HP 3563A

PROGRAMMING MANUAL

Control Systems Analyzer

HEWLETT PACKARD

HP Part No. 03563-90005
Microfiche No. 03563-90205

Printed in U.S.A.
Print Date: December 1989

©Hewlett-Packard Company, 1989. All rights reserved.
8600 Soper Hill Road
Everett, Washington 98205-1298 U.S.A.

HEWLETT PACKARD

CERTIFICATION

Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Bureau of Standards, to the extent allowed by the Bureau's calibration facility, and the calibration facilities of other International Standards Organization Members.

WARRANTY

This Hewlett-Packard product is warranted against defects in material and workmanship for a period of one year from date of shipment. During the warranty period, Hewlett-Packard Company will, at its option, either repair or replace products which prove to be defective.

For warranty service or repair, this product must be returned to a service facility designated by HP. Buyer shall prepay shipping charges to -hp- and -hp- shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to -hp- from another country.

HP software and firmware products which are designated by HP for use with a hardware product, when properly installed on the hardware product, are warranted not to fail to execute their programing instructions due to defects in materials and workmanship. If HP receives notice of such defects during their warranty period, HP shall repair or replace software media and firmware which do not execute their programming instructions due to such defects. HP does not warrant that the operation of the software, firmware or hardware shall be uninterrupted or error free.

LIMITATION OF WARRANTY

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HEWLETT-PACKARD SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR PARTICULAR PURPOSE.

EXCLUSIVE REMEDIES

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HEWLETT-PACKARD SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

ASSISTANCE

Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products.

For any assistance, contact your nearest Hewlett-Packard Sales and Service Office. Addresses are provided at the back of this manual.

HEWLETT PACKARD

SAFETY SUMMARY

The following general safety precautions must be observed during all phases of operation, service, and repair of this instrument. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of the instrument. Hewlett-Packard Company assumes no liability for the customer's failure to comply with these requirements. This is a Safety Class 1 instrument.

GROUND THE INSTRUMENT

To minimize shock hazard, the instrument chassis and cabinet must be connected to an electrical ground. The instrument is equipped with a three-conductor ac power cable. The power cable must either be plugged into an approved three-contact electrical outlet or used with a three-contact to two-contact adapter with the grounding wire (green) firmly connected to an electrical ground (safety ground) at the power outlet. The power jack and mating plug of the power cable meet International Electrotechnical Commission (IEC) safety standards.

DO NOT OPERATE IN AN EXPLOSIVE ATMOSPHERE

Do not operate the instrument in the presence of flammable gases or fumes. Operation of any electrical instrument in such an environment constitutes a definite safety hazard.

KEEP AWAY FROM LIVE CIRCUITS

Operating personnel must not remove instrument covers. Component replacement and internal adjustments must be made by qualified maintenance personnel. Do not replace components with power cable connected. Under certain conditions, dangerous voltages may exist even with the power cable removed. To avoid injuries, always disconnect power and discharge circuits before touching them.

DO NOT SERVICE OR ADJUST ALONE

Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.

DO NOT SUBSTITUTE PARTS OR MODIFY INSTRUMENT

Because of the danger of introducing additional hazards, do not install substitute parts or perform any unauthorized modification to the instrument. Return the instrument to a Hewlett-Packard Sales and Service Office for service and repair to ensure the safety features are maintained.

DANGEROUS PROCEDURE WARNINGS

Warnings, such as the example below, precede potentially dangerous procedures throughout this manual. Instructions contained in the warnings must be followed.

Warning **Dangerous voltages, capable of causing death, are present in this instrument. Use extreme caution when handling, testing, and adjusting.**

HEWLETT PACKARD

SAFETY SYMBOLS

General Definitions of Safety Symbols Used On Equipment or In Manuals.

⚠	Instruction manual symbol: the product will be marked with this symbol when it is necessary for the user to refer to the instruction manual in order to protect against damage to the instrument.
ϟ	Indicates dangerous voltage (terminals fed from the interior by voltage exceeding 1000 volts must be so marked.)
⏚ OR (⏚)	Protective conductor terminal. For protection against electrical shock in case of a fault. Used with field wiring terminals to indicate the terminal which must be connected to ground before operating equipment.
(⏚)	Low-noise or noiseless, clean ground (earth) terminal. Used for a signal common, as well as providing protection against electrical shock in case of a fault. A terminal marked with this symbol must be connected to ground in the manner described in the installation (operating) manual, and before operating the equipment.
⊥ OR ⊥	Frame or chassis terminal. A connection to the frame (chassis) of the equipment which normally includes all exposed metal structures.
~	Alternating current (power line.)
⎓	Direct current (power line.)
≂	Alternating or direct current (power line.)

