2710	1-7
Placing the 2710 Online	1-7
Setting the GPIB Device Address	1-9
The Power-on SRQ	1-9
Setting the Message Terminator	1-10
Setting the TALK ONLY Option	1-10
Configuring the Device Driver	1-11
Installing the Device Driver	1-13
Configuring the (Optional) Printer and/or Plotter	1-13
Printer Configuration	1-14
Plotter Configuration	1-14
Communicating With the 2710	1-14
Readying the Software	1-15
Preparing the QuickBASIC Environment	1-16
A Simple Control Program	1-16
Section 2 - 2710-Specific Message Structure	2-1
What Is a Message	2-1
Input Message	2-2
Output Message	2-3
Message Unit	2-3
Message Unit Delimiter	2-3
Message Terminator	2-3
Command	2-4

//////////////////////////////////// 2710 GPIB Programmers Manual //////////////////////////////////////

Query	2-4
Response	2-4
Mnemonic or Header	2-4
Header Delimiter	2-4
Argument	2-5
Digit	2-5
Number Argument	2-5
Units	2-5
Character Argument	2-6
String Argument	2-6
Link Argument	2-6
Binary Block Argument	2-6
Argument Delimiter	2-7
Message Buffering	2-7
Message Format	2-8
Message	2-8
Commands	2-9
Queries	2-10
Responses	2-11
Headers	2-12
Space	2-13
Comma	2-13
Semicolon	2-13
Colon	2-13
Line Feed	2-13
Section 3 - 2710-Specific Commands and Queries	3-1
The Command/Query List	3-1
Functional Grouping of Commands and Queries	3-4
Frequency/markers Commands	3-6
Span and resolution BW commands	3-10
Vertical Display and Reference Level Commands	3-12
Input Menu Commands	3-14
Sweep/trigger Commands	3-16
Display Storage Commands	3-20
Application Menu Commands	3-24
Utility Menu Commands	3-26
Detector/Generator Commands	3-28
Curve and Waveform Commands	3-30
System-related Commands	3-31
Miscellaneous Commands	3-32

//////////////////////////////////// **2710 GPIB Programmers Manual** //////////////////////////////////////

Section 4 - 2710 Command and Query Definitions	4-1
Typographical Conventions	4-1
List of Commands and Queries	4-3
Section 5 - Status Reporting	5-1
The Service Request	5-2
Status Byte	5-3
Event Codes	5-7
Section 6 - Programming for GPIB Operations	6-1
Before You Start Programming	6-1
Beginning Your Program	6-2
Error Trapping	6-4
GPIB System Software	6-7
Sample Subroutines	6-7
Curve Transfers	6-8
Transferring Files	6-10
Plotting 2710 Screen Data	6-12
Returning the On-screen Readouts	6-16
Saving and Restoring Equipment Settings	6-17
Waiting For Results	6-19
Sample Program	6-21
Appendix A - IEEE STD 488 (GPIB) System Concepts	A-1
Mechanical Elements	A-1
Electrical Elements	A-2
Functional Elements	A-2
A Typical GPIB System	A-4
Talkers, Listeners, and Controllers	A-6
Interface Control Messages	A-7
Device-Dependent Messages	A-8
GPIB Signal Line Definitions	A-12
Transfer Bus (Handshake)	A-13
Management Bus	A-14
Interface Functions and Messages	A-16
RL (Remote-Local Function)	A-17
T/TE (Talker and Listener Functions)	A-18
SH and AH (Source and Acceptor Handshake Functions)	A-19
DC (Device Clear Function)	A-20
DT (Device Trigger Function)	A-20
C, SR, and PP (Controller, SRQ, and Parallel Poll Functions)	A-21
Taking Control (Asynchronous or Synchronous)	A-22
Performing a Serial Poll	A-22
Performing a Parallel Poll	A-23

List of Illustrations

Figure 1-1. Typical small instrument system.	1-4
Figure 1-2. Connection examples for multiple instruments.	1-8
Figure 1-3. The 2710 GPIB Port Configuration Menu.	1-9
Figure 1-4. The National Instruments PC-2 Board Characteristics screen from IBCONF.	1-11
Figure 1-5. The National Instruments PC-2A Board Characteristics screen from IBCONF.	1-12
Figure 1-6. The TEK_SA Device Characteristics screen from IBCONF.	1-12
Figure 3-1. Frequency/marker front panel commands.	3-7
Figure 3-2. Marker/Frequency Menu commands.	3-9
Figure 3-3. Span and resolution front panel commands.	3-11
Figure 3-4. Vertical scale and resolution BW front panel commands.	3-13
Figure 3-5. Input Menu commands.	3-15
Figure 3-6. Sweep/trigger front panel commands.	3-17
Figure 3-7. Sweep/Trigger Menu commands.	3-19
Figure 3-8. Display storage front panel commands.	3-21
Figure 3-9. Display Menu commands.	3-23
Figure 3-10. Applications Menu commands.	3-25
Figure 3-11. Utility Menu (abbreviated) commands.	3-27
Figure 3-12. Detector/Generator Menu commands.	3-29
Figure 4-1. Format of curve data.	4-16
Figure 4-2. 2710 graticule coordinates.	4-17
Figure 6-1. Possible data acquisition scheme.	6-13
Figure 6-2. Possible data print/plot scheme.	6-14
Figure A-1. IEEE Std 488 (GPIB) connector.	A-3
Figure A-2. A typical GPIB system.	A-5
Figure A-3. ASCII and GPIB code chart.	A-11
Figure A-4. An example of data byte traffic on the GPIB.	A-15
Figure A-5. A typical handshake timing sequence (idealized).	A-15

List of Tables

Table 1-1. A simple 2710 control program.	1-19
Table 3-1. List of 2710-specific commands and queries.	3-1
Table 3-2. Frequency/marker front panel commands.	3-6
Table 3-3. Marker/Frequency Menu commands.	3-8
Table 3-4. Span and resolution front panel commands.	3-10
Table 3-5. Vertical scale and resolution BW front panel commands.	3-12
Table 3-6. Input Menu commands.	3-14
Table 3-7. Sweep/trigger front panel commands.	3-16
Table 3-8. Sweep/Trigger Menu commands.	3-18
Table 3-9. Display storage front panel commands.	3-20
Table 3-10. Display Menu commands.	3-22
Table 3-11. Applications Menu commands.	3-24
Table 3-12. Utility Menu (abbreviated) commands.	3-26
Table 3-13. Detector/Generator Menu commands.	3-28
Table 3-14. Curve and waveform commands.	3-30
Table 3-15. System-related commands.	3-31
Table 3-16. Miscellaneous commands.	3-32
Table 4-1. The 2710 file system.	4-27
Table 4-2. Arguments fo the KEY command.	4-37
Table 4-3. Arguments of the WFMpre? query.	4-97
Table 4-4. Formulas related to Table 4-3.	4-98
Table 5-1. General system status bytes.	5-4
Table 5-2. General device-dependent status bytes.	5-4
Table 5-3. Specific system status bytes.	5-5
Table 5-4. Specific device-dependent status bytes.	5-5
Table 5-5. Event priorities.	5-6
Table 5-6. Subroutine for reading the status byte.	5-6
Table 5-7. Event code categories.	5-7
Table 5-8. Subroutine for reading event codes.	5-8
Table 5-9. GPIB event codes.	5-9
Table 6-1. Variable names.	6-3
Table 6-2. GPIB system software callabel subroutines.	6-6
Table 6-3. Program header statements.	6-7
Table 6-4. Subroutines to return (GET.CURVE) or transmit (PUT.CURVE) curve data.	6-9

////// 2710 GPIB Progammers Manual ////

Table 6-5. Subroutines to return (GET.FILE) or transmit (PUT.FILE) data files.	6-11
Table 6-6. Subroutines to return (GET.PLOT) or send (SEND.PLOT) screen plot dateto a 4-pen HPGL-compatible plotter.	6-15
Table 6-7. Subroutine (READOUTS) for returning the 2710 on-screen readouts.	6-16
Table 6-8. Subroutines for saving (GET.SET) and restoring (PUT.SET) settings groups.	6-18

Safety Summary

Refer all servicing to qualified service personnel

The following general safety information is for both operating and service personnel. Other specific warnings may occur throughout the manual even though absent from this summary.

Conformance to Industry Standards

This instrument complies with the following Industry Safety Standards and Regulatory Requirements.

Safety

CSA 556B -- Electrical Bulletin

FM -- Electrical Utilization Standard Class 3820

ANSI/ISA DS82 (1988) -- Safety Standard for Electrical and Electronic Test, Measuring, Controlling and Related Equipment.

IEC 348 (2nd edition) -- Safety Requirements for Electronic Measuring Apparatus.

VDE 0411 -- German interpretation of IEC 348.

UL 1244 (second edition) -- Standard for Electrical and Electronic Measuring and Testing Equipment.

Regulatory Requirements

VDE 0871 Class B -- Regulations for RFI Suppression of High Frequency Apparatus and Installations.

FCC Part 15, Sub-part J, Class A -- Emission requirements for computing equipment.

Mil Standard 461B, Part 7 -- Susceptibility of Test Equipment in Non-critical Ground Areas.

Terms

In This Manual

CAUTION statements identify conditions or practices that could result in damage to the equipment or other property.

WARNING statements identify conditions or practices that could result in personal injury or loss of life.

As Marked on Equipment

CAUTION indicates a personal injury hazard not immediately accessible as one reads the marking, or a hazard to property, including the equipment itself.

DANGER indicates a personal injury hazard immediately accessible as one reads the marking.

Symbols

In This Manual



This symbol indicates where applicable cautionary or other information is to be found.

As Marked on Equipment



DANGER -- High voltage.



Protective ground (earth) terminal.



ATTENTION -- refer to manual.

ⓘ Refer to manual

Power

Power Source

This product is intended to operate from a power source that will not apply more than 250 V rms between the supply conductors or between either supply conductor and ground. A protective ground connection by way of the grounding conductor in the power cord is essential for safe operation.

Grounding the Product

This product is grounded through the protective grounding conductor in the power cord. To avoid electrical shock, plug the power cord into a properly wired receptacle before using the instrument. A protective ground connection is essential for safe operation. Note that no automatic protective grounding is provided during battery operation.

Danger From Loss of Ground

Upon loss of the protective-ground connection, all accessible conductive parts (including knobs and controls that may appear to be insulating) can render an electric shock.

Use the Proper Power Cord

Use only the power cord and connector specified for your product.

Use only a power cord that is in good condition.

CSA certification applies to the spectrum analyzer with CSA-certified power cords only (the power cord shipped

with your instrument and Tektronix Option A4). International power cords (Tektronix Options A1, A2, A3, and A5) are approved only for the country of use, and are not included in the CSA certification.

Refer cord and connector changes to qualified service personnel.

For information on power cords, see the General Information section in the Operators Manual.

Use the Proper Fuse

To avoid fire hazard or equipment damage, use only the fuse of correct type, voltage rating, and current rating for your product (as specified in the Replaceable Electrical Parts list in Volume 2 of the Service Manual). Refer fuse replacement to qualified service personnel.

Operational Precautions

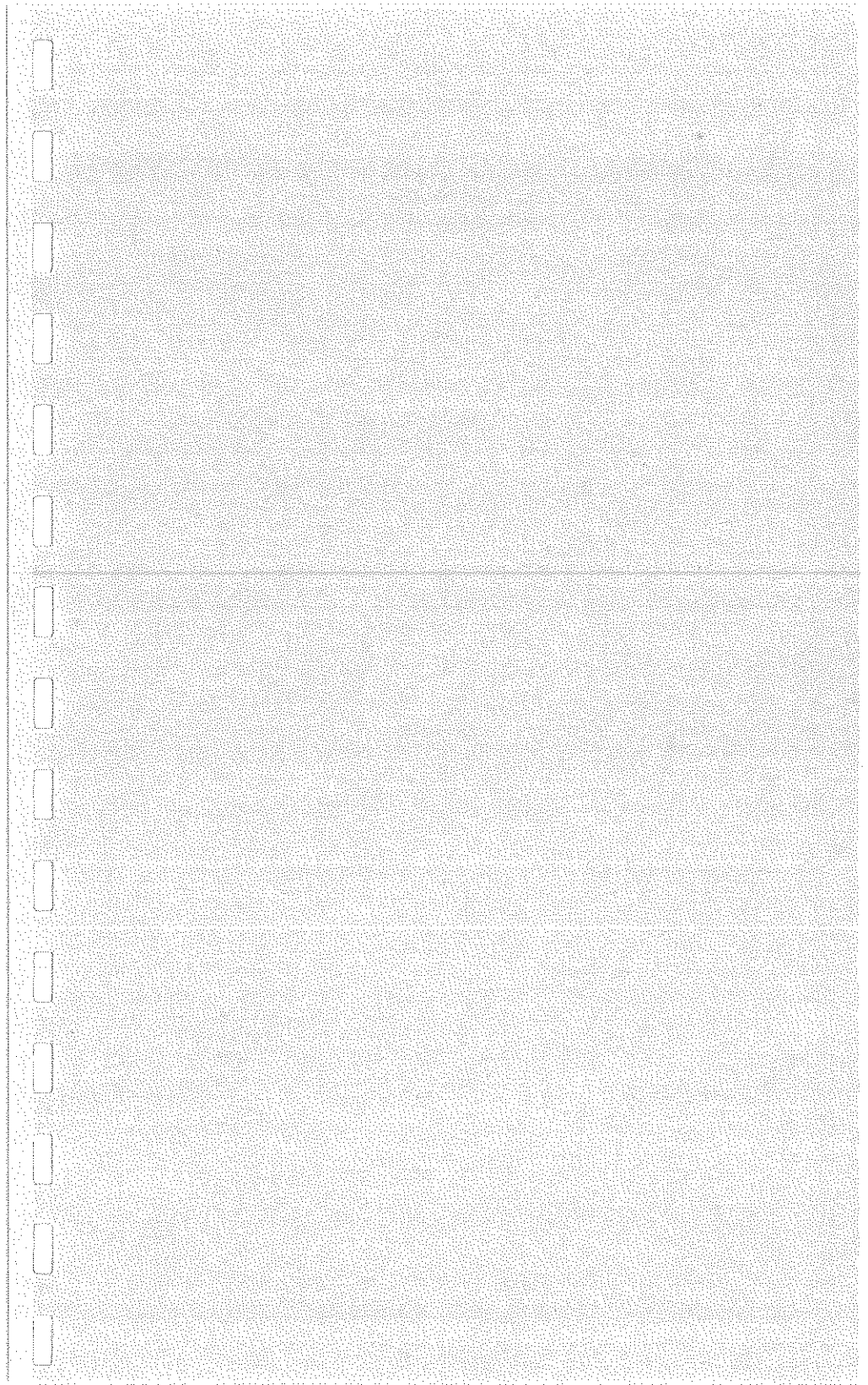
Do Not Operate in Explosive Atmospheres

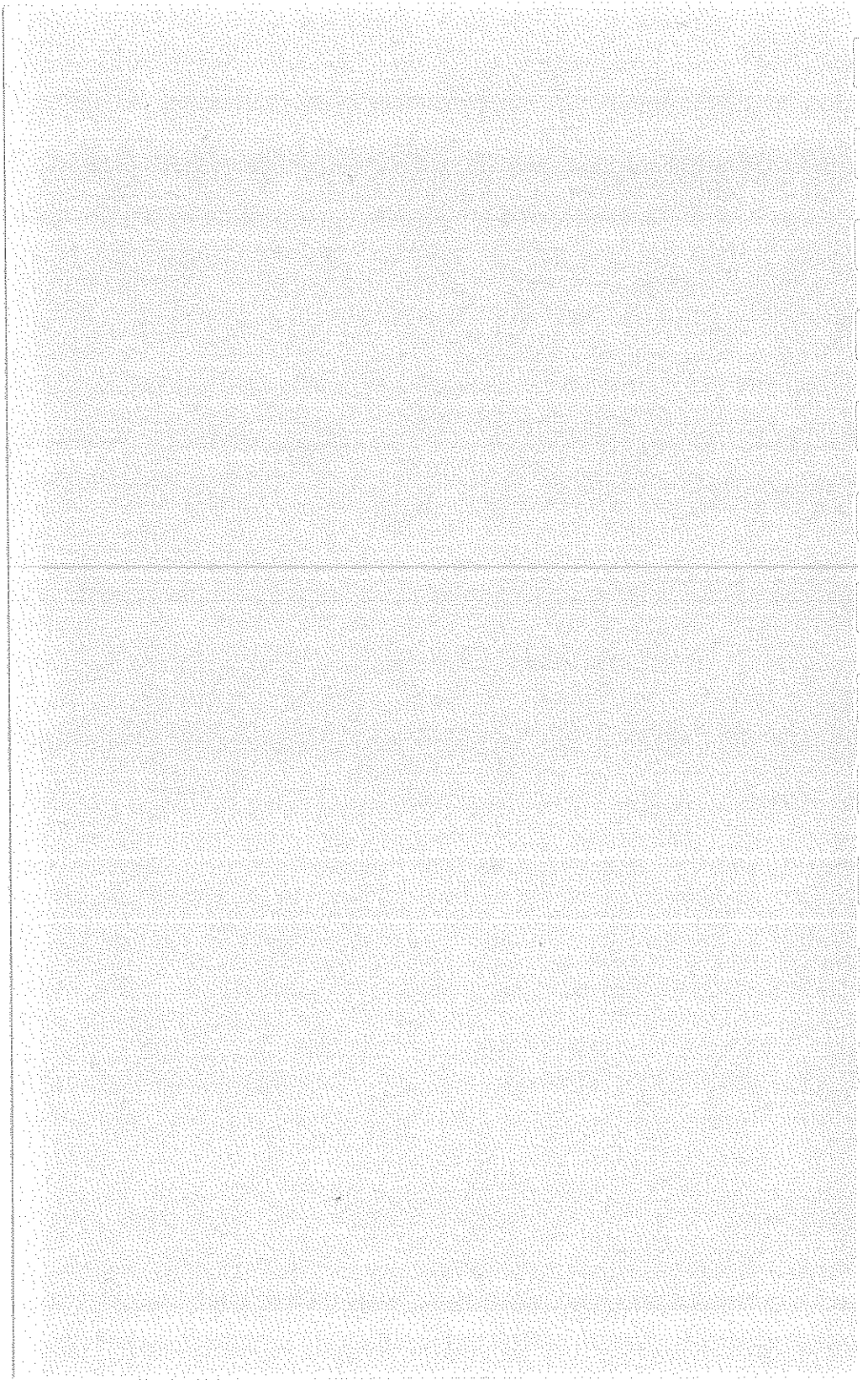
To avoid explosion, do not operate this product in an explosive atmosphere.

Do Not Remove Covers or Panels

To avoid personal injury, do not remove the product covers or panels unless you are qualified to do so. Do not operate the product without the covers and panels properly installed.

Dangerous voltages exist at several points in this product. To avoid personal injury, do not touch exposed connections and components while power is on. REFER ALL SERVICING TO QUALIFIED SERVICE PERSONNEL.





Section 1 Introduction To GPIB Operation

The Tektronix 2710 Spectrum Analyzer allows automated control of instrument functions via the IEEE Standard 488 General Purpose Interface Bus (GPIB). Front panel settings can be controlled (except those intended for local use only, such as INTENSITY) and data can be acquired, transferred, processed, and analyzed remotely. The 2710 follows the Tektronix Interface Standard for GPIB Codes, Formats, Conventions, and Features. This standard promotes ease of operation and, so far as possible, makes this spectrum analyzer compatible with other Tektronix instruments and with GPIB instruments from other manufacturers.

What is the GPIB and How Does It Work?

IEEE Standard 488 establishes electrical levels, connector configuration, and signal protocols for communication between two or more electronic instruments using a common multi-line bus structure. The bus structure, which is known as the GPIB, consists of eight data lines, eight dedicated control signal lines, a shield, and various grounds.

Data are transferred via the eight data lines in a bit parallel, byte serial fashion. That is, the eight bits of a data byte are placed on the eight data lines simultaneously. As soon as they are transferred, the next eight-bit data byte is placed on the lines and transferred. Data can consist of instrument commands and queries, control settings, parameter values, or display information.

The eight control lines are divided into three transfer control (handshake) lines and five interface management lines. Handshaking and interface management are necessary because the bus operates asynchronously, meaning that signals can be generated by one instrument

without regard for what another may be doing or the rate at which the instrument can carry out an operation. For instance, two instruments may try to send information simultaneously, or a high speed instrument may try to send data to a low speed instrument.

Instruments connected to the bus are designated as talker, listener, or both talker and listener. A listener can only receive information over the bus and a talker can only send information. A talker and listener can do both (but not simultaneously). Furthermore, one instrument is usually designated as the system controller. This is generally a computer which determines through software when specific instruments are activated as talkers or listeners. Each instrument is assigned a unique address between 0 and 30, but only 15 instruments can be connected to the bus simultaneously.

Except in the case of abnormal events (see *Status Reporting* in Section 5), here is how an actual data transfer typically takes place:

1. The instrument on the bus that is designated as system controller determines (through operator intervention or program control) that it needs to send a message to one of the other instruments.
2. Using the data and interface management lines, the controller first addresses the desired instrument as a listener. This is called LISTENING an instrument.
3. Normally the instruments on the bus are idle and signal via the handshake lines that they are ready to receive data. The controller then places the first byte of the message indicating the type of information it wants (for instance, the current signal amplitude in the case of the spectrum analyzer) on the bus.
4. The controller then signals, also via the handshake lines, that the data byte is ready.
5. As the listener accepts the data byte, it signals over the handshake lines that it has done so. The controller then removes it from the data lines.
6. The process from step 3 onward is repeated until the entire message has been transferred.

7. The controller indicates that the last data byte has been sent. Depending on the option selected, this is done by signaling over the EOI interface management line simultaneously with the last data byte, or by appending the ASCII codes for carriage return (CR) and line feed (LF) to the end of the message and simultaneously signaling EOI.
8. When the message is complete, the controller normally UNLISTENS the instrument. If a message requires a response, the controller then addresses the instrument as a talker (TALKS the instrument).
9. Now the instrument places the first byte of the response on the data bus and signals that it is ready.
10. After the controller reads the byte, it signals (over the handshake lines) that it has done so and is ready to receive more. The process repeats until EOI is detected, at which point the controller normally UNTALKS the instrument.

This process is carried out behind the scenes by the 2710, the GPIB board in your controller, and a piece of software (generally supplied with the GPIB board) called a device driver; the process is transparent to you. In the following subsections you will learn how to set up your 2710 for GPIB operation. For additional information concerning IEEE 488 and the GPIB, see Appendix A.

What You Need To Operate Over the GPIB

In order to operate the 2710 Spectrum Analyzer over the General Purpose Interface Bus (GPIB), you need the following:

- System controller
- Software device driver
- 2710 equipped with the GPIB option (Option 03)
- Interconnecting cable
- Application software
- (Optional) Printer or Plotter

Figure 1-1 shows a typical small system including both a printer and a plotter.

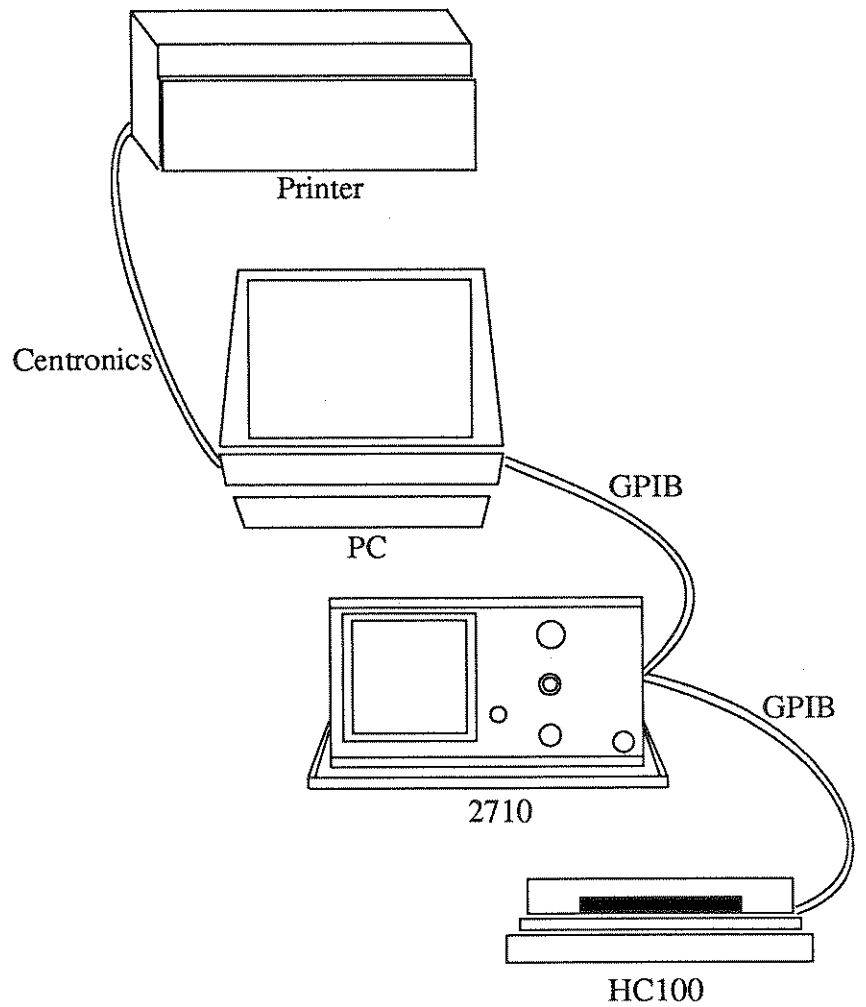


Figure 1-1. Typical small instrument system.

System Controller

The system controller can be any general purpose computer equipped with a GPIB board. Specially built controllers can also be used, but are beyond the scope of this manual. The techniques and programs discussed in this manual are appropriate to the IBM family of PC computers and their function-alike counterparts, which support the MS-DOS, PC-DOS, and OS/2 environments. Your computer must be equipped with a GPIB board to function as a controller. Tektronix supplies the National Instruments PC-2A GPIB board along with associated software and an interconnecting cable as the GURU II package (Tek P/N S3FG100). The board is also available separately as Tek P/N S3FG120.

Software Device Driver

The device driver is a program, usually supplied with the GPIB board by the board's manufacturer, which tells your computer how to access the board. In the case of the National Instruments PC-2 or PC-2A GPIB boards, it is a file named GPIB.COM. A separate program is usually also supplied which enables you to correctly configure the driver by telling it such things as the instrument address and the type of message terminator. The National Instruments program is named IBCONF.EXE.

2710 Equipped With the GPIB Option

Your 2710 must be equipped with Option 03, the GPIB option, to operate over the General Purpose Interface Bus. If your 2710 is not so equipped, contact your local Tektronix field office or the Tektronix factory. If your 2710 is equipped with a Centronics interface (Option 09), it must be removed before Option 03 can be installed. Press [UTIL MENU]/[4]/[9] to see a list of the installed options.

Interconnecting Cable

You must have an appropriate cable to connect the controller to the spectrum analyzer. The cable is supplied as part of the Tektronix GURU II package or can be purchased separately as Tek P/N 012-0991-01 (1 meter) or 012-0630-01 (2 meter).

Application Software

Application software is the program or programs which control and acquire data from the 2710. Using the information in this manual you can construct your own programs. You will, however, need the applications interfacing software supplied by the board manufacturer. In the case of the PC-2/PC-2A board and the QuickBASIC language, these programs have names like QBIB4.OBJ, QBIB4728.OBJ, GPIB.QLB, QBDECL4.BAS, etc. The programs contain, amongst other items, the BASIC device function calls which enable you to communicate easily over the GPIB. The function calls become an integral part of your application programs.

(Optional) Printer or Plotter

Either a printer or a plotter or both can be added to your system to provide hard-copy output. The printer is the preferred instrument for character-based data such as parameter values or instrument settings. The plotter provides superior results when displaying graphical data. A convenient approach is to install a printer on a parallel port of the controller and a GPIB-compatible plotter on the bus. With this approach, you can plot graphical data directly from the 2710 when the controller is not available. See *Setting the Talk Only Option*.

Setting Up For GPIB Operation

Before you can begin GPIB operations, your equipment must be connected, the 2710 and the device driver configured, the device driver installed in controller memory, and the (optional) printer and/or plotter correctly configured.

Connecting the Equipment

If your system consists only of the controller and the 2710, you can simply connect one end of the interconnecting cable to each instrument. If you have more instruments on the bus, you can use a star configuration, a daisy chain configuration, or combination of both. See Figure 1-2 for examples. Up to 15 instruments can be connected at one time. To maintain bus electrical performance, no more than one 2-meter cable per instrument should be allowed, and at least 2/3 of the connected instruments should be turned on.

Configuring the 2710

Turn on the power to the 2710. Then press:

[UTIL MENU]/[4]/[0]/[0]

A menu appears similar to the one shown in Figure 1-3. It enables you to configure the 2710's GPIB parameters.

Placing the 2710 Online

The first item on the menu (item 0) controls the 2710's GPIB online/offline status. After all preparations have been completed and GPIB operations are ready to begin, toggle item 0 until the end of the line indicates ONLINE. The 2710 is then ready to exchange information over the GPIB.

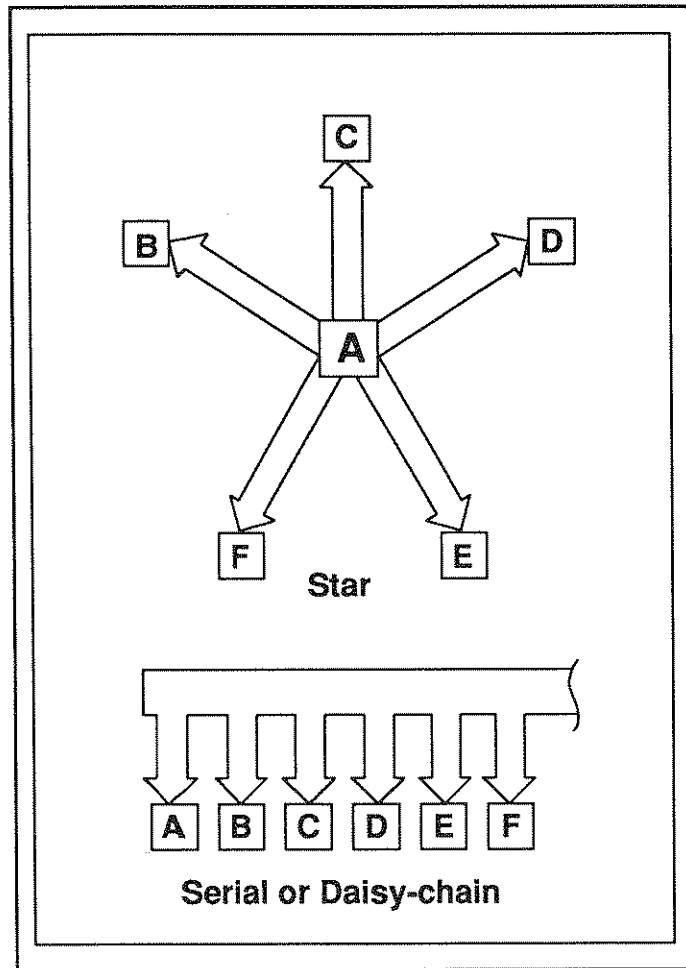


Figure 1-2. Connection examples for multiple instruments.

Setting the GPIB Device Address

You must assign a primary address to the 2710 (the 2710 does not support secondary addresses). The address can have a value from 0 through 30. However, addresses 0 and 30 are usually reserved for system controllers. The address you assign is not critical, but it must not be the same as the address of any other instrument on the bus. Further, you must ensure that the device address you use for the 2710 when configuring the device driver is the same that you assign to the 2710 when you configure it.

To assign the address, select item 1, GPIB ADDRESS, from the GPIB port Configuration Menu. Follow the 2710 on-screen prompts to enter the desired address. If the 2710 is the only instrument on the bus, we suggest you use 1 as the address. The address you set is read immediately by the 2710 and retained in NVRAM until you change it.

GPIB PORT CONFIGURATION	
0 STATUS	ON/OFFLINE
1 GPIB ADDRESS	0 - 30
2 POWER ON SRQ	ON/OFF
3 EOI/LF MODE	LF/EOI
4 TALK ONLY MODE	ON/OFF

Figure 1-3. The 2710 GPIB Port Configuration Menu.

The Power-on SRQ

Normally there is no need to have the 2710 generate an SRQ when it powers up. Therefore, the default setting of item 2, POWER ON SRQ, is OFF. However, if your test sequence requires unpowering the analyzer, you may want to sense the return of power before continuing your test. For these cases you can toggle the POWER ON SRQ to ON by pressing [2].

Setting the Message Terminator

Whenever a message is transmitted over the bus, the instrument sending the message must signify to other instruments on the bus (including the system controller) that the message has been completed. This can be done in one of two possible ways.

- The interface management line named End Or Identify (EOI) is brought to its low state simultaneously with the last data byte that is transmitted.
- The ASCII codes for carriage return (CR) and line feed (LF) are appended to the message string. EOI is still asserted (brought to its low state) simultaneously with the transmission of LF.

All Tektronix instruments and controllers are equipped to use the first option. You should, therefore, toggle item 3 of the GPIB Port Configuration Menu until the end of the line indicates EOI. The LF OR EOI option is included for controllers which do not use the EOI signal line. The option selected is retained in NVRAM until you change it.

Setting the TALK ONLY Option

The 2710 TALK ONLY option (item 4 of the GPIB Port Configuration Menu) must be disabled when the spectrum analyzer is used with a controller, because the analyzer both talks and listens to the controller. Since you will normally use the analyzer with a controller, turn off the TALK ONLY option by pressing [4] until the end of line 4 in the menu indicates OFF. The system controller will determine when the analyzer should be addressed as a talker and when as a listener.

To send the spectrum analyzer output directly to a plotter without the need of a controller, select the TALK ONLY option, and disconnect all instruments except the 2710 and the plotter from the bus. Place the plotter in the listen only mode (usually done with controls on the

plotter). With the talk only and listen only modes selected, you can send screen data directly from the 2710 to the plotter by pressing [UTIL MENU]/[6].

Configuring the Device Driver

Detailed instructions for configuring the device driver should be included with the material you received with your GPIB board. However, if you are using a National Instruments PC-2 or PC-2A board, run IBCONF.EXE. Follow the on-screen prompts and ensure that the .Board Characteristics screen resembles Figure 1-4 or 1-5. Next, create/edit the Device Characteristics screen for a device named "TEK_SA". Make it look like Figure 1-6. Most importantly, ensure that the device address used for the 2710 is the same as that you set up when you configured the 2710, and that the message terminator is set for EOI.

Primary GPIB Address	0
Secondary GPIB Address	NONE
Timeout setting	T30s
EOS byte	00H
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
GPIB-PC Model	PC2
Board is System Controller	yes
Local Lockout on all devices	no
Disable Auto Serial Polling	yes
High-speed timing	no
Interrupt jumper setting	7
Base I/O Address	2B8H
DMA channel	1
Internal Clock Freq (in MHz) ...	8

Figure 1-4. The National Instruments PC-2 Board Characteristics screen from IBCONF.

Primary GPIB Address	0
Secondary GPIB Address	NONE
Timeout setting	T30s
EOS byte	00H
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
GPIB-PC Model	PC2A
Board is System Controller	yes
Local Lockout on all devices	no
Disable Auto Serial Polling	yes
High-speed timing	no
Interrupt jumper setting	7
Base I/O Address	02E1H
DMA channel	1
Internal Clock Freq (in MHz) ...	6

Figure 1-5. The National Instruments PC-2A Board Characteristics screen from IBCONF.

Primary GPIB Address	1
Secondary GPIB Address	NONE
Timeout setting	T30s
EOS byte	00H
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

Figure 1-6. The TEK_SA Device Characteristics screen from IBCONF.

Installing the Device Driver

Before your IBM/MS-DOS computer can transfer information over the GPIB, it must know how to access the GPIB board and the 2710. The device driver tells it how. If you are using a National Instruments PC-2 or PC-2A board, the device driver is a program named GPIB.COM created and modified by another National Instruments program named IBCONF.EXE. If you are using a board from another manufacturer, the appropriate driver should have accompanied your board.

The device driver program must be installed whenever you wish to use the GPIB. Use this procedure:

1. Copy GPIB.COM to your computer's root directory
2. Add the following line to your controller's CONFIG.SYS file. If CONFIG.SYS does not already exist in the controller's root directory, create it (see your DOS manual if you need help creating or modifying files):

```
device=GPIB.COM
```

3. Reboot your controller.

The GPIB device driver is loaded into memory whenever you boot your computer. It then remains in memory until the computer is turned off.

Configuring the (Optional) Printer and/or Plotter

A variety of printers and plotters are available which can be used with your system. We recommend a serial or parallel printer of your choice connected to the appropriate computer port, and a HPGL-compatible plotter connected to the GPIB. This arrangement enables you to send data directly from the 2710 to the plotter when the system controller is unavailable. The Tektronix HC100 plotter is recommended; its four pens provide a useful complement to the 2710's four trace capability.

typically implement routine housekeeping chores such as instrument addressing, handshaking, requesting service, terminating messages, and so on.

The 2710-specific messages are also referred to as device-dependent messages. They are generally understandable by, and meaningful to, only the instrument or class of instruments for which they are designed. The organization of the 2710-specific messages is explained in the next section of this manual, and Section 3 provides a summary of the messages. Section 4 describes the individual messages in detail, and Section 6 provides some programming examples for the National Instruments GPIB/2710 combination working in the QuickBASIC environment.

To send or receive messages, the 2710 is addressed as a talker or listener, depending on whether messages are being sent to or received from it, by the system controller. The GPIB system software provided with your GPIB card automatically addresses the analyzer as a talker or listener depending on the callable subroutine used. Then the device-dependent messages are transferred between the controller and the spectrum analyzer over the data lines of the GPIB as one or more eight-bit bytes of information. The art of controlling the analyzer is the art of efficiently programming these messages.

Readying the Software

After you have completed the set up procedures, your equipment is essentially ready for GPIB operation, but you must still provide the software needed to control the 2710. If you are creating new software, this is usually a 2-step process. First, establish the programming environment. Then create and run the actual control program. If you are using ready-made control software, simply follow the supplier's instructions.

Preparing the QuickBASIC Environment

When creating your own QuickBASIC software, you must ensure that QuickBASIC has the necessary GPIB information. Use this procedure:

1. Copy the files QBIB4.OBJ, GPIB.QLB, GPIB.LIB, BQLB45.LIB, and QBIB4728.OBJ from the National Instrument disk to the QuickBASIC directory.
2. Create the Quick library by typing this command from the DOS command line:

```
LINK /Q QBIB4.OBJ QBIB4728.OBJ,  
GPIB.QLB,, BQLB45.LIB
```

3. To make your QuickBASIC program a stand-alone *.EXE file, you need an additional library file. Type this command from the DOS command line:

```
LIB GPIB.LIB + QBIB4.OBJ + QBIB4728.OBJ;
```

4. Start QuickBASIC using this command:

```
QB /L GPIB.QLB
```

This procedure ensures that the National Instruments GPIB subroutines needed to control devices on the bus are present in the QuickBASIC environment. If you are using another version of QuickBASIC, use the analogous files and procedures indicated in the READ-QB.DOC document file from National Instruments.

A Simple Control Program

Although you will learn more about controlling the 2710 in Sections 4-6, we have provided a very simple program here so you can immediately check out the operation of your system and observe the typical interactions of the controller and 2710. Be aware that the program is very unsophisticated; it contains no error checking and may hang up the controller (requiring you to reboot it) if you enter incorrect or unacceptable commands or queries. It will, however, accept upper- or lower-case entries.

2710 GPIB Programmers Manual

Alternately, you can use the IBIC program supplied with the National Instruments GPIB board. It enables you to communicate with the 2710, but requires that you learn to use a few simple subroutines like IBWRT() and IBRD(). See your National Instruments documentation.

To use the program in Table 1-1, follow this procedure:

1. Start QuickBASIC according to the instructions in the preceding subsection. Enter the program listed in Table 1-1. Be sure to enter the program exactly as written. The 2710 must be named TEK_SA by the IBCONF program.
2. Place the 2710 on line by selecting item 0 from the GPIB Port Configuration Menu (press [UTIL MENU]/[4]/[0]/[0]). The instrument is nominally online, but is not yet handshaking with the controller.
3. Start the program. The computer display should show:

```
2710 SHOULD NOW BE HANDSHAKING  
NDAC SHOULD BE DISPLAYED
```

```
PRESS ANY KEY TO CONTINUE
```

When the 2710 is handshaking with the controller, NDAC (Not Data ACcepted) is displayed at the lower right of the 2710 screen. NDAC is asserted almost all of the time. In fact, it is not asserted only briefly following receipt of a message to indicate that the message has been accepted (see Appendix A).