Warning — **The WARNING sign denotes a hazard. It calls attention to a procedure, practice, condition or the like, which if not correctly performed or adhered to, could result in injury or death to personnel.**

Caution — The CAUTION sign denotes a hazard. It calls attention to an operating procedure, practice, condition or the like, which, if not correctly performed or adhered to, could result in damage to or destruction of part or all of the product.

Note — The NOTE sign denotes important information. It calls attention to procedure, practice, condition or the like, which is essential to highlight.

Table of Contents

Chapter 3: The Data Transfer Group

Chapter 4: Signal Processing Group

Appendix A: Quick Reference Guide

Appendix B: Example Programs

1

HP 3563A and the HP-IB

Purpose of This Chapter

This chapter describes the HP-IB capabilities of the HP 3563A and explains how it interacts with the HP-IB. It is assumed that you are familiar with the operation of the HP 3563A and with HP-IB programming. The topics discussed in this chapter are:

- Connecting the HP-IB System
- The HP-IB capabilities of the HP 3563A
  - Interface capabilities
  - Controller capabilities
  - Interrupts and instrument status
- The HP 3563A's command set
- The HP 3563A's response to bus management commands
- Programming hints

For general information on the HP-IB, contact your HP Sales Representative for a copy of the following document :

Tutorial Description of the Hewlett-Packard Interface Bus
Part Number: 5952-0156

Connecting the HP-IB System

The following describes how to connect the HP 3563A to an HP-IB system:

1. Remove power from the HP 3563A.
2. Connect the HP 3563A to a computer using an HP-IB cable.

Caution

The HP 3563A has metric-threaded HP-IB cable mounting studs, as opposed to English threads. Metric-threaded HP-IB cables must be used. Metric fasteners are colored black, while English fasteners are colored silver. DO NOT attempt to connect black and silver fasteners, or damage to cable and instrument will result.

3. If required, load the BASIC language operating system, following the instructions in the computer's operating manual.
4. Apply power to the HP 3563A.
5. Set the HP-IB address of the analyzer. The HP-IB address is stored in nonvolatile memory. To check the current HP-IB address of the analyzer: press the HP-IB FCTN key in the HP 3563A's HP-IB group, press the SELECT ADDRESS softkey, and then press the HP-IB ADDRESS softkey. The analyzer's current address is displayed in the lower left corner of the display. If you want to enter a new address, use the 0-9 numeric keypad and the ENTER softkey. The examples in this manual use address 20 for the HP 3563A.

To verify proper system connection, type in the following command on the computer:

REMOTE 720

and press the computer's **Return**, **EXECUTE**, or **EXEC** key (depending on the type of keyboard you have). After you have issued the remote command, the green REMOTE indicator LED on the HP 3563A's front panel should light up. This indicates that the analyzer is now under the remote control of the computer. Try pressing some keys on the HP 3563A to verify that the keyboard is disabled. If you want to return local (front panel) control to the analyzer, press the LOCAL key in the HP-IB group. You can also issue the "LOCAL 7" command from the computer.

HP-IB Capabilities of the HP 3563A

The HP 3563A can be selected as the HP-IB system controller or as an addressable-only device. When it is the system controller, the analyzer directs the flow of commands and data on the bus. This also allows it to directly control digital plotters, access disc drives, and output HP-IB command strings. When it is the only controller in an HP-IB system, the analyzer is usually operated as the system controller.

When operated on the bus with another controller (a desktop computer, for example), the analyzer generally operates in addressable-only mode. When it is addressable-only, it responds to commands and data from the system controller. When the HP 3563A needs to be in control of the bus, it can accept control from the system controller, then automatically pass back control when finished. Refer to "Passing Control" in Chapter 6 for more information.

Interface Capabilities

The HP 3563A has the following interface capabilities, as defined by IEEE Standard 488-1978:

SH1	Complete Source handshake
AH1	Complete Acceptor handshake
T6	Basic Talker; serial poll; unaddress if MLA; no Talk Only
TE0	No Extended Talker capability
L4	Basic Listener; unaddress if MTA; no Listen Only
SR1	Complete Service Request capability
RL1	Complete Remote/Local capability
PP0	No Parallel Poll capability
DC1	Complete Device clear capability
DT1	Complete Device Trigger capability
C1	System Controller capability
C2	Send IFC and Take Charge Controller capability
C3	Send REN Controller capability
C12	Send IF messages; receive control; pass control capability
E1	Open-collector drivers (250 kBytes/s maximum)