4. Press any key. The word REMOTE should appear at the lower left of the 2710 screen and the controller should display:

```
2710 SHOULD NOW BE IN REMOTE MODE
```

```
PRESS ANY KEY TO CONTINUE
```

The National Instruments software places the 2710 in

remote mode whenever a message is sent (the message HDR ON was transmitted). Unless the GPIB local lockout command is issued, pressing any 2710 key which alters its measurement status (the menu keys don't, for instance, but selecting an item from the menu may) returns the 2710 to local mode. Press [VID FLTR] twice. The REMOTE message should disappear from the 2710 screen.

5. Press any key. This message should appear:

ENTER MESSAGE TO SEND

Enter the message you want to send, which can be either a command or query. For example, enter this query requesting the 2710 to identify itself:

ID?

6. REMOTE should reappear. You may also see the words TALKER or LISTENER appear momentarily on the 2710. They are displayed when the analyzer enters the indicated mode, but because of timing considerations, don't always appear for short commands, queries, and responses. You should see a response similar to this on the controller:

THE REPLY IS :

ID TEK/2710,V81.1,"VERSION 12.7.89 FIRM WARE", "300HZ,1,10,100KHZ,1MHZ RBW FL TR", "PHASE LOCK", "300KHZ FILTER", "VIDEO MONITOR", "GPIB", "COUNTER", "NVM 12.88";

The actual response depends on the firmware version and options installed in your instrument.

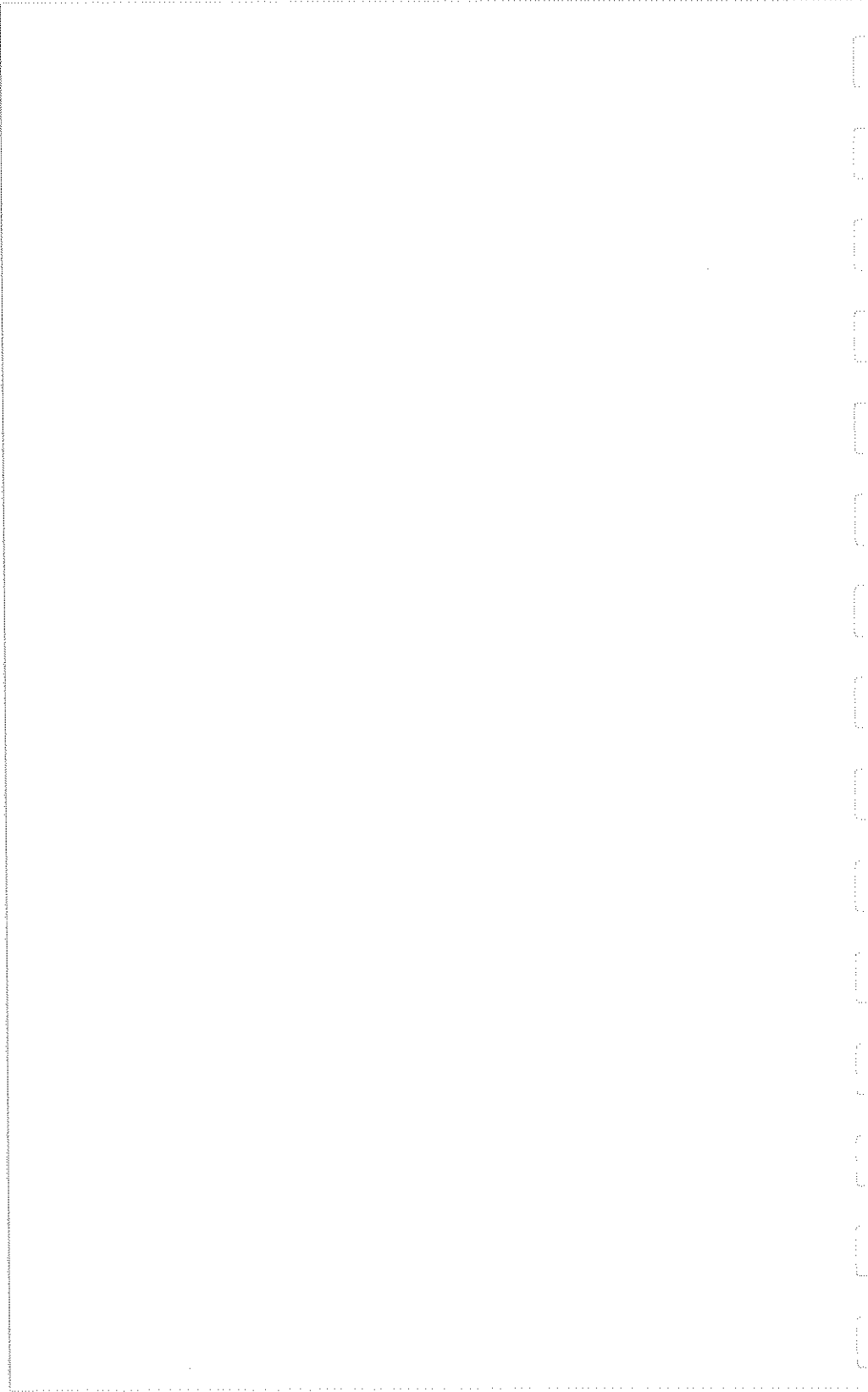
7. You will then be asked:

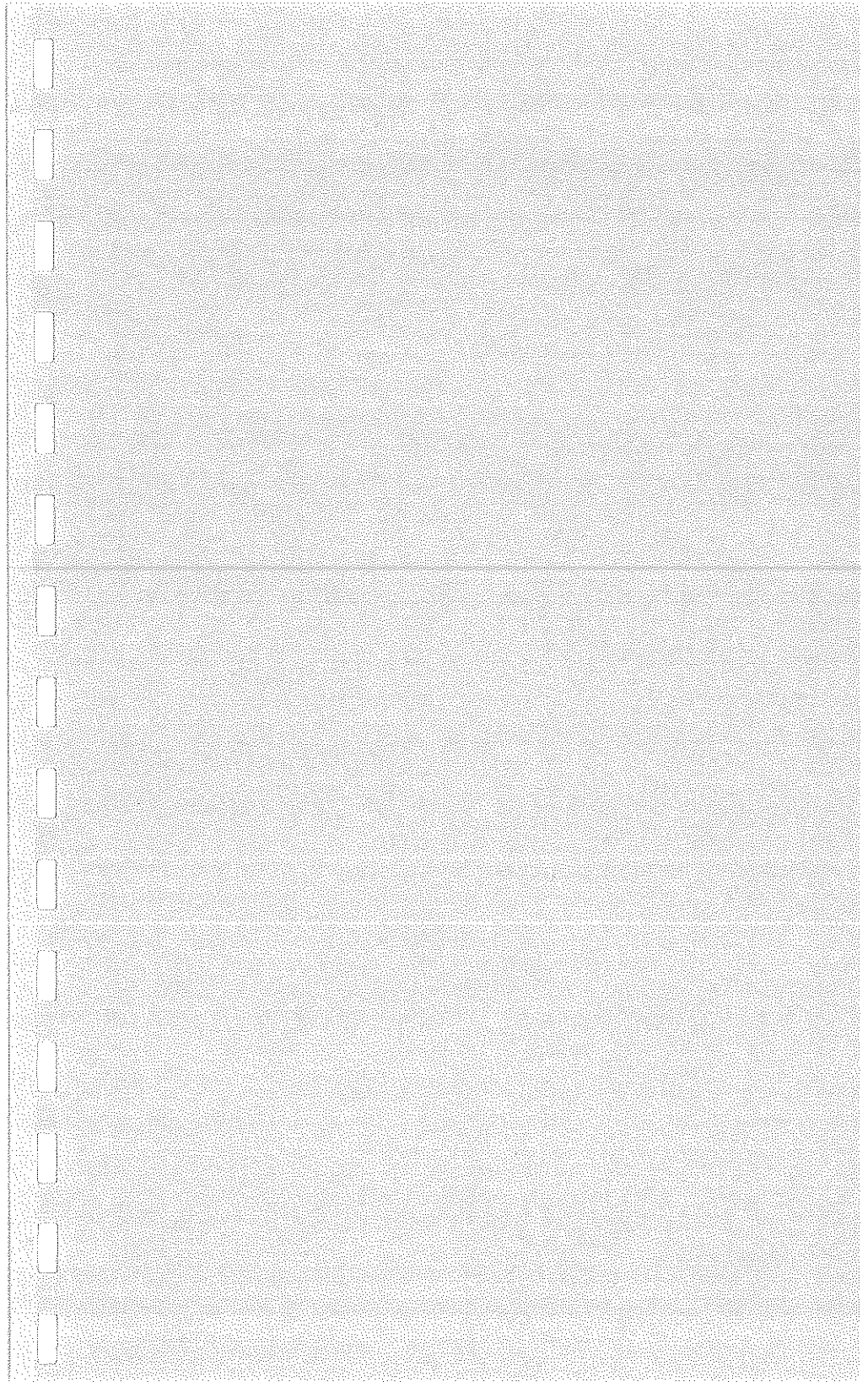
SEND MORE (Y/N)?

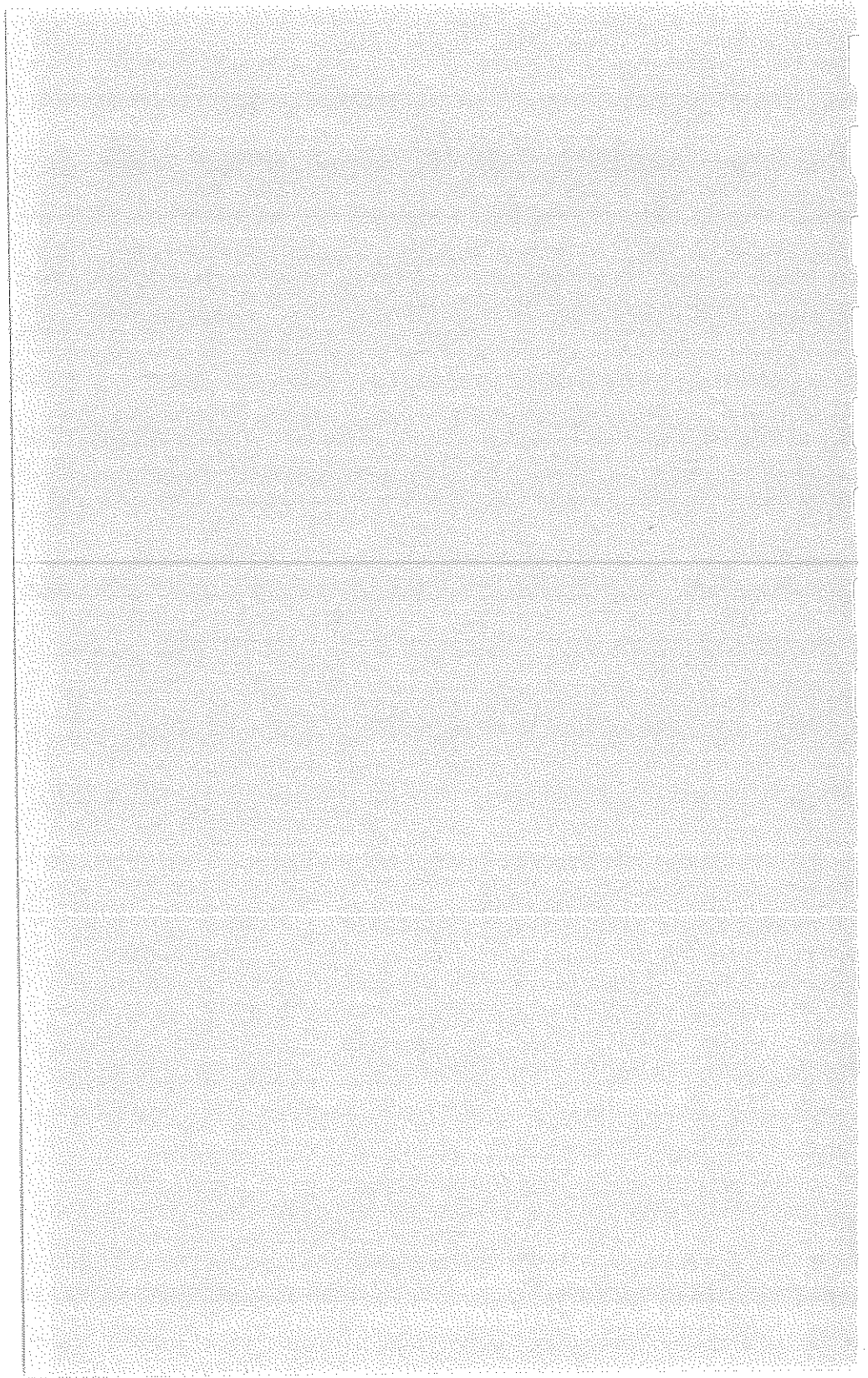
Enter "Y" to send more; other answers end the program.

Table 1-1. A simple 2710 control program.

```
REM $INCLUDE: 'QBDECL4.BAS'
RD$ = SPACE$(3000)
CLS
  CALL IBFIND("GPIB0", BD%)
V% = 0
  CALL IBSRE(BD%, V%)
  CALL IBFIND("TEK_SA", BD%)
PRINT "2710 SHOULD NOW BE HANDSHAKING"
PRINT "NDAC SHOULD BE DISPLAYED"
PRINT:PRINT "PRESS ANY KEY TO CONTINUE":PRINT
DO WHILE INKEY$ = ""
  LOOP
WRT$ = "HDR ON"
  CALL IBWRT(BD%, WRT$)
PRINT "2710 SHOULD NOW BE IN REMOTE MODE"
PRINT:PRINT "PRESS ANY KEY TO CONTINUE"
DO WHILE INKEY$ = ""
  LOOP
SEND.RCV:
CLS
PRINT : PRINT "ENTER MESSAGE TO SEND"
PRINT : INPUT WRT$
  CALL IBWRT(BD%, WRT$)
QUES = INSTR(1, WRT$, "?")
HOLD.TIME = TIMER
DO WHILE TIMER < HOLD.TIME + 1
  LOOP
IF QUES = 0 THEN GOTO MORE
  CALL IBRD(BD%, RD$)
PRINT : PRINT "THE REPLY IS:"
PRINT : PRINT MID$(RD$, 1, IBCNT%)
MORE:
PRINT : PRINT
INPUT "SEND MORE (Y/N)? "; Y$
IF Y$ = "Y" THEN GOTO SEND.RCV
END
```







Section 2 2710-Specific Message Structure

Both generic GPIB messages and 2710-specific messages are exchanged between the system controller and the 2710 over the GPIB.

Generic GPIB messages exercise control over the bus itself and carry out routine system operations such as instrument addressing, handshaking, requesting service, terminating messages, and so on. GPIB messages may be transmitted over the handshake lines or interface management lines (uni-line messages), or the data lines (multi-line messages). The sending and receiving of these messages is usually carried out by the GPIB hardware and software, and is transparent to the system operator or programmer. Additional information about uni- and multi-line messages is included in Appendix A.

The 2710-specific messages, on the other hand, control the measurement and display functions of the spectrum analyzer, and are always (with the exception of the EOI message terminator) transmitted over the data lines. These are the messages that control parameters such as center frequency, span/division, reference level, resolution BW and so on. It is the system programmer's task to efficiently compile a series of these messages in a script designed to implement specific tests and measurements. The script, which is written in a conventional computer language and embodies specific 2710 commands and queries, is called a control program.

What Is a Message ?

A 2710-specific message consists of three or more 8-bit bytes of information transferred between the 2710 and the system controller. Each byte represents an ASCII character or binary data. A message may be an input message or an output message, and it may contain one or more message units.

For instance, here is a message from Mary to John:

John

dinner - put in oven; washing machine - start;
bank - withdraw how much?; cat - let her in

Bye bye

His response might look like either of these:

Mary	Mary
Withdraw - \$100	\$100
Bye bye	Bye bye

These messages both have salutations (John and Mary) which are similar to the GPIB device addresses. Both messages consist of one or more message units. Mary's message to John has four units, one of which is a query and so indicated by a question mark. Each of the message units begins with a "header" describing what it's about (dinner, cat, etc.) separated from its object or argument by a dash -- the argument delimiter. The message units themselves are separated or delimited by semicolons. John's message to Mary consists of a single response indicating how much she should withdraw. If John thinks Mary will remember her own question, he may reply with only "\$100". However, to further remind her he may answer "Withdraw \$100". As you will soon discover, the latter form is equivalent to receiving a response from the 2710 with HDR ON. Both messages close with a message terminator (Bye bye).

The 2710-specific GPIB messages are similarly constructed. The following definitions clarify the structure.

Input Message

An input message is one or more message units along

with any necessary message unit delimiters and a message terminator transmitted from the controller to the spectrum analyzer (input to the analyzer).

Output Message

An output message is one or more message units along with any necessary message unit delimiters and a message terminator transmitted from the spectrum analyzer to the controller (output from the analyzer).

Message Unit

A message unit is a single command or query, or a single response.

Message Unit Delimiter

A semicolon (;) must be used to delimit or separate message units in a message. Its use after the last message unit is optional except in the case of responses when the analyzer always appends a message unit delimiter as the last data byte. However, you may optionally substitute the line feed character for the semicolon in the case of responses (see the MSGDLM command). The line feed character is not to be confused with the optional message terminator discussed later.

Message Terminator

Messages can be terminated in two ways:

The EOI interface management line can be brought low (asserted) simultaneously with the last byte of the message.

The ASCII codes for carriage return (CR) and line feed (LF) can be appended to the end of the message and the EOI interface management line can be asserted simultaneously with transmission of the LF character.

Tektronix instruments all assert the EOI line. The CR-LF terminator option is provided for instruments which do not use the EOI line. If you are using such an instrument, select the CR-LF option (see *Setting the Message Terminator* in Section 1). Terminator control is handled automatically by the GPIB hardware and software.

Command

A command consists generally of a command mnemonic or header, header delimiter, argument(s), and argument delimiter. However, some commands have no arguments, or only one argument, and hence, may not require header or argument delimiters.

Query

A query consists of a query mnemonic or header, a question mark, header delimiter, and argument. However, many queries do not require an argument.

Response

A response consists of an optional response mnemonic or header, header delimiter, argument(s), and argument delimiter.

Mnemonic or Header

A header is a short name associated with each command, query, or response (viz., *FREQ*, *REF*, *MAR*, etc., etc.). So far as possible, headers are chosen to be mnemonic in nature, the name reminding you of the function.

Header Delimiter

Command headers, response headers, and the question mark following a query header are delimited or separated from any following arguments by a header delimiter which is a space. The space is optional if there are no arguments.

Argument

An argument is the value(s) that a given command, query, or response transfers to or from its associated 2710 setting(s). For instance, in the command `FREQ 200 MHZ`, the value of 200 MHZ is transferred to the center frequency setting. Arguments may be numbers (with or without units), characters, strings, or linked with a colon (:). The argument of some waveform commands and responses can also be a block of binary data.

Digit

A digit is one of the Arabic figures 0-9.

Number Argument

Number refers to a decimal number consisting of one or more digits. Three number formats are possible:

The nr1 format is an integer (no decimal point).

The nr2 format is a floating point number (decimal point required).

The nr3 format is an integer or floating point number in scientific notation ($2.0E+3$ or $2.000E+3$, for example, instead of 2000).

Units

Engineering units can be appended to certain number arguments. Except in the case of dB's, only the first letter of the units is used, the remainder being determined by context. For instance, the command `FREQ 10 M` is the same as `FREQ 10 MHZ`, the M being interpreted as MHz. On the other hand, the M is interpreted as msec in the command `TIME 10 M`. With commands like `REFlvl`, the entire dB unit (DBMV, DBM, etc.) must be used to avoid confusion. You can place as many spaces as desired between the number and

its units.

Note that responses with numerical arguments DO NOT append units to the argument; you must either keep track of the units yourself or use a query which specifically responds with the units currently in use (see, for instance, the RUnit? query).

Character Argument

A character argument consists of one or more letters usually expressing a word or mnemonic. For instance, ON and OFF are arguments for commands that control 2710 functions like ARES (auto resolution BW).

String Argument

String arguments consist of one or more characters, including space, enclosed in quotation marks. They are used with commands like TITLE to convey messages meant to be displayed or plotted. If you intend the quotation mark itself to be part of the message, then double quotation marks (") must be used.

Link Argument

Link arguments provide a method of passing multiple but related arguments. Linked arguments are separated by a colon (:). An example is the VRTdsp command in which you set both the vertical display mode and its related scale factor. For instance, the command VRT LOG:5 selects a logarithmic display at 5 dB per division.

Binary Block Argument

A binary block argument is a sequence of binary numbers preceded by the ASCII code for percent (%) and a two-byte binary integer representing the number of binary numbers plus one (the extra byte is the checksum) and followed by the checksum. The checksum provides an error check of the binary block transfer.

Argument Delimiter

A comma (,) must be used to delimit or separate multiple arguments in a message unit, but it should not be used as the last character in a message.

Message Buffering

The spectrum analyzer buffers each input message it receives, but begins processing the message as it is received; it does not wait for the terminator. The instrument remains busy until it is done executing the commands in its input buffer, unless the process is stopped by the DCL (Device Clear) or SDC (Selected Device Clear) GPIB messages. If an error is detected while transferring a command or query, the rest of the message (up to the the message terminator) is thrown away.

Output data are ready following execution of each query and are passed to the spectrum analyzer's output buffer prior to transmission over the bus. The spectrum analyzer begins transmission of an output message after it is addressed as a talker and as soon as the data become available, but the response terminator is not sent until the command terminator is seen, in case there are more queries in the input message.

Output continues until the end of the information in the output buffer is reached or until it is interrupted by a DCL (Device Clear), UNT (Untalk), or IFC (Interface Clear) GPIB message. If the spectrum analyzer is interrupted before the buffer is cleared, the analyzer will resume output if it is readdressed as a talker. The buffer is cleared by the DCL message, or by the SDC message. If not interrupted, the spectrum analyzer terminates the output according to the selected message terminator (EOI or CR/LF/EOI).

Message Format

Messages are formatted along structural lines, each message consisting of one or more message units separated by semicolons (an exception: line feeds can be optionally selected as message unit delimiters in responses -- see the MSGDLM command).

Message

Each input or output message can be represented as:

Message Unit 1;[Message Unit 2];...:[Message Unit N][;]

message delimiters
(optional at end of message)

Items enclosed in square brackets denote optional quantities. Each message unit in turn consists of an individual command, query, or response along with any necessary arguments and delimiters. Simple messages may consist of individual commands or queries, such as these:

```
FREQ 200MHZ  
SAVE A:ON  
CURVE?
```

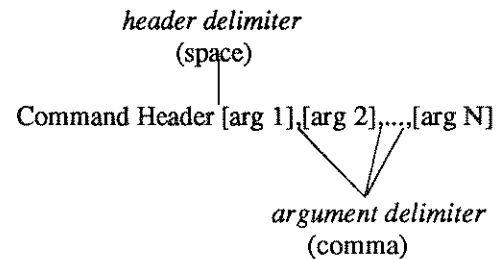
However, you can combine message units to form more complex messages like this:

```
FREQ 200 MHZ;SAVE A:ON;CURVE?
```

Notice that commands and queries can be mixed in a single message.

Commands

We represent a command like this:



Although multiple arguments are shown, it is worth remembering that the majority of commands have only a single argument. Some specific commands look like this:

```
SPAN 5.5E+6
REFLVL -12 DBMV
VRTDSP LIN:100MV
RESBW 1.0M
SAVE A:ON,B:ON,C:OFF
```

These examples illustrate several characteristics of command formatting:

1. Requirement for a header delimiter (space) following the header.
2. Variable number formatting: arguments are expressed as integers (-12 and 100), floating point numbers (1.0) and in scientific notation (5.5E+6).
3. Absence of units: the appropriate units are inferred from the command header. For instance, TIME implies seconds; FREQ, RESBW, SPAN, etc. imply Hz, and so forth. Thus, 5.5E+6 in the FREQ command implies 5.5×10^6 Hz or 5.5 MHz.
4. Use of a space between an argument and its units. Use of a space between an argument and its units is optional.
5. Shortened form for the units (M instead of MHZ in RESBW): only the first letter of the unit is read. The

VIEW?

VIEW WATERFALL:OFF,A:ON,B:OFF,
C:OFF,D:ON,MINUSA:OFF;
WATERFALL:OFF,A:ON,B:OFF,
C:OFF,D:ON,MINUSA:OFF;

MAMPL? DELTA

MAMPL DELTA:18.5;
18.5;

Headers

Thus far we've written headers in capital letters, but the 2710 understands lower-case letters (or a mixture of upper- and lower-case) just as well. Furthermore, we've used the long form of the command and query headers. Most (but not all!) command and query headers can be written using as few as the first three letters. That is, FRE means the same as FREQ and TTLM means the same as TTLMODE. As you become more familiar with the commands and queries, you'll find the shortened forms quicker to use. However, the long forms are less ambiguous and are always used in response headers.

Throughout the remainder of this manual we'll continue to use the long form for headers, but we will print the required letters in capitals and the optional characters of the longer form in lower-case letters. For example:

FREq
VRTdsp
MAMP!
CALSig

You can use any of the possible forms between the required short form and the optional long form. That is, VRT, VRTd, VRTds, and VRTdsp are all acceptable header forms for the vertical display command or query.

Space

The space character is used to separate a command or response header from its first argument, or to separate the question mark following a query header from its argument. The space is optional when there are no arguments.

Comma

Commas are used to separate or delimit multiple arguments in commands and responses. They should not be used elsewhere.

Semicolon

Semicolons are normally used to separate or delimit multiple message units in a single message. They may also (optionally) follow the last argument of a command or query. They should not be used elsewhere. The line feed character can be optionally substituted for the semicolon.

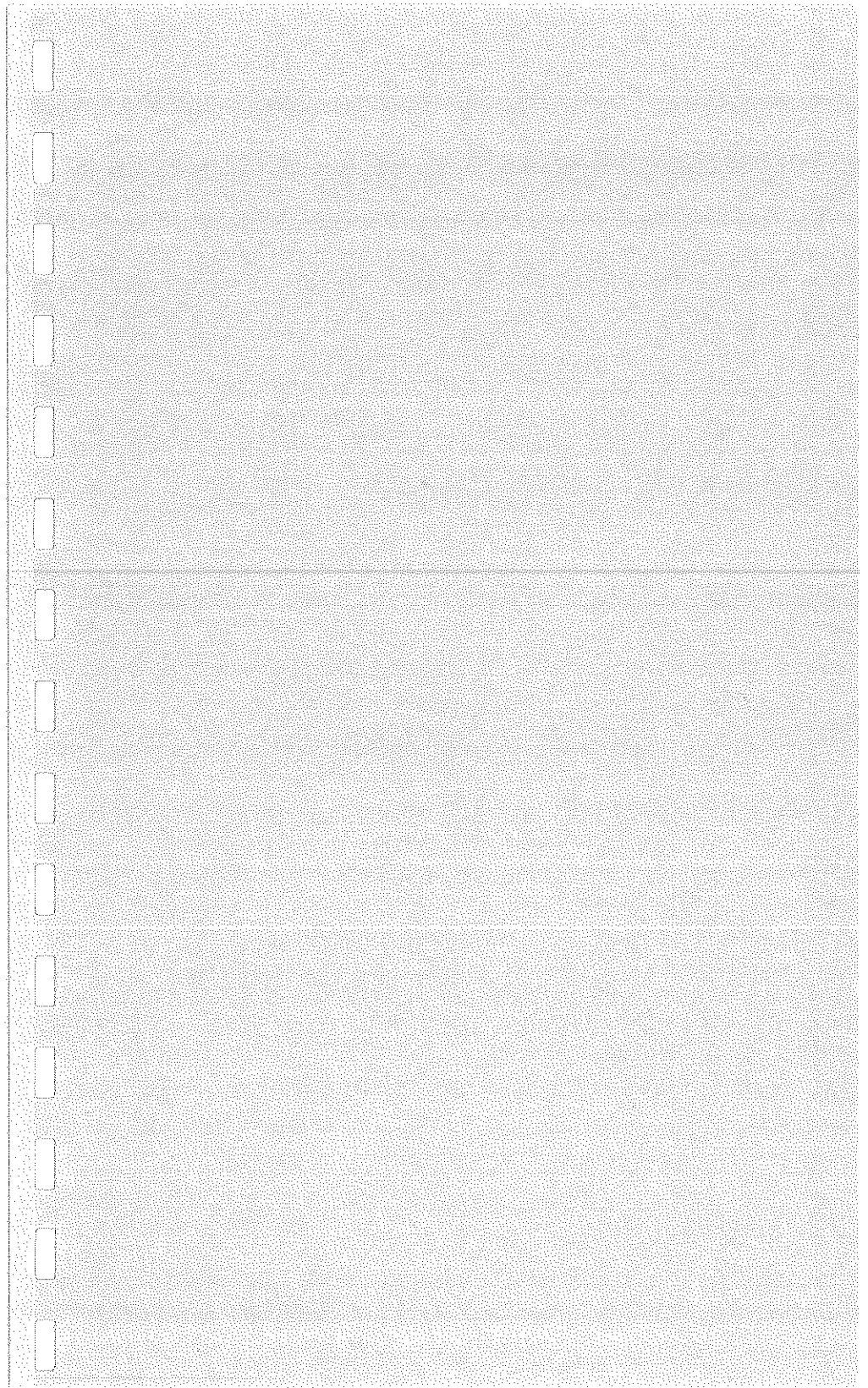
Colon

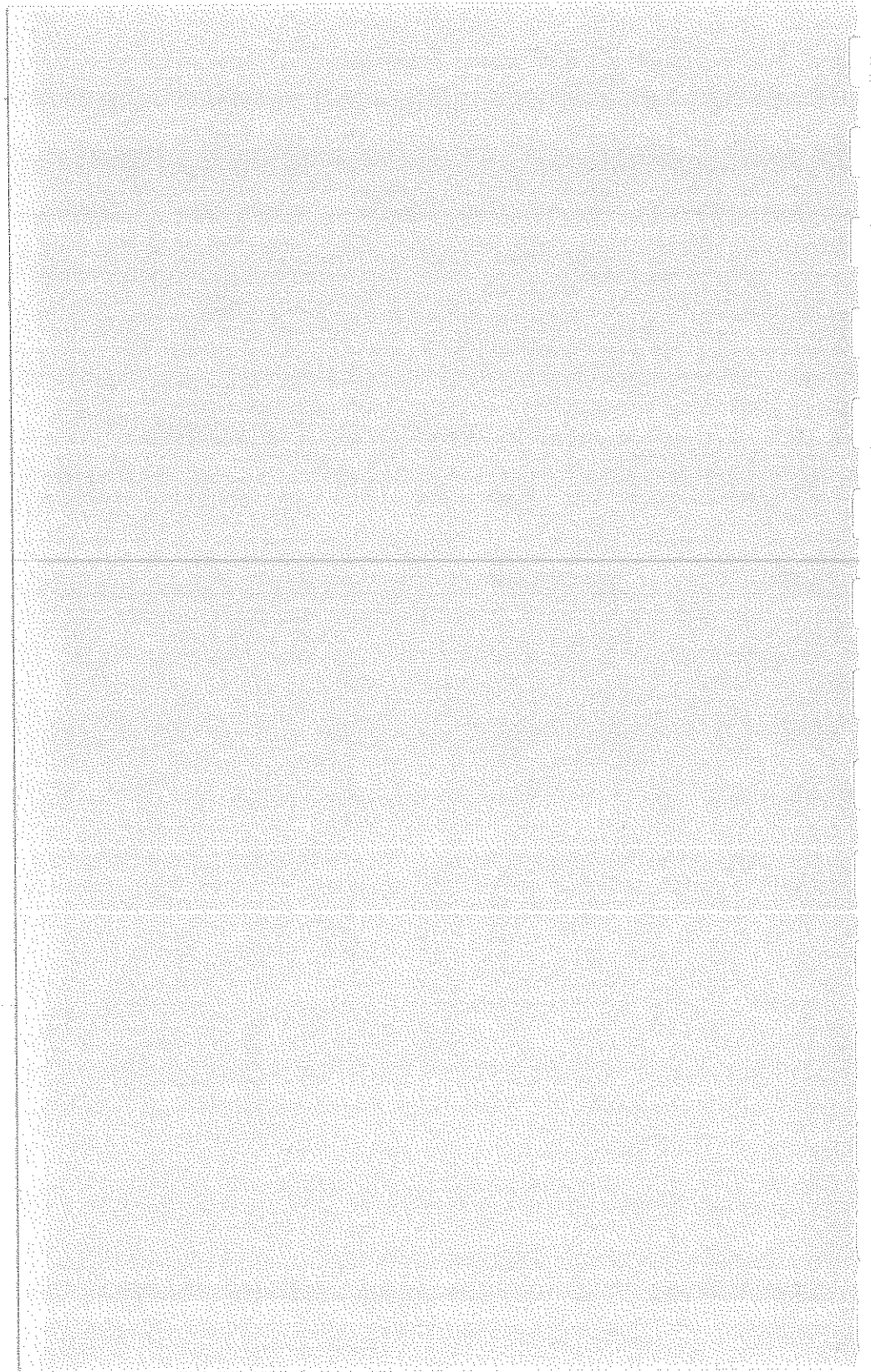
Colons are used to connect the two parts of a linked argument. They should not be used elsewhere.

Line Feed

The line feed character can be used instead of a semicolon to delimit message units in a single message. The 2710 will substitute line feeds for semicolons in its responses when MSGdlm is set to LF. Line feed can also be used as a message terminator with controllers not supporting the GPIB EOI protocol. The two uses are separate and not exclusive.







Section 3 2710-Specific Commands and Queries

To make accessing and using the various 2710-specific commands and queries more convenient to a range of users from the occasional to the frequent, we've listed them in several ways. The level of detail increases as you progress through the manual.

Table 3-1 is intended as a quick reference to jog your memory should you forget the name or header for a particular command. Tables 3-2 through 3-16 show how the GPIB commands are related to the 2710 front panel controls and menus.

Detailed descriptions of the commands and queries are put off until the next section, which contains a complete alphabetical list of all the 2710 commands and queries, including discussion, syntax examples, data formats, and so on

Capital letters in a header indicate the minimum number of letters that must be supplied for the 2710 to recognize the header. The lower-case letters indicate optional additional letters which may be used to make the header less ambiguous to a programmer or operator. In no case will the 2710 accept headers which contain letters other than those indicated.

The Command/Query List

Table 3-1 displays all of the commands available for controlling the 2710 Spectrum Analyzer. It is a short alphabetic list intended to show the experienced user the correct form of the individual command and query headers. Once you are familiar with what each header does, leaving the manual open to Table 3-1 is a handy way to have the mnemonics for each command or query readily available while you are programming.

Table 3-1. List of 2710-specific commands and queries.

ACQmode	DETEctor	MAMpl?
ACQmode?	DETEctor?	MARker
AREs	DIR?	MARker?
AREs?	DIScor	MEMory?
ARFatt	DIScor?	MEXchg
ARFatt?	DLIne	MFReq
ATBI?	DLIne?	MFReq?
ATHrhd	DLLimit	MKTime
ATHrhd?	DLLimit?	MKTime?
AVDest	DLValue	MLFtnxt
AVDest?	DLValue?	MMAx
AVG	DSRc	MNHld
AVG?	DSRc?	MNHld?
AVMode	EOS	MPOS?
AVMode?	EOS?	MRGTnxt
AVNum	ERAsE	MSGdlm
AVNum?	ERr?	MSGdlm?
BWMode	EVENt?	MSTep
BWMode?	FILE	MTUNE
BWNum	FILE?	MVPos?
BWNum?	FINe	MXHld
BWResult?	FINe?	MXHld?
CALSig	FOFFset	MXRlvl
CALSig?	FOFFset?	MXRlvl?
CENsig	FOMode	MXSpn
CFSF	FOMode?	MXSpn?
CFSF?	FREQ	NNBw
CMEas	FREQ?	NNBw?
CNBw	GRAt	NNMode
CNBw?	GRAt?	NNMode?
CNMode	HDR	NNResult?
CNMode?	HDR?	NORM
CNResult?	HELp?	NORM?
CNTtrak	HRAmpl	PKHeight
CNTtrak?	ID?	PKHeight?
COUnt?	IMPCor	PLLmode
CREs	IMPCor?	PLLmode?
CREs?	INIT	PLOT?
CURve	KEY	POFset
CURve?	LRAmpl	POFset?

PRDouts?	STEp	TVLine
PREamp	STEp?	TVLine?
PREamp?	STOre	TVLMode
PROTset	STPinc	TVLMode?
PROTset?	STPinc?	TVLStd
PSTep	STStop	TVLStd?
PTYPE	TABLE	VDMode
PTYPE?	TABLE?	VDMode?
RECall	TAMpl?	VFEnab
REDout	TEXT	VFEnab?
REDout?	TEXT?	VFMode
REFlvI	TFReq?	VFMode?
REFlvI?	TGEnab	VIDflt
RESbw	TGEnab?	VIDflt?
RESbw?	TGLevel	VIEW
RFAtt	TGLevel?	VIEW?
RFAtt?	TGMan	VMAnttbl
RLUnit	TGMan?	VMAnttbl?
RLUnit?	TGOMode	VMDEst
ROFset	TGOMode?	VMDEst?
ROFset?	TGOOffset	VMDIst
ROMode	TGOOffset?	VMDIst?
ROMode?	TGTMode	VMMkrunit
RQS	TGTMode?	VMMkrunit?
RQS?	TGTRack	VMOnitor
SAVe	TGTRack?	VMOnitor?
SAVe?	THRhd	VPolarity
SET?	THRhd?	VPolarity?
SGErr	TIME	VRTdsp
SGErr?	TIME?	VRTdsp?
SGSrCh	TIMMode	VSync
SGTrak	TIMMode?	VSync?
SGTrak?	TITLE	WAIt
SIGswp	TITLE?	WAVfrm?
SIGswp?	TMOde	WFMpre
SPAn	TMOde?	WFMpre?
SPAn?	TOPsig	ZERosp
SSBegin	TRigger	ZERosp?
SSBegin?	TRigger?	
SSEnd	TTLMode	
SSEnd?	TTLMode?	
SSResult?	TUNE	

This page is intentionally blank.

Functional Grouping of Commands and Queries

Tables 3-2 through 3-16 show how the various commands and queries available for programming the spectrum analyzer are related to the 2710 menus or front panel function blocks. An illustration of the function block or menu is shown. The command is then placed adjacent to the feature which it controls or affects. Major groupings include in order:

- Frequency/markers function block
- Marker/Frequency Menu
- Span and Resolution BW function blocks
- Vertical Display and Reference Level function blocks
- Input Menu
- Sweep/trigger function block
- Sweep/Trigger Menu
- Display Storage function block
- Display Menu
- Application Menu
- Utility Menu
- Detector/Generator Menu
- Curve and Waveform commands
- System-related commands
- Miscellaneous commands

Table 3-2. Frequency/marker front panel commands.

HEADER	FUNCTION
CMEas	Perform a center measure.
COUnt?	What is the counter reading?
FREq	Set the start or center frequency.
FREq?	What is the start or center frequency?
MARKer	Turn one or both markers on and off.
MARKer?	What is the current marker status?
MFReq	Set the marker frequency.
MFReq?	What is the frequency of either or both markers?
MLFtnxt	Move the marker to the next signal peak left.
MMAx	Move the marker to the highest data point on screen.
MRGTnxt	Move the marker to the next signal peak right.
MSTep	Equivalent to turning the knob 1 click to the left.
PSTep	Equivalent to turning the knob 1 click to the right.
SGTrak	Turn signal tracking on and off.
SGTrak?	Is signal tracking on or off?
TUNe	Change frequency.

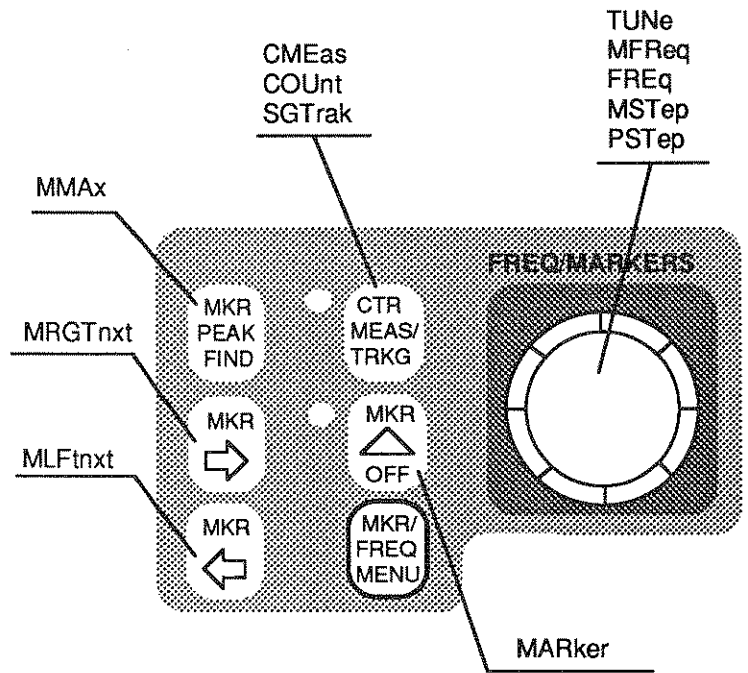


Figure 3-1. The frequency/marker front panel commands.

Table 3-3. Marker/Frequency Menu commands.

HEADER	FUNCTION
ATHrhld	Turn the auto threshold on and off.
ATHrhld?	Is the auto threshold on or off?
CENsig	Set frequency to the marker frequency.
CFSF	Select center or start frequency.
CFSF?	Is center or start frequency being used?
CNTtrak	Turn counter on and off during signal track.
CNTtrak?	Is counter on or off during signal track?
CREs	Set the counter resolution.
CREs?	What is the counter resolution?
FOFset	Set the frequency offset.
FOFset?	What is the frequency offset?
FOMode	Turn frequency offset mode on and off.
FOMode?	Is frequency offset mode on or off?
FREq	Set the center or start frequency.
FREq?	What is the center or start frequency?
HRAmpl	Move the marker to the next higher amplitude peak.
LRAmpl	Move the marker to the next lower amplitude peak.
MAMpl?	What is the amplitude of either or both markers?
MEXchg	Exchange markers.
MKTime	Set the marker time in zero span mode.
MKTime?	What is the time of either or both markers?
MTUNE	Change the marker frequency by a specified amount.
MPOs?	What is the hor. position of either or both markers?
MVPos?	What is the vert. position of either or both markers?
SPAn	Set the span/div.
SPAn?	What is the span/div?
STEp	Set the frequency increment step size.
STEp?	What is the frequency increment step size?
STPinc	Set the type of frequency increment.
STPinc?	What type of frequency increment is being used?
STStop	Set start and stop frequencies to marker frequencies.
TABLE	Select tabular tuning table.
TABLE?	What tabular tuning table is selected?
TAMpl?	What is amplitude of tracked signal?
TFReq?	What is frequency of tracked signal?
THRhld	Replace the auto threshold with the specified value.
THRhld?	What is the threshold value?

Table 3-3. (continued)

HEADER	FUNCTION
TMOde	Select the knob function.
TMOde	What is the knob functin.
TOPsig	Change the reference level to the marker amplitude.

MARKER/FREQUENCY MENU			
0	FREQUENCY ENTRY	96.000MHZ	FREq
1	SPAN/DIV ENTRY	1.0000MHZ	SPAN
2	KNOB FUNCTION	MARKER	TMOde
3	MARKER TO REFERENCE LEVEL		TOPsig
4	MOVE MARKER TO NEXT PEAK		LRAmpl, HRAmpl
5	TRANSPOSE MARKERS		MEXchg
6	MARKER START/STOP		STStop
7	FREQUENCY START/STOP		STStop
8	TUNING INCREMENT	AUTO	STPinc
9	SETUP TABLE		
0	CENTER/START FREQ	CENTER	CFSF
1	THRESHOLD	AUTO -16.3DBM	THRhld
2	COUNTER RESOLUTION	1HZ	CREs
3	PROGRMD TUNING INC		STEp
4	TABULAR TUNING TABLES		TABLE
5	FREQ OFFSET	0.000HZ	FOFset
6	FREQ OFFSET MODE	OFF	FOMode

MAMpl?, MKTime?, TAMpl?, and TFReq? return on-screen measurement parameters. MPOs? and MVPos? have no direct 2710 analog.

Figure 3-2. The Marker/Frequency Menu commands.

Table 3-4. Span and resolution front panel commands.

HEADER	FUNCTION
AREs	Turn AUTO resolution BW on and off.
AREs?	Is AUTO resolution BW on or off?
MXSpn	Turn MAX SPAN mode on and off.
MXSpn?	Is MAX SPAN on or off?
RESbw	Select resolution BW filter.
RESbw?	What is the resolution BW?
SPAn	Select the frequency span per division..
SPAn?	What is the frequency span?
VFEnab	Turn video filter on and off.
VFEnab?	Is video filter on or off?
ZERosp	Turn ZERO SPAN on and off.
ZERosp?	Is ZERO SPAN on or off?

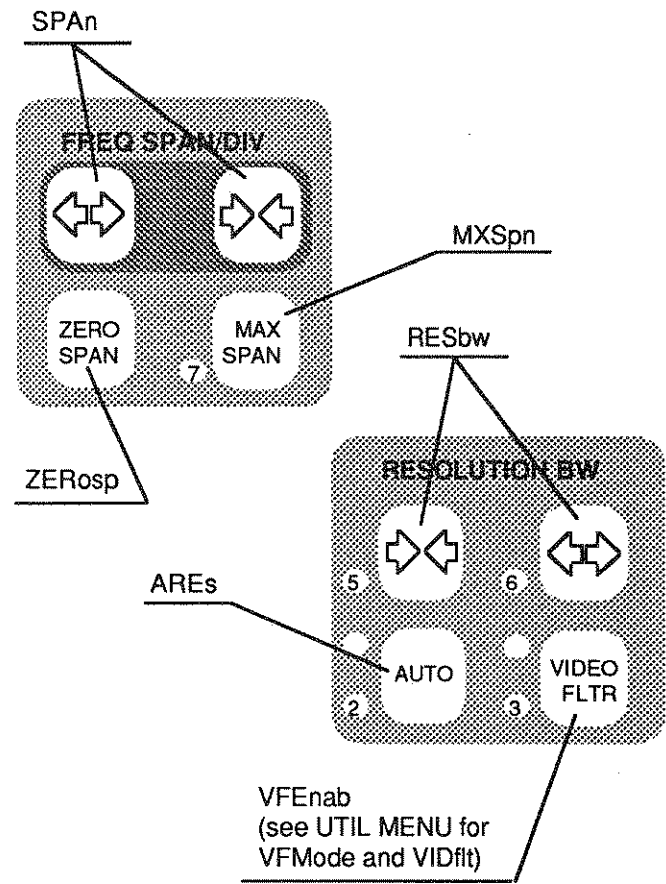


Figure 3-3. Span and resolution front panel commands.

Table 3-5. Vertical scale and resolution BW front panel commands.

HEADER	FUNCTION
FINE	Selects 1 dB or 10 dB reference level steps.
FINE?	Is the reference level step 1 dB or 10 dB?
REFlvl	Set/increment/decrement reference level.
REFlvl?	What is the reference level?
VRTdsp	Select the vertical scale factor.
VRTdsp?	What is the vertical scale factor?

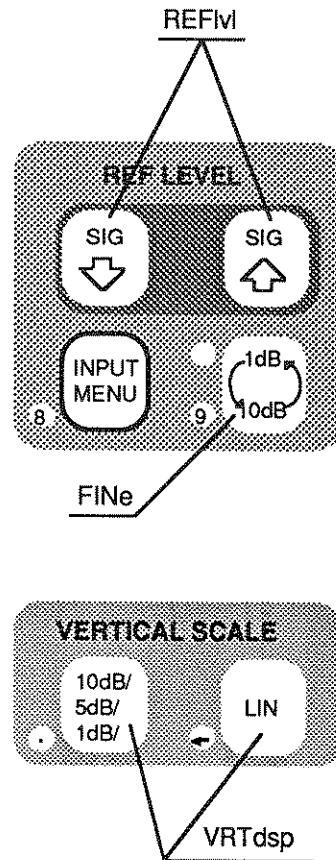


Figure 3-4. Vertical scale and resolution BW front panel commands.

Table 3-6. Input Menu commands.

HEADER	FUNCTION
ARFatt	Turn auto RF attenuation on and off.
ARFatt?	Is auto RF attenuation on or off?
ATBl?	Provide a listing of an antenna correction table.
CALsig	Turn the internal calibration signal on and off.
CALsig?	Is the internal calibration signal on or off?
IMPCor	Corrects indicated amplitude for 50/75 ohm source.
IMPCor?	Amplitude corrected for 50 or 75 ohm source?
MXRlvl	Select 1 st mixer level.
MXRlvl?	What is 1 st mixer level?
PREamp	Turn the preamp on and off.
PREamp?	Is the preamp on or off?
REFlvl	Set the reference level.
REFlvl?	What is the reference level?
RFatt	Set the RF attenuation to a specific value.
RFatt?	What is the RF attenuation?
RLUnit	Select reference level unit.
RLUnit?	What is the reference level unit?
ROFset	Set the reference level offset and turn it on and off.
ROFset?	What is the reference level offset?
ROMode	Turn reference level offset mode on and off.
ROMode?	Is reference level offset mode on or off?
VMAnttbl	Select an antenna table.
VMAnttbl?	What antenna table is selected?
VMDIst	Select measurement distance in dBuV/m mode.
VMDIst?	What is the measurement distance?
VMDEst	Select destination register in dBuV/m mode.
VMDEst?	What is the destination register?
VMMkrunit	Select marker unit in dBuV/m mode as dBuV/m or Volts/m.
VMMkrunit?	What is the marker unit in dBuV/m mode?

INPUT MENU			
0	REF LEVEL ENTRY	20.0DBM	REFIvl
1	PREAMP	OFF	PREamp
2	50 OHM DBM/75 OHM DBMV	50	IMPCor
3	REF LEVEL UNIT	DBUVM	RLUnit
4	1ST MXR INPUT LVL	-30DBM	MXRvl
5	RF ATTENUATION	AUTO 50DB	ARFatt, RFatt
6	EXTERNAL ATTEN/AMPL	NONE	ROFset, ROMode
9	CAL SIG @ 100MHZ -30DBM	OFF	CALsig

VMAnttbl, VMDEst
VMD1st, VMMkrunit

Figure 3-5. Input Menu commands.

Table 3-7. Sweep/trigger front panel commands.

HEADER	FUNCTION
SIGswp	Selects and arms the single sweep mode.
SIGswp?	What is the status of the single sweep mode?
TIME	Select/increment/decrement the sweep speed.
TIME?	What is the sweep speed?
TIMMode	Select auto, manual, or programmed sweep mode.
TIMMode?	What sweep mode is selected?

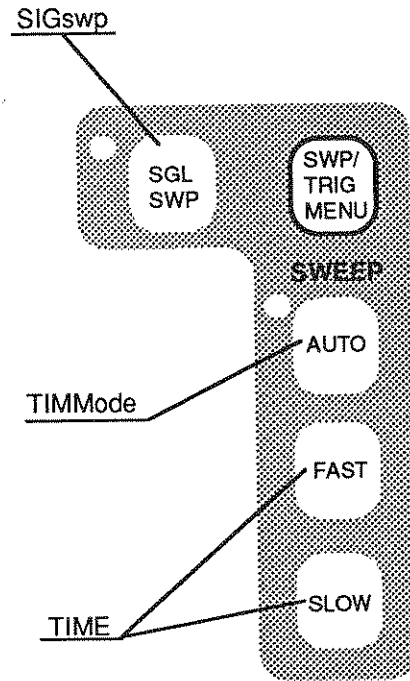


Figure 3-6. Sweep/trigger front panel commands.

Table 3-8. Sweep/Trigger Menu commands.

HEADER	FUNCTION
TIME	Set the sweep rate.
TIME?	What is the sweep rate?
TIMMode	Select auto, manual, or programmed sweep.
TIMMode?	What is the sweep mode?
TRIGGER	Select the trigger mode.
TRIGGER?	What is the trigger mode?
TVLine	Selects the number of the video raster line to trigger on when TV line triggering is selected.
TVLine?	What is the number of the TV line to trigger on?
TVLMode	Selects continuous or programmed TV line trigger.
TVLMode?	Is continuous or programmed TV line trigger used?
TVLStd	Selects TV standards used in various countries.
TVLStd?	What TV standard is being used?
VDMode	Selects broadcast or satellite video demodulation.
VDMode?	Is broadcast or satellite video demodulation used?
VMonitor	Turns the video monitor mode on and off.
VMonitor?	Is the video monitor mode on or off?
VPolarity	Selects positive or negative video polarity.
VPolarity?	Is positive or negative video polarity selected?
VSync	Selects positive or negative video sync polarity.
VSync?	Is positive or negative sync polarity being used?

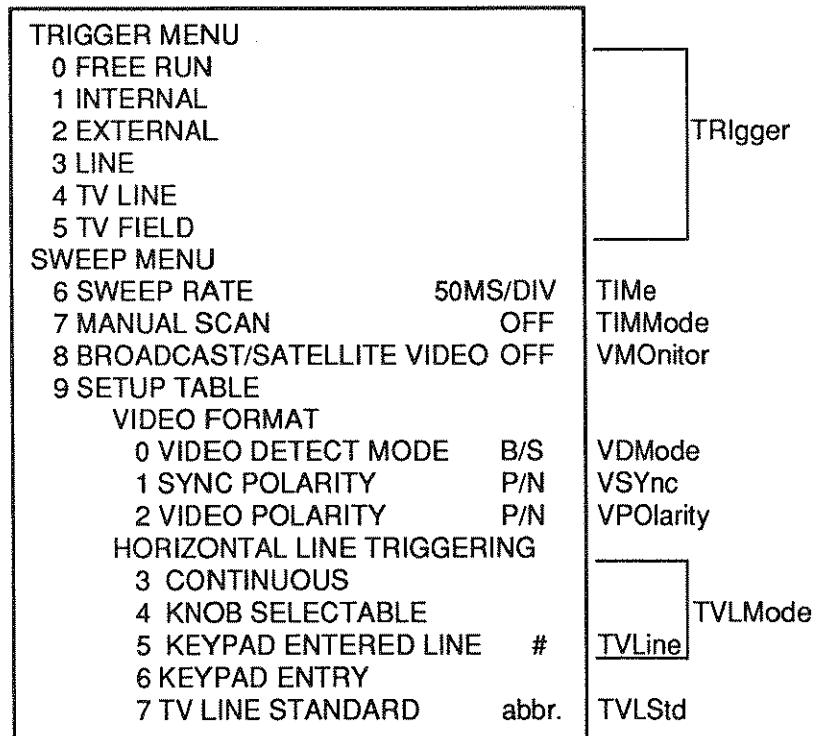


Figure 3-7. Sweep/Trigger Menu commands.

Table 3-9. Display storage front panel commands.

HEADER	FUNCTION
MXHId	Turn max hold function on and off.
MXHId?	Is the max hold function on or off?
SAVe	Turn display storage on or off in any or all registers.
SAVe?	Is storage on or off in any or all registers?
VIEW	Turn display on and off in any or all registers. Also turns waterfall and B,C minus A modes on and off.
VIEW?	What is the display status of any or all registers?

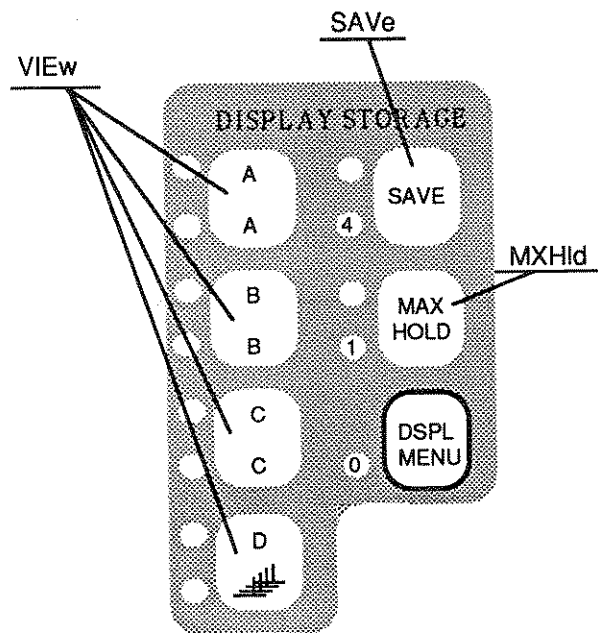


Figure 3-8. Display storage front panel commands.

Table 3-10. Display Menu commands.

HEADER	FUNCTION
ACQmode	Selects peak or max/min acquisition mode.
ACQmode?	What is the acquisition mode?
AVDest	Select destination register for ensemble averaging.
AVDest?	What is the destination register for averaging?
AVG	Turn ensemble averaging on and off.
AVG?	Is ensemble averaging on or off?
AVMode	Select the ensemble averaging mode.
AVMode?	What is the ensemble averaging mode?
AVNum	Select number of sweeps averaged .
AVNum?	What is the number of sweeps averaged?
DLIne	Turn the display line on and off.
DLIne?	Is the display line on or off?
DLLimit	Control the limit detector.
DLLimit?	What is the limit detector status?
DLValue	Set the display line value and turn it on.
DLValue?	What is the display line value?
DSRc	Select the detection mode.
DSRc?	What is the detection mode?
MNHld	Turn min hold function on and off
MNHld?	Is the min hold function on or off?
POFset	Offset B,C-A mode to center or top of screen.
POFset?	Is B, C -A offset to top or center of screen?
REDout	Turn the on-screen readouts on and off.
REDout?	Are the on-screen readouts on or off?
TEXT	Display the indicated text on line 8 of display.
TEXT?	What is the text string being displayed?
TITLE	Display the indicated text as a title in title mode.
TITLE?	What is the title?
TTLMode	Turn title mode on and off.
TTLMode?	Is title mode on or off?
VIEW Minusa:	Turn B,C MINUS A mode on and off

DISPLAY MENU			
1	ENSEMBLE AVERAGING		
1	INITIATE AVERAGING		AVG
2	TERMINATE AVERAGING		AVG
3	MAX		AVMode
4	MEAN		
5	MIN		
6	MAX/MIN		
7	NUMBER OF AVERAGES	#	AVNum
8	SAVE RESULTS IN DISPLAY	C	AVDest
2	B,C MINUS A	OFF	VIEW Minusa:
3	B,C OFFSET TO	CENTER	POFset
4	ACQUISITION MODE	PEAK	ACQmode
5	TITLE MODE	OFF	TITLE, TTLMode
6	READOUT	ON	REDout
7	DISPLAY SOURCE	AM	DSRc
8	DISPLAY LINE		
1	ON/OFF	OFF	DLline
2	VALUE ENTRY	20.0DBM	DLValue
3	DISPLAY LINE TO MARKER		DLValue
4	LIMIT DETECTOR	OFF	DLLimit
9	MIN HOLD IN WFM	C OFF	MNHld

TEXT is related to the User Defined Program function [USER DEF]/[9]/[3].

Figure 3-9. Display Menu commands.

Table 3-11. Applications Menu commands.

HEADER	FUNCTION
BWMode	Turn bandwidth mode on and off.
BWMode?	Is bandwidth (BW) mode on or off?
BWNum	Set the number of dB down for BW mode.
BWNum?	What is the dB down setting in BW mode?
BWResult?	What is the BW at the specified dB down?
CNBw	Set noise BW for carrier-to-noise (C/N) mode.
CNBw?	What is the noise BW in C/N mode?
CNMode	Turn carrier-to-noise mode on and off.
CNMode?	Is carrier-to-noise mode on or off?
CNResult?	What is the C/N ratio?
NNBw	Set the noise BW for normalized noise mode.
NNBw?	What is the noise BW in normalized noise mode?
NNMode	Turn normalized noise mode on and off.
NNMode?	Is normalized noise mode on or off?
NNResult?	What is the normalized noise in the specified BW?
SSBegin	Set the beginning signal search frequency.
SSBegin?	What is the beginning signal search frequency?
SSEnd	Set the ending signal search frequency.
SSEnd?	What is the ending signal search frequency?
SGSrch	Search for all signals greater than the threshold (see THRhd) between the beginning and ending search frequencies.
SSResult?	What is the result of the signal search?

APPLICATION MENU		
0	BANDWIDTH MODE @	-3DBC
1	CARRIER TO NOISE @	5.0MHZBW
2	NOISE NORM'D @	1.0HZBW
3	SIGNAL SEARCH MENU	
0	BEGIN FREQ	88.000MHZ
1	END FREQ	108.000MHZ
2	START TEST	
3	DISPLAY RESULTS	# SIGNALS
	DESTINATION DEVICE	CRT
9	SETUP TABLE	
0	DB DOWN FOR BW MODE	-3DBC
1	NORM BW FOR C/N	5.0MHZBW
2	NOISE NORM'D BW	1.0HZBW

BWMode
CNMode
NNMode
SSBegin
SSEnd
SGSrch
SSResult?
BWNum
CNBw
NNBw

BWResult?, CNResult?, and NNResult? have no direct analog within the 2710. They merely return the results that are normally displayed on screen.

Figure 3-10. Applications Menu commands.

Table 3-12. Utility Menu (abbreviated) commands.

HEADER	FUNCTION
DIR?	Return a 2710 file system directory.
DIScor	Turn the frequency corrections on and off.
DIScor?	Are the frequency corrections on or off?
ERase	Erase the stored settings in a particular register.
ID?	List the 2710 firmware version and installed options.
INIT	Reset to user-defined or factory power-up settings.
NORM	Carry out the indicated normalizations.
NORM?	Return a list of current normalization parameters.
PKHeight	Set the minimum signal height for marker functions.
PKHeight?	What is min signal height for marker functions?
PLLmode	Turn phase lock on and off.
PLLmode?	Is phase lock on or off?
PLOT?	Return screen plot data.
PROTset	Turn stored settings files protection on and off.
PROTset?	Is stored settings files protection on or off?
PTYPE	Specify the plotter type for screen plots.
PTYPE?	What is the specified plotter type?
RECall	Recall a stored settings file.
STOre	Store the current settings in a stored settings file.
VFMode	Selects auto or fixed video filter mode.
VFMode?	Is auto or fixed video filter mode selected?
VIDflt	Sets and turns the video filter on and off.
VIDflt?	What video filter is selected?

UTILITY MENU		
0 INITIALIZE INSTR SETTINGS		INIT
1 STORED SETTINGS/DISPLAYS		RECall
2 KEYPAD ENTERED SETTINGS		STORe
5 VIDEO FILTER	WIDE	ERAsE
3 NORMALIZATIONS		VFMode, VIDfft
4 SYSTEM CONFIGURATION		NORM
1 SCREEN PLOT CONFIGURATION		PTyPe
3 INSTRUMENT CONFIGURATION		
1 MINIMUM SIGNAL SIZE	20	PKHeight
4 PHASELOCK	ON	PLLmode
5 FREQUENCY CORRECTIONS	ON	DIScor
5 STORED SETTINGS PROTECT	OFF	PROTset
6 FILE SYSTEM DIRECTORY		DIR?
9 INSTALLED OPTIONS DISPLAY		ID?
5 INSTR DIAGNOSTICS/ADJUSTMENTS		
6 SCREEN PLOT		PLOT?

Figure 3-11. Utility Menu (abbreviated) commands.

Table 3-13. Detector/Generator Menu commands.

HEADER	FUNCTION
DETECTOR DETECTOR?	Turn on/select which audio detector is used. Which audio detector is being used?
TGENAB TGENAB?	Turns the tracking generator on and off. Is the tracking generator on or off?
TGLEVEL TGLEVEL?	Sets the tracking generator output level. What is the tracking generator output level?
TGMAN TGMAN?	Enables and disables manual control of tracking gen. Is manual tracking gen. control enabled or disabled?
TGOMODE TGOMODE?	Turn tracking generator output level offset on or off. Is tracking generator output level offset on or off?
TGOOFFSET TGOOFFSET?	Sets the tracking generator output level offset. What is the tracking generator output level offset?
TGTMODE TGTMODE?	Turns the tracking generator tracking on and off. Is the tracking generator tracking on or off?
TGTRACK TGTRACK?	Sets the tracking generator tracking. What is the tracking generator tracking?

DETECTOR/GENERATOR MENU		
0	OFF	
1	AM DETECTOR	
2	FM DETECTOR	
3	AM & FM DETECTOR	
4	TRACKING GENERATOR	OFF
5	TG FIXED LEVEL	#
6	TG VARIABLE LEVEL	OFF
7	TG TRACKING	OFF
8	TG EXT ATTEN/AMPL	#

DETECTOR

TGEnab
TGLevel
TGMan
TGMode, TGTRack
TGOMode, TGOOffset

Figure 3-12. Detector/Generator Menu commands.

Table 3-14. Curve and waveform commands.

HEADER	FUNCTION
CURve	Transfer the waveform data to the register specified in the WFMpre command using the encoding set by WFMpre.
CURve?	Transfer the waveform data from the specified register or from the register set with the WFMpre command using the encoding specified by the WFMpre command.
WAVfrm? WFMpre	Same as WFMpre? followed by CURVE? Specifies which register to use as the source or destination for transferring waveform data using the CURve command/query. Also specifies the encoding to be used on the waveform data.
WFMpre?	Can be used to request the complete waveform preamble, or to ask which register and encoding are to be used for waveform transfers.

Waveform transfers are not reflected on any menu or function block; they transfer data representing on-screen spectra and their formatting between the 2710 and the controlling computer.

Table 3-15. System-related commands.

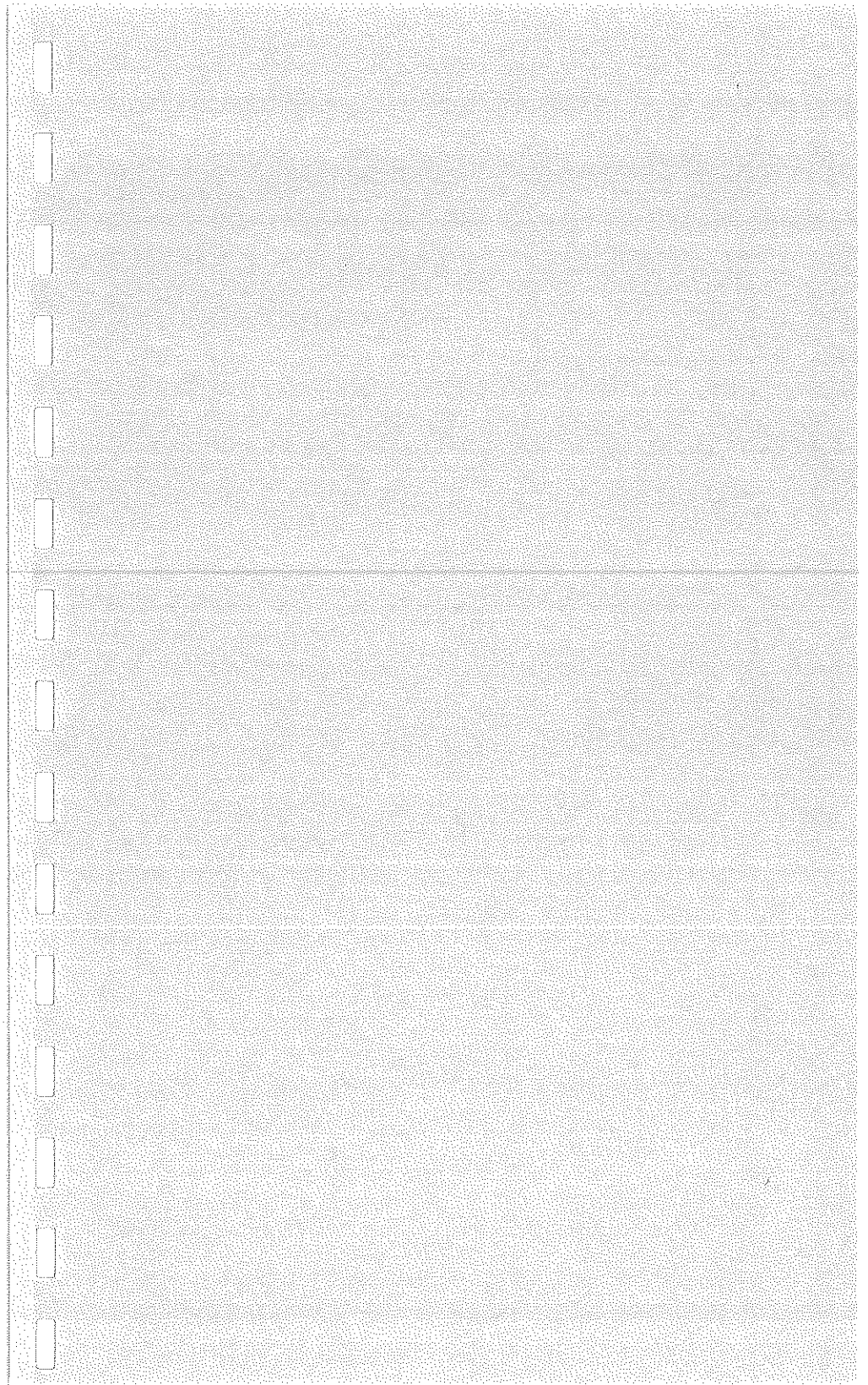
HEADER	FUNCTION
EOS	Enables and disables SRQ on end-of-sweep.
EOS?	Is SRQ enabled or disabled at end-of-sweep?
ERr?	What is the error code?
EVEnt?	What is the event code?
HDR	Turns the response header on and off.
HDR?	Is the response header on or off.
HELp?	2710 sends a list of legal GPIB command headers.
MSGdlm	Selects a semicolon or line feed as the response delimiter.
MSGdlm?	What is the response delimiter?
RQS	Enables or disables SRQ's (except power-on SRQ).
RQS?	Are SRQ's enabled or disabled?
SET?	What are the current 2710 settings?
SGErr	Enables and disables the SRQ when the marker cannot find a signal.
SGErr?	Is the marker-can't-find-signal SRQ enabled or disabled?
WAIt	Commands the 2710 to wait for the end of sweep.

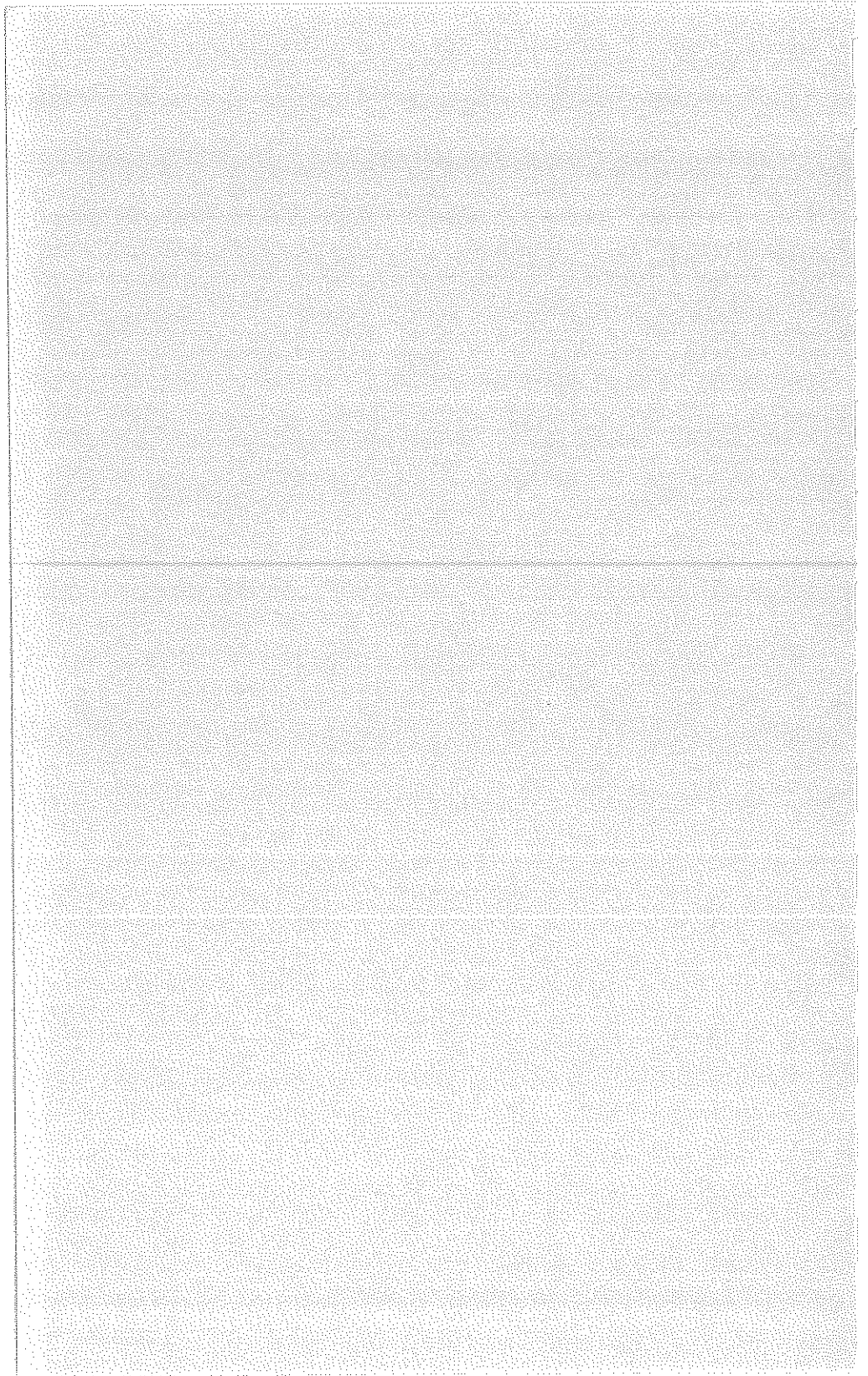
These commands and queries are independent of any 2710 menu or function block. They represent functions which effect the interaction of the 2710 and the system controller.

Table 3-16. Miscellaneous commands.

HEADER	FUNCTION
FILE	Store a binary block under a given file name.
FILE?	Return the named file as a binary block.
GRAt	Turn the graticule light on and off.
GRAt?	Is the graticule light on or off?
KEY	Simulate pressing a key.
MEMory?	How much NVRAM is free?
PRDouts?	Return the 2710 on-screen readouts.

The few remaining miscellaneous commands and queries in this table are not reflected on any of the 2710 menus.





Section 4 2710 Command and Query Definitions

This section contains an alphabetical list of all 2710-specific commands and queries. The list completely defines each command or query; it contains all the information you need to send messages to the 2710 or to interpret the responses from the analyzer.

Typographical Conventions

Each 2710 command is discussed in a format like that below. Upper-case letters are required. Lower-case letters may be optionally supplied. In no case will letters other than those shown be accepted by the 2710.

COMmand <arg> (If no argument is needed, <arg> is deleted)

Arguments: argument 1, argument 2,...

If no argument is required, "none" is listed.

General discussion of the command or query, its arguments, specific precautions, etc. Optional tables as required explaining the function provided by specific arguments, or other detailed information.

Actual messages shown in their syntactically correct form. Where the number of possible arguments is limited (such as commands that turn features on and off), all messages are shown.

COMmand ON
COMmand OFF

Where there is a large range of arguments (such as numeric values), typical examples are shown.

COMmand 10.5 KHz (*for example*)

Typical examples are always followed by the term (*for example*).

Each 2710 query is discussed in a format like that below.

QUERy? <arg> (In most cases no argument is needed, and
<arg> is omitted)

Arguments: argument 1, argument 2,...
If no argument is required, "none" is listed.

General discussion of the query, its arguments, specific precautions, etc. Optional tables as required explaining the function provided by specific arguments, or other detailed information. Detailed description of the response to the query.

Actual messages: the query is shown along with its arguments. The response to the query is shown indented on the following line. The response is always shown with HDR ON. Where the number of possible responses is limited (such as queries that report the on/off status of features), all responses are shown.

```
QUERy?  
  QUERY ON  
  QUERY OFF
```

Where there is a large range of responses (such as numeric values), typical examples are shown.

```
QUERy?  
  QUERY 10.500E+3 (for example)
```

Typical examples are always followed by the term (*for example*).

List of Commands and Queries

The following list of commands and queries provides detailed information about the 2710-specific GPIB instruction set. It does not attempt to explain the operation of the spectrum analyzer. For descriptions of the 2710, its features, and functions, see the *27710 Operators Manual* or *2710 Spectrum Analyzer User's Guide*.

ACQmode <arg>

Arguments: MAXMin, PEAK

Single-argument command designating the 2710 display storage acquisition mode.

```
ACQmode PEAK  
ACQmode MAXMin
```

ACQmode?

Arguments: none

Simple query returning the currently selected acquisition mode.

```
ACQmode?  
ACQMODE PEAK  
ACQMODE MAXMIN
```

AREs <arg>

Arguments: ON, OFF

Single-argument command turning 2710 automatic selection of resolution BW on and off.

```
AREs ON  
AREs OFF
```

AREs?

Arguments: none

Simple query returning status of 2710 automatic resolution BW selection mode.

```
AREs?  
ARES ON  
ARES OFF
```

ARFatt <arg>

Arguments: ON, OFF

Single-argument command turning 2710 automatic selection of RF attenuation on and off. The attenuation, linear scale factor, and reference level may change when turning on auto selection, but not when turning it off.

```
ARFatt ON  
ARFatt OFF
```

ARFatt?

Arguments: none

Simple query returning status of 2710 automatic RF attenuation selection.

```
ARFatt?  
ARFATT ON  
ARFATT OFF
```

ATBI? <arg>

Arguments: none, integer in range of 1 to 5

Query with one or no argument that returns a listing of the specified antenna table. The argument is the number of the antenna table to be listed. If a number outside the

range is indicated, the last table in the range is returned (i.e., an argument of 6 returns table number 5, an argument of 0 returns table number 1). If no argument is specified, the currently selected table is returned.

```
ATBI? 3
ATBL "ANTENNA 3
Cal Distance = 3.0 Meters
Frequency      Factor(dB)
-----
100.0MHz      1.0
200.0MHz      2.0
300.0MHz      3.0
.
.
1.8GHz        18.0
-----
"; (for example)
```

ATHrhld<arg>

Arguments: ON, OFF

Single-argument command turning 2710 automatic selection of signal threshold on and off. The threshold may change when turning on auto selection, but not when turning it off.

```
ATHrhld ON
ATHrhld OFF
```

ATHrhld?

Arguments: none

Simple query returning status of 2710 automatic signal threshold selection.

```
ATHrhld?
ATHrhld ON
ATHrhld OFF
```

AVDest <arg>

Arguments: A, B, C

Single-argument command designating the 2710 display register used as destination for ensemble average and minimum hold operations. The destination register cannot be changed while a min hold or ensemble average operation is in progress.

```
AVDest A
AVDest B
AVDest C
```

AVDest?

Arguments: none

Simple query returning the 2710 display register currently selected as the destination for ensemble averaging and minimum hold functions.

```
AVDest?
AVDEST A
AVDEST B
AVDEST C
```

AVG <arg>

Arguments: ON,OFF

Single argument command turning the currently selected 2710 ensemble averaging mode on and off. Ensemble averages will terminate themselves after the requested number of sweeps is averaged, but AVG OFF is used to terminate a continuous average. You cannot turn ensemble averaging on if the 2710 is in analog display mode, or there is a destination register conflict.

```
AVG ON
AVG OFF
```

AVG?

Arguments: none

Simple query returning the on/off status of the currently selected 2710 ensemble averaging mode.

```
AVG?  
AVG ON  
AVG OFF
```

AVMode <arg>

Arguments: MAX, MAXMin, MEAN, MIN

Single-argument command designating the 2710 ensemble average mode.

```
AVMode MAX  
AVMode MAXMin  
AVMode MEAN  
AVMode MIN
```

AVMode?

Arguments: none

Simple query returning the currently selected 2710 ensemble averaging mode.

```
AVMode?  
AVMODE MAX  
AVMODE MAXMIN  
AVMODE MEAN  
AVMODE MIN
```

AVNum <arg>

Arguments: integer number in the range of 0 to 1024

Single-argument command designating the number of sweeps to be averaged by the currently selected ensemble averaging mode. If zero is specified, a continuous average is performed. The 2710 default is 16.

AVNum 128 (for example)

AVNum?

Arguments: none

Simple query returning the integer number of sweeps the current ensemble averaging mode will average.

AVNum?
AVNUM 128 (for example)

BWMode <arg>

Arguments: ON, OFF, IDLE

Single-argument command turning the 2710 bandwidth measurement mode on and off. Turning on bandwidth mode turns off marker mode if it is on. Bandwidth mode is not allowed if the 2710 is in analog display mode or video monitor mode. Bandwidth mode is not possible in the D-register if waterfall mode is enabled.

BWMode ON
BWMode OFF
BWMode IDLE

BWMode IDLE has the same effect as BWMode ON.

BWMode?

Arguments: none

Simple query returning the status of the 2710 bandwidth measurement mode.

```
BWMode?  
BWMODE ON  
BWMODE OFF  
BWMODE IDLE
```

BWNum <arg>

Arguments: number in the range -1 to -70

Single argument command that specifies the integer number of dB below the signal peak at which the bandwidth will be measured by the 2710 bandwidth measurement mode. Units are not allowed; dBc are assumed. Non-integer values are truncated.

```
BWNum -20 (for example)
```

BWNum?

Arguments: none

Simple query returning the integer number of dB below the peak at which a signal's bandwidth will be measured by the 2710 bandwidth measurement mode.

```
BWNum?  
BWNUM -20 (for example)
```

BWResult?

Arguments: none

Simple query returning the result of the most recent bandwidth measurement in Hertz using the 2710's bandwidth measurement feature. The result is updated at the end of the current sweep if bandwidth mode is not idle.

```
BWResult?  
BWRESULT 5.238E+3 (for example)
```

CALSig <arg>

Arguments: ON, OFF

Single-argument command turning the 2710 calibration signal on and off. The 2710 input signal is disconnected when the cal signal is on.

```
CALSig ON  
CALSig OFF
```

CALSig?

Arguments: none

Simple query returning the on/off status of the 2710 calibration signal.

```
CALSig?  
CALSIG OFF  
CALSIG ON
```

CENsig

Arguments: none

Command with no argument which sets the 2710 center frequency to the frequency of the primary marker.

```
CENsig
```

CFSF <arg>

Arguments: CENter, STArT

Single-argument command designating the displayed frequency as the center or start frequency. The indicated frequency may be adjusted depending on the current

frequency and span to ensure the resulting 2710 condition is permissible.

```
CFSF CENTER  
CFSF START
```

CFSF?

Arguments: none

Simple query whose response indicates whether the 2710 on-screen frequency is the center or start frequency.

```
CFSF?  
CFSF CENTER  
CFSF START
```

CMEas

Arguments: none

Command with no argument that causes the 2710 to perform a center measure.

```
CMEas
```

CNBw <arg>

Arguments: frequency in the range 1 Hz to 1.8 GHz

Single-argument command specifying the bandwidth used by the 2710 carrier-to-noise (C/N) feature to perform a C/N measurement. Hertz are assumed if no units are appended.

```
CNBw 4.0 Mhz (for example)
```

CNBw?

Arguments: none

Simple query returning the noise bandwidth in Hertz used by the 2710 carrier-to-noise (C/N) feature to perform a C/N measurement.

```
CNBw?  
CNBW 4.0E+6 (for example)
```

CNMode <arg>

Arguments: ON, OFF, IDLE

Single-argument command that turns the 2710 carrier-to-noise (C/N) mode on and off. Turning on C/N mode turns off marker mode.

```
CNMode ON  
CNMode OFF  
CNMode IDLE
```

CNMode IDLE has the same effect as CNMode ON.

CNMode?

Arguments: none

Simple query returning the status of the 2710 carrier-to-noise mode.

```
CNMode?  
CNMODE OFF  
CNMODE ON  
CNMODE IDLE
```

CNMode IDLE means there is no signal, the AM detector is not selected, the noise is too close to the analyzer noise floor, or analog mode is selected.

CNResult?

Arguments: none

Simple query returning the result in dB of the most recent carrier-to-noise (C/N) measurement performed by the 2710 C/N feature. The measurement is updated at the end of the current sweep if C/N mode is enabled. CNMode must be ON for the result to be meaningful.

```
CNResult?  
CNRESULT -4.65E+1 (for example)
```

CNTtrak<arg>

Arguments: ON, OFF

Single-argument command which enables and disables the frequency counter in signal track mode. The counter option must be installed in the 2710. The command sets counter resolution to 1 Hz when the counter is enabled.

```
CNTtrak ON  
CNTtrack OFF
```

CNTtrak?

Arguments: none

Simple query whose response indicates whether the counter is on or off in signal track mode.

```
CNTtrak?  
CNTTRAK OFF  
CNTTRAK ON
```

COUnt?

Arguments: none

Simple query that returns the result in Hertz of the last frequency count.

```
COUnt?  
COUNT 55.250E+6 (for example)
```

CREs <arg>

Arguments: 1 Hz, 1Khz

Single-argument command designating the 2710 frequency counter resolution. The optional counter must be installed.

```
CREs 1 Hz  
CREs 1Khz
```

CREs?

Arguments: none

Simple query that returns the currently selected 2710 counter resolution. The optional counter must be installed in the 2710. Only values of 1 Hz and 1 kHz are possible.

```
CREs?  
GRES 1.E+0  
GRES 1.E+3
```

CURve <arg>

Arguments: complex block of data described in the following discussion.

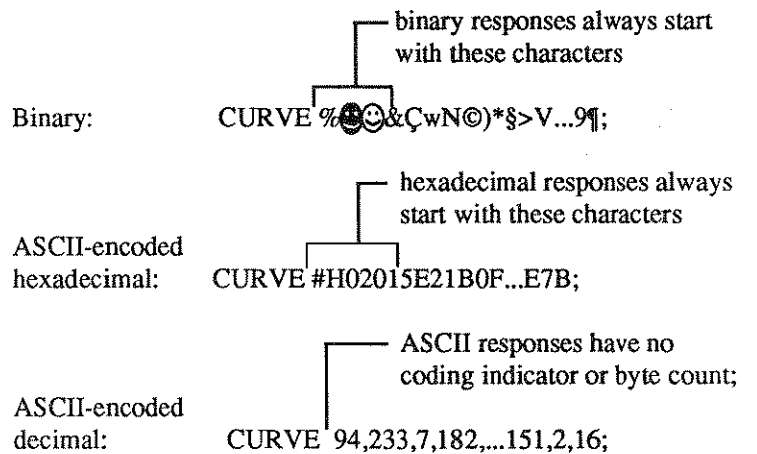
Single-argument command enabling you to send a block of curve data to a 2710 display register.

The data block represents the 512 points in a 2710 waveform. The encoding of the data points (ASCII-encoded decimal, ASCII-encoded hexadecimal, or binary) is determined by the current waveform preamble (see the WFMpre command). The register (A, B, C, or D) to which the data are to be sent and is also determined by the preamble.

Data transferred to the analyzer can be previously returned waveforms or artificially generated curves. To ensure that the transferred data are not immediately overwritten, you should transfer them to a saved register (other methods are also possible, such as transferring to an active register in single sweep mode).

The format of curve data is shown in Figure 4-1. The checksum definition ensures that the sum (modulo 256) of the data points plus point count plus checksum equals zero.

If you were to display the message on your controller screen (for instance, by printing the response to the CURve? query with HDR ON), you would see something similar to this:



CURVE<space><ind><bc_{hi}><bc_{lo}><d₁><d₂>...<d₅₁₂><checksum>;

where:

CURVE	command header
space	header delimiter
ind	absent when ASCII-encoded decimal is used; equals #H when ASCII-encoded hexadecimal is used; equals the % sign when binary is used
bc _{hi}	high order byte of the number of data points (always 512) plus one in the data block: absent in decimal, 00000010 in binary, 02 in hexadecimal
bc _{lo}	low order byte of the number of data points (always 512) plus one in the data block: absent in decimal, 00000001 in binary, 01 in hexadecimal
d ₁ d ₂ ...d ₅₁₂	1 st data point, 2 nd data point,...512 th data point of the curve; each data point may be represented by one to four bytes depending on encoding: binary: one byte hexadecimal: two bytes representing the hexadecimal numerals 0-F decimal: two-four bytes representing a comma delimiter plus one to three decimal numerals 0-9
checksum	absent for ASCII-encoded decimal; otherwise computed as below:

$$\text{chksum} = 256 - \text{remainder of } \left[\frac{\text{sum of all data points} + \text{bc}_{hi} + \text{bc}_{lo}}{256} \right]$$

Figure 4-1. Format of curve data.

Here are several more points worth noting about the various encodings:

- Binary encoding uses one byte per data point, making it more compact and faster than the other techniques.
- Hexadecimal always uses two bytes per data point, thus, it requires no delimiter to separate the points.
- Decimal uses a variable number of bytes (1 to 3) to encode each data point, and therefore requires a data point delimiter (the comma).
- The use of delimiters in decimal encoding makes this form compatible with many spread sheets and word processors, enabling you to create custom waveforms for later transmission to the analyzer, or to edit waveforms previously returned from the 2710.

The area of the 2710 screen covered by the graticule is represented by 500 intervals horizontally and 240 intervals vertically. The graticule corners are represented as shown in Figure 4-2. Horizontally the sixth data point corresponds to the leftmost graticule line; the 505th point corresponds to the right-most graticule line. Generally, the points are numbered from 0 - 511, so the graticule is

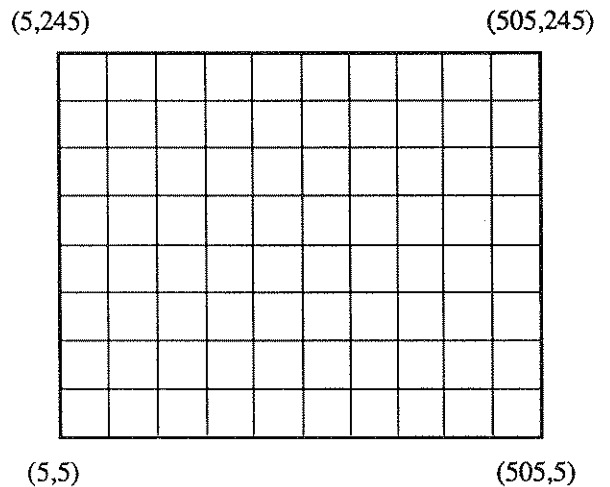


Figure 4-2. 2710 graticule coordinates.

represented horizontally by points 5 - 505 inclusive. Vertically, the data are digitized into 256 values from 0 to 255. Values of 5 and 245 represent the bottom and top graticule lines, respectively. The curve data extend slightly beyond the graticule area, and data points (except the first) beyond the graticule are valid. See Section 6 for programming examples using CURve and CURve?.

CURve? or CURve? <arg>

Arguments: none, A, B, C, D

Query with either one or no arguments returning a complex response representing the contents of a 2710 waveform display register.

```
CURve?  
CURve? A  
CURve? B  
CURve? C  
CURve? D  
CURVE 94,233,7,...151,2,16; (for example)
```

The format of the response is determined by the previous WFMpre command. If no argument is specified, the source register of the curve data is also determined by the previous WFMpre command. See the CURve command discussion for data formats and other details.

The CURve? response with HDR ON will resemble one of the forms shown in the CURve command discussion.

You can return data from an active or inactive register whether it is saved or not (the register always contains data even if it is not displayed). Contrast with the FILE command which enables you to return a stored curve file whether it is currently in a register or not (in binary only, and not in curve data format).

See Section 6 for programming examples using the CURve? query.

DETEctor <arg>

Arguments: AM, AMFm, FM, OFF

Single-argument command which determines the type of signal detector used for the 2710 audio output. Depending on the argument, the AM, FM or both detector outputs are presented at the 2710 audio output. The audio output cannot be used in video monitor mode.

```
DETEctor AM
DETEctor AMFm
DETEctor FM
DETEctor OFF
```

DETEctor?

Arguments: none

Simple query returning the current status of the audio source (whether the output of the AM detector, FM detector, neither, or both are being presented at the 2710 audio output).

```
DETEctor?
DETECTOR AM
DETECTOR AMFM
DETECTOR FM
DETECTOR OFF
```

DIR?

Arguments: none

Simple query that returns a formatted system file directory listing similar to pressing [UTIL MENU]/[4]/[6]. Each line in the listing (except the first and last) is formatted like this:

filename, read/write enabled (R or W), size in bytes

```
DIR?  
DIR * 12.88, . 0  
TMPDBG, RW, 16380  
12.88, . 0  
DSET00, RW, 370  
SET0BU, RW, 370  
:  
UDP2, R , 160  
"; (for example)
```

DIScor <arg>

Arguments: ON, OFF

Single-argument command enabling and disabling the 2710 frequency corrections. When DIScor is ON (frequency corrections are disabled), the message "FREQ COR OFF" appears on the 2710 screen.

```
DIScor ON  
DIScor OFF
```

DIScor?

Arguments: none

Simple query returning the current on/off status of the 2710 frequency corrections. Note that DISCOR ON means the frequency corrections are off.

```
DiScor?  
DISCOR ON  
DISCOR OFF
```

DLine <arg>

Arguments: ON, OFF

Single-argument command that turns the 2710 display line feature on and off. DLINE cannot be turned on if the A-register is being used (as by waterfall mode, min hold, ensemble averaging, etc.); attempting to do so generates an event 787, destination waveform conflict.

```
DLine ON  
DLine OFF
```

DLine?

Arguments: none

Simple query returning the current on/off status of the 2710 display line feature.

```
DLine?  
DLINE ON  
DLINE OFF
```

DLLimit <arg>

Arguments: OFF, OVER, OVUNder, UNDER

Single-argument command controlling the status of the 2710 display line limit detector. When the limit detector is not off, an SRQ and an event 895 are generated whenever the limit condition is exceeded.

argument	condition
OVER:	alarm when signal > display line
UNDER:	alarm when signal < display line
OVUNder:	alarm when signal > display line or signal < threshold
OFF:	no alarm

```
DLLimit OFF
DLLimit OVER
DLLimit OVUNder
DLLimit Under
```

DLLimit?

Arguments: none

Simple query returning the current status of the display line limit detector.

```
DLLimit?
DLLIMIT OFF
DLLIMIT OVER
DLLIMIT OVUNDER
DLLIMIT UNDER
```

DLValue <arg>

Arguments: MARKer
amplitude in the range -150 to +100 dBm

Single-argument command turning on the 2710 display

line and setting its amplitude. A numeric argument sets the amplitude to the value of the argument. The units are the currently selected reference level units. However, the argument must be within a range equivalent to -150 to +100 dBm. The MARKer argument sets the display line amplitude to the amplitude of the primary maker.

```
DLValue MARKer  
DLValue -30 (for example)
```

DSRc <arg>

Arguments: AM, EXTERNAL, FM

Single-argument command designating the source of the signal displayed by the 2710 as the internal AM detector (normal display), internal FM detector (useful for FM deviation checks), or an external input.

If FM or EXTERNAL are selected at any time, the 2710 is placed in zero span mode and max/min signal acquisition is selected. FM and EXTERNAL are not allowed in DBUVM mode.

```
DSRc AM  
DSRc EXT  
DSRc FM
```

DSRc?

Arguments: none

Simple query returning the currently selected source of the signal displayed by the 2710.

```
DSRc?  
DSRC AM  
DSRC FM  
DSRC EXTERNAL
```

EOS <arg>

Arguments: ON, OFF

Single-argument command enabling and disabling the end-of-sweep SRQ. When EOS is ON, an end-of-sweep SRQ is normally generated at the end of each 2710 sweep. However, intermediate end-of-sweep SRQ's are suppressed in the case of normalization, User Defined Programs, signal searches, plots, and ensemble averages, until the process is complete and then a single SRQ is issued indicating "end-of-process".

For instance, if you compile a 10-sweep ensemble average, you will not get an SRQ following each sweep. Rather, you will get a single SRQ accompanied by event 882, ensemble average complete, after the averaging process is finished.

```
EOS ON
EOS OFF
```

EOS?

Arguments: none

Simple query returning current on/off status of the end-of-sweep generator.

```
EOS?
EOS ON
EOS OFF
```

ERAsE <arg>

Arguments: Numeral in the range 2 to 9

Single-argument command specifying a stored settings register to erase. Registers 2 - 9 can be designated. If the settings are protected (locally or because PROTSET is ON), only the waveforms associated with the indicated

settings are erased; an SRQ and event 839 are generated.

```
ERase 2  
.  
ERase 9
```

ERr?

Arguments: none

Simple query used to return an integer event code. ERr? is equivalent to EVenT? and preserved in the 2710 for consistency with the command sets of other instruments.

```
ERr?  
ERR 878 (for example)
```

EVENt?

Arguments: none

Simple query used to return an integer event code. If RQS is ON, a serial poll must be performed following the SRQ and prior to sending the EVenT? query to obtain the correct event code. If RQS is OFF, an EVenT? query may be carried out at any time without a serial poll.

```
EVENt?  
EVENT 878 (for example)
```

FILE <arg>

Arguments: <filename>,<data block>

Complex single- or multiple-argument command transferring a previously stored 2710 file from the controller to the analyzer. When used with only the <filename> argument, the command establishes the name of the file to be transferred to the controller by the next FILE? query. The allowable filenames are listed in Table 4-1.

The intent of the FILE command and FILE? query is to transfer files, through the mechanism of up-loading and subsequent down-loading, between different 2710's. For instance, you might develop a User Definable Program (UDP) in one analyzer, transfer it to the controller using the FILE? query, and subsequently down-load it to a number of other analyzers using the FILE command. Be aware, however, that if files are transferred between analyzers with different installed options, uncertain results may occur.

You can also use this technique with a single analyzer to back up important settings files, or the reference normalizations, in preparation for changing the 2710 NVRAM battery.

When files are originally returned from the 2710 with the FILE? query, HDR is usually set ON so the ASCII strings FILE and <filename> precede the actual data and are stored as the first bytes of the disk file. It is then unnecessary to explicitly transmit the FILE header or the <filename> when restoring the file to the analyzer. Read the disk file into a string variable called, say, FILEDAT\$. The string variable will be of exactly the form needed to send the file to the 2710:

```
FILEDAT$ = "FILE <filename>,<data block>"
```

Simply transmit FILEDAT\$ to the analyzer.

Table 4-1. The 2710 file system.

Settings files: Each file saves the 2710 control settings for a particular register (A, B, C, D) in a designated location (00-09). BSET03 saves B-register settings in the number 3 location.

Curve files: Each file saves the curve data from a particular register (A, B, C) in a designated location (00-09). D-register curves are never saved. AWF04 saves the A-register curve in location 4.

User-Definable Program files: Each file saves a keystroke command sequence in a designated location (00-09) representing a particular user-defined program. UDP01 is the second user-defined program.

Antenna Table files: Each file saves the antenna table in a designated location (1-5) representing antenna data for a particular antenna. ACF3 saves the antenna information in location 3.

Normalization file: The normalization file saves the data generated by normalizing the 2710, including the reference normalizations.

FILE NAMES

Curves	Settings	User-Definable Programs	Antenna Tables	Normalizations
nWFM00	nSET00	UDP00	ACF1	NORM
:	:	:	:	
nWFM09	nSET09	UDP08	ACF5	
n=A,B,C	n=A,B,C,D			

Several other files of little use to the average user may be present, but should not be altered:

NAME	DESCRIPTION	NAME	DESCRIPTION
12.88	Version	S_PLOT	Plotter configuration
SETUP	Instrument configuration	S_GPIB	GPIB configuration
S_CENT	Centronics configuration	S_RTC	Real-time clock configuration
SEARCH	Signal search configuration		

Other files of a temporary or transitory nature may be created by the 2710 for internal purposes. These files should not be altered.

If HDR was OFF when the file was returned, then it is necessary to precede the disk file with "FILE". Transmit the message:

```
FILE FILEDAT$
```

See Section 6 for programming examples.

FILE? or FILE? <arg>

Arguments: none, <filename>

Simple query which returns a file stored in the 2710 to the controller. When used without an argument, it returns the file specified by the previous FILE command. When used with a <filename> argument, it returns the named file. In either case, the filename must match, including case, one of those listed in Table 4-1.

The file names in the 2710 are established by its firmware. A directory of currently created files can be viewed by pressing [UTIL MENU]/[4]/[6] or [UTIL MENU]/[5]/[4]/[1]/[0]. Files are created within the 2710's memory only as required. That is, you won't find a BSET03 settings file unless you have previously saved the B-register settings in the third storage location.

DSET00 and SET0BU are two special files. They are created automatically by the 2710 and contain the D-register settings used when the analyzer was last turned off. They are listed as LAST POWER-DOWN under [UTIL MENU]/[1]/[0]. SET0BU is a backup in case DSET00 becomes corrupted at the next power-up.

The FILE? query is intended to enable you to store a 2710 file on disk for later restoration to the same or another 2710. The file is in binary format, and the first bytes of the response are the ASCII character codes for <filename>. If HDR is ON before issuing the FILE? message, then the query and response look like this:

```
FILE? <filename>  
FILE <filename>,<data block>
```

The response is in exactly the format needed to send a file to the 2710. The general approach to file transfer, then, is to read the response, including header and filename, into a string variable and write the variable to a disk file. The presence of FILE <filename> within the disk file makes it possible to restore the file to a 2710 without having to explicitly send the FILE command or specify the 2710 file name. Follow this sequence to store a 2710 file:

```
Send HDR ON to the 2710
Send FILE? <filename> query to 2710
Read response into string variable FILEDAT$
Write FILEDAT$ to disk file MYPROG
```

and use this sequence to restore it:

```
Read file MYPROG to string variable FILEDAT$
Send FILEDAT$ to the 2710
```

Be aware that all files except UDPs (and even most UDPs) occupy less than 5 kbytes, but a UDP file can theoretically occupy up to 100 kbytes.

See Section 6 for programming examples.

FINE <arg>

Arguments: ON, OFF

Single-argument command selecting 1dB reference level steps when ON and 10 dB steps when OFF.

```
FINE ON
FINE OFF
```

FINE?

Arguments: none

Simple query returning the current on/off status of the reference level steps: ON equals 1dB/step, OFF equals 10dB/step

```
FINE?  
FINE ON  
FINE OFF
```

FOFset <arg>

Arguments: ON,OFF, frequency in the range -1000 GHz to +1000 GHz

Single-argument command controlling the 2710 frequency offset. Frequency units may be appended; otherwise Hertz are assumed.

Argument	Action
OFF	Turn frequency offset off.
ON	Turn on frequency offset, using last offset frequency.
<freq>	Turn on frequency offset and set the value. If the value is 0, turn it off.

When enabled, the value of the frequency offset affects the display and any subsequent FREQ, MFREQ, STSTOP, SSBEGIN, and SSEND commands.

```
FOFset ON  
FOFset OFF  
FOFset 5.15 GHz (for example)
```

FOFset?

Arguments: none

Simple query returning the value of the 2710 frequency offset in Hertz. If the frequency offset is disabled, it returns 0.

```
FOFset?  
FOFset 0  
FOFset 5.15E+9 (for example)
```

FREq <arg>

Arguments: frequency in the range -10 MHz to 1.8 GHz

Single-argument command that sets the center or start frequency to the indicated value. Frequency units may be appended; otherwise Hertz are assumed. Interpreted as center or start frequency depending on the setting of CFSF. If start frequency is selected, the span is checked and the start frequency may be adjusted to ensure the center frequency is never more than 1.8 GHz. The argument is offset by FOFset if FOMode is enabled.

```
FREq 193.25 MHz (for example)
```

FREq?

Arguments: none

Simple query returning the currently selected center or start frequency.

```
FREq?  
FREQ 193.25E+6 (for example)
```

GRAI <arg>

Arguments: ON, OFF

Single-argument command that turns the graticule illumination on and off. GRAI must be ON for graticule lines to appear on screen prints or plots.

```
GRAI OFF
GRAI ON
```

GRAI?

Arguments: none

Simple query returning the current on/off status of the 2710 graticule illumination.

```
GRAI?
  GRAT OFF
  GRAT ON
```

HDR <arg>

Arguments: ON, OFF

Single-argument command that turns the header on and off in a query response. When HDR is ON, a command header describing the nature of the response precedes the response proper. This also makes the response an executable command.

The following table lists several queries and their potential responses with HDR ON and HDR OFF.

Notice in particular, that in the case of most linked numerical arguments (but not those resulting from VRTdsp?, WAVfrm?, and WFMpre? queries), the link is turned off along with the command header, but not in the case of linked character arguments (see MFREQ? and VIEW? in the table).

HDR ON	HDR OFF
FREq? FREQ 193.25E+6	FREq? 193.25E+6
GRAI? GRAT ON	GRAI? ON
MFREq? DELta MFREQ DELTA:4.5E+6	MFREq?DELta 4.5E+6
VIEW? A VIEW A:ON	VIEW? A A:ON

```
HDR OFF
HDR ON
```

HDR?

Arguments: none

Simple query that returns the current on/off status of the response header.

```
HDR?
HDR OFF
HDR ON
```

HELp?

Arguments: none

Simple query that returns a list of GPIB commands separated by commas. Commands affecting all 2710 options are included.

```
HELp?
HELP ACQMODE,AQP,...WFMPRE,ZEROSP
```

HRAmpl

Arguments: none

Command requiring no argument. It moves the primary marker from its current position to the peak of the next higher signal on screen. If the marker is not enabled, the command does so. If signal track is enabled, HRAmpl turns it off, enables the marker, and assigns the knob function to marker control. If SGErr is ON and a higher peak doesn't exist, an SRQ and event 896 are generated.

```
HRAmpl
```

ID?

Arguments: none

Simple query returning the instrument identification, firmware version, and installed options.

```
ID?  
ID TEK/2710,V81.1,"VERSION 11.11.89  
FIRMWARE";"300HZ,1,10,100KHZ,1MHZ  
RBW FLTR";"PHASE LOCK",300KHZ  
FILTER;"VIDEOMONITOR";"GPIB";  
"COUNTER";"NVM 12.88"; (for example)
```

The items in quotes indicate the firmware version and options installed in the particular instrument and may vary depending on how your analyzer is equipped.

IMPCor <arg>

Arguments: integer or floating point number

Single-argument command instructing the 2710 to scale its measurement results for 50 or 75 ohm input. Only the values 50 and 75 are valid. Numbers of 60 or below are rounded to 50; numbers above 60 are rounded to 75. Units are not allowed.

The 2710 has a fixed 50 ohm input, but it can scale its measurements to reflect what would be measured if a 75 ohm instrument were used under the same conditions. See the *2710 Spectrum Analyzer User's Guide* for a detailed description of the 50/75 ohm corrections.

IMPCor 75
IMPCor 50

IMPCor?

Arguments: none

Simple query returning the input impedance value in ohms for which measurements are scaled.

IMPCor?
IMPCOR 50
IMPCOR 75

INIT

Arguments: none

Command requiring no argument that places the 2710 in its power-up configuration. Factory default power-up settings are used unless user-defined power-up settings have been implemented.

INIT

KEY <arg>

Arguments: various mnemonics as listed in Table 4-2.

Single-argument command that simulates pressing a key on the 2710 front panel. The general form of the command is:

KEY <arg>

where <arg> is the mnemonic for the key you want to simulate pressing. All permissible mnemonics are listed in Table 4-2.

For instance, to simulate pressing the Input Menu key, you send the message:

KEY INPutmenu *(for example)*

To turn on the 2710 calibrator signal using the KEY command, you can send this message:

KEY INPutmenu;KEY M9 *(for example)*

The KEY command is less efficient from both a speed and memory standpoint than using a dedicated 2710--specific command, such as CALSig ON, to achieve the same result.

Nevertheless, the command does provide a fallback position if you experience difficulty implementing a dedicated command. In fact, you can in principal perform all GPIB programming using only the KEY command. However, because of its increased memory requirement and decreased speed, use of the KEY command is discouraged as a general purpose GPIB programming technique.

Table 4-2. Arguments of the KEY command.

MNEMONIC	KEY	MNEMONIC	KEY
APplmenu	Application Menu	MKRMenu	Marker
A	Display A	MKRLeft	Marker Left
B	Display B	MKRPeak	Marker Peak
BS	Backspace key	MKRRight	Marker Right
C	Display C	PERIOD	Period
D	Display D	RESAuto	Auto ResBW
CTRMeas	Center Measure	RESNarrow	ResBW narrower
DETMenu	Detector Menu	RESWide	ResBW wider
DIspmenu	Display Menu	SAve	Save
Fine	Fine Ref Lvl steps	SGLswp	Single Sweep
GRAT	Grat lights	SIGUp	Signal Up
INPutmenu	Input menu	SIGDown	Signal Down
M0	0 key	SPANDown	Span Down
M1	1 key	SPANUp	Span Up
M2	2 key	SWPAuto	Sweep Auto
M3	3 key	SWPFast	Sweep Faster
M4	4 key	SWPMenu	Swp/Trig Menu
M5	5 key	SWPSlow	Sweep Slower
M6	6 key	USERdef	User Def Menu
M7	7 key	UTImenu	Utility Menu
M8	8 key	VRTLog	Log Mode
M9	9 key	VRTLin	Lin Mode
MAXHold	Max Hold	VIDflt	Video Filter
MAXSpan	Max Span	Zerospan	Zero Span
MKREnab	Marker On/Off/Delta		

LRAmpl

Arguments: none

Command requiring no argument. It moves the primary marker from its current position to the next lower on-screen signal peak. If the marker is not on, the command turns it on. If signal track is enabled, LRAmpl turns it off, enables the primary marker, and assigns the knob to marker control. If SGErr is ON and a lower peak does not exist, an SRQ and event 896 are generated.

LRAmpl

MAMpl? <arg>

Arguments: none, PRImary, SECond, DELta

Simple query with one or no arguments that returns a linked response indicating the amplitude of the primary (<arg> = none or PRImary) or secondary (<arg> = SECond) marker, or their amplitude difference (<arg> = DELta). The applicable units are those currently selected for the reference level unit.

```
MAMpl?
MAMPL PRIMARY:6.8 (for example)
MAMpl? SECond
MAMPL SECond:2.4 (for example)
MAMpl? DELta
MAMPL DELTA:4.4 (for example)
```

MARker <arg>

Arguments: ON, OFF, SINGle, DELta

Single-argument command that turns markers on and off. Turning on a marker places the knob function in marker control and disables signal track mode, bandwidth measurement mode, noise measurement mode, and C/N measurement mode. The markers cannot be turned on in analog display or video monitor modes, on in waterfall mode unless the D register is enabled.

Argument	Action
ON	Turn on primary marker
SINGle	Turn on primary marker
DELta	Turn on both markers
OFF	Turn off all markers

```
MARker ON
MARker SINGle
MARker DELta
MARker OFF
```

MARker?

Arguments: none

Simple query returning the on/off status of the 2710 markers.

```
MARker?  
MARKER SINGLE  
MARKER DELTA  
MARKER OFF
```

MEMory?

Arguments: none

Query returning two integer numbers separated by a comma. The first number represents the total amount of free NVRAM. The second number represents the largest contiguous block of free NVRAM. The values depend on the options installed and the number of waveforms, settings, programs, etc. stored in the 2710. Values are always multiples of 16.

```
MEMory?  
MEMORY 16464,3296 (for example)
```

MEXchg

Arguments: none

Command requiring no argument which interchanges the primary (stationary) and secondary (moveable) markers. If delta marker mode is not active, the command generates an SRQ and event code 825.

```
MEXchg
```

MFreq <arg>

Arguments: number in the range -10 MHz to 1.8 GHz

Single argument command that sets the frequency of the primary marker. Units may be appended; otherwise Hertz are assumed. Although the over all range is -10 MHz to 1.8 GHz, the value specified must be currently within the analyzer on-screen frequency span or an SRQ and event code are generated. Not valid in zero span mode.

MFreq 193.25 Mhz (for example)

MFreq? <arg>

Arguments: none, PRImary, SECond, DELta

Simple query with one or no arguments that returns a linked response indicating the frequency of the primary (<arg> = none or PRImary) or secondary (<arg> = SECond) marker, or their frequency difference (<arg> = DELta). The units are Hertz

*MFreq?
MFREQ PRIMARY:193.25E+6 (for example)
MFreq? SECond
MFREQ SECOND:197.75E+6 (for example)
MFreq? DELta
MFREQ DELTA:4.5E+6 (for example)*

MKTime <arg>

Arguments: number in the range 0 to 20

Single argument command that sets the time of the primary marker in zero span mode. Units may be appended; otherwise seconds are assumed. Although the over all range is 0 to 20 seconds, the value specified must be currently within the analyzer on-screen time span or an SRQ and event code are generated. Not valid

unless in zero span mode.

MKTime 204 Msec (for example)

MKTime? <arg>

Arguments: none, PRImary, SECond, DELta

Simple query with one or no arguments that returns a linked response indicating the time of the primary (<arg> = none or PRImary) or secondary (<arg> = SECond) marker, or their time difference (<arg> = DELta). The units are seconds.

MKTime?
MKTIME PRIMARY:4.67E-4 (for example)
MKTime? SECond
MKTIME SECOND:8.98E-4 (for example)
MKTime? DELta
MKTIME DELTA:4.31E-4 (for example)

MLFtnxt

Arguments: none

Command requiring no argument. It moves the primary marker from its current position to the next signal peak to the left. If signal track is enabled, MLFtnxt turns it off, enables the primary marker, and assigns the knob function to marker control. If SGErr is ON and a peak does not exist, an SRQ and event code are generated.

MLFtnxt

MMAx

Arguments: none

Command requiring no argument. It moves the primary marker from its current position to the highest signal peak on screen. If signal track is enabled, MMAx turns it off, enables the primary marker, and assigns the knob function to marker control. If SGErr is ON and a higher peak does not exist, an SRQ and event code are generated.

MMAx

MNHid <arg>

Arguments: OFF, ON

Single-argument command that turns the minimum hold feature on and off. This command is not allowed if:

- Analog mode is being used
- Waterfall mode is enabled
- The destination register is A and display line is on dBuV/m is enabled and the destination register is the same as for min hold
- Ensemble averaging is enabled

MNHid ON
MNHid OFF

MNHid?

Arguments: none

Simple query returning the on/off status of the minimum hold feature.

MNHid?
MNHL ON
MNHL OFF

MPOs? <arg>

Arguments: none, PRImary, SECOnd, DELta

Query with one or no argument that returns a linked integer response indicating the horizontal position of the primary (<arg> = none or PRImary) or secondary (<arg> = SECOnd) marker, or their horizontal difference (<arg> = DELta). See the CURve command for an explanation of screen coordinates.

```
MKTime?  
MKTIME PRIMARY:356 (for example)  
MKTime? SECOnd  
MKTIME SECOnd:233 (for example)  
MTime? DELta  
MKTIME DELTA:123 (for example)
```

MRGTnxt

Arguments: none

Command requiring no argument. It moves the primary marker from its current position to the next signal peak to the right. If signal track is enabled, MRGTnxt turns it off, enables the primary marker, and assigns the knob function to marker control. If SGErr is ON and a peak does not exist, an SRQ and event code are generated.

```
MRGTnxt
```

MSGdlm <arg>

Arguments: Lf (line feed), Semicolon

Single-argument command that selects a semicolon or line feed character as a message delimiter.

```
MSGdlm Lf  
MSGdlm Semicolon
```

MSGdlm?

Arguments: none

Simple query whose response indicates the currently selected message delimiter.

```
MSGdlm?  
MSGDLM LF  
MSGDLM SEMICOLON
```

MSTep

Arguments: none

Command requiring no argument that is equivalent to turning the frequency/markers knob one click CCW. The effect on the 2710 depends on the currently selected knob function.

```
MSTep
```

MTUNE <arg>

Arguments: value in the range +/- 1.8 GHz

Single argument command that changes the frequency of the primary marker by the indicated amount. Negative values indicate a decrease in frequency. Although the over all range is +/- 1.8 GHz, the value specified must result in the new marker frequency being within the analyzer on-screen frequency span or an SRQ and event code are generated.

```
MTUNE 546 Khz (for example)
```

MVPos? <arg>

Arguments: none, PRIMary, SECond, DELta

Query with one or no argument that returns a linked inte-

ger response indicating the vertical position of the primary (<arg> = none or PRImary) or secondary (<arg> = SECOnd) marker, or their vertical difference (<arg> = DELta). The 0-point is established as the center or top of the screen by POFFset. See the CURve command for an explanation of screen coordinates.

```
MVPos?  
MVPOS PRIMARY:356 (for example)  
MVPos? SECOnd  
MVPOS SECOnd:233 (for example)  
MVPos? DELta  
MVPOS DELTA:123 (for example)
```

MXHId

Arguments: OFF, ON

Single-argument command that turns the 2710 maximum hold feature on and off.

```
MXHId ON  
MXHId OFF
```

MXHId?

Arguments: none

Simple query returning the current on/off status of the 2710 maximum hold feature.

```
MXHId?  
MXHLD ON  
MXHLD OFF
```

MXRvl <arg>

Arguments: NOMinal, number in range -50 to -20

Single-argument command setting the level of the signal at the input to the 2710 first mixer required for full-screen (top graticule line) deflection in 2 dB steps. Odd values are rounded. Units are not allowed; the number is interpreted as dBm. NOMinal selects the factory default -30 dBm value.

```
MXRvl NOMinal
MXRvl -24 (for example)
```

MXRvl?

Arguments: none

Simple query returning the signal amplitude at the input to the 2710 first mixer required to deflect the display to the top graticule line.

```
MXRvl?
MXRLVL -30 (for example)
```

MXSpn <arg>

Arguments: OFF, ON

Single-argument command turning the 2710 maximum span mode on and off. The 2710 returns to the previously selected span/division when max span is turned off.

```
MXSpn ON
MXSpn OFF
```

MXSpn?

Arguments: none

Simple query returning the current on/off status of the

2710 maximum span feature.

```
MXSpn?  
MXSPN ON  
MXSPN OFF
```

NNBw <arg>

Arguments: number in the range 1 Hz to 1.8 GHZ

Single-argument command that sets the noise bandwidth for normalized noise mode measurements. Units may be appended; otherwise Hertz are assumed.

```
NNBw 4 Mhz (for example)
```

NNBw?

Arguments: none

Simple query that returns the noise bandwidth in Hertz to be used for 2710 normalized noise mode measurements.

```
NNBw?  
NNBW 4.0E+6 (for example)
```

NNMode <arg>

Arguments: OFF, ON, IDLE

Single-argument command that turns the 2710 normalized noise measurement mode on and off. The command also sets TMODE to MARKER, ACQMODE to MAXMIN, and SGTRAK to OFF. The command cannot be used in analog mode, video monitor mode, waterfall mode when the D-register is off, or in linear display mode. Normalized noise mode measurements cannot be made on a saved waveform. The NNMode IDLE command is equivalent to NNMode ON. The IDLE version is provided because Tektronix Codes and Formats requires that you be able to send any response back to the instrument as a command, and IDLE is a possible response indicating the mode is enabled but no signal is present to measure.

```
NNMode ON  
NNMode OFF  
NNMode IDLE
```

NNMode IDLE has the same effect as NNMode ON.

NNMode?

Arguments: none

Simple query that returns the current status of the 2710 normalized noise measurement mode. IDLE indicates the noise is too close to the analyzer noise floor, the AM detector is not enabled, MAX SPAN is active, the waveform is saved, or the analyzer is in analog mode.

```
NNMode?  
NNMODE ON  
NNMODE OFF  
NNMODE IDLE
```

NNResult?

Arguments: none

Simple query that returns the result of the most recent normalized noise measurement. The result is updated at the end of each sweep when the normalized noise measurement mode is enabled. The units are those selected for the reference level units.

```
NNResult?  
NNRESULT -93.5 (for example)
```

NORM

Arguments: ALL, AMPlitude, FREquency, TG

Single-argument command instructing the 2710 to carry out the indicated normalizations.

```
NORM ALL (all normalization  
except reference)  
NORM AMPlitude (amplitude normalizations)  
NORM FREquency (frequency normalizations)  
NORM TG (tracking generator  
normalizations)
```

NORM?

Arguments: none

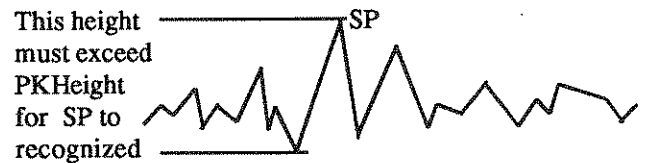
Simple query that returns a formatted listing of the current normalization parameters.

```
NORM?
NORM TEK 2710
CURRENT NORMALIZATION VALUES:
=====
MISCELLANEOUS - NORM VALUES
:
-----
CF NORMALIZATIONS
MAIN DAC SENS      460.94kHz
MAIN DAC OFFSET    94
FM DAC SENS        558.01Hz
MAXIMUM CF ERROR   1.51MHz
-----
REFERENCES
CALIBRATOR FREQ    99.997700MHz
CF REFERENCE       99.997700MHz
CALIBRATOR AMPLTD -30.00
GAIN STEP REFERENCE 10.40
-----
LOG NORMALIZATIONS
:
-----
SENSITIVITY
:
-----
FILTER AMPLITUDES
:
-----
VR GAIN
:
-----
RF ATTEN AND PREAMP GAIN
:
=====
:
```


PKHeight <arg>

Arguments: integer in the range 2 to 255

Single-argument command that specifies how high a signal peak must be in vertical display increments relative to the nearest local minimum in its skirts to be recognized by the "next lower" and "next higher" marker functions. 20 is the default value. Units are not allowed.



PKHeight 50 (for example)

PKHeight?

Arguments: none

Simple query that returns an integer representing the signal height in vertical display increments that must exist for the peak to be recognized by the "next lower" and "next higher" marker functions.

PKHeight?
PKHEIGHT 20 (for example)

PLLmode <arg>

Arguments: OFF, ON

Single-argument command that turns the 2710 phase lock option on and off. The phase lock option must be installed. Phase lock can only be enable for spans of 20 kHz/div and less.

PLLmode ON
PLLmode OFF

PLLmode?

Arguments: none

Simple query that returns the current on/off status of the 2710 phase lock option. The option must be installed.

```
PLLmode?  
PLLMODE ON  
PLLMODE OFF
```

PLOT?

Arguments: none

Simple query returning a complex response that transfers screen plot information from the 2710 to a plotter or printer. It is similar to the Utility Menu SCREEN PLOT option ([UTIL MENU]/[6]). The printer or plotter must speak either the HPGL language or be compatible with Epson FX codes. The appropriate printer type must first be specified either locally or with the PTyPe command.

```
PLOT?  
<screen data array up to 61.1Kbyte long>
```

The data array can be up to 61.1 kbytes long for Epson printers and up to 37 kbytes for HPGL plotters. PLOT? never produces a response header, even if HDR is ON.

See section 6 for programming examples.

POFset <arg>

Arguments: CENter, TOP

Single-argument command specifying whether to offset the result of the 2710 B,C MINUS A register arithmetic feature to the top or center of the display.

```
POFset CENter  
POFset TOP
```

POFset?

Arguments: none

Simple query that returns whether the result of the 2710 B,C MINUS A function is offset to the center or top of the display screen.

```
POFset?  
POFSET CENTER  
POFSET TOP
```

PRDouts?

Arguments: none

Simple query that returns a list of the 2710 on-screen readouts. There are up to 13 arguments depending on the current status and mode of operation of the 2710. The arguments, in order, are:

- Title
- Center/start frequency
- Reference level
- Span/division
- Resolution bandwidth
- Attenuation or marker/delta frequency/normalized noise/carrier-to-noise/occupied bandwidth/frequency count
- Video filter or marker/delta amplitude/noise bandwidth/dB down for bandwidth mode
- Vertical scale
- Video line/TV channel number/average count/D line
- Single sweep mode/arm
- Tracking generator amplitude or amplitude offset CALIBRATOR or tracking generator frequency offset UNCAL or FREQ COR OFF

The arguments are enclosed in quotation marks and separated by commas, and, as usual, the response ends in a semicolon. If an argument is missing, a null string ("") is returned. In principal, each argument can be up to 32 characters long (the string length is dimensioned for a

maximum of $13 \times 32 = 416$ characters), but this length is never achieved in practice. The query does not return the 2710 general purpose message line, GPIB status line, or user-defined "DISPLAY MESSAGE" line.

The response below would be returned after initialization of the 2710 with the factory power-up.

```
PRDouts?  
PRDOUTS "" "900MHZ" "20.0DBM"  
"180MHZ/MAX" "5MHZ RBW" "ATTN  
50DB" "VF WIDE" "10DB" "" "" "" ""
```

See section 6 for a programming example.

PREamp <arg>

Arguments: OFF, ON

Single-argument command that turns the 2710 built-in preamplifier on and off.

```
PREamp ON  
PREamp OFF
```

PREamp?

Arguments: none

Simple query returning the current on/off status of the 2710's built-in preamplifier.

```
PREamp?  
PREAMP ON  
PREAMP OFF
```

PROTset <arg>

Arguments: OFF, ON

Single-argument command that turns stored settings

protection on and off. Stored settings cannot be erased when PROTset in ON.

```
PROTset ON
PROTset OFF
```

PROTset ?

Arguments: none

Simple query returning the current on/off status of the stored settings protection. Stored settings cannot be erased when they are protected.

```
PROTset?
PROTSET ON
PROTSET OFF
```

PSTep

Arguments: none

Command requiring no argument that is equivalent to turning the frequency/markers knob one click CW. The effect on the 2710 depends on the currently selected knob function.

```
PSTep
```

PTYPE <arg>

Arguments: EPSON, HPGL2, HPGL4

Single-argument command that specifies the type of printer/plotter encoding to use for screen data transferred from the 2710 to the controller in response to the PLOT? query.

```
PTYPE EPSON
PTYPE HPGL2
PTYPE HPGL4
```

PTYPE?

Arguments: none

Simple query returning the type of printer/plotter currently selected for use with the PLOT? query.

```
PTYPE?  
PTYPE EPSON  
PTYPE HPGL2  
PTYPE HPGL4
```

RECALL <arg>

Arguments: integer in the range 0 to 9

Single-argument command instructing the 2710 to recall the stored settings in the location indicated by the argument.

```
RECALL 0  
:  
RECALL 9
```

REDOUT <arg>

Arguments: OFF, ON

Single-argument command that turns the 2710 on-screen readouts on and off.

```
REDOUT ON  
REDOUT OFF
```

REDOUT?

Arguments: none

Simple query returning the on/off status of the 2710 on-screen readouts.

```
REDout?  
REDOUT ON  
REDOUT OFF
```

REFMl <arg>

Arguments: DEC,INC
reference level in the range -70 to +20 dBm

Single-argument command that increases, decreases, or sets the reference level.

If INC or DEC is the argument, the command increases or decreases the reference level by 1 or 10 dB depending on the FINE command or the local 1dB/10dB setting.

If a numeric argument is used, it must be in the range -70 to +20 dBm. However, alternate units (DBM,DBMV, DBV, DBUV, DBUW, DBUVM) may be used as long as their dBm equivalent is within the acceptable range. If no units are used, the current reference level units are assumed. If units other than the current units are used, the value will be converted to the current units.

If automatic RF attenuation is enabled, the amount of attenuation may be changed. If linear display mode is active, the scale factor will be recomputed. All values are interpreted according to the current reference level offset and impedance correction.

```
REFMl INC  
REFMl DEC  
REFMl 10 DBMV (for example)
```

REFM?

Arguments: none

Simple query that returns the current reference level in the currently selected reference level units.

```
REFM?  
REFLVL -35.0 (for example)
```

RESbw <arg>

Arguments: INC, DEC
bandwidth of a 2710 resolution BW filter

Single-argument command that increases, decreases, or selects the resolution BW.

If INC or DEC is the argument, the command increases or decreases the resolution BW to the next installed resolution BW filter.

If a numeric argument is used, the installed resolution BW filter closest in value is selected. Bandwidths available depend on the installed options. Use of this command disables automatic Resolution BW selection. If units are not attached, Hertz are assumed.

```
RESbw INC  
RESbw DEC  
RESbw 30 KHz (for example)
```

RESbw?

Arguments: none

Simple query that returns the currently selected resolution BW in Hertz.

```
RESbw?  
RESBW 3 0E+4 (for example)
```

RFAtt <arg>

Arguments: number in the range 0 to 50

Single-argument command that sets the 2710 RF attenuation to a fixed value between 0 and 50 dB in 2 dB steps. Values other than even integers are rounded. Units are not allowed.

```
RFAtt 34 (for example)
```


RFAtt?

Arguments: none

Simple query that returns the current RF attenuation in dB whether it is fixed or automatically selected.

```
RFAtt?  
RFATT 34 (for example)
```

RLUnit <arg>

Arguments: DBM,DBMV,DBV,DBUV,DBUW,DBUVM

Single-argument command that specifies the indicated units for the reference level.

The DBUVM is not allowed under the following conditions:

- linear display mode
- DBUVM result is already saved
- display source is the FM detector or external source
- there is a destination conflict with ensemble average, minimum hold, or display line

```
RLUnit DBM  
RLUnit DBMV  
RLUnit DBV  
RLUnit DBUV  
RLUnit DBUW  
RLUnit DBUVM
```

RLUnit?

Arguments: none

Simple query that returns the currently selected reference level units.

```
RLUnit?  
RLUNIT DBM  
RLUNIT DBMV  
RLUNIT DBV  
RLUNIT DBUV  
RLUNIT DBUW  
RLUNIT DBUVM
```

ROFset <arg>

Arguments: OFF, ON, value in the range +/- 100 dB

Single-argument command that turns the reference level offset on and off and sets its value. The offset must be in the range -100 to +100 dB. Units are not allowed.

```
ROFset ON  
ROFset OFF  
ROFset -7.5 (for example)
```

ROFset?

Arguments: none

Simple query that returns the current reference level offset. If the offset is disabled, the query returns 0. The units are dB.

```
ROFset?  
ROFSET -7.5 (for example)
```

RQS <arg>

Arguments: OFF, ON

Single-argument command that enables and disables the generation of service requests (SRQ) by the 2710. The user request is effected but power-on SRQ is not.

```
RQS ON
RQS OFF
```

RQS?

Arguments: none

Simple query that returns the on/off status of 2710 service request (SRQ) generation.

```
RQS?
RQS ON
RQS OFF
```

SAVe <arg>

Arguments: A:ON, A:OFF, B:ON, B:OFF, C:ON, C:OFF
combinations of above

Command with a single or multiple linked arguments that saves and deletes waveforms in NVRAM. ON saves the indicated display register to NVRAM. OFF deletes a saved display register from NVRAM.

For instance, SAVE A:ON saves the current contents of the A-register to NVRAM and halts the A-register from updating. SAVE A:OFF deletes the saved A-register waveform from NVRAM and permits the A-register to be updated on each sweep.

Single or multiple arguments can be used in a single command:

```
SAVe A:ON (for example)
SAVe C:OFF (for example)
SAVe A:ON,B:OFF (for example)
SAVe A:ON,B:OFF,C:OFF (for example)
```

Using the SAVE <link>:OFF command with the

destination register for an ensemble average or minimum hold operation also terminates those operations. Also, you cannot turn off the DBUVM destination register without first exiting DBUVM mode.

SAVe? <arg>

Arguments: none, A, B, C

Query with one or no argument that returns the storage state of the indicated register or all registers. If no argument is used, the state of the A, B, and C register is returned. If an argument is used, only the state of the indicated register is returned.

```
SAVe?  
SAVE A:ON,B:OFF,C:OFF (for example)  
SAVe? A  
SAVE A:ON (for example)  
SAVe? B  
SAVE B:OFF (for example)  
SAVe? C  
SAVE C:OFF (for example)
```

SET?

Arguments: none

Simple query returning a group of command headers and arguments representing the current operating environment of the 2710. The string of commands can be retained for transfer to the same or another 2710 at a later time when it is desirable to reproduce the same operating environment. Only those functions necessary for recreating the operating environment are returned.

HDR has no effect on the SET? query. Individual command headers are always returned and the group header (SET) never is. Each header and argument is separated by a semicolon. This ensures that the response always represents a functional message group.

Although the SET? response is lengthy, it is straight forward and easy to interpret. The response shown below includes all 95 command headers that can be returned. Fewer headers are possible depending on the options installed on your instrument and its particular configuration at the time of the SET? query. For example:

- If the phaselock, tracking generator, or video monitor options are not installed, all the commands related to them will be absent (PLLMODE and all commands beginning TG..., TVL..., and V... except VSYNC).
- If the instrument is not in single sweep mode or the display line is turned off, the SIGSWP and DLVALUE commands will be missing, settings that can be returned.

SET?

```
VIEW WATERFALL:OFF;RECALL 1;DSRC AM;VRTDSP LOG:
10;DETECTOR OFF;NNBW 1.0E+0;NNMODE ON;CNBW
1.0E+0;CNMODE ON;BWNUM -3;BWMODE ON;ACQMODE
MAXMIN;SPAN 180.E+6;MXSPN ON;ZEROSP OFF;
FOFFSET 0.000;FOMODE OFF;CFSF CENTER;FREQ
900.E+6;PLLMODE ON;DISCOR OFF;VMANTTBL 1;VMDIST
3.0E+0;VMDEST C;VMMKRUNIT DBUVM;IMPCOR 50;
ROFSET 0.0;ROMODE OFF;RLUNIT DBM;WAIT;PREAMP
OFF;MXRLVL -30;REFLVL 20.0;RFATT 50;ARFATTON;FINE
OFF;THRHL D -20.0;ATHRHL D ON;PKHEIGHT 20;DLVALUE
-20.0;DLINE ON;DLLIMIT OFF;RESBW 5.0E+6;ARES ON;
VIDFLT 5.0E+6;VFMODE AUTO;VFENAB OFF;MARKER
OFF;CRES1.E+3;SGTRAK OFF;AVDEST C;AVMODE MEAN;
AVNUM 16;AVG OFF;MNHLD OFF;MXHLD OFF;POFSET
CENTER;VIEW A:OFF;B:OFF;C:OFF;D:ON;MINUSA:OFF;
STEP 3.600E+6;STPINC AUTO;TABLE 0;TGTRACK 0.000;
TGTMODE OFF;TGGOFFSET 0.0;TGOMODE OFF;TGMAN
OFF;TGLEVEL -48.0;TGENAB OFF;TITLE "";TTLMODE OFF;
TEXT "";TEDOUT ON;GRAT OFF;PROTSET OFF;PTYPE
HPGL2;CAL SIG OFF;MSGDLM SEMICOLON;HDR ON;EOS
OFF;SGERR OFF;RQS ON;TVLSTD NTSC;VDMODE BROAD
CAST;VPOLARITY NEGATIVE;VSYNC POSITIVE;TVLINE 6;
TVLMODE CONT;TIME 50.E-3;TIMMODE AUTO;TRIGGER
FRERUN;SIGSWP:TMODE FREQUENCY;VMONITOR ON;
TIME 50.E-3;SSBEGIN -10.000E+6;SSEND 1.036E+9;
```

Although the intent of the SET? response is to enable you to replicate equipment setups, it is also possible to view and edit the response with most text editors and word processors. This is a handy characteristic for checking, or even modifying equipment settings.

SGErr <arg>

Arguments: OFF, ON

Single-argument command that enables and disables the generation of a service request (SRQ) when a marker function is unable to find a signal. SGErr ON enables SRQ generation for event 896.

```
SGErr ON
SGErrOFF
```

SGErr?

Arguments: none

Simple query that returns the on/off status of service request (SRQ) generation when a marker function cannot find the intended signal.

```
SGErr?
SGERR ON
SGERR OFF
```

SGSrch

Arguments: none

Command requiring no argument that instructs the 2710 to perform a signal search between the current begin and end frequencies for all signals greater than the threshold (see THRhld). The beginning and end frequencies can

be set locally or by using the SSBEGIN and SSEND commands. Results of the search are returned by SSRESULT?

SGsrch

SGTrak <arg>

Arguments: OFF, ON

Single-argument command that enables and disables signal track mode.

SGTrak ON
SGTrak OFF

SGTrak?

Arguments: none

Simple query that returns the on/off status of the 2710 signal track mode.

SGTrak?
SGTRAK ON
SGTRAK OFF

SIGswp

Arguments: none

Command requiring no argument that selects and arms the 2710 single sweep mode. The sweep doesn't actually occur until the trigger conditions for the currently selected trigger mode are satisfied. Any TRIGGER command cancels single sweep mode.

SIGswp

SIGswp?

Arguments: none

Simple query that returns the current status of the 2710 single sweep mode.

```
SIGswp?  
SIGSWP ON,  
SIGSWP OFF  
SIGSWP ARM
```

SPAn <arg>

Arguments: 0, INC, DEC
value in the range 10 kHz to 180 MHz
(1 kHz to 180 MHz for 2710s with Option 1)

Single-argument command that increases, decreases, or sets the 2710 span/division.

When used with the INC or DEC argument, the command changes the span/division in the indicated direction in the 2710's normal 1-2-5 sequence.

For a numeric argument other than zero, the span is set to the indicated value. If the value is out of range, the end point is substituted and an SRQ and event code are generated. If 0 (zero) argument is used, zero span mode is activated.

```
SPAn INC  
SPAn DEC  
SPAn 0  
SPAn 25 KHz (for example)
```

SPAn?

Arguments: none

Simple query returning the current span/div in Hertz.

SPAn?
SPAN 2.5E+4 (for example)

SSBegin <arg>

Arguments: value in the range 10 kHz to 1.8 GHz

Single-argument command that specifies the beginning frequency for the 2710 signal search mode. For the search to be meaningful, the beginning frequency must be less than the end frequency. Units may be appended; otherwise Hertz are assumed. The value is assumed to be offset by FOFset if FOMode is ON.

SSBegin 54 Mhz (for example)

SSBegin?

Arguments: none

Simple query that returns the currently specified beginning frequency in Hertz for the 2710 signal search mode.

SSBegin?
SSBEGIN 54.000e+6 (for example)

SSEnd <arg>

Arguments: value in the range 10 kHz to 1.8 GHz

Single-argument command that specifies the end frequency for the 2710 signal search mode. For the search to be meaningful, the end frequency must be greater than the beginning frequency. Units may be appended; otherwise Hertz are assumed. The value is assumed to be offset by FOFset if FOMode is ON.

SSEnd 300 Mhz (for example)

SSEnd?

Arguments: none

Simple query that returns the currently specified end frequency in Hertz for the 2710 signal search mode.

```
SSEnd?  
SSEND 300.00E+6 (for example)
```

SSResult?

Arguments: none

Simple query that returns the number of signals detected during a 2710 signal search operation, and lists the frequency and amplitude of each. Up to 50 frequency - amplitude pairs can be returned. The frequency and amplitude values are separated by commas, and the pairs of values are also delimited by commas. The list begins with the lowest frequency signal detected and proceeds to the highest. The amplitude units are those currently selected as reference level units; frequency is in Hertz.

The general form of the response with HDR ON looks like this:

```
SSRESULT <N>,<freq1>,<ampl1>,...,<freqN>,<amplN>;
```

where:

<N> = number of signals detected
($N \leq 50$)

<freq1>,...,<freqN> = frequency of 1st,...,Nth
detected signal

<ampl1>,...,<amplN> = amplitude of 1st,...,Nth
detected signal

If no signals were detected during the search, the SSResult? query returns zero (i.e., with HDR ON, the response is SSRESULT 0;). If the amplitude of a

detected signal is off-screen, it is listed as 1.0E+6.

```
SSResult?  
SSRESULT 55.250E+6,7.3,...299.75E+6,-4.0;  
      (for example)
```

See Section 6 for programming examples.

STEp <arg>

Arguments: CF, MARKer
number in the range 1 Hz to 1.8 GHz

Single-argument command specifying the programmed frequency tuning increment stepsize. Specifying the step size also turns on the 2710 programmed tuning mode. CF argument selects the current center frequency as the step size, MARKer selects the current marker frequency, and a numeric argument specifies the increment in Hertz. Units may be appended.

```
STEp CF  
STEp MARKer  
STEp 30 KHz (for example)
```

STEp?

Arguments: none

Simple query that returns the currently specified programmed frequency tuning increment in Hertz.

```
STEp?  
STEP 3.0E+4 (for example)
```

STOre <arg>

Arguments: integer in the range 2 to 9

Single-argument command that stores the current 2710 control setup in the location designated by the argument. Locations 0 and 1 are reserved for the last 2710 power-down and factory default power-up settings respectively.

```
STOre 2
```

```
STOre 9
```

STPinc <arg>

Arguments: AUTO, TABular, PROg

Single-argument command that selects the tuning increment mode as automatic, tabular, or programmed. To set the programmed increment, see STEp command.

```
STPinc AUTO  
STPinc TABular  
STPinc PROg
```

STPinc?

Arguments: none

Simple query returning the currently selected 2710 tuning increment mode.

```
STPinc?  
STPINC AUTO  
STPINC TABULAR  
STPINC PROG
```

STStop <arg>

Arguments: MARKer
pair of values in range -10 MHz to 1.8 GHz

Single- or double argument command that sets the start and stop frequencies of the 2710 display.

If MARKer is the argument, the start and stop frequencies are set to the current marker frequencies (delta marker mode must be enabled).

If the argument is a pair of numbers, the first number specifies the start frequency and the second specifies the stop frequency. Units may be appended; otherwise Hertz are assumed.

The second number must be at least 100 kHz greater than the first (10 kHz if option 1 is installed) to satisfy 2710 span requirements. If the second number is greater than the first, but by less than 100 kHz (10 kHz/option 1), the 2710 will set the stop frequency to the start frequency plus 100 kHz (10 kHz/option1).

If the stop frequency is set lower than the start frequency, the 2710 is tuned to the lower frequency and zero span mode is activated.

STStop MARKer
STStop 192 Mhz, 198 Mhz (for example)

TABLE <arg>

Arguments: integer number in the range 0 to 9

Single-argument command that selects the tuning table to be used when tabular tuning increment mode is active. Does not turn on tabular tuning, only selects the table. The number of an empty table can be entered, but an SRQ and event code will be generated.

TABLE 0
TABLE 7

TABLE?

Arguments: none

Simple query that returns the currently selected tabular tuning table.

```
TABLE?  
TABLE 3 (for example)
```

TAMp?

Arguments: none

Simple query that returns the amplitude of the signal being tracked. Updated at the end of each sweep when signal track mode is enabled. Otherwise, returns the the amplitude of the signal last tracked. The units are those currently selected as reference level units.

```
TAMp?  
TAMPL -34.0 (for example)
```

TEXT <arg>

Arguments: string of up to 32 ASCII characters

Single-argument command that displays a message on line 8 of the 2710 screen (line 9 if title mode is active). The message is the argument of the command which may be up to 32 characters long. Quotation marks must be used. Only upper-case characters can be displayed, although lower case can be sent. To erase the message, transmit a null string (TEXT "").

```
TEXT "MY MESSAGE" (for example)
```

TEXT?

Arguments: none

Simple query that returns the current contents of the message buffer in the 2710. If lower-case letters were originally sent, lower case is returned even though the on-screen message is upper case.

```
TEXT?  
TEXT "MY MESSAGE" (for example)
```

TFReq?

Arguments: none

Simple query that returns the frequency of the signal being tracked in Hertz. Updated at the end of each sweep when signal track mode is enabled. Otherwise, returns the the frequency of the signal last tracked.

```
TFReq?  
TFREQ 101.36E+6 (for example)
```

TGEnab <arg>

Arguments: OFF, ON

Single-argument command that turns the optional tracking generator on and off. The tracking generator must be installed.

```
TGEnab ON  
TGEnab OFF
```

TGEnab?

Arguments: none

Simple query that returns the current on/off status of the optional tracking generator.

```
TGEnab?  
TGENAB ON  
TGENAB OFF
```

TGLevel <arg>

Arguments: number in range equivalent to 0 to -48 dBm

Single-argument command that sets the output amplitude of the 2710 optional tracking generator. The value specified may be in DBM, DBMV, DBV, DBUV, or DBUW (DBUVM defaults to DBUV), but its equivalent value must fall in the range -48 to 0 dBm. The level can be changed in 0.1 dB steps. If units are not appended, current reference level units are assumed.

```
TGLevel 1.2 DBMV (for example)
```

TGLevel?

Arguments: none

Simple query that returns the currently specified output amplitude of the optional 2710 tracking generator. The units are those currently selected as the reference level units.

```
TGLevel?  
TGLEVEL 1.2 (for example)
```

TGMan <arg>

Arguments: OFF, ON

Single-argument command that enables and disables fine manual adjustment of the 2710 optional tracking generator output amplitude. In manual mode, the 2710 trigger level knob adjusts the tracking generator output amplitude several dB about the level established with TGLevel. When TGMan is ON, the level returned by TGLevel? may differ slightly from the actual level.

```
TGMan ON  
TGMan OFF
```


TGMan?

Arguments: none

Simple query that returns the on/off status of manual tracking generator amplitude control.

```
TGMan?  
TGMAN ON  
TGMAN OFF
```

TGOMode <arg>

Arguments: OFF, ON

Single-argument command that turns the output level offset of the 2710 optional tracking generator on and off.

```
TGOMode ON  
TGOMode OFF
```

TGOMode?

Arguments: none

Simple query that returns the current on/off status of the tracking generator output offset.

```
TGOMode?  
TGOMODE ON  
TGOMODE OFF
```

TGOffset <arg>

Arguments: value in the range +/- 100 dB

Single-argument command that specifies the output level offset of the 2710 optional tracking generator. The offset can range from -100 dB to +100 dB. Units are not allowed. A non-zero argument turns offset mode on and a 0 argument turns the offset mode off.

```
TGOffset -10.5 (for example)
```

TGOffset?

Arguments: none

Simple query returning the currently specified output level offset of the optional tracking generator in dB.

```
TGOffset?  
TGOFFSET -10.5 (for example)
```

TGMode <arg>

Arguments: OFF, ON

Single-argument command that turns the frequency offset of the 2710 optional tracking generator on and off.

```
TGMode ON  
TGMode OFF
```

TGMode?

Arguments: none

Simple query that returns the current on/off status of the tracking mode of the 2710 optional tracking generator.

```
TGMode?  
TGMODE ON  
TGMODE OFF
```

TGTRack <arg>

Arguments: value in the range -5.01 kHz to +60 kHz

Single-argument command that specifies the tracking adjustment of the 2710 optional tracking generator. Units may be appended; otherwise Hertz are assumed.

Specifying a non-zero value also turns on the tracking generator tracking mode; a zero value turns the tracking mode off.

TGTRack 9.7 Khz (for example)

TGTRack?

Arguments: none

Simple query that returns the currently specified tracking value of the 2710 optional tracking generator in Hertz.

*TGTRack?
TGTRACK 9.7E+3 (for example)*

THRhd <arg>

Arguments: number within the range -174 to 20 dBm

Single-argument command that specifies the value of the threshold above which the 2710 automatically detects signals. Units of dBm, dBmV, dBV, dBuV, dBuW, and dBuV/m can be used, but the equivalent value must fall within the range -174 to +20 dBm. If units are not supplied, the current reference level units are assumed. This command also turns off the 2710 automatic threshold selection mode.

THRhd -10 DBMV (for example)

THRhd?

Arguments: none

Simple query that returns the current 2710 threshold value whether it is fixed or automatically selected. The units are the currently selected reference level units.

```
THRhd?  
THRHL -10.0 (for example)
```

TIME <arg>

Arguments: INC, DEC
value in the range 1 microsec to 2 sec

Single-argument command that increases, decreases, or sets the 2710 sweep speed. Also turns off 2710 automatic sweep speed selection. Units of NS, US, MS, or S may be appended; otherwise seconds are assumed.

If an INC or DEC argument is used, the sweep speed is increased or decreased in the 2710's normal 1-2-5 sequence. Sweep times which are not in sequence are increased to the next permissible setting.

If a numeric argument is used, it must be in the range of 1 microsecond to 2 seconds. Sweep times less than 100 microseconds are not permitted in digital display mode.

```
TIME INC  
TIME DEC  
TIME 25 US (for example)
```

TIME?

Arguments: none

Simple query that returns the currently selected time base (sweep speed). Units are in seconds.

```
TIME?  
TIME 25.E-6 (for example)
```

TIMMode <arg>

Arguments: AUTO, MANUAL, FIXED

Single-argument command that enables (AUTO) and disables (FIXED) 2710 automatic sweep speed selection, or enables manual sweep positioning (MANUAL) using the 2710 LEVEL control. In FIXED mode, the sweep speed is controlled locally or with the TIME command.

```
TIMMode AUTO  
TIMMode FIXED  
TIMMode MANUAL
```

TIMMode?

Arguments: none

Simple query that returns the currently selected time base mode.

```
TIMMode?  
TIMMODE AUTO  
TIMMODE FIXED  
TIMMODE MANUAL
```

TITLe <arg>

Arguments: string of up to 32 ASCII characters

Single-argument command that displays a title on line 1 of the 2710 screen. The title is the argument of the command which may be up to 32 characters long. Quotation marks must be used. Only upper-case characters can be displayed. To erase the title, transmit a null string (TITLe ""). Use the TTLMode command to turn the title on or off.

```
TITLe "SCREEN 1" (for example)
```

TITLe?

Arguments: none

Simple query that returns the 2710 screen title if one currently exists. If the title was sent lower case, the returned string will be lower case even though the title can only be displayed upper case on the 2710.

TITLe?
TITLE "SCREEN 1" (for example)

TMOde <arg>

Arguments: FREquency, MARker, TG, VIDline

Single-argument command that selects the 2710 frequency/marker knob function.

Argument	Function
FREquency	adjust start or center frequency
MARker	adjust marker frequency
TG	adjust tracking generator tracking
VIDline	select video line number if knob selectable TV line triggering is enabled

TMOde FREquency
TMOde MARker
TMOde TG
TMOde VID line

TMOde?

Arguments: none

Simple query that returns the currently selected function of the 2710 frequency/markers knob.

```
TMOde?  
TMODE FREQUENCY  
TMODE MARKER  
TMODE TG  
TMODE VIDLINE
```

TOPsig

Arguments: none

Command requiring no argument that instructs the 2710 to change the reference level to the amplitude of the primary marker. Marker must be enabled.

```
TOPsig
```

TRigger <arg>

Arguments: EXTERNAL, FRERun, INTERNAL, LINE, TVField, TVLine

Single-argument command that selects the 2710 trigger type. TVLine also sets the knob function to vidline if knob-selectable TV line mode is enabled.

```
TRigger EXTERNAL  
TRigger FRERun  
TRigger INTERNAL  
TRigger LINE  
TRigger TVField  
TRigger TVLine
```

TRIGGER?

Arguments: none

Simple query that returns the currently selected 2710 trigger type.

```
TRIGGER?  
TRIGGER FRERUN  
TRIGGER EXTERNAL  
TRIGGER INTERNAL  
TRIGGER LINE  
TRIGGER TVFIELD  
TRIGGER TVLINE
```

TTLMode <arg>

Arguments: OFF, ON

Single-argument command that turns the 2710 screen title on and off.

```
TTLMode ON  
TTLMode OFF
```

TTLMode?

Arguments: none

Simple query that indicates whether the 2710 screen title is being displayed (ON) or not (OFF).

```
TTLMode?  
TTLMODE ON  
TTLMODE OFF
```

TUNE <arg>

Arguments: number in the range -10^7 to 1.8×10^9

Single-argument command that changes the start or center frequency by the amount of the argument. The resultant frequency must also remain within the range of -10 MHz to 1.8 GHz. Units may be appended; otherwise Hertz are assumed.

TUNe 10.8 Mhz (for example)

TVLine<arg>

Arguments: integer in the range 1 to 1024

Single-argument command that specifies the number of the TV line that the 2710 is to trigger on when programmed TV line triggering is enabled. Also turns off video monitor mode if it is enabled. The minimum argument is 1 and the maximum value depends on the TV line standard. It is 525 for NTSC, 625 for PAL and SECAM and 1024 for OPEN. This command also sets TRIGGER to TVLine and enables the programmed mode.

TVLine 17 (for example)

TVLine?

Arguments: none

Simple query that returns the (integer) TV line number that the 2710 is to trigger on when TV line trigger mode is selected.

TVLine?
TVLINE 17 (for example)

TVLMode <arg>

Arguments: CONT, KNOB, PROg

Single-argument command that designates the specific 2710 TV line trigger mode, and enables TVLine trigger mode. Turns of the video monitor mode if it is enabled.

Argument	Function
CONT	trigger on every line
KNOB	trigger line number selected with the 2710 frequency/markers knob
PROg	trigger line number entered locally or with the TVLine command

TVLMode KNOB (for example)

TVLMode?

Arguments: none

Simple query that returns the currently selected 2710 specific TV line trigger mode.

```
TVLMode?
TVLMODE CONT
TVLMODE KNOB
TVLMODE PROG
```

TVLStd <arg>

Arguments: NTSC, OPEN, PAL, SECAM

Single-argument command that designates the TV standard when using the TV line trigger mode. Also turns off the video monitor mode if it is enabled, and sets the trigger mode to TV Line. The maximum value of the argument of the TVLine command is influenced by the TV standard.

```
TVStd NTSC
TVStd OPEN
TVStd PAL
TVStd SECAM
```

TVStd?

Arguments: none

Simple query that returns the currently selected TV standard.

```
TVStd?
TVSTD NTSC
TVSTD OPEN
TVSTD PAL
TVSTD SECAM
```

VDMode <arg>

Arguments: BROADCAST, SATELLITE

Single-argument command that designates the type of detection to be used in the 2710 video monitor mode. BROADCAST selects AM detection for use with broadcast television; SATELLITE selects FM detection for use with satellite transponders.

```
VDMode BROADCAST
VDMode SATELLITE
```

VDMode?

Arguments: none

Simple query that returns the currently selected detection for video monitor mode.

```
VDMode?
VDMODE BROADCAST
VDMODE SATELLITE
```

VFMode <arg>

Arguments: AUTO, FIXEd

Single-argument command that enables (AUTO) and disables (FIXEd) automatic selection of the video filter bandwidth by the 2710. When fixed mode is first entered, the automatically selected video filter bandwidth is made the current fixed filter bandwidth. A new fixed filter bandwidth can then be selected locally or by using the VIDflt command.

```
VFMode AUTO
VFMode FIXEd
```

VFMode?

Arguments: none

Simple query whose response indicates whether the video filter bandwidth is fixed or automatically selected by the 2710

```
VFMode?
VFMODE AUTO
VFMODE FIXEd
```

VFEnab <arg>

Arguments: OFF, ON

Single-argument command that turns the 2710 video filter on and off.

```
VFEnab ON
VFEnab OFF
```

VFEnab?

Arguments: none

Simple query that returns the current on/off status of the 2710 video filter.

```
VFEnab?  
VFENAB ON  
VFENAB OFF
```

VIDflt <arg>

Arguments: OFF, ON, floating point number

Single-argument command that turns the 2710 video filter on or off, or specifies the filter bandwidth.

ON and OFF arguments enable and disable the currently selected video filter in the same way as the VFEnab command.

A numeric argument is used to specify a particular video filter bandwidth and turn on the video filter. Units may be appended; otherwise Hertz are assumed. The 2710 video filter bandwidths follow a 1-3 sequence. The video filter bandwidth closest to the specified filter width is selected.

```
VIDflt ON  
VIDflt OFF  
VIDflt 30 Khz (for example)
```

VIDflt?

Arguments: none

Simple query that returns the currently selected video filter bandwidth in Hertz whether the filter is enabled or not.

```
VIDflt?  
VIDFLT 3.0E+4 (for example)
```

VIEW <arg>

Arguments: A:ON, A:OFF, B:ON, B:OFF, C:ON, C:OFF, D:ON, D:OFF, Minusa:ON, Minusa:OFF, Waterfall:ON, Waterfall:OFF, combinations of above

Command with single or multiple linked arguments that enables and disables digital display mode in the indicated register.

For instance, VIEW A:ON turns on the digital display in the A-register; VIEW A:OFF turns it off.

Single or multiple arguments can be used in a single command:

```
VIEW B:ON (for example)
VIEW A:ON,B:OFF (for example)
VIEW A:ON,B:OFF,C:OFF (for example)
VIEW Waterfall:ON,A:OFF (for example)
```

Separate multiple arguments with commas.

VIEW?

Arguments: none, A, B, C, D, Minusa, Waterfall

Query with one or no argument that returns the on/off status of the indicated register or all registers. If no argument is used, the status of the A, B, C, and D registers, and the waterfall and B,C minus A display modes is returned. If an argument is used, only the state of the indicated register is returned.

```
VIEW?
VIEW Waterfall:OFF,A:ON,B:OFF,C:OFF,
D:ON,Minusa:OFF (for example)
VIEW? B
VIEW B:OFF (for example)
```

VMAnttbl <arg>

Arguments: integer in the range 1 to 5

Single-argument command that designates, by number, the antenna table to be used for DBUVM measurements. Units are not allowed.

VMAnttbl 3 (for example)

VMAnttbl?

Arguments: none

Simple query that returns the number of the antenna table currently selected for use when making DBUVM measurements.

*VMAnttbl?
VMATTBL 3 (for example)*

VMDEst <arg>

Arguments: A, B, C

Single-argument command that designates the 2710 display register used as the destination for DBUVM measurements.

VMDEst B (for example)

VMDEst?

Arguments: none

Simple query that returns the currently selected destination register for DBUVM measurements.

*VMDEst?
VMDEST B (for example)*

VMDist <arg>

Arguments: floating point number

Single-argument command specifying the source-antenna distance at which a DBUVM measurement is actually performed. You can enter the distance in feet (FT), meters (M), kilometers (KM), or miles (MI) but the 2710 converts to meters or kilometers.

```
VMDist 3 M (for example)
```

VMDist?

Arguments: none

Simple query that returns the currently specified source-antenna distance for DBUVM measurements. Units are meters.

```
VMDist?  
VMDIST 3.0 (for example)
```

VMMkrunit <arg>

Arguments: DBUVM, VM

Single-argument command that specifies the amplitude units for marker readouts in DBUVM mode as DBUVM or volts/m.

```
VMMkrunit DBUVM  
VMMkrunit VM
```

VMMkrunit?

Arguments: none

Simple query that returns the currently selected units for marker readouts in the DBUVM mode.


```
VMMkrunit?  
VMMKRUNIT DBUVM  
VMMKRUNIT VM
```

VMOonitor <arg>

Arguments: OFF, ON

Single-argument command that turns the 2710 optional video monitor on and off.

```
VMOonitor ON  
VMOonitor OFF
```

VMOonitor?

Arguments: none

Simple query that returns the current on/off status of the 2710 optional video monitor.

```
VMOonitor?  
VMONITOR ON  
VMONITOR OFF
```

VPOlarity <arg>

Arguments: NEGative, POSitive

Single-argument command that specifies the polarity of the video signal to be received with the 2710 optional video monitor.

```
VPOlarity NEGative  
VPOlarity POSitive
```

VPOlarity?

Arguments: none

Simple query that returns the currently specified video signal polarity.

```
VPOlarity?
VPOLARITY NEGATIVE
VPOLARITY POSITIVE
```

VRTdsp <arg>

Arguments: LOG:<num>, LIN:<num>, FM:<num>
EXT:<num>

Command with a single linked argument that specifies the vertical scale factor for the indicated display mode. If the default AM source is displayed, LOG and LIN also change display modes. The DSRC command must be used to enter FM or EXT modes, although VRTDsp still sets the scale factor.

Link	<num>	Units	Display Type
LOG:	10, 5, 1	dB/div	logarithmic
LIN:	8.839 to 279.5*	uV/div mV/div	linear
FM:	10, 5, 1	kHz/div	linear
EXT:	17.5, 87.5, 175	mV/div	linear

* corresponds to reference levels of -70 to +20 dBm

If units are not supplied, the LIN argument assumes volts (i.e., LIN:1 = LIN:100 Mv), FM assumes Hertz, and EXT assumes volts.

```
VRTdsp LOG:5 (for example)
VRTdsp LIN:50 Uv (for example)
VRTdsp FM:5 KHz (for example)
VRTdsp EXT:175E-3 (for example)
```

VRTdsp? <arg>

Arguments: none, LOG, LIN, FM, EXT

Query with one or no argument that returns a linked response indicating a vertical scale factor. When the query is used without an argument, it returns the currently selected scale factor. Units are dB for LOG, volts for LIN or EXT, and Hertz for FM. When used with an argument, it returns the scale factor used when the indicated mode was last entered.

```
VRTdsp?  
VRTDSP LOG:5 (for example)  
VRTdsp? LOG  
VRTdsp? LOG:5 (for example)  
VRTdsp? LIN  
VRTDSP LIN:50.0E-3 (for example)  
VRTdsp? FM  
VRTDSP FM:5.E+3 (for example)  
VRTdsp? EXT  
VRTDSP EXT:175.E-3
```

VSync <arg>

Arguments: NEGative, POSitive

Single-argument command that specifies the polarity of the video sync to be received with the 2710 optional video monitor.

```
VSync NEGative  
VSync POSitive
```

VSYnc?

Arguments: none

Simple query that returns the currently specified video sync polarity.

```
VSYnc?  
VSYNC NEGATIVE  
VSYNC POSITIVE
```

WAI

Arguments: none

Command requiring no argument that causes the 2710 to wait for an end of sweep to occur before processing any more GPIB commands. **WAI** is ignored in single sweep mode. **WAI** can be cancelled by the GPIB DCL or SDC commands. See also the EOS command.

```
WAI
```

WAVfrm?

Arguments: none

Simple query that is functionally equivalent to the combination of a **WFMpre?** and **CURve?** query. The **WAVfrm** header is never returned, even if **HDR** is **ON**.

```
WAVfrm?
```

WFMpre <arg>

Arguments: ENCdg:Asc, ENCdg:Bin, ENCdg:Hex,
WFId:A, WFId:B, WFId:C, WFId:D

Command with one or more linked arguments used to designate the source/destination register for the **CURve**

command/query, and the data encoding to be used during a CURve transfer. Multiple arguments separated by commas may be used. See the CURve command for an explanation of data encoding.

In its simplest form, an argument(s) always follows the command header and the general forms look like this:

```
WFMpre WFID:<register>  
WFMpre ENCdg:<type>
```

```
WFMpre WFID:<register>,ENCdg:<type>
```

where:

```
<register> = A, B, C or D  
<type> = Asc, Bin, or Hex
```

For instance, the following are all possible WFMpre commands:

```
WFMpre WFid:A  
WFMpre WFid:B  
WFMpre WFid:C  
WFMpre WFid:D  
WFMpre ENCdg:Asc  
WFMpre ENCdg:Bin  
WFMpre ENCdg:Hex  
  
WFMpre WFid:D,ENCdg:Asc
```

The last command string is a typical message. It says register D is the source/destination for future CURve transfers, and that ASCII encoding is to be used for the data.

WFMpre? <arg>

Arguments: none, ENCDg, WFId

Query with one or no argument whose response provides information necessary to:

- determine the currently selected source/destination register for CURve transfers
- determine the data encoding for CURve transfers
- interpret the result of a CURve query

Used with the WFId argument, the query returns the currently selected source/destination register.

```
WFMpre? WFId  
WFMPRE WFID:B (for example)
```

Used with the ENCDg argument, the query returns the currently selected CURve data encoding.

```
WFMpre? ENCDg  
WFMPRE ENCDG:ASC (for example)
```

When the query is issued without an argument, it returns all the information necessary to interpret the response to a CURve? query. The response, with HDR ON, looks like this:

```
WFMPRE WFID:<register>,ENCDG:<type>,  
NR.PT:512,PT.FMT:Y,PT.OFF:<nr1>,XINCR:<nr3>,  
XZERO:<nr3>,XUNIT:<xunit>,YOFF:<nr1>,  
YMULT:<nr3>,YZERO:<nr3>,YUNIY:<yunit>,  
BN.FMT:RP,BYT/NR:1,BIT/NR:8,CRVCHK:  
CHKSM0, BYTCHK:None
```

The response identifies the waveform, specifies the data encoding, and provides the offsets, scale factors, and units necessary to plot or interpret the curve data. Table 4-3 and its associated formulas define the terms in the response.

Table 4-3. Arguments of the WFMpre? query.

ARGUMENT	NAME	DESCRIPTION
1 WFID:<id>	Waveform ID	ID may be A, B, C or D.
2 ENCDG:<enc>	Encoding	The possibilities are ASC, BIN or HEX.
3 NR.PT:512	Number of points	There are 512 data points on every 2710 curve.
4 PT.FMT:Y	Point format	Only Y-data is transmitted. X-data is implicit by the position of the point.
5 PT.OFF:<nr1>	Point offset	See following formulas.
6 XINCR:<nr3>	X increment	See following formulas.
7 XZERO:<nr3>	X zero	See following formulas.
9 XUNIT:<xunit>	X units	HZ (Hertz) or S(seconds).
10 YOFF:<nr1>	Y offset	See following formulas.
11 YMULT:<nr3>	Y multiplier	See following formulas.
12 YZERO:<nr3>	Y zero	See following formulas.
13 YUNIT:<yunit>	Y units	DBM, DBMV, DBV, DBUV, DBUW, DBUVM, V or HZ.
14 BN.FMT:RP	Binary format	It will always be a binary positive integer.
15 BYT/NR:1	Bytes per number	There is always 1 byte per number.
16 BIT/NR:8	Bits per number	There are always 8 significant bits per number.
17 CRVCHK: CHKSM0	Curve checksum	The last byte of binary or hex transfer is the 2's complement of the modulo-256 sum of the bytes following the header which starts the block. The header is % in the case of a binary block and #H in the case of an ascii hex block.
18 BYTCHK: NONE	Byte check	There is no per-byte error checking.

Table 4-4. Formulas related to Table 4-3.

Let <valN> be the value of the Nth data point of the CURVE query. Then the X-value of that point is computed as:

$$XN = XZERO + XINCR * (N - PT.OFF).$$

The Y-value is computed as:

$$YN = YZERO + YMULT * (<valN> - YOFF)$$

Below is a WFMpre? query and the response obtained for the factory default power-up settings:

```
WFMpre?
WFMpre WFID:A,ENCDG:BIN,NR:PT:512,
PT.FMT:Y,PT.OFF:5,XINCR:3.6e+6,XZERO:0,
XUNIT:HZ,YOFF:245,YMULT:3.333E1,YZERO:
2.0E+1,YUNIT:DBM,BN.FMT:RP,BYT/NR:1,BIT/
NR:8,CRVCHK:CHKSM0,BYTCHK:NONE
(for example)
```

Using this preamble, let's compute the value in engineering units of a data point in a CURve? response. Let's choose the 255th point and suppose that the CURve? response indicates it has an integer value of 125 (remember that curve data always have integer values). From the formulas above, we note that the X and Y values of the point are given by:

$$XN \text{ (in xunits)} = XZERO + XINCR * (N - PT.OFF)$$

and

$$YN \text{ (in yunits)} = YZERO + YMULT * (VALN - YOFF)$$

where N = 255 and VALN = 125.

From the preamble,

XZERO = 0	YZERO = 2.0×10^1
XINCR = 3.6×10^6	YMULT = 3.333×10^{-1}
PT.OFF = 5	YOFF = 245
XUNIT = HZ	YUNIT = DBM

Evaluating the expressions, we find:

$XN = 0 + 3.6 \times 10^6 * (255 - 5) = 900 \times 10^6 \text{ Hz}$
$YN = 20 + .3333 * (125 - 245) = -20 \text{ DBM}$

which is, of course, the center of the screen for the factory default power-ups.

ZERosp <arg>

Arguments: OFF, ON

Single-argument command that turns the 2710 zero span mode on and off.

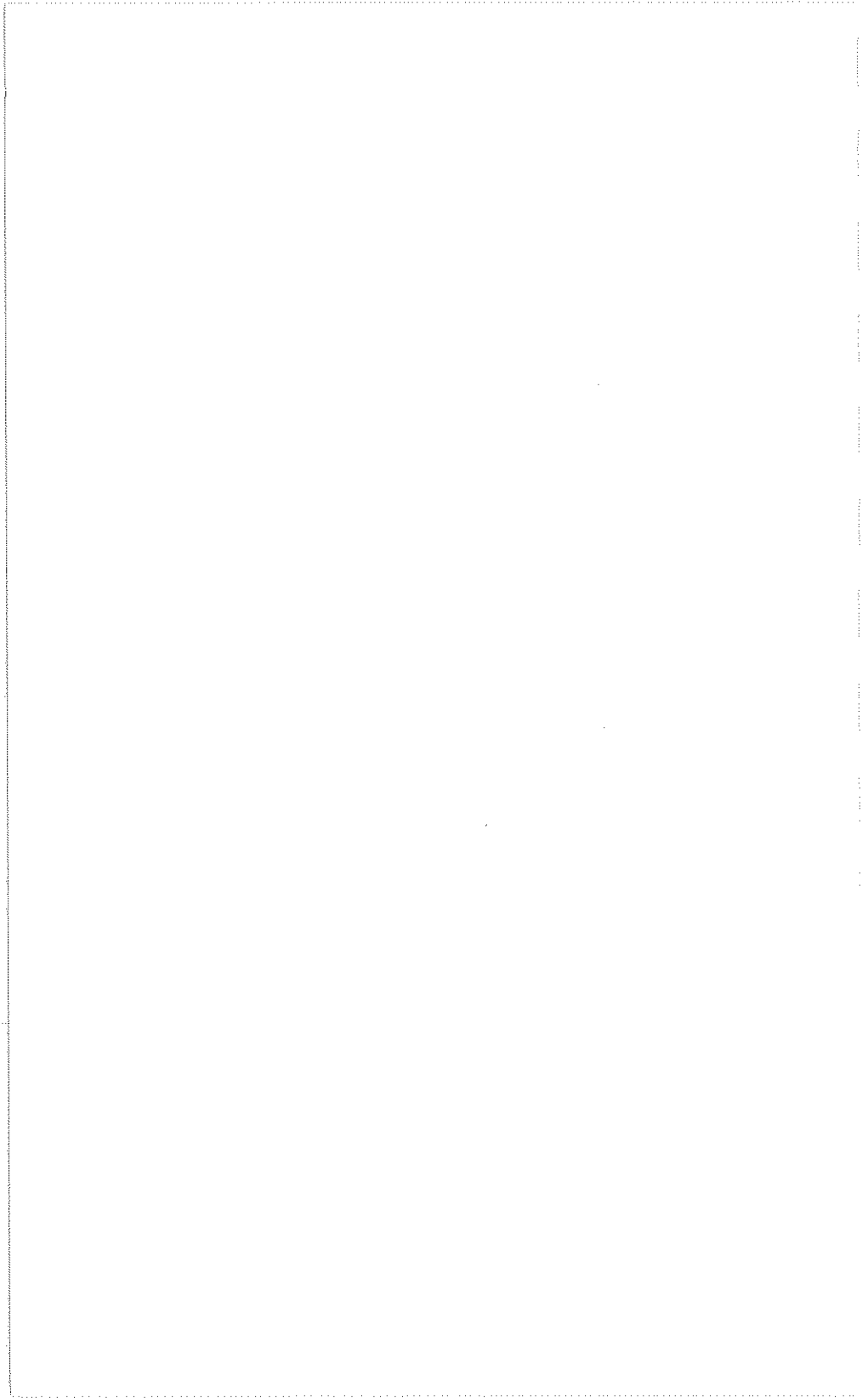
```
ZERosp ON  
ZERosp OFF
```

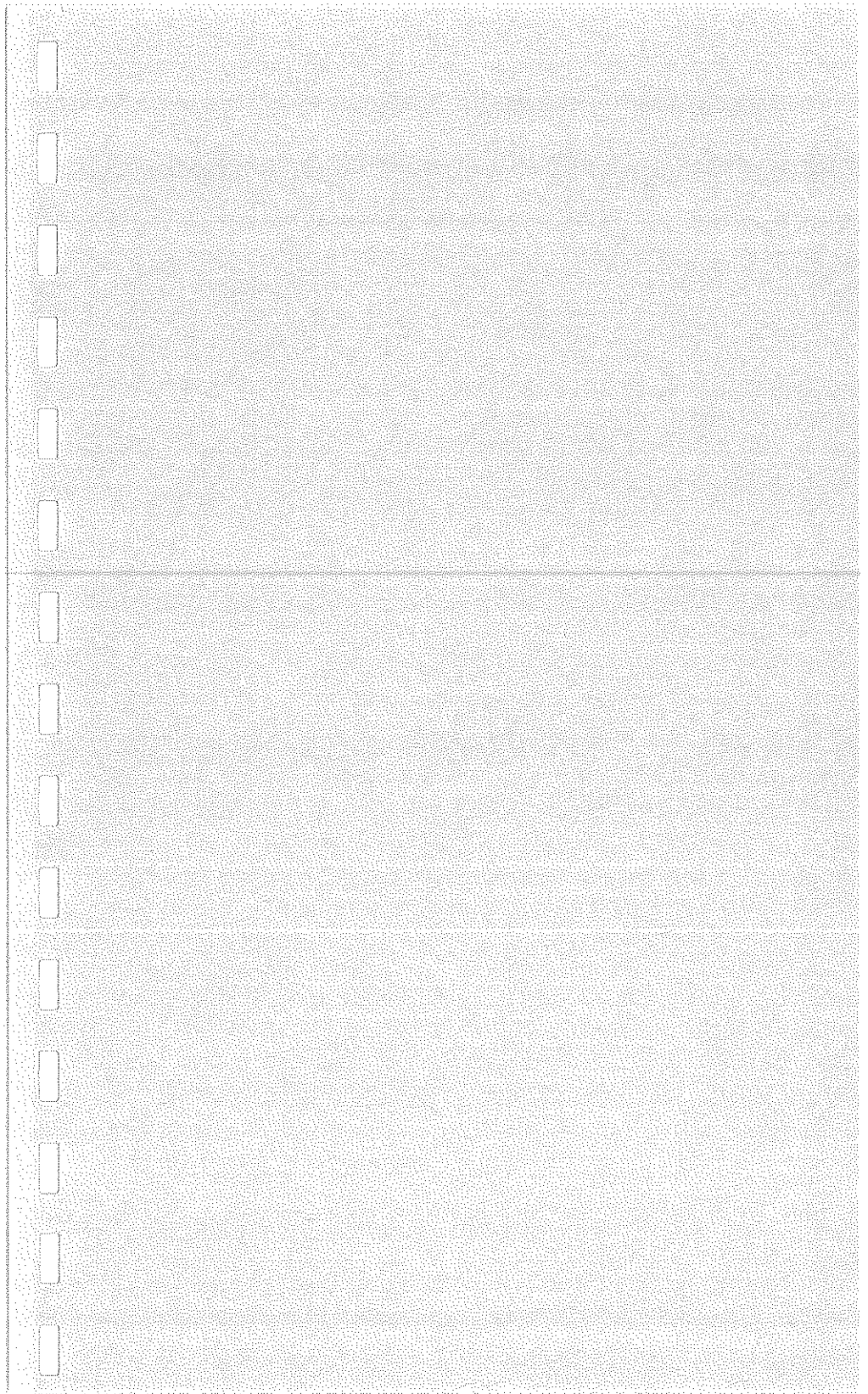
ZERosp?

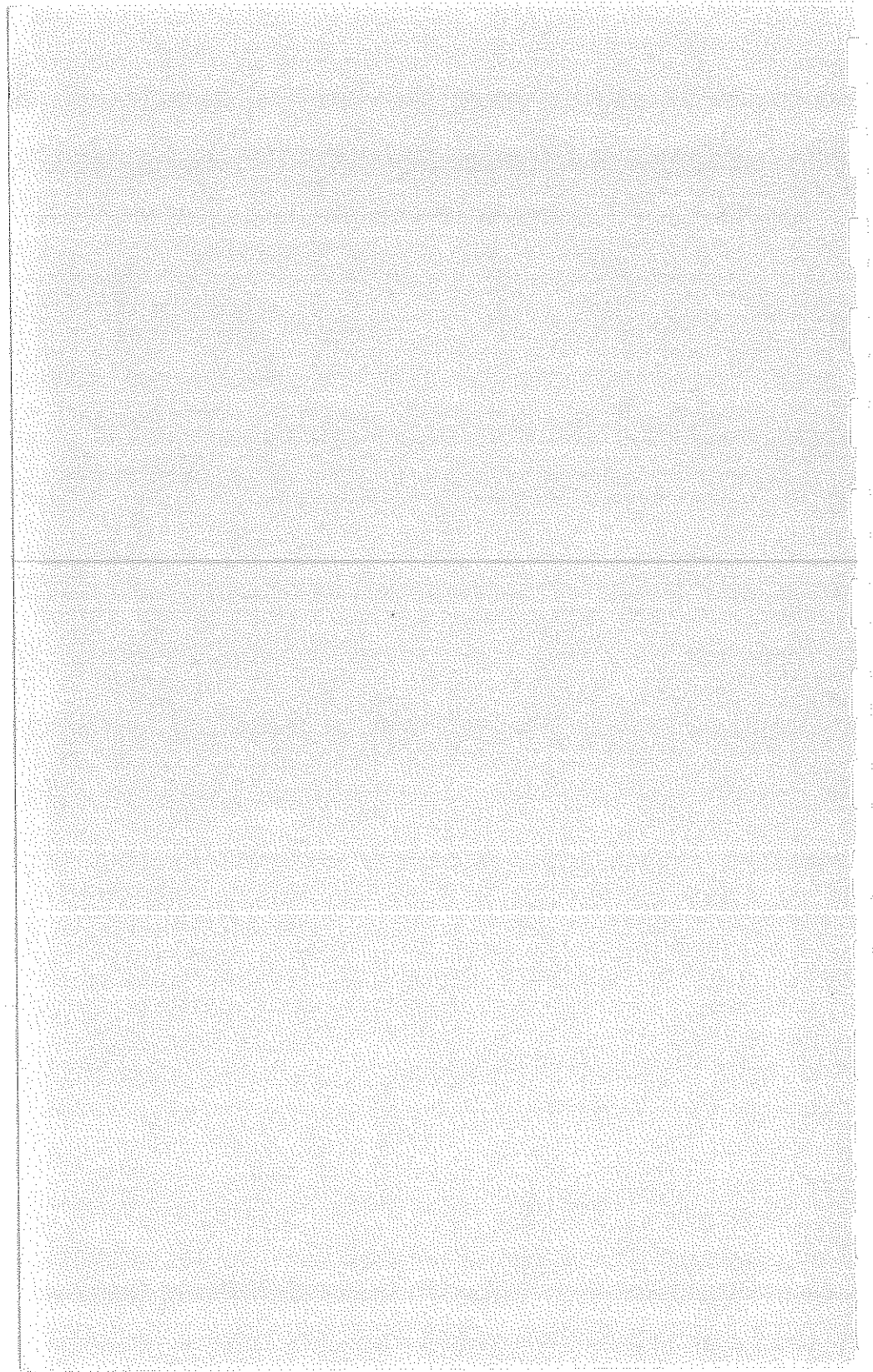
Arguments: none

Simple query that returns the current on/off status of the 2710 zero span mode.

```
ZERosp?  
ZEROSP ON  
ZEROSP OFF
```







Section 5 - Status Reporting

The 2710 reports its status to the controller by means of the status byte and event codes. The status byte is a reporting feature provided in the IEEE-488 standard. When the 2710 is serially polled by the controller, it places a byte representing its general condition on the data bus to be subsequently read by the controller. (Note: the 2710 supports serial polling only; it does not support parallel polling). Event codes, on the other hand, are generated by the spectrum analyzer for transmission to the controller over the data bus in response to a 2710-specific `EVEnt?` or `ERr?` query. Both the status byte and event codes can indicate normal or abnormal conditions. Typically, either or both are used to alert the system user to abnormal conditions.

Decoding the status byte usually provides the more efficient approach to detecting general classes of instrument conditions, but event codes represent more detailed information. For instance, suppose the signal on the analyzer screen is larger than the current reference level, and you attempt to use the `MMAx` command to place the marker on the signal peak. The status byte following the command (`1110000`) indicates only a "device-dependent failure or warning" condition exists, while the event code (`810`) indicates "signal out of range".

To monitor the status of the 2710 at any time, you can use two techniques:

Issue an `EVEnt?` or `ERr?` query and interpret the numerical response.

Serially poll the instrument and decode the status byte.

When an instrument supporting the IEEE service request function (all Tek GPIB instruments do) detects an abnormal condition, it asserts (pulls to its low state) the

dedicated service request (SRQ) line of the GPIB's interface management bus to indicate it requires attention. Therefore, to classify abnormal conditions, construct a subroutine which monitors the SRQ line and then use either of the foregoing techniques to determine the nature of the event causing the SRQ.

The Service Request

The 2710 has complete support for the IEEE-488 service request function (see *Appendix A*). SRQs may be created by an instrument on the GPIB in response to certain equipment states including all abnormal conditions. However, SRQ generation may also be disabled with the RQS command. In addition to this general capability for inhibiting SRQs, certain SRQs may be independently masked.

The following events can generate SRQs:

- Pressing the front panel button sequence [UTIL MENU]/[7]. This is the 'user request' event. It cannot be independently masked.
- Completing certain operations: End of sweep, normalization, User-Defined routine, signal search, plot, or ensemble average. Note that intermediate operation complete events are blocked. The plot complete event is generated after the plot has been formatted and sent to the plot device, not necessarily when the plotter is finished. These SRQs can be masked using the EOS command.
- Device dependant failures or warnings. These SRQs include all the 2710 error messages which can appear on screen. They cannot be independently masked.
- Powering up the 2710. The power-up SRQ can be enabled or disabled by toggling [UTIL MENU]/[4]/[0]/[0]/[2].
- Illegal commands (command, execution, or internal errors). This SRQ cannot be independently masked.

- Failure of find-signal commands MMAx, MRGTnx, MLFtnxt, HRAmpl, LRAmpl. This SRQ can be masked with the SGErr command.
- Crossing of the display line limit. This SRQ is enabled only when the display line limit detector is on (DLLimit). There is no way to independently mask this SRQ.
- Internal hardware/firmware errors. This SRQ cannot be independently masked.

Status Byte

The status byte generally provides information about instrument conditions according to category (normal, abnormal, busy, command error, execution error, etc.) as opposed to the ERr?/EVEnt? queries which provide detailed information about the cause of the current status. For instance, the response to an EVEnt? query might describe what kind of error or warning prompted the spectrum analyzer to assert SRQ and report abnormal status.

The status byte is stored in the status byte register, which is updated as conditions in the instrument change. The spectrum analyzer responds to a serial poll by placing the status byte on the data bus (see *Appendix A*) to be read by the controller. Only the most recent status byte is placed on the bus. The status byte is cleared by a serial poll of that instrument, or by the DCL or (if the instrument is first addressed) SDC GPIB commands.

Status bytes can be divided into 2 categories: device-dependent and system. Device-dependant status bytes represent conditions specific to the 2710, while system status bytes have a fixed definition for all Tek devices and identify most of the events common to any IEEE-488 system. Bit 8 is always set (1) for device-dependent status bytes and is always reset (0) for system status bytes. Tables 5-1 through 5-4 show general and specific encodings of device-dependent and system status bytes.

The status coding scheme provides that only one condition can be reported in a single status byte. Since more than one condition may exist in the 2710 at any given time, the following rules are used to determine which condition is reported by any status byte:

- Each condition is assigned a priority according to Table 5-5. Only one occurrence of each condition is retained in memory.
- When RQS is ON, the status byte following an SRQ represents the condition that caused the SRQ (not necessarily the highest priority).
- When RQS is OFF, the normal device-dependent status shown in Table 5-3 is reported.
- After a status byte is read by the controller, the status byte is updated with the highest priority condition which has not yet been reported.
- All status conditions which have occurred but have not been reported (except power on) can be cleared by a device clear (DCL).
- Bit 5 represents the message processor current status.

Table 5-1. General system status bytes.

STATUS BYTE BITS	8	7	6	5	4	3	2	1
BIT VALUE	0	R	E	B	S	S	S	S

Where:

- R = SRQ pending (1= pending, 0 = not pending)
- B = Instrument busy (1= busy, 0= not busy)
- S = System status
- E = Normal (0)/Abnormal (1)

Table 5-2. General device-dependent status bytes.

STATUS BYTE BITS	8	7	6	5	4	3	2	1
BIT VALUE	1	R	D	D	D	D	D	D

Where:

- R = SRQ pending (1= pending, 0 = not pending)
- D = 2710-specific

Table 5-3. Specific system status bytes.

STATUS BYTE BITS								BYTE*	DESCRIPTION
8	7	6	5	4	3	2	1	VALUE	
0	0	0	B	0	0	0	0	00,10	No status to report
0	1	0	B	0	0	0	1	41,51	Power on
0	1	0	B	0	0	1	1	43,53	User Request
0	1	1	B	0	0	0	1	61,71	Command error
0	1	1	B	0	0	1	0	62,72	Execution error
0	1	1	B	0	0	1	1	63,73	Internal error

* Byte value depends on B=1 or B=0

Table 5-4. Specific device-dependent status bytes.

STATUS BYTE BITS								BYTE*	DESCRIPTION
8	7	6	5	4	3	2	1	VALUE	
1	0	U	B	N	S	A	P		Normal device dependent status
1	1	0	B	0	0	1	0	C2,D2	Device dependent operation complete (EOS, Norm, ..)
1	1	1	B	0	0	0	0	E0,F0	Device dependent failure or warning
1	1	1	B	0	0	1	1	E3,F3	Signal find error (MFBIG, MRGTNX, ..)
1	1	1	B	0	1	0	0	E4,F4	Display line limit exceeded
1	1	1	B	0	1	0	1	E5,F5	Firmware error

* Byte value depends on B=1 or B=0

Where:

- U = UDP executing (1)
- N = Normalization(s) executing (1)
- S = Signal search executing (1)
- A = Ensemble average executing (1)
- P = Plot executing (1)
- B = Busy (1)

Table 5-5. Event priorities.

Event	Priority*
Power on	1
Command error	2
Execution error	3
Internal error	4
User Request	5
Signal find error (MFBIG, MRGTNX, ..)	6
Display line limit exceeded	6
Device dependent failure or warning	7
Device dependent operation complete (EOS, Norm, ..)	8
Firmware error	9
Normal device dependent status	10
No status to report	10

* Highest priority = 1

Table 5-6 shows a short QuickBASIC subroutine which can be used with the National Instruments GPIB board and software to report the current status byte. The first five lines must be part of the parent program. IBFIND and IBRSP are callable subroutines supplied by National Instruments. See Section 6 for explanations and additional programming instructions.

Table 5-6. Subroutine for reading the status byte.

```

REM $INCLUDE 'IBDCL4.BAS'
COMMON SHARED BDNAME$,BD%,SPR$
BDNAME$ = "TEK_SA"
CALL IBFIND (BDNAME$, BD%)
GOSUB SERIAL.POLL
:
:
SERIAL.POLL:
    CALL IBRSP (BD%,SPR$)
    PRINT SPR$
RETURN
    
```

Event codes

Not all GPIB applications need the capability provided by the SRQ function and the serial poll sequence. Often the complexity of the SRQ service routine is more than the application demands. For this reason the Request for Service (RQS) command is implemented in the 2710 to allow the controller to prevent the instrument from asserting SRQ. In this mode of operation, the EVENT? and ERr? queries provide for the transmission of error/status information. In the 2710 both EVENT? and ERr? return the same codes. The ERr? query is included simply for compatibility with the 49X series analyzers.

The EVENT? and ERr? queries provide more information about the cause of an event than can be encoded in the status byte. For this reason, the event query can be useful in both the RQS ON and RQS OFF modes.

The event codes are grouped into categories as shown in Table 5-7. Individual event codes are listed in Table 5-9.

Table 5-7. Event code categories.

NUMERIC RANGE	EVENT
0 - 99	Local events (not used)
100 - 199	Command errors
200 - 299	Execution errors
300 - 399	Internal errors
400 - 499	System events
500 - 599	Execution warnings (not used)
600 - 699	Internal warnings (not used)
700 - 899	2710 dependant events

Event codes are assigned priorities according to Table 5-5, but only the first event code of a given priority is accumulated in the pending event table. The EVENT? and ERr? queries, on the other hand, always return a single code. Therefore, to ensure that all pending event

codes are reported, continue to issue EVen? or ERr? queries until event code zero (0) is returned.

When RQS is OFF, EVen? or ERr? returns the highest priority event in the table. With RQS ON, either query returns the code corresponding to the event reported in the status byte (not necessarily the highest priority).

Reading an event code also clears it from the pending event table, but does not clear the status byte, and vice versa. In either RQS mode, the DCL or (if the instrument is first addressed) SDC GPIB command also clears all event codes except POWER ON.

Table 5-8 shows a QuickBASIC subroutine which can be used with the National Instruments GPIB board and software to report all pending event codes. However, when RQS is ON, you must first call SERIAL.POLL to ensure a valid event code is returned. Further, the response header must be turned off so EVENT.CODE\$ is returned exclusively as a number string. IBFIND, IBWRT, and IBRD are callable subroutines supplied by National Instruments. See Section 6 for explanations and additional programming instructions.

Table 5--8. Subroutine for reading event codes.

```
REM $INCLUDE 'IBDCL4.BAS'  
COMMON SHARED BDNAMES$, BD%,  
                                     EVENT.CODE$  
BDNAMES$ = "TEK_SA"  
CALL IBFIND (BDNAMES$, BD%)  
GOSUB EVENT.FIND  
:  
EVENT.FIND:  
  WRT$ = "HDR OFF;EVE?"  
  DO  
    CALL IBWRT (BD%,WRT$)  
    CALL IBRD (BD%,EVENT.CODE$)  
    PRINT EVENT.CODE$  
    WHILE VAL (EVENT.CODE$) <> 0  
  RETURN
```

Table 5-9. GPIB event codes.

Event Code	Status Dec	Byte Hex	Description
0	128	80	No device-dependent status to report
0	0	00	No system status to report
101	97	61	Command header error
102	97	61	Header delimiter error
103	97	61	Command argument error
104	97	61	Argument delimiter error
105	97	61	Non-numeric Arg.(numeric expected)
106	97	61	Missing argument
107	97	61	Invalid message unit delimiter
108	97	61	Binary block checksum error
109	97	61	Binary block byte count error
121	97	61	Illegal Hex Character
122	97	61	Unrecognized argument type
123	97	61	The argument is too large
124	97	61	Non-binary Arg. (binary or hex expected)
151	97	61	Illegal response value in query
201	98	62	Remote command received while in local mode
202	98	62	Command aborted - (rtl) return to local
203	98	62	I/O Deadlock detected
205	98	62	Argument out of range
206	98	62	Group execute trigger ignored
252	98	62	System error (Illegal command)
253	98	62	Integer overflow (range 0-65535)
371	99	63	Output buffer full (too many queries)
372	99	63	Input buffer full (command too Long)
401	65	41	Power On
403	67	43	User Request
700	224	E0	Error
701	224	E0	Illegal Parameter Passed
702	224	E0	Tracking Generator Not Installed
703	224	E0	300Hz Filter Not Installed
704	224	E0	Illegal Command
705	224	E0	malloc: Ran out of memory
706	224	E0	RunTask: Cannot Start process
707	224	E0	Interrupt Fault at FF

Table 5-9. GPIB event codes.
(continued)

Event Code	Status Dec	Byte Hex	Description
708	224	E0	Interrupt Fault
709	224	E0	Command Not Implemented
710	224	E0	Markers are Off
711	224	E0	Signal Cannot Be Set Properly
712	224	E0	No Signal at Counter Input
713	224	E0	Counter frequency unstable
714	224	E0	Normalization Suggested
715	224	E0	Timer Interrupt Fault
716	224	E0	No Signal (Normalizations)
717	224	E0	Ampl out of range (Normalizations)
718	224	E0	Freq out of range (Normalizations)
719	224	E0	Func Not Avail in Current Mode
720	224	E0	Frequency Normalization Failed
721	224	E0	Amplitude Normalization Failed
722	224	E0	Reference Normalization Failed
723	224	E0	Internal Ref Freq too inaccurate
724	224	E0	Internal Ref Ampl too inaccurate
725	224	E0	Selected Stored Setting is Empty
726	224	E0	Video Monitor Not Installed
727	224	E0	Satellite Video Mntr Not Instld
728	224	E0	Not Installed
729	224	E0	Counter Not Installed
730	224	E0	Cannot overwrite saved display
731	224	E0	NVM Checksum Error
732	224	E0	Non-Compatible NVM Format
733	224	E0	First Step Must Be Done First
734	224	E0	FREQ Norm Suggested (Inner PLL)
735	224	E0	FREQ Norm Suggested (Set VCO)
736	224	E0	Polynomial Has No Solution
737	224	E0	Last Pwr Down Reg Checksum Err
738	224	E0	Storage Register Empty
739	224	E0	Normalized Result Out Of Range
740	224	E0	Function not avail. in LIN mode
741	224	E0	Cannot Store - NV Memory Full
742	224	E0	AMPL Norm Suggested (VR Pin DAC)

Table 5-9. GPIB event codes.
(continued)

Event Code	Status Dec	Byte Hex	Description
743	224	E0	Cannot Calc. Vert. Sensitivity
744	224	E0	Cannot Count (VCO,IF)
745	224	E0	Cannot Normalize PLL VCO
746	224	E0	Cannot Count Beat Frequency
747	224	E0	FREQ Norm Suggested (Set Beat)
748	224	E0	FREQ Norm Suggested (1st LO)
749	224	E0	Setting Corrupted
750	224	E0	NVM Fragmentation Err
751	224	E0	NVM Segmentation Error
752	224	E0	Comm Port Not Installed
753	224	E0	Real Time Clock HW Failure
754	224	E0	Real Time Clock Not Installed
755	224	E0	FREQ Norm Suggested (Find Side)
756	224	E0	FREQ Norm Suggested (Span DAC)
759	224	E0	Insufficient Memory Available
760	224	E0	Not Avail in Short Holdoff Mode
761	224	E0	Short Holdoff Mode Not Instld
762	224	E0	Cannot Overwrite Stored Setting
763	224	E0	Cannot Overwrite Stored Waveform
764	224	E0	Delete Existing Program First
765	224	E0	Editing Buffer Is Empty
766	224	E0	Remove Protection First
767	128	80	Wait Aborted, Sweep Not Armed
768	224	E0	Selected Program Is Empty
769	224	E0	Program Not Executable
770	224	E0	Not Avail in Waterfall Mode
771	224	E0	Amplitude out of Calibration
772	224	E0	Illegal Start/Stop/Inc Values
773	224	E0	Delete Existing Table First
774	224	E0	Selected Table Is Empty
775	224	E0	Use ANTENNA SETUP Menu First
776	224	E0	Table Is Too Large To Edit
777	224	E0	Default Data Loaded
778	224	E0	Delete Editing Buffer First
779	224	E0	Warning: Using Empty Ant Table

Table 5-9. GPIB event codes.
(continued)

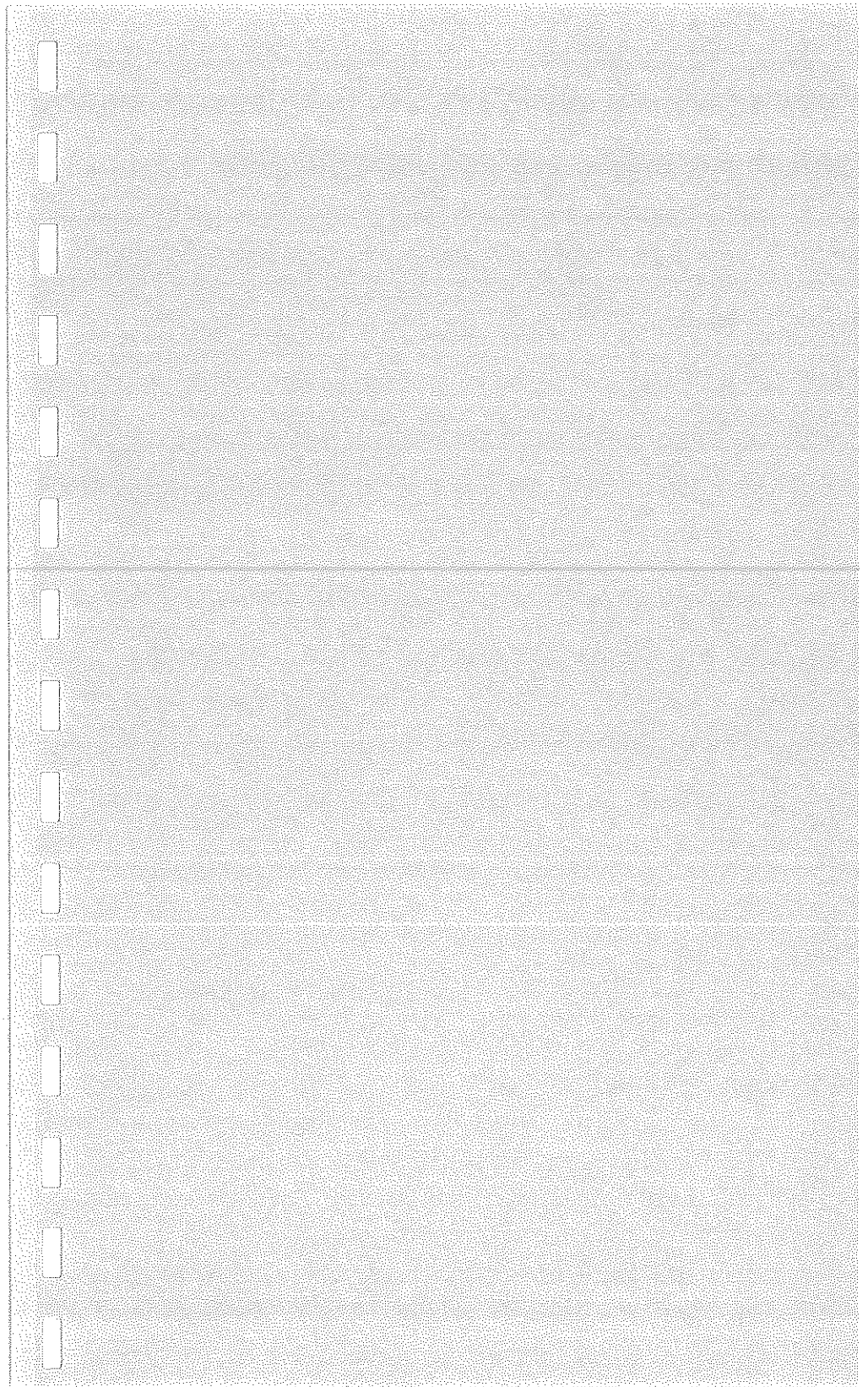
Event Code	Status Dec	Byte Hex	Description
780	224	E0	Not Available with DBUV/M Idle
781	224	E0	Mkr Would Overwrite Noise Value
782	224	E0	Function Not Avail in DBUVM Mode
783	224	E0	No Listener
784	224	E0	Select TALK ONLY mode first
785	224	E0	Tracking Generator Norm. Failed
787	224	E0	Destination waveform conflict
788	224	E0	TG normalization suggested
790	229	E5	Input buffer empty (Firmware error)
791	229	E5	Illegal event code (firmware error)
792	229	E5	Illegal command received from CP
793	229	E5	Illegal byte count in command
800	224	E0	Exiting Quasi-Peak Detector
801	224	E0	Out Of Range
802	224	E0	None of the Traces are Active
803	224	E0	Uncal Off
804	224	E0	Uncal On
805	128	80	Single Sweep Mode
806	128	80	Single sweep armed
807	128	80	Single sweep trigger
808	224	E0	No signal Found Above Threshold
809	224	E0	Inactive marker off screen
810	224	E0	Signal Over Range
811	224	E0	Function Not Avail in Max Span
813	224	E0	Normalization Complete
814	224	E0	No signal at center of display
815	224	E0	Not Avail w/ Display Storage On
816	224	E0	500KHz RBW used for Counting
817	224	E0	Noise Level Less Than 2dB
818	224	E0	Start Frequency Changed
819	224	E0	Stop Frequency Changed
820	224	E0	Signal out of IF passband
821	224	E0	No Modulation on signal
822	224	E0	1ST Measurement Complete
823	224	E0	Disconnect Input Signal

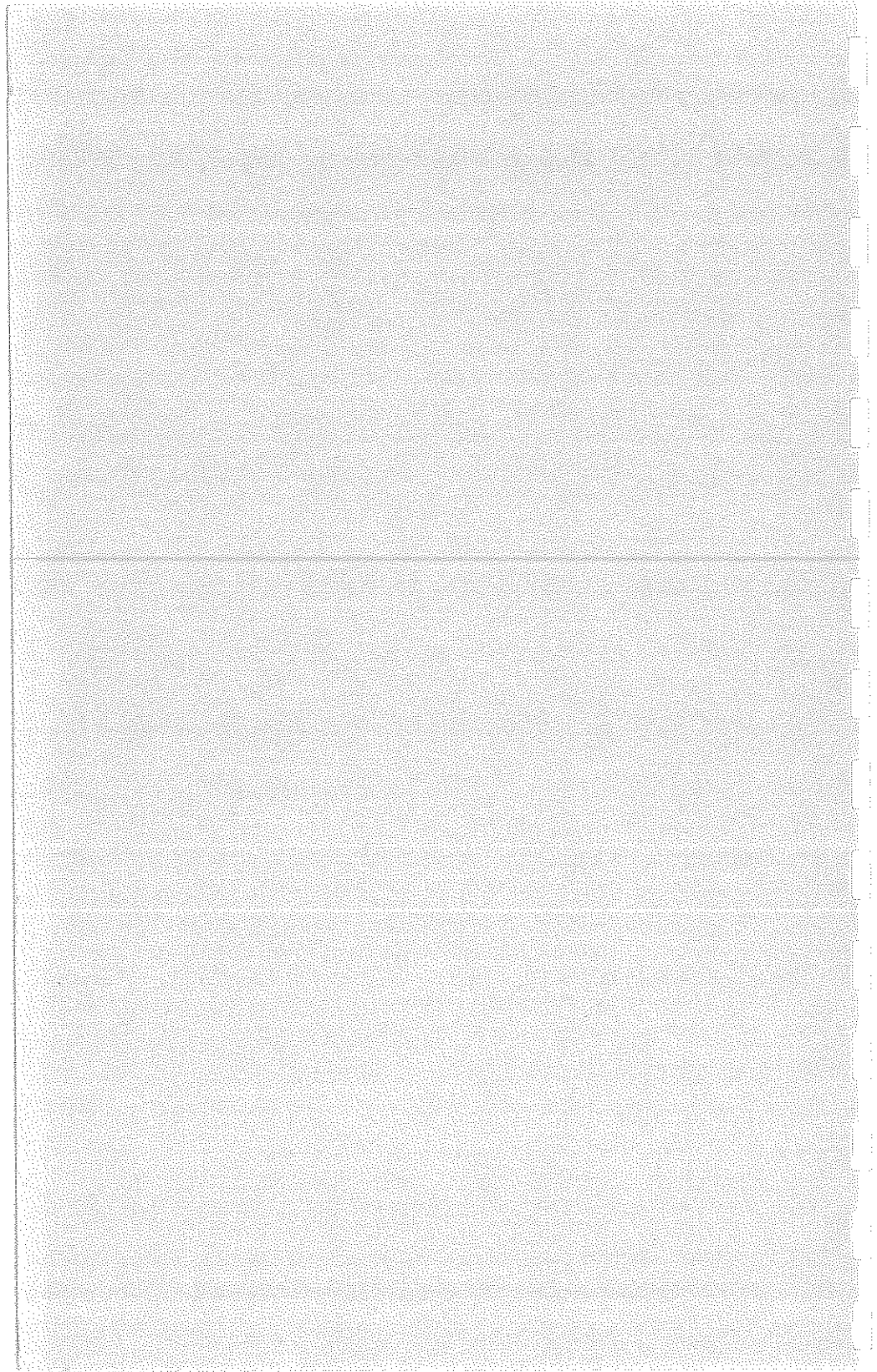
Table 5-9. GPIB event codes.
(continued)

Event Code	Status Dec	Byte Hex	Description
824	224	E0	ZERO SPAN Entered
825	224	E0	Must Be In Delta Marker Mode
826	224	80	Stand By
827	224	E0	Printer Error
828	224	E0	Printer Out Of Paper
829	224	E0	Printer Is Not Connected
830	224	E0	Port Offline
831	128	80	Formatting Plot
832	224	E0	Plot Aborted
833	224	E0	Can't Count With Corrections Off
834	224	E0	Cntr Signal Out of IF Passband
835	224	E0	Vert Mode/Scale Mismatch on Diff
836	224	E0	Query Not Available
837	224	E0	Average Noise Too Low
838	224	E0	Only Waveforms Saved
839	224	E0	Only Waveforms Deleted
840	224	E0	File System Full
841	224	E0	File System Directory Full
842	224	E0	File Size Error
843	224	E0	Too Many Files Open
844	224	E0	File Not Found
845	224	E0	Protected File
846	224	E0	Cannot Delete File While in Use
847	224	E0	Additional NVRAM Not Installed
848	224	E0	Invalid File Number
849	224	E0	Invalid Device Number
850	224	E0	End of File
851	224	E0	NVM Version Mis-Match
852	224	E0	Fatal Error in File
853	224	E0	Directory Error in File
854	224	E0	Data Error in File
855	224	E0	Table Currently In Use
856	128	80	Clear Event
857	224	E0	Calibrator Doesn't Match Readout
858	128	80	Return to Local Request

Table 5-9. GPIB event codes.
(continued)

Event Code	Status Dec	Byte Hex	Description
859	224	E0	Display Line Off Screen
860	224	E0	DBUV/M Measurement Mode Idle
861	224	E0	Search Terminated, Max Signals
862	128	80	Lock Event
863	128	80	Unlock Event
864	128	80	DCL End
865	128	80	User Defined Program in Process
866	128	80	Plot in Process
867	128	80	Average in Process
868	128	80	Signal Search In Process
869	128	80	Normalizing
880	194	C2	User Defined Program Complete
881	194	C2	Plot Complete
882	194	C2	Ensemble Average Complete
883	194	C2	Signal Search Complete
884	194	C2	Normalization Process Finished
885	194	C2	End of Sweep Detected
895	228	E4	Display Line Limit Exceeded
896	227	E3	Signal Find Error





Section 6 Programming For GPIB Operations

In this section we will discuss general approaches to GPIB programming on IBM-compatible MS-DOS/PC-DOS computers and their function-alike counterparts. The intent is not to teach you programming, but to provide a few useful subroutines and a demonstration program. If you are an experienced programmer, you may find you don't need to read this section.

Unfortunately, there are no uniform standards for all GPIB boards and all programming languages. Therefore, the examples that we use will be both device and language specific.

The devices considered are the National Instruments PC-2 and PC-2A GPIB boards, which are supplied as part of the Tektronix GURU II package, or are available directly from National Instruments dealers. Consult Section 1 of this manual and the material supplied with your board for detailed instructions.

The GPIB system software is that supplied by National Instruments along with their boards, or as part of the GURU II package.

The language used is Microsoft's QuickBASIC, version 4.5. The routines also work with earlier versions of QuickBASIC providing you use the appropriate GPIB system software and make the changes to function calls as outlined in the READ-QB.DOC document file from National Instruments.

Before You Start Programming

Before you begin programming there are several steps which must be performed to ensure QuickBASIC has the necessary GPIB information. See *Preparing the QuickBASIC Environment* in Section 1 for the necessary procedure.

Beginning Your Program

Generally speaking, one begins a BASIC or QuickBASIC program with a list of declarations. This establishes variable types and the names of global variables.

A number of variable names are used by the National Instruments software. To prevent any confusion or misunderstanding regarding these variables, National Instruments supplies a program file which declares them for you. To properly declare the reserved National Instruments variables, place this line at the beginning of your programs:

```
REM $INCLUDE 'QBDECL4.BAS'
```

Because QuickBASIC supports the sub-program module concept, traditional COMMON statements are replaced with COMMON SHARED statements, meaning that the variable names so declared may be shared by all modules. Further, sub-program modules themselves must be declared. Here are the lines which declare the global variables used by our subroutines and demonstration program:

```
COMMON SHARED BD%,BDNAME$,RD$,  
WRT$, EVENT.CODE$  
COMMON SHARED NUMBYT%
```

Table 6-1 lists the global variables declared by QBDCL4.BAS and the COMMON SHARED statements, and what they are used for.

You may also want to dimension any arrays or variables you plan to use later at the beginning of your program. For instance, CUR%() is the integer array used for curve data in our demonstration program. RD\$ is a string variable which we use as the destination for a variety of data transfers. If memory size is not a limiting factor in your controller, you may wish to set up the maximum size of RD\$ at the head of the program with a SPACES\$

command like this:

```
DIM SHARED CUR%(512)
RD$ = SPACE$(<max anticipated size>)
```

For all queries except PLOT? and FILE?, the number of bytes in the response is less than 5000.

Table 6-1. Variable names.

BD%	An integer value associated with a particular device name by IBFIND(BDNAME\$,BD%). Other names can be used (MYDEV%, FRST.DEV%, A1%, etc.)
BDNAME\$	A string variable set equal to a specific device name such as TEK_SA. The specific device name is the one used when you set up the particular device using IBCONF. Any string variable name can be used (MYNAME\$, DEVNAME\$, DNS\$, etc.)
EVENT.CODE\$	A string variable argument of IBRD containing the 2710 event code. Any string variable name can be used (MYNAME\$, DEVNAME\$, DNS\$, etc.)
IBCNT%	An integer variable updated by the GPIB system software following each read, write, or command function to indicate the number of bytes actually transferred. This name is reserved.
IBERR%	An integer variable returned by the GPIB system software when the error bit of the status word is set. Its value ranges from 0 to 7 or 10 to 16. See your GPIB manual for meanings of IBERR% values. The name is reserved.
IBSTA%	An integer variable updated by all GPIB system software functions. See <i>Status Byte</i> in Section 5.
NUMBYT%	Integer argument of DEBLK which indicates the number of bytes converted. Any integer variable name can be used (MYNUM%, INTEG%, etc.).
RD\$	A string variable used with the IBDR function to contain the data returned by the 2710. Other names can be used (RET.DAT\$, MYDAT\$, S1\$, etc.).
WRT\$	A string variable used with the IBWRT function to contain the data to be sent to the 2710. Other names can be used (WRT.DAT\$, MESSAGE\$, B4\$, etc.).

For PLOT? the number can be as large as 37,000 for HPGL plots and 61,100 for Epson plots. The number of points in a PLOT? response depends on the number of waveforms present and the acquisition mode.

A user-defined program (UDP) can theoretically occupy all available memory making UDPxx files larger than 100,000 bytes possible. However, in most cases, 5000 bytes are probably adequate for all but the largest UDPs.

Error Trapping

Three types of errors can occur in your program. Different techniques are used to deal with each type. In general the errors are trapped so corrective action can be taken or the program caused to end gracefully.

DOS errors are the first type. These typically happen when trying to access a device which is missing or not ready. Traditionally they are handled with the BASIC ON ERROR statement. Following the declarations at the head of the program, add this line:

```
ON ERROR GOTO ERR.TRAP
```

Then construct a subroutine to deal with the problem. To end the program gracefully, this will work:

```
ERR.TRAP:  
  PRINT "DOS ERROR HAS OCCURRED."  
  PRINT "CHECK YOUR SYSTEM SETUP."  
  END  
RESUME NEXT
```

Using this routine, the program simply stops if a DOS error is encountered. Check your system and restart. Consult a BASIC programming manual for selective error trapping techniques and other approaches.

The GPIB system error is the second type. These errors are detected by examining IBSTA% following each

GPIB function call. If the value of IBSTA% is equal to or greater than 32768, then a GPIB error has occurred. You then look at the value of IBERR% to see what type of error has occurred. You can either take corrective action, proceed without doing anything (if this doesn't hang the program or yield erroneous results), or end the program gracefully. To determine the type of error and end the program gracefully, call this subroutine following each call to a GPIB function:

```
GPIB.ERR:
  IF IBSTA%<32768 THEN RETURN
  PRINT "GPIB ERROR HAS OCCURRED."
  PRINT "GPIB ERROR CODE IS ";IBERR%
  END
RETURN
```

Errors directly related to the 2710 or its interface with the bus are the third type. Any time the 2710 detects an abnormal condition, it issues a service request (SRQ) if RQS is set to ON. The SRQ causes the GPIB board to generate a light pen interrupt. By branching to an interrupt handler whenever the interrupt occurs, you can automatically detect and decode abnormal events. Add these lines to your program:

```
ON PEN GOSUB ABNORM.EVE
PEN ON
:
ABNORM.EVE:
  CALL SERIAL.POLL
  CALL EVENT.FIND
RETURN
```

SERIAL.POLL and EVENT.FIND are the sample subroutines you learned in Section 5.

6-2. GPIB system software callable subroutines.

SUBROUTINE (WITH ARGUMENTS)	DESCRIPTION
DEBLK(CUR%(),CUR%(), CNT%,8,NUMBYT%)	convert first CNT% elements of array CUR%() from binary block to 2-byte integer format in same array and return the number of bytes converted
IBFIND(BD%,BDNAME\$)	open device indicated by BDNAME\$ and return unit descriptor BD%
IBRD(BD%,RD\$)	read data from device indicated by BD% to string RD\$
IBRDF(BD%,FILENAME\$)	read data from device indicated by BD% and store to disk in file named <FILENAME\$>
IBRDI(BD%,CUR%(),CNT%)	read CNT% data bytes from device indicated by BD% into integer array CUR%()
IBRSP(BD%,SPR\$)	perform serial poll of device indicated by BD%
IBSRE(BD%,V%)	enable/disable remote mode in the device indicated by BD%. V%=0 disables.
IBWRT(BD%,WRT\$)	send contents of string variable WRT\$ to device indicated by BD%
IBWRTF(BD%,FILENAME\$)	send disk file named <FILENAME\$> to device indicated by BD%

GPIB System Software

All of the programming examples in this manual make use of the subroutines supplied by National Instruments as part of the GPIB or GURU II software. If you are using a board from another manufacturer, you should have received equivalent software with your board. The software supplies the interface between the programming language and the installed GPIB board.

Table 6-2 lists the National Instruments system subroutines used in this manual. Consult your GPIB documentation for detailed information about these and the many other subroutines available.

Sample Subroutines

This subsection contains a collection of subroutines intended to illustrate simple approaches to transferring data of various types between the system controller and the 2710. You may wish to incorporate them into your own software, or modify them so they are more appropriate to your needs.

The header statements in Table 6-3 can be placed at the beginning of a program, and used as a basis for the subroutines in this section. If you modify the subroutines, or add new ones, the statements may no longer be adequate. Be aware that neither the statements nor any of the sample subroutines make provision for error trapping and event reporting. See Section 5 and the demonstration program at the end of this section for error/event reporting routines.

Table 6-3. Program header statements.

```
REM $INCLUDE: 'QBDECL4.BAS'  
COMMON SHARED BD%,BDNAME$,RD$,WRT$  
RD$=SPACE$(5000)  
BDNAME$ = "TEK_SA"  
CALL IBFIND (BDNAME$, BD%)
```

Curve Transfers

Curves transferred from the 2710 to the controller are important for data analysis, archiving, and reporting purposes. Curves transferred to the 2710 can be used for comparison, or to establish references. The *CURve* command transfers a block of data from the controller to the analyzer and the *CURve?* query returns a block of data from the analyzer to the controller. The data block represents the 512 points in a 2710 waveform (see Section 4 for *CURve?* response formats). Before you can transfer curve data, you must specify which digital display register the curve is coming from or going to, and the type of data encoding to be used. The encoding (ASCII-encoded decimal, ASCII-encoded hexadecimal, or binary) is determined by the waveform preamble (see the *WFMpre* command). The destination of the transmitted data or origin of the returned data is also determined by the preamble except in cases where the A, B, C, or D argument is used with the *CURve?* query. In these cases, the origin is specified by the argument (i.e., *CURve? C* returns data from the C register).

Table 6-4 lists two QuickBASIC subroutines that can be used with the National Instruments board and software to send and return ASCII-encoded curve data. The returned data are displayed on the controller screen, and will resemble the ASCII example in the *CURve* command in Section 4. The *WFMpre* command is used to establish the encoding and source/destination registers in both subroutines. Another approach to transferring curve data (as packed integers) is illustrated in the demonstration program at the end of this section.

The curve transferred to the analyzer is the previously returned waveform, but you can also send artificially generated curves. Such curves can be generated in ASCII format using a spreadsheet. To ensure that curves sent to the analyzer are not immediately overwritten, you should transfer them to a saved register.

Note that if you generate a curve, you must substitute something like the following for the *WRT\$ = MID...line*

in the PUT.CURVE subroutine:

```
WRT$ = "CURve "+IND$+BC$+DATA$+CK$
```

where:

```
IND$ = Null for ASCII, #H for hex, % for binary
BC$ = Null for ASCII, 0201 for hex,
      CHR$(2)+CHR$(1) for binary
DATA$ = 512 data points, appropriately encoded,
        representing the curve
CK$ = Null for ASCII, HEX$(chksum) for hex
      or "0"+HEX$(chksum) if chksum < 16,
      CHR$(chksum) for binary
```

Table 6-4. Subroutines to return (GET.CURVE) or transmit (PUT.CURVE) curve data.

```
GET.CURVE:
'establish source register and encoding,ensure header is
'turned on
WRT$ = "WFMPRE WFID:D,ENCDG:ASCII;HDR ON"
      CALL IBWRT(BD%,WRT$)
'reserve space for the data
RD$ = SPACE$(2056)
'return the curve data
WRT$ = "CURVE?"
      CALL IBWRT(BD%,WRT$)
      CALL IBRD(BD%,RD$)
'Trim and display the data on screen
PRINT MID$(RD$,1,IBCNT%)
RETURN
:
PUT.CURVE:
'establish encoding and destination register
'save the register
WRT$ = "WFMPRE WFID:A,ENCDG:ASCII;SAVE A:ON"
      CALL IBWRT(BD%,WRT$)
'Trim curve data and send it
'CURve command header is included in RD$
WRT$ = MID$(RD$,1,IBCNT%)
      CALL IBWRT(BD%,WRT$)
RETURN
```

Transferring Files

The FILE command/query transfers named data files between the analyzer and controller. The intent of FILE/FILE? is to enable you to return information from one instrument for disk storage and subsequent transmission to another instrument -- or possibly to the same one in the event of NVRAM failure. It is not intended for curve viewing or settings editing.

Permissible file names are established by the 2710, and are listed under the *FILE* discussion in Section 4. The files are created within the 2710's memory only as required. That is, a BSET03 file exists only if you have previously saved the B-register settings in the third storage location. The currently created files can be viewed by pressing [UTIL MENU]/[4]/[6].

Table 6-5 lists QuickBASIC subroutines which can be used with the National Instruments GPIB board and software to transmit or return data files. You must supply the name of the file to be returned or transmitted.

The response header is turned on when the file is returned to the controller. The returned data string (FILEDAT\$) then becomes:

```
FILEDAT$ = FILE "<filename>",<data block>
```

This is exactly the form required for the FILE command, so upon transmission you need only send FILEDAT\$. Alternately, you can set HDR OFF in GET.FILE and then set WRT\$ = "FILE "+FILEDAT\$ in PUT.FILE.

Notice that you can send similar files to different locations, but you cannot transfer one type of file to another type. That is, you can return BSET04 and send it to CSET05, but you can't return ACF2 and rename it UDP07. Because the file name is embedded in the response, this subroutine transmits the file to the same location it came from. This limitation can be

circumvented by inserting this line at the beginning of PUT.FILE:

```
MID$(FILEDAT$,1) = "FILE <new filename>"
```

These subroutines neither store nor retrieve the files to/ from disk. You can do so with the usual BASIC OPEN, PRINT #, and INPUT # statements. Alternately, you can use the National Instruments IBWRT() and IBRDF() calls to transfer data directly to and from the 2710 and disk. See the demonstration program for an example of this approach.

Table 6-5. Subroutines to return (GET.FILE) or transmit (PUT.FILE) data files.

GET.FILE:

```
'enter the 2710 filename for the file you want uploaded
'see Table 4-1 for a list of the names
FILE2710$ = <2710 filename>
'turn on Header, request file
WRT$ = "HDR ON;FILE? "+FILE2710$
CALL IBWRT(BD%,WRT$)
'read file into string variable RD$
CALL IBRD(BD%,RD$)
'trim string variable to number of characters transferred
FILEDAT$ = MID$(RD$,1,IBCNT%)
RETURN
```

PUT.FILE:

```
'send the file in memory back to the 2710
WRT$ = FILEDAT$
CALL IBWRT(BD%,WRT$)
RETURN
```

Plotting 2710 Screen Data

The PLOT? query enables you to transfer data representing an image of the 2710 screen from the analyzer to a plotter or printer. It performs a function similar to the Utility Menu SCREEN PLOT option ([UTIL MENU]/[6]). The printer or plotter must speak the HPGL language or be compatible with Epson FX codes, and the appropriate printer type must be specified either locally or using the PTYpe command.

It is possible to have the 2710 send data directly to a plotter by holding the GPIB ATN line high while addressing the 2710 as a talker and the plotter as a listener, and then asserting ATN. You must also set EOS ON and WAlt for an end-of-sweep SRQ. This approach requires no computer memory.

However, an alternate approach (returning the screen plot data to the controller, then sending it to the designated output device) enables you to create independent input and output subroutines for use with specific devices, and to share those devices with other instruments on the GPIB. Because the program itself does not proceed until the plot data are received, it is not necessary to WAlt for an end-of-sweep SRQ. Further, it enables you to do the printing/plotting at a more convenient time, or even using an entirely different controller and output device.

Figures 6-1 and 6-2 show how you might return and print/plot instrument data. Each "Do it" in the diagrams can represent a separate subroutine for a specific device.

Table 6-6 lists a subroutine for obtaining screen plot data from the 2710 using the PLOT? query. It does not store the data on disk (you can add that capability if desired), but holds them in memory as the string variable PLOT.DAT\$. The length of the string required depends on the display acquisition mode, the plotter type, and the number of registers displayed. 12 kbyte of memory is enough for a single sweep in PEAK acquisition mode if an HPGL plotter is used. If an Epson FX printer is used, as

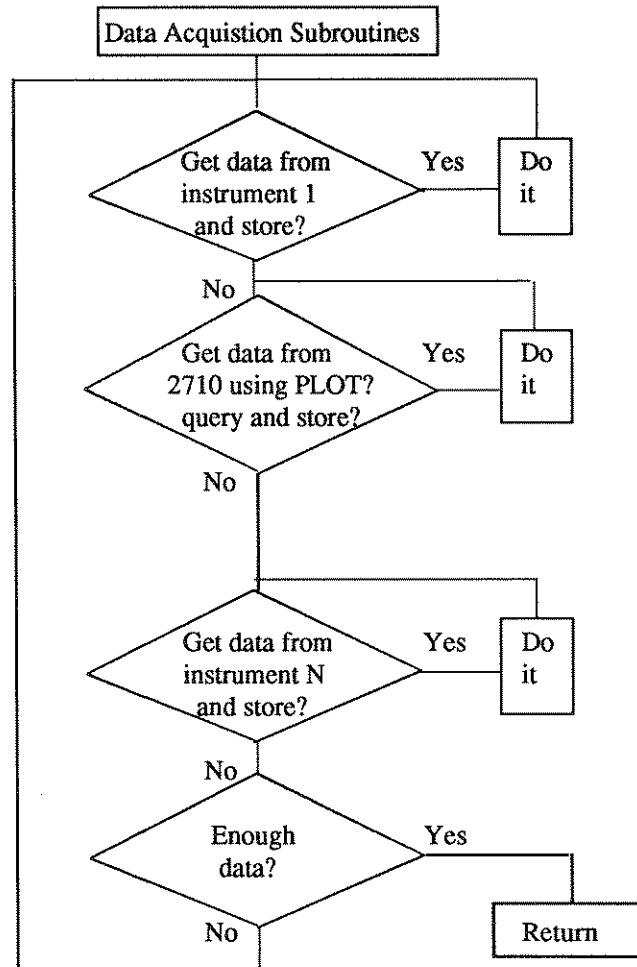


Figure 6-1. Possible data acquisition scheme.

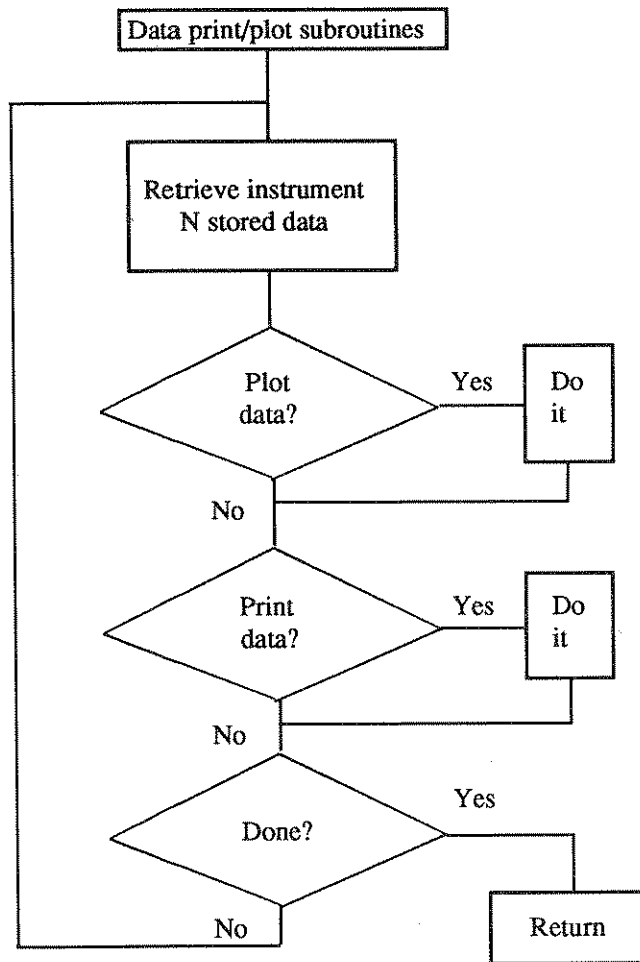


Figure 6-2. Possible data print/plot scheme.

many as 61.1 kbyte may be required for 4 sweeps in MAX/MIN mode. Substitute 61100 for 12000 and change HPGL4 to EPSON. Specify HPGL2 if you have a 2-pen plotter. You can send data to a parallel Epson type printer on the controller's parallel port, but to send it over the GPIB, the printer must be equipped with a GPIB board (an unlikely but possible configuration).

The SEND.PLOT subroutine transmits PLOT.DAT\$ to a plotter matching the type specified in GET.PLOT. Note that the controller timeout is disabled, and that you restart execution following the plot by pressing any key.

Table 6-6. Subroutines to return (GET.PLOT) or send (SEND.PLOT) screen plot data to a 4-pen HPGL-compatible plotter.

```

GET.PLOT:
'reserve space for screen data
    PLOT.DAT$ = SPACE$(12000)
'set plotter type and request screen data
    WRT$ = "PTYPE HPGL4;PLOT?"
    CALL IBWRT (BD%,WRT$)
'get screen data
    CALL IBRD (BD%,PLOT.DAT$)
'Trim data to number of bytes transferred
    PLOT.DAT$=MID$(PLOT.DAT$,1,IBCNT%)
    RETURN
:
SEND.PLOT:
'disables time out to give plotter ample time to finish
    CALL IBTMO(BD%,0)
    CLS
'send screen data to plotter
    CALL IBWRT(BD%,PLOT.DAT$)
'press a key after plotter finishes
    PRINT "PRESS ANY KEY TO CONTINUE"
    DO WHILE INKEY$ = ""
        LOOP
'reestablishes 30-second time out
    CALL IBTMO (BD%,14)
    RETURN
    
```

Returning the On-screen Readouts

The 2710 on-screen readouts provide a synopsis of the the important operational parameters of the analyzer. The PRDouts? query enables you to return most of those readouts. It does not return the 2710 general purpose message line, the GPIB status line, or the user-defined "DISPLAY MESSAGE" line. It does return the PRDOOTS header (if HDR is ON) and up to 13 arguments depending on the mode of operation of the 2710 and its current status. The arguments are listed in Section 4 under *PRDouts?* The response is ASCII-encoded and can be read and displayed on the controller screen using the subroutine in Table 6-7.

Table 6-7. Subroutine (READOUTS) for returning the 2710 on-screen readouts.

```
READOUTS:
'reserve space for the readouts
  READOUT.DAT$ = SPACE$(416)
'request the readouts
  WRT$ = "PRDOOTS?"
  CALL IBWRT (BD%,WRT$)
'return the readouts
  CALL IBRD (BD%,READOUT.DAT$)
'trim data to number of bytes returned
  READOUT.DAT$ =
    MID$(READOUT.DAT$,1,IBCNT%)
'display the readouts
  PRINT READOUT.DAT$
RETURN
```

Saving and Restoring Equipment Settings

The SET? query can be used to return the current 2710 control settings. The settings are returned in a message format containing the command headers and arguments necessary to place the analyzer in its current configuration. The settings can be saved in a controller disk file for later transfer to the same or another 2710 when you want to restore the same operating environment. Because the SET header is always suppressed in the response, the response can be retransmitted as received.

The SET? query is generally used in preference to settings file transfers because:

- The same command and format can be used with a variety of Tektronix instruments for the same purpose.
- The 2710 implements the retransmitted settings as soon as they are received, rather than requiring a separate RECALL command.
- The returned settings are in ASCII format, and can be directly edited if necessary.

See the SET? query in Section 4 for more details.

Table 6-8 lists QuickBASIC subroutines for storing and reestablishing the settings group. The routine is suitable for use with the National Instruments GPIB board and software. Notice that you must actually substitute the disk file name that you want to store the settings in for "filename".

Table 6-8. Subroutines for saving (GET.SET) and restoring (PUT.SET) settings groups.

```
GET.SET:
'request the settings
  WRT$ = "SET?"
  CALL IBWRT (BD%,WRT$)
'read the settings from the 2710
  CALL IBRD (BD%,RD$)
'trim the settings to the number of bytes returned
  SETTINGS$ = MID$(RD$,1,IBCNT%)
'display the settings for verification purposes
  PRINT SETTINGS$
'save the settings on disk -- substitute the diskfile name
'that you want to store the settings under for filename
  OPEN "O",#1,"filename"
  PRINT #1, SETTINGS$
  CLOSE #1
RETURN
:
PUT.SET:
'open the disk file and read in the stored settings
  OPEN "I",#1,FILENAME$
  INPUT #1,SETTINGS$
  CLOSE #1
'display the settings for verification purposes
  PRINT SETTINGS$
'send the settings to the 2710
  WRT$ = SETTINGS$
  CALL IBWRT (BD%,WRT$)
RETURN
```

Waiting For Results

A number of functions in the 2710 require a wait period between the time they are requested or begin execution, and the time at which the results of the operation are available. These include:

delta marker readouts	marker readouts
counter readouts	ensemble averages
signal searches	plots
User-Definable Programs	normalizations

We will use a signal search (with the 2710 calibration signal as the source) as an example of how to program for these events. You can use the same basic approach for a general signal search program; simply change the parameters to meet your needs.

We'll then use the `SSResult?` query to return a list of the amplitudes and frequencies of the signals detected. Although the results of a signal search are held in 2710 memory until overwritten by another search or until power is turned off, it is generally good practice to return the results immediately after the search is completed.

The subroutine in Table 6-9 returns and prints the signals detected. Six should be present (at 100, 200, 300, 400, 500, and 600 MHz) ranging in amplitude from -30 dBm to about -60 dBm). The subroutine first recalls the factory default power-ups and then changes the equipment settings as required for the 50 - 650 MHz signal search.

Notice that we load `WRT$` with the desired message when the message is lengthy, but use the actual message as the argument of `IBWRT()` when the message is short. You can decide for yourself which approach to use.

Following the subroutine, you will have to reset the 2710 to whatever conditions you require.

Table 6-9. Subroutine (SSRESULTS) for returning and printing signal search results.

```
SSRESULTS:
'initialize 2710, turn on calibrator, set center freq
  WRT$ = "RECALL 1;CALSIG ON;FREQ 250 MHZ"
  CALL IBWRT(BD%,WRT$)
'set span/div and reference level
  WRT$ = "SPAN 20 MHZ;REFLVL -30 DBM;"
  CALL IBWRT(BD%,WRT$)
'set signal search begin and end frequencies
  WRT$ = "SSBEGIN 50 MHZ;SSEND 650 MHZ"
  CALL IBWRT(BD%,WRT$)
'ensure EOS is ON
  CALL IBWRT(BD%,"EOS ON")
'perform signal search
  CALL IBWRT(BD%,"SGSRCH")
'wait for end-of-sweep before continuing -- the search
' must be completed before you can return results
  CALL IBWRT(BD%,"WAIT")
'request results
  CALL IBWRT(BD%,"SSRESULTS")
'read results from 2710
  CALL IBRD (BD%,RD$)
'trim results to number of bytes returned
  RESULT.DAT$ = MID$(RD$,1,IBCNT%)
'display results on controller screen
  PRINT RESULT.DAT$
RETURN
```


Sample Program

The following COMM2710 program is a simple utility for communicating with the 2710. It contains some of the subroutines (or elements of them) that were discussed earlier as well as new material. It is not very sophisticated, but it does show how to satisfactorily command and interrogate the spectrum analyzer, enabling you to perform several useful operations.

```

                                COMM2710
'
'program to communicate with a Tektronix 2710 Spectrum Analyzer
'via the GPIB
'
'declare GPIB system software reserved variables
    REM $INCLUDE: 'QBDECL4.BAS'
'
'declare common global variables
    COMMON SHARED BD%,BDNAME$,RD$,WRT$
    COMMON SHARED EVENT.CODE$,NUMBYT%
'
'dimension max size of returned data string
    RD$ = SPACE$(5000)
'
'dimension an integer array for packed integer CURve? response
    DIM SHARED CUR%(512)
'
'obtain bus device unit descriptor (BD%). BDNAME$ must match
'the name established for the 2710 in the IBCONF program
    BDNAME$ = "TEK_SA"
    CALL IBFIND(BDNAME$, BD%)
'
'establish link to abnormal event handler; enable interrupt line
    ON PEN GOSUB ABNORM.EVE
    PEN ON
'
'enables SRQ generation in case of abnormal event
    CALL IBWRT(BD%, "RQS ON")

```

```
'trap DOS errors
  ON ERROR GOTO ERR.TRAP
```

```
'generate the menu
```

```
MENU:
  CLS
  PRINT "F1  SEND COMMAND OR QUERY"
  PRINT
  PRINT "F2  SAVE CURRENT SETTINGS TO FILE"
  PRINT
  PRINT "F3  RESTORE SETTINGS"
  PRINT
  PRINT "F4  SAVE AN INSTRUMENT FILE"
  PRINT
  PRINT "F5  RESTORE AN INSTRUMENT FILE"
  PRINT
  PRINT "F6  ACQUIRE CURVE DATA"
  PRINT
  PRINT "F10 EXIT"
  PRINT
  PRINT "PRESS F1-F6 OR F10 TO MAKE SELECTION"
  PRINT
```

```
'chk keyboard for keypress, decode, branch to correct subroutine
```

```
KYBD.CHK:
  IN$ = INKEY$
  IF IN$ = "" GOTO KYBD.CHK
  IF LEN(IN$) = 1 THEN BEEP: GOTO KYBD.CHK
  IN.KEY = ASC(MID$(IN$, LEN(IN$), 1)) 'decoding complete
  SELECT CASE IN.KEY 'branch to subroutine
    CASE 59 'F1 pressed
      GOSUB SEND.RCV
      GOTO MENU
    CASE 60 'F2 pressed
      GOSUB SAVE.SET
      GOTO MENU
    CASE 61 'F3 pressed
      GOSUB RES.SET
      GOTO MENU
    CASE 62 'F4 pressed
      GOSUB SAVE.FILE
      GOTO MENU
```

//////////////////////////////////// 2710 GPIB Programmers Manual //////////////////////////////////////

```

CASE 63                                'F5 pressed
    GOSUB RES.FILE
    GOTO MENU
CASE 64                                'F6 pressed
    GOSUB INT.CUR
    GOTO MENU
CASE 68                                'F10 pressed
    SYSTEM                             'returns to dos
END SELECT
GOTO MENU                              'regenerates the menu

```

'subroutine to send a command or query and receive the response

```

SEND.RCV:
CLS
PRINT : PRINT "ENTER MESSAGE TO SEND"
PRINT
INPUT WRT$
    CALL IBWRT(BD%, WRT$)
    GOSUB GPIB.ERR
HOLD.TIME = TIMER          'slight delay for srq checking
DO WHILE TIMER < HOLD.TIME + 1
    LOOP
QUES = INSTR(1, WRT$, "?")    'if ques=0, there
IF QUES = 0 THEN GOTO SEND.RCV 'is no response; if
    CALL IBRD(BD%, RD$)      'message contains "?"
    GOSUB GPIB.ERR          'get response and print it
PRINT : PRINT "THE RESPONSE IS:"
PRINT : PRINT MID$(RD$, 1, IBCNT%)
PRINT : PRINT
INPUT "SEND MORE (ENTER Y OR N)?"; Y$
IF Y$ = "Y" THEN GOTO SEND.RCV
RETURN

```

'subroutine to fetch current instrument settings from 2710 and
'save to disk

```

SAVE.SET:
CLS : PRINT
PRINT "ENTER NAME FOR SETTINGS FILE"
PRINT "USE PATH IF NOT IN CURRENT DIRECTORY"
INPUT FILENAME$
WRT$ = "SET?"
    CALL IBWRT(BD%, WRT$)    'request settings

```

//////////////////////////////////// 2710 GPIB Programmers Manual //////////////////////////////////////

```

GOSUB GPIB.ERR
CALL IBRD(BD%, RD$)           'read settings
GOSUB GPIB.ERR
SETTINGS$ = MID$(RD$, 1, IBCNT%) 'eliminate extra
PRINT : PRINT SETTINGS$      'characters & print
PRINT : PRINT
INPUT "OK TO STORE (ENTER Y OR N)? "; Y$
IF Y$ = "N" THEN RETURN      'store if everything
OPEN "O", #1, FILENAME$, IBCNT% 'looks OK
PRINT #1, SETTINGS$
CLOSE #1
RETURN

```

'subroutine to restore a group of instrument settings from disk
'to the 2710

```

RES.SET:
CLS : PRINT
PRINT "ENTER NAME OF SETTINGS FILE"
PRINT
INPUT FILENAME$
OPEN "I", #1, FILENAME$      'read settings file
INPUT #1, SETTINGS$
CLOSE #1
PRINT : PRINT SETTINGS$
INPUT "OK TO RESTORE (ENTER Y OR N)? "; Y$
IF Y$ = "N" THEN RETURN      'if displayed settings
WRT$ = SETTINGS$            'OK, then restore
CALL IBWRT(BD%, WRT$)       'to 2710
GOSUB GPIB.ERR
RETURN

```

'subroutine to fetch a file from the 2710 and store it on disk

```

SAVE.FILE:
CLS : PRINT
PRINT "ENTER NAME OF 2710 FILE TO STORE"
INPUT FILE2710$ 'see FILE command for 2710 file names
PRINT
PRINT "ENTER NAME OF DISK FILE FOR STORING"
INPUT FILENAME$
FILE2710$ = UCASE$(FILE2710$)
WRT$="HDR ON;FILE? "+CHR$(34)+FILE2710$+CHR$(34)

```

//////////////////////////////////// 2710 GPIB Programmers Manual //////////////////////////////////////

```

CALL IBWRT(BD%, WRT$)      'request file transfer
GOSUB GPIB.ERR
CALL IBRDF(BD%, FILENAME$) 'read and store
GOSUB GPIB.ERR            '2710 file to disk
RETURN                    'as FILENAME$
'
'subroutine to restore a 2710 file from disk to the 2710
RES.FILE:
CLS : PRINT
PRINT "ENTER DISK FILE TO RESTORE TO 2710"
INPUT FILENAME$          'note: the file named
CALL IBWRTF(BD%, FILENAME$) 'FILENAME$
GOSUB GPIB.ERR          'contains the name of
RETURN                  'the 2710 file to be restored
'
'subroutine to fetch curve data in packed binary form and convert
'it to 2-byte integer format
INT.CUR:
PRINT
PRINT "GET CURVE FROM WHICH REGISTER?"
INPUT " (ENTER A, B, C, OR D) "; REG$
PRINT
'ensure response header is on
CALL IBWRT(BD%, "HDR ON")
GOSUB GPIB.ERR
'tell 2710 which register and encoding to use
WRT$ = "WFMPRE WFID:" + REG$ + ",ENCDG:BINBLK"
CALL IBWRT(BD%, WRT$)
GOSUB GPIB.ERR
CALL IBWRT(BD%, "CUR?")
CALL IBRDI(BD%, CUR%(), 9)      'fetch curve data in
GOSUB GPIB.ERR                'packed binary;write
CALL IBRDI(BD%, CUR%(), 512)   'over header char-
GOSUB GPIB.ERR                'acters (1st9) with data
CALL DEBLK(CUR%(), CUR%(), 512, 8, NUMBYT%)
PRINT "# OF BYTES CONVERTED = "; NUMBYT%
RETURN                          'DEBLK unpacks binary data and re-stores
                                'as 2-byte integers in same array.
                                'NUMBYT$ should always equal 512

```

'subroutine to find and display event code following an SRQ
'created by an abnormal event

```

ABNORM.EVE:
  CLS
  PRINT "AN ABNORMAL EVENT HAS OCCURRED."
  PRINT
    GOSUB SERIAL.POLL      'call subroutines to poll
    GOSUB EVENT.FIND      '2710 and find event code
  PRINT
  PRINT "R TO RESTART; ANY OTHER KEY TO END"
  INPUT KEY$              'pressing R returns to
  IF KEY$ <> "R" THEN END 'menu; variables
    GOTO MENU              'are not erased
RETURN
    
```

'subroutine to serially poll the 2710, read the status byte, and print
'it; used as part of abnormal event handler, but can be used at any
'time to obtain status byte; see your 'GPIB documentation for more
'info re serial polling

```

SERIAL.POLL:
  CALL IBRSP(BD%, SPR%)    'read, print status
  PRINT "STATUS BYTE = "; SPR% 'byte and reset SRQ
RETURN
    
```

'Subroutine to find the event code at any time using the EVE?
'query, but result is valid only after a serial poll 'if RQS is ON

```

EVENT.FIND:
  PRINT "EVENT CODE(S) IS:";
  WRT$ = "HDR OFF;EVE?"    'command to turn off header
  EVENT.CODE$ = SPACE$(5)  'and request event code
  CALL IBWRT(BD%, WRT$)    'send command
  CALL IBRD(BD%, EVENT.CODE$) 'read code
  PRINT EVENT.CODE$        'print code
  GOSUB GPIB.ERR
RETURN
    
```

'subroutine to print the GPIB error code if a GPIB error occurs, and
'end the program gracefully

```

GPIB.ERR:
  IF IBSTA% < 32768 THEN RETURN 'no GPIB error if GPIB
  CLS : PRINT                  'status word<32768
  PRINT "GPIB ERROR HAS OCCURRED."
    
```

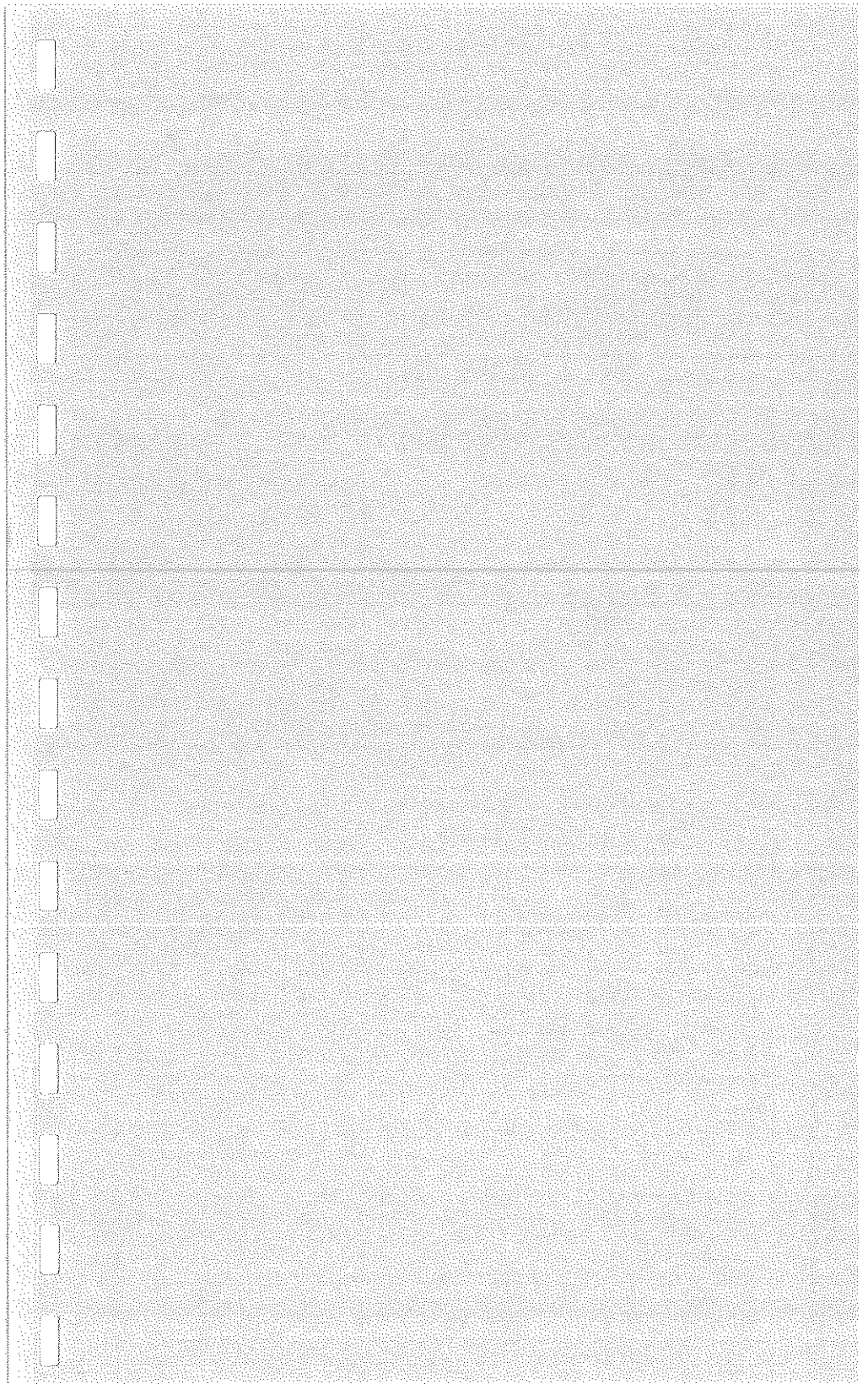
////// 2710 GPIB Programmers Manual ////

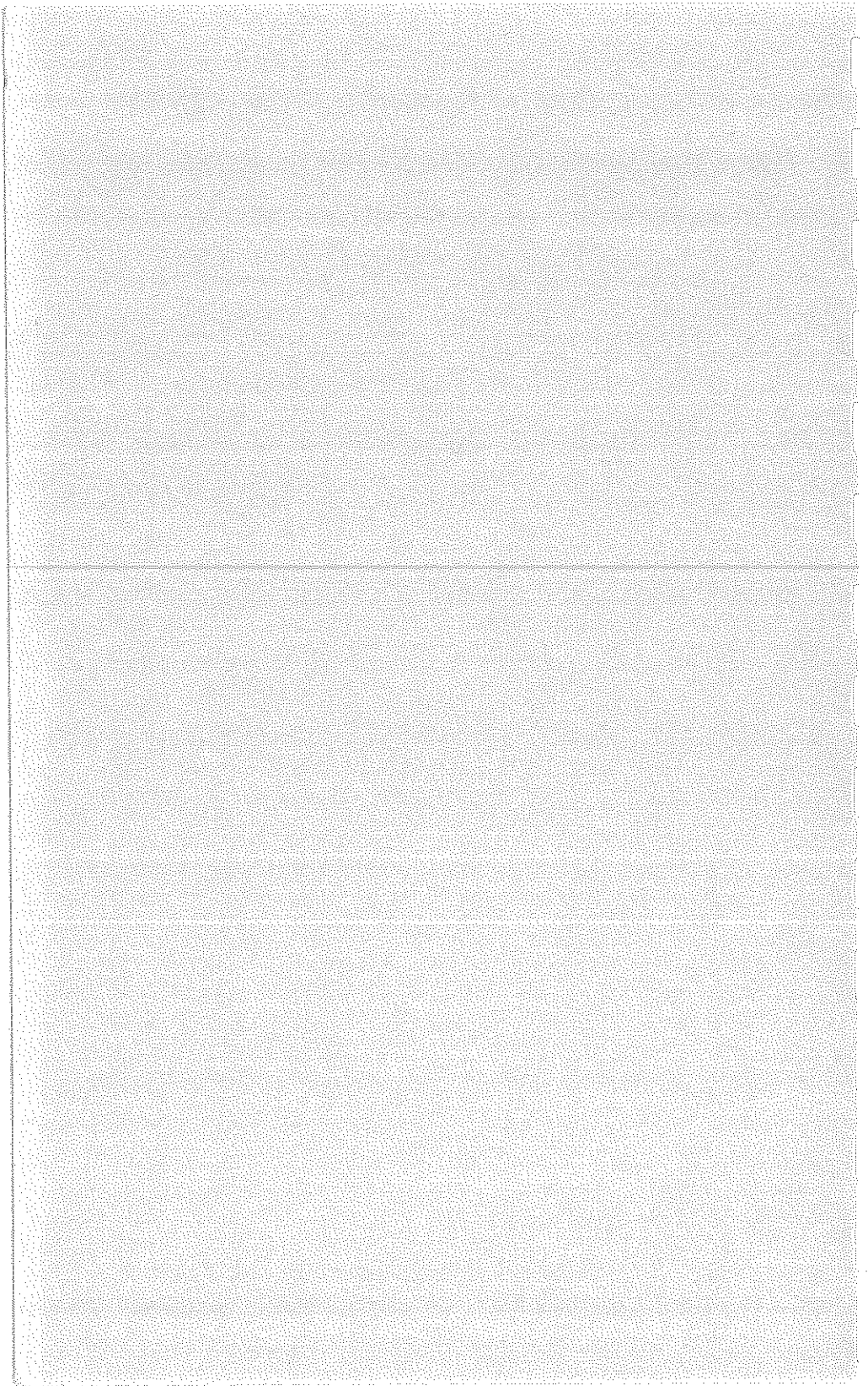
```
PRINT : PRINT "GPIB ERROR CODE IS "; IBERR%  
PRINT : PRINT  
PRINT "CHECK YOUR SYSTEM AND RESTART."  
END  
RETURN
```

'subroutine to end program gracefully if a DOS error occurs

```
ERR.TRAP:  
CLS : PRINT  
PRINT "DOS ERROR HAS OCCURRED."  
PRINT : PRINT  
PRINT "CHECK YOUR SYSTEM AND RESTART."  
END  
RESUME NEXT
```







Appendix A - IEEE STD 488 (GPIB) System Concepts

The General Purpose Interface Bus (GPIB) is a digital control bus that allows efficient communications between self-contained instruments or devices interconnected in an instrumentation system. The GPIB is an interface system independent of the stimulus or measurement functions incorporated in any instrument.

Instruments or devices designed to operate on the digital control bus must be developed according to the specifications contained in IEEE Std 488-1978, "IEEE Standard Digital Interface for Programmable Instrumentation." The IEEE 488 digital interface is commonly known as the General Purpose Interface Bus (GPIB). This section discusses the basic concepts of the GPIB. (For complete specifications, refer to the IEEE Std 488-1978 standard, published by the Institute of Electrical and Electronics Engineers, Inc.).

The GPIB has four elements; mechanical, electrical, functional, and operational. Of these four, only the last is device-dependent. Operational elements state the way in which each instrument reacts to a signal on the bus.

Mechanical Elements

The IEEE Std 488 defines the GPIB connector and cable assembly as the mechanical elements of the instrumentation system. Standardizing the connector and cable assembly ensures that GPIB-compatible instruments can be physically linked together with complete pin compatibility. The connector has 24 pins; sixteen active signal lines, seven interlaced grounds, and 1 shield connection. Standard connector pin arrangement and nomenclature for the digital control signals are illustrated in Figure A-1.

The cable that attaches to the GPIB connector must be

no longer than 20 meters maximum with no more than fifteen peripheral devices (including a GPIB controller) connected at one time. The interconnecting cable assembly, which is offered as an optional accessory to the spectrum analyzer, is provided with both a plug and receptacle connector type at each end of the cable to allow either a star or linear bus structure. Contact your local Tektronix Field Office or representative for cable ordering information. Connectors may be rigidly stacked, using standard counter-bored captive screws.

Electrical Elements

The voltage and current values required at the connector nodes on the bus are based on TTL technology. The power source is not to exceed +5.25 V referenced to logic ground. The standard defines the logic levels as follows.

1. Logical 1 is a true state, low voltage level ($\leq +0.8$ V), signal line is asserted.
2. Logical 0 is a false state, high voltage level ($\geq +2.0$ V), signal line is not asserted.

Messages can be sent over the GPIB as either active-true or passive-true signals. Passive-true signals occur at a high voltage level and must be carried on a signal line using open-collector devices. Active-true signals occur at a low voltage level.

Functional Elements

The functional elements of the GPIB cover three areas.

1. The ten major interface functions of the GPIB, are listed in Table A-1. Each interface function is a system element that provides the basic operational facility through which an instrument can receive, process, and send messages over the GPIB.
2. The second functional element is the specific protocol by which the interface functions send and receive

Table A-1. Major GPIB interface functions.

INTERFACE FUNCTION	SYMBOL
Source Handshake	SH
Acceptor Handshake	AH
Talker or Extended Talker	T or TE
Listener or Extended Listener	L or LE
Service Request	SR
Remote-Local	RL
Parallel Poll	PP
Device Clear	DC
Device Trigger	DT
Controller	C

Note that while the IEEE Std 488 standard defines the ten interface functions, the specific protocol, and timing relationships, not every instrument on the bus will have all ten interface functions incorporated. Only those functions important to a particular instrument's purpose need to be implemented.

A Typical GPIB System

A typical GPIB instrumentation system is illustrated in Figure A-2, and it includes the nomenclature for the sixteen active signal lines. Only four instruments are shown connected directly to the control bus, but the GPIB can support up to fifteen instruments connected directly to the bus. However, more than fifteen devices can be interfaced to a single bus if they do not connect directly to the bus, but are interfaced through a primary device. Such a scheme can be used for programmable plug-ins housed in a mainframe where the mainframe is addressed with a primary address code and the plug-ins are addressed with a secondary address code.

To maintain the electrical characteristics of the bus, a device load should be connected for each two meters of cable length. Although instruments are usually spaced no

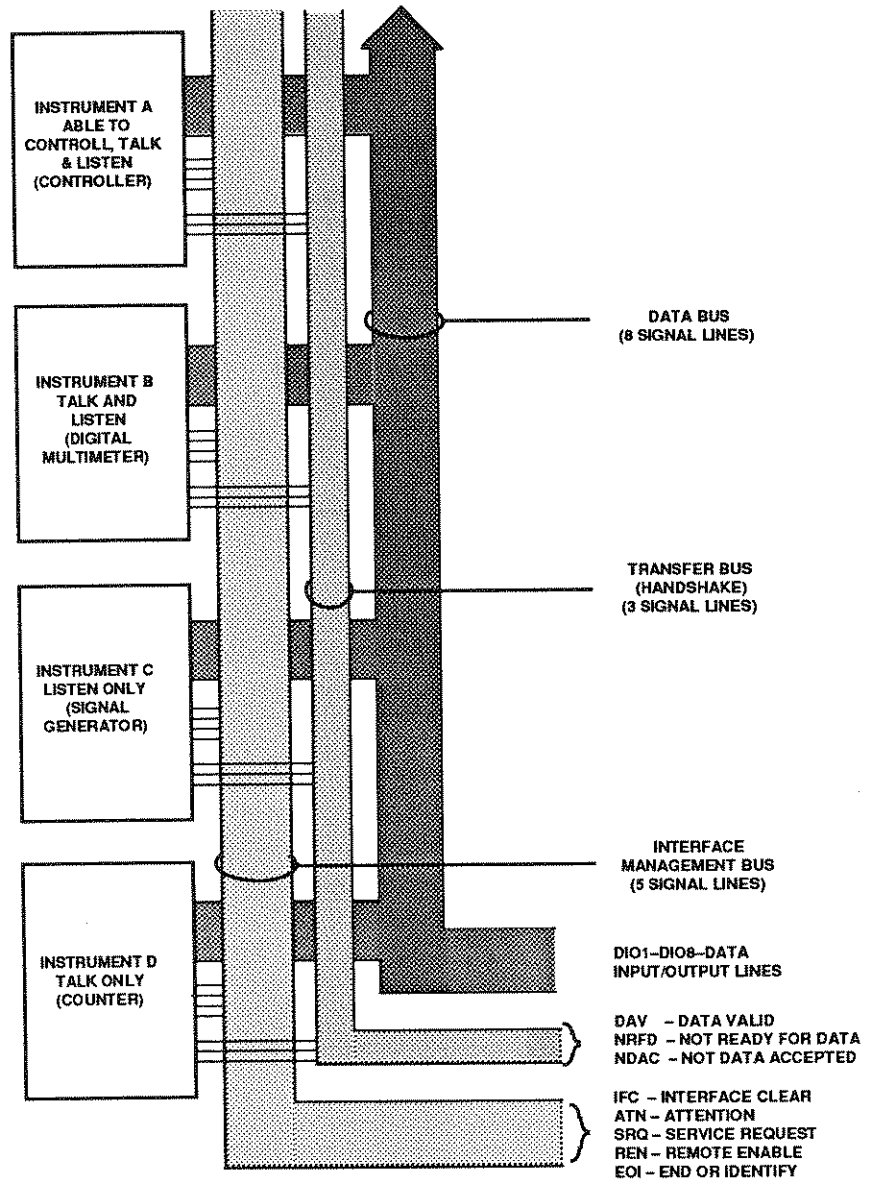


Figure A-2. A typical GPIB system.

more than two meters apart, they can be separated farther apart if the required number of device loads are lumped at any given point. For proper operation, at least two-thirds of the instruments connected directly to the bus must be in the power-on state.

Talkers, Listeners, and Controllers

A talker is an instrument that can send messages and data over the bus; a listener is an instrument that can accept messages and data from the bus. An instrument can be a talker only, listener only, or be both a talker and a listener. Unless a device is in the talk-only or listen-only mode, it can only communicate with other devices on the bus when it is enabled to do so by the controller in charge of the instrumentation system.

A controller is an instrument that determines, by software routines, which instrument will talk and which instruments will listen during any given time interval. The controller has the ability to assign itself as a talker or a listener whenever the program routine requires it. In addition to designating the current talker and listeners for a particular communication sequence, the controller is assigned the task of sending special codes and commands (called interface control messages) to any or all instruments on the bus. A complete operating system may contain more than one controller. The IEEE standard has provisions for a system controller that operates with another controller in charge of the bus. The controller that is in charge of the bus can take control only when it is directed to do so by the system controller. The system controller itself may be, but is not necessarily, the controller in charge of the bus.

Interface Control Messages

The two types of interface control messages are multi-line messages sent over the data bus and uni-line messages.

A message that shares a group of signal lines with other messages, in some mutually exclusive set, is called a multi-line message (only one multi-line message (message byte) can be sent at one time).

A message sent over a single line is called a uni-line message (two or more of these messages can be sent concurrently.)

Only multi-line messages are discussed here; uni-line messages are discussed later under GPIB Signal Line Definitions.

The interface control messages (refer to Figure A-3) are sent and received over the data bus only with the ATN (attention) line asserted (true). Interface message coding can be related to the ISO (International Standards Organization) 7-bit code by relating data bus lines DIO1 through DIO7 to bits B1 through B7, respectively, in the Bits column in Figure A-3.

Interface control messages (refer to Table A-2) include the primary talk and listen addresses for instruments on the bus, addressed commands (only instruments previously addressed to listen will respond to these commands), universal commands (all instruments, whether they have been addressed or not, will respond to these commands), and secondary addresses for devices interfaced through a primary instrument. Parallel Poll Enable (PPE) messages are derived from the characters in the first column under Lower Case letters in Figure A-3 (decimal coded characters 96 through 111). The standard recommends the use of decimal code 112 (lower case letter p) for the Parallel Poll Disable (PPD) command. All parallel poll configured instruments respond with status information at the same time when the EOI line is asserted with ATN true.

Device-Dependent Messages

The IEEE standard does not specify the coding of device-dependent messages (messages that control the internal operating functions of a device). After addressing a talker and the required number of listeners via interface control messages, the controller unasserts the ATN line (false) on the bus. When ATN becomes false (high), any commonly understood 8-bit binary code may be used to represent a device-dependent message.

However, the standard recommends that the alphanumeric codes associated with the numbers, symbols, and upper case characters (decimal 32 to decimal 94) in the ASCII Code Chart (Figure A-3) be used to compose device-dependent messages. One example of a device-dependent message could be the following ASCII character string:

```
MODE V;VOLTS 2.5E-3;FREQ 1.0E3
```

The ASCII character string, sent with the ATN line unasserted, tells the instrument to set its front panel controls to the voltage mode and output a 2.5 mV signal at a frequency of 1000 Hz.

When 8-bit binary codes other than the ISO 7-bit are used for device-dependent messages, the most significant bit should be on data line DI08 (for bit-8).

To summarize the difference between interface control messages and device-dependent messages on the data bus, remember that any message sent or received when the ATN line is asserted (low) is an interface control message. Any message (data bytes) sent or received when the ATN line is unasserted (high) is a device-dependent message.

Table A-2. Interface messages and functions referred to in this appendix.

Remote Messages Received		
Mnemonic	Message	Interface Function
ATN	Attention	AH,C,L,LE,PP,SH,T,TE
DAC	Data Accepted	SH
DAV	Data Valid	AH
DCL ^a	Device Clear	DC
GET ^a	Group Execute Trigger	DT
GTL ^a	Go To Local	RL
IFC	Interface Clear	C,L,LE,T,TE
LLO ^a	Local Lockout	RL
MSA ^a	My Secondary Address	LE,TE
MTA ^a	My Talk Address	T,TE
PPC ^a	Parallel Poll Configure	PP
PPD ^a	Parallel Poll Disable	PP
PPE ^a	Parallel Poll Enable	PP
PPU ^a	Parallel Poll Unconfigure	PP
REN	Remote Enable	RL
RFD	Ready For Data	SH
SDC ^a	Selected Device Clear	DC
SPD ^a	Serial Poll Disable	T,TE
SPE ^a	Serial Poll Enable	T,TE
SRQ	Service Request	(via C)
TCT ^a	Take Control	C
UNL ^a	Unlisten	L,LE

^a Multi-line messages

Table A-2 (continued)

Remote Messages Sent		
Mnemonic	Message	Interface Function
ATN	Attention	C
DAC	Data Accepted	AH
DAV	Data Valid	SH
DCL ^a	Device Clear	(via C)
GET ^a	Group Execute Trigger	(via C)
GTL ^a	Go To Local	(via C)
IFC	Interface Clear	C
LLO ^a	Local Lockout	(via C)
MSA ^a	My Secondary Address	(via C)
MTA ^a	My Talk Address	(via C)
PPC ^a	Parallel Poll Configure	(via C)
PPD ^a	Parallel Poll Disable	(via C)
PPE ^a	Parallel Poll Enable	(via C)
PPU ^a	Parallel Poll Unconfigure	(via C)
REN	Remote Enable	C
RFD	Ready For Data	AH
SDC ^a	Selected Device Clear	(via C)
SPD ^a	Serial Poll Disable	(via C)
SPE ^a	Serial Poll Enable	(via C)
SRQ	Service Request	SR
TCT ^a	Take Control	(via C)
UNL ^a	Unlisten	(via C)
UNT ^a	Untalk	(via C)

^a Multi-line messages

GPIB Signal Line Definitions

Figure A-2 shows how the sixteen active signal lines on the GPIB are functionally divided into three component buses: an eight-line data bus, a three-line data byte transfer control (handshake) bus, and a five-line general interface management bus.

The data bus contains eight bidirectional signal lines, DI01 through DI08. Information, in the form of data bytes, is transferred over this bus. A handshake timing sequence between the enabled talker and the enabled listeners on the three-line data transfer control bus transfers one data byte (eight bits) at a time. These data bytes are sent and received in a byte-serial, bit-parallel fashion.

Since the handshake sequence is an asynchronous operation (no clock signal on the bus), the data transfer rate is only as fast as the slowest instrument involved in a data byte transfer. A talker cannot place data bytes on the bus faster than the slowest listener can accept them.

Figure A-4 illustrates the flow of data bytes on the bus when a typical controller sends ASCII data to an assigned listener. The first data byte (decimal 44) enables an instrument at address 12 as a primary listener. The second data byte (decimal 108) is optional; for example, enabling a plug-in device at secondary address 12 as the final destination of the data to follow. The data is the two ASCII characters A and B (decimal 65 and decimal 66). Note that the ATN line is asserted for the first two data bytes and unasserted for the device-dependent character to indicate the last data byte in the message.

The controller activates the ATN line again and sends the universal unlisten (UNL) and untalk (UNT) commands to clear the bus. Six handshake cycles on the data transfer control bus are required to send the six data bytes.

Transfer Bus (Handshake)

Each time a data byte is transferred over the data bus, an enabled talker and all enabled listeners execute a handshake sequence via signal lines DAV, NRFD, and NDAC (see Figure A-5 -- the ATN line is shown to illustrate the controller's role in the process).

DAV (Data Valid). The DAV signal line is asserted by the talker after the talker places a data byte on the data bus. When asserted (low), DAV tells each assigned listener that a new data byte is on the bus. The talker is inhibited from asserting DAV as long as any listener holds the NRFD signal line asserted.

NRFD (Not Ready For Data). An asserted NRFD signal line indicates one or more of the assigned listeners are not ready to receive the next data byte from the talker. When all of the assigned listeners for a particular data byte transfer have released NRFD, the NRFD line becomes unasserted (high). When NRFD goes high, the RFD message (Ready For Data) tells the talker it may place the next data byte on the data bus.

NDAC (Not Data Accepted). Each assigned listener holds the NDAC signal line asserted until the listener accepts the data byte currently on the bus. When all assigned listeners have accepted the current data byte, the NDAC signal line becomes unasserted (high), telling the talker to remove the data byte from the bus. The DAC message (Data Accepted) tells the talker that all assigned listeners have accepted the current data byte.

Note that one handshake cycle transfers one data byte; then, the listeners reset the NRFD line high and the NDAC line low before the talker asserts DAV for the next data byte transfer. Both NRFD and NDAC high at the same time is an invalid state on the bus.

Management Bus

The management bus is a group of five signal lines that are used to control the operation of the IEEE Std 488 (GPIB) Digital Interface.

IFC (Interface Clear)

The system controller is the only instrument on the bus allowed to assert IFC. IFC is asserted for greater-than 100 μ s to place all instruments in a predetermined state. While IFC is being sent, only the DCL (Device Clear), LLO (Local Lockout), PPU (Parallel Poll Unconfigure), and REN (Remote Enable) interface messages (universal commands) will be recognized.

ATN (Attention)

The controller in charge is the only instrument on the bus allowed to assert ATN. ATN is asserted when an instrument connected to the bus is being enabled as a talker or listener, or when sending other interface control messages. As long as the ATN line is asserted (low), only instrument address codes and interface control messages are sent over the bus. When the ATN line is unasserted, only those instruments enabled as a talker and listener can send and receive data over the bus.

SRQ (Service Request)

Any instrument connected to the bus can request the controller's attention by asserting the SRQ line. The controller responds by asserting ATN and executing a serial poll routine to determine which instrument is requesting service. The instrument requesting service responds with a device-dependent status byte with bit seven asserted. When the instrument requesting service is found, program control is transferred to a service routine for that instrument. When the service routine is completed, program control returns to the main program. (The controller does not have to see the SRQ line asserted to perform a polling routine; it may do so whenever a program requires it).

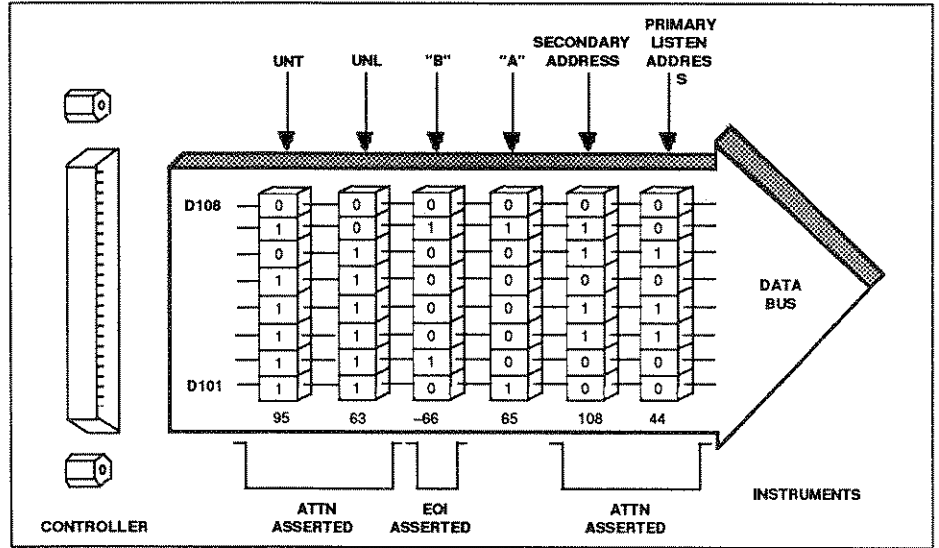


Figure A-4. An example of data byte traffic on the GPIB.

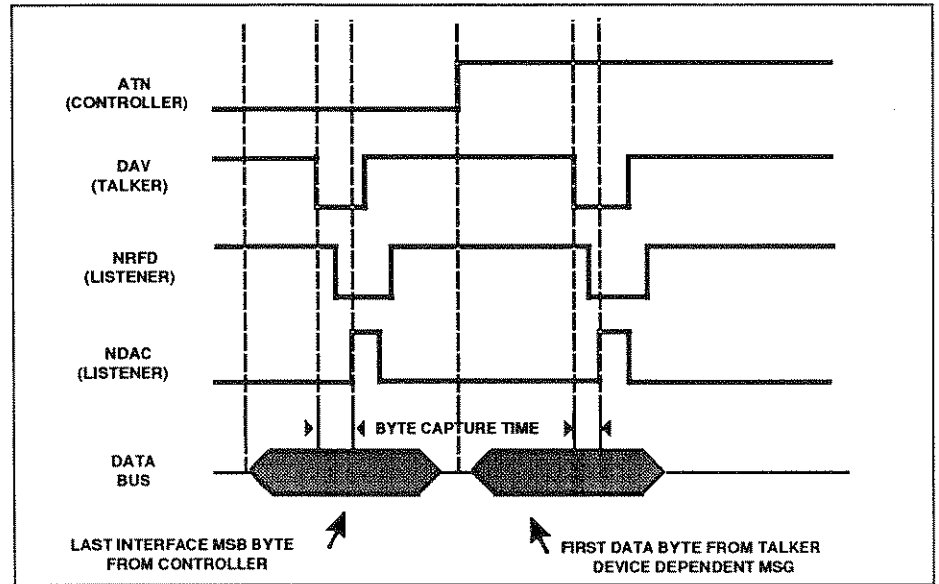


Figure A-5. A typical handshake timing sequence (idealized).

REN (Remote Enable)

The system controller asserts the REN signal line whenever the interface system operates under remote program control. Used with other interface control messages, such as LLO (Local Lockout) or GTL (Go To Local), the REN signal causes an instrument on the bus to select between two alternate sources of programming data. A remote-local interface function indicates to an instrument that the instrument will use either information input from the interface (remote) or to information input by the operator via the front panel controls (local).

EOI (End Or Identify)

A talker can use the EOI signal line to indicate the end of a data transfer sequence. The talker asserts EOI as the last byte of data is transmitted. In this case, the EOI line is essentially a ninth data bit and must observe the same settling time as the data on the data bus. When an instrument controller is listening, it assumes that a data byte sent with EOI asserted is the last data byte in the complete message. When the instrument controller is talking, it may assert the EOI signal line as the last data byte is transferred. The EOI line is also asserted with the ATN line true if the controller conducts a parallel polling sequence on the bus. The EOI line is not used for a serial polling sequence.

Interface Functions and Messages

The ten major interface functions listed in Table A-1 provide a variety of capabilities and options for an instrumentation system. These functions may be implemented in, or for, any particular instrument with instrument hardware or with a programming routine (software).

Only those functions necessary for an instrument's purpose need be implemented by the instrument's designer; it is not likely that one single instrument will have all ten interface functions. For example, an

instrument generally doesn't need to implement the Parallel Poll (PP) function if the instrument can respond to a serial polling sequence from the controller in charge of the GPIB system.

The following discusses the interface functions and their relationship to the interface control messages shown in Figure A-3. All the interface control messages discussed are sent and received over the GPIB with the ATN line asserted (low).

RL (Remote-Local Function)

The RL function provides an instrument with the capability to select between two sources of input information. This function indicates to the instrument that its internal device-dependent functions are to respond to information input from the front panel (Local) or to corresponding programming information from the GPIB (Remote). Only the system controller is permitted to assert the REN (Remote Enable) line, whether or not it is the controller in charge at the time.

When the system controller asserts the REN line, an instrument on the GPIB goes to a remote mode when it is addressed as a listener with its listen address, not before. An instrument remains in a remote mode until the REN line is released (high), or an optional front-panel switch on the instrument is activated to request the local mode, or a GTL (Go To Local) command is received while the instrument is enabled as a listener.

However, the controller can disable the instrument's front-panel "return to local" switch(es) by sending a LLO (Local Lockout) command. The LLO command must be preceded or followed by a listen address (MLA) to cause the instrument to go to a remote mode with front-panel lockout. The UNL (Unlisten) command does not return an instrument to the local mode.

When the REN line goes false, it must be recognized by all instruments on the bus and they must go to the local

mode within 100 μ s. If data bytes are still being placed on the bus when REN goes false, the system program should assure that the data bytes are sent and received with the knowledge that the system is in a local mode, not remote.

T/TE and L/LE (Talker and Listener Functions)

Although discussed under one heading, the T/TE and L/LE functions are independent of each other.

The T (Talker) and TE (Talker Extended) functions provide an instrument and its secondary devices, if any, with the capability to send device-dependent data over the GPIB (or, in case of a controller, the capability to send device-dependent program data) over the GPIB. The Talker (T) function is a normal function for a talker and uses only a one-byte primary address code called MTA (My Talk Address). The Talker Extended (TE) function requires a two-byte address code; an MTA code followed by the second byte called MSA (My Secondary Address).

Only one instrument in the GPIB system can be in the active talker state at any given time. A non-controller commences talking when ATN is released and continues its talker status until an Interface Clear (IFC) message occurs or an Untalk (UNT) command is received from the controller in charge. The instrument will stop talking and listen any time that the controller in charge asserts ATN.

One or more instruments on the bus can be programmed for the L (Listener) function by use of their specific primary listen address (called MLA). Some of the instruments interfaced to the bus may be programmed for the LE (Listener Extended) function, if implemented. The LE function requires a two-byte address code. No L or LE function is active during the time that ATN is asserted.

All talker and listener functions must respond to ATN

within 200 ns. They must also respond to IFC in less than 100 μ s.

An instrument may be a talker only, a listener only, or implement all functions. In any case, its address code has the form X10TTTTT for a talker and X01LLLLL for a listener. For instruments with both T and L functions, the T-bit binary values are usually equal to the binary value of the L bits. Before applying power to the system, the system operator sets these five least significant bits by means of an address switch on each instrument. The controller's address code may be implemented in software.

The system program, run from the controller, designates the primary talker and primary listener status of the desired instruments by coding data bits 6 and 7; 1, 0, respectively, for a talker and 0, 1, respectively, for a listener. Secondary talk and listen addresses (or commands) are represented by the controller sending both data bits (6 and 7) as a logical 1. The controller may listen to bus traffic without actually addressing itself over the bus.

SH and AH (Source and Acceptor Handshake Functions)

Although discussed under one heading, the SH and AH functions are independent of each other.

The SH (Source Handshake) function guarantees proper transmission of data, while the AH (Acceptor Handshake) function guarantees proper reception of data. The interlocked handshake sequence between these two functions guarantees asynchronous transfer of each data byte. The handshake sequence is performed via the NRFD, DAV, and NDAC signal lines on the bus (see Figure A-5). Both functions must respond to ATN within 200 nsec.

The SH function must wait for the RFD (Ready For Data) message plus wait at least 2 μ s before asserting DAV; this allows the data to settle on the data bus. If

three-state drivers are used, the settling time is reduced to RFD plus 1.1 μ s. Faster settling times are allowed under special conditions and warning notes in the standard. The time it takes for the AH function to accept an interface message byte is dependent on how the designer implemented the function.

DC (Device Clear Function)

The DCL (Device Clear) function allows the controller in charge to "clear" any or all instruments on the bus. The controller (under program direction) asserts ATN and sends either the universal DCL (Device Clear) command or the SDC (Selected Device Clear) command.

When the DCL message is received, all instruments on the bus must clear or initialize their internal device functions. When the controller sends the SDC command, only those instruments that have been previously addressed to listen must respond. The IEEE 488 standard does not specify the settings an instrument must go to as a result of receiving the DCL or SDC command. (In general, these commands are used only to clear the GPIB interface circuits within an instrument.)

DT (Device Trigger Function)

The DT (Device Trigger) function allows the controller in charge to start the basic operation specified for an instrument or group of instruments on the bus. The IEEE 488 standard does not specify the basic operation an instrument is to perform when it receives the GET (Group Execute Trigger) command. To issue the GET command, the controller asserts ATN, sends the listen addresses of the instruments that are to respond to the trigger, and then sends the GET message.

Once an instrument starts its basic operation in response to GET, the instrument must not respond to subsequent trigger-state transitions until the current operation is complete. Only after completing the operation can the instrument repeat its basic operation in response to the

next GET message. Thus, the basic operating time is the major factor that determines how fast the instrument(s) can be repeatedly "triggered" by commands from the bus.

C, SR, and PP (Controller, Service Request, and Parallel Poll Functions)

The C (Controller) function provides the capability to send primary talk and listen addresses, secondary addresses, universal commands, and addressed commands to all instruments on the bus. The Controller function also provides the capability to respond to a service request message (SRQ) from an instrument or to conduct a parallel poll routine to determine the status of any or all instruments on the bus that have the Parallel Poll (PP) function implemented.

If an instrumentation system has more than one controller, only the system controller is allowed to assert the IFC (Interface Clear) and REN (Remote Enable) lines at any time during system operation, whether or not it is the controller in charge at the time.

If a controller requests system control from another controller and it receives a message from another controller to send REN, the system controller must verify that the REN line remains unasserted (false) for at least 100 μ s before asserting REN. The time interval that REN is asserted depends on the remote programming sequence and will vary with the program. The IFC line must be asserted for at least 100 μ s.

The Controller function has specified time intervals for certain operations. For example, the execution time for parallel polling instruments on the bus cannot be less than 2 μ s. If the controller is in the controller active wait state and does not receive an internal message to conduct a parallel poll, it must wait for at least 1.5 μ s before going to the controller active state in order to give the NRFD, NDAC, and EOI lines sufficient time to assume their valid states.

The controller must also have a delay of at least 2 μ s (1.1 μ s for tri-state drivers) in order for the instruments to see the ATN line asserted before the controller places the first data byte on the bus.

Taking Control (Asynchronous or Synchronous)

All data bytes transmitted over the GPIB with the ATN line asserted are interpreted as system control information. Asserting ATN directly at any moment is an asynchronous operation with respect to the bus and may cause loss of data if a handshake cycle is in progress. To prevent loss of data, a controller can take control synchronously, that is, it can monitor the Transfer Bus and only assert ATN when DAV is unasserted (false). As a controller in charge, the system controller (program) may pass control to any other instrument in the system capable of acting as a controller. The controller in charge first addresses the other controller as a talker and then sends the TCT (Take Control) command. The other controller then becomes the controller in charge when ATN is released.

Performing a Serial Poll

The controller-in-charge may conduct a serial poll at any time, whether or not an instrument on the bus has asserted the SRQ line. (Most, but not all, instruments have the Service Request (SR) function.)

To perform a serial poll, the controller first asserts ATN and issues the Untalk (UNT) and Unlisten (UNL) commands. The controller then sends the Serial Poll Enable (SPE) command, followed by the talk address of the first instrument to be polled. The controller then releases ATN and the addressed talker responds by sending its status byte over the bus. If the addressed talker has requested service, it must assert bit seven of the status byte and encode the remaining seven bits to indicate the reason for asserting SRQ. Status bytes are device-dependent and are not specified in the IEEE 488 standard. An addressed instrument will release its SRQ

line when serial polled, but other instruments may still hold it asserted. When the controller has read the status byte of an addressed instrument, it reasserts ATN and addresses the next instrument to talk, then releases ATN and receives the instrument's status byte. The routine continues until the controller no longer detects the SRQ line asserted. At this time, the controller should send the Serial Poll Disable (SPD) message and, optionally, send the UNT message to release the last active talker.

Performing A Parallel Poll

The Parallel Poll (PP) function provides an instrument with the capability to present one, and only one, bit of status information to the controller without being previously addressed to talk. The parallel polling capability requires a commitment by the system program to periodically conduct a parallel poll sequence.

When an instrument responds to a parallel poll, the single data bit presented to the controller may or may not indicate a need for service. If the data bit is used as a service request indication, the controller should perform a serial poll in order to obtain a complete status byte with more information (if the device has the SR function implemented).

Before an instrument can respond to a parallel poll, the GPIB system must first be configured. In a typical sequence, the controller first sends an UNL command to clear the bus of listeners, then the listen address of the device to be configured. Following this, the controller sends the PPC (Parallel Poll Configure) command followed by a PPE (Parallel Poll Enable) message. The PPE message contains coded information that tells the selected instrument which data line will carry the PP status bit for that device. This entire sequence is repeated for each instrument to be configured.

The PPE message(s) sent by the controller has the form X110SPPP. Bit 4 (S) is called the sense bit and the three least significant bits (PPP) represent an octal number (0

through 7) that corresponds to a specific line on the data bus that an instrument must assert if its internal status has the same value as the sense bit (S may equal 1 or 0).

The actual parallel poll takes place after each instrument has been completely configured. The concept is to have the controller receive one data byte that contains status information on all of the addressed instruments. To receive this status byte, the controller asserts the EOI line and the ATN line. The assertion of EOI may be coincident with ATN or later, so long as both are asserted. This may occur any time after the last PPE message. The controller then reads the data bus lines while ATN and EOI are asserted to interpret the status of all selected instruments.

To conclude the parallel poll, the controller releases EOI and then ATN. The instrument(s) do not need to be reconfigured for each subsequent parallel poll. The PPU (Parallel Poll Unconfigure) command will clear all device configurations and prevent them from responding to future polls. The PPD (Parallel Poll Disable) command accomplishes essentially the same thing, except that the PP function remains in the "configured" state. PPU is a universal command (all instruments) while PPD is used with PPC and becomes an addressed command (only those devices selected with PPC will accept PPD.)