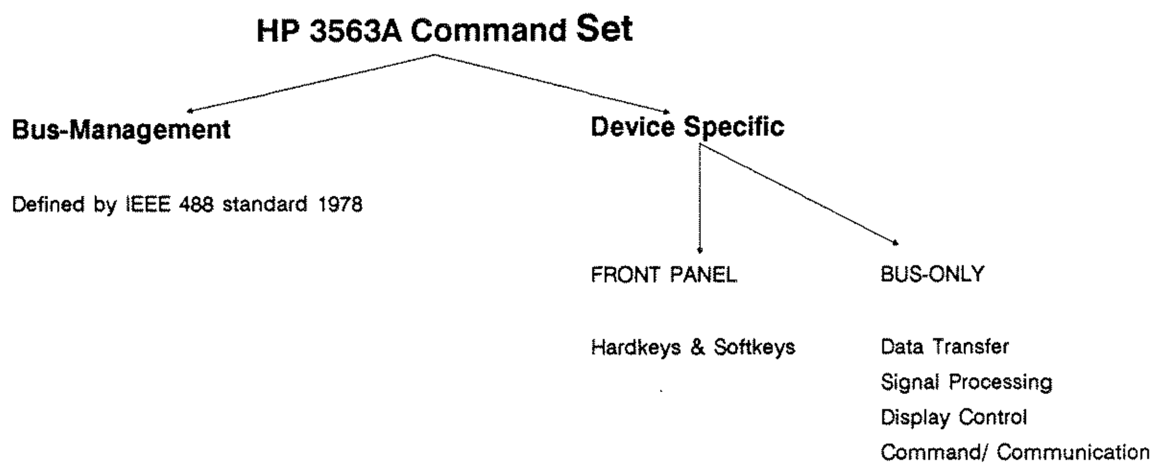
Refer to IEEE Standard 488-1978 if you need more detailed information.

Interrupt and Instrument Status Features

When the HP 3563A analyzer is in addressable-only mode, conditions within the analyzer can cause it to generate service requests (SRQs). SRQs are usually generated for two reasons either the analyzer needs control of the bus to perform an operation, or there is a change in the status of the analyzer that the controller might want to know about. Refer to chapter 6 for complete information on service requests and status registers.

The HP 3563A's Command Set

The HP 3563A responds to two different types of commands — bus management commands (covered in the next section) and device specific commands. The device-specific command set can be further divided into commands that emulate front panel keystrokes (Front Panel Commands) and commands that are available only via HP-IB (Bus-only commands). Bus-only commands are further broken down into the following four groups: Data Transfer, Signal Processing, Display Control and Control Communication.



Front Panel Group

As the name implies, this group emulates the hardkeys and softkeys on the analyzer's front panel.

Chapter 2 provides mnemonics for the front panel group. It is alphabetized by hardkey, with the softkeys associated with each key listed in order of appearance. Chapter 2 is designed to help you easily emulate front panel operation via HP-IB.

Data Transfer Group

This group allows you to transfer information to and from the analyzer. This information includes:

- Data traces
- Coordinate transform data blocks
- Synthesis tables
- Throughput/capture data
- Instrument states

Chapter 3 explains how to use Data Transfer commands. It also explains how to interpret data headers and convert data traces.

Signal Processing Group

This group provides access to the analyzer's signal processing primitives. It allows you to set up data blocks in memory, operate on these blocks (for example, using FFT or averaging), then transfer the blocks back to the controller or to display them on the analyzer's display.

Chapter 4 explains how to use the commands in the Signal Processing Group.

Display Control Group

This group provides control of the HP 3563A's vector display. The display can be controlled by:

- Using individual HP-GL (Hewlett-Packard Graphics Language) commands
- Loading an entire display from a controller
- Defining the display as the plotter and using HP BASIC graphics commands

Chapter 5 shows how to program the display using these three approaches.

Command/Communication Group

This final group provides:

- Control and communication functions, (including service requests)
- Instrument and activity status
- Reading marker values
- Communication with the front panel

Chapter 6 explains how to use these commands. It also describes the status byte, instrument status, and activity status parameters.

Bus Management Commands

Bus management commands are used to control all interface hardware connected to the bus. This section describes the HP 3563A's response to the primary bus commands. Your controller's programming or interfacing manual should contain information on these commands from the controller's perspective.

Abort I/O

This command instructs the HP 3563A to abort input or output. It is an unconditional assumption of control of the bus by the system controller. All bus activity halts and the HP 3563A becomes unaddressed. This does not, however, clear the analyzer's HP-IB command buffer or clear any pending input or output data. The HP 3563A does not relinquish bus control when it receives this command.

BASIC example: ABORT 7

where: 7 is the interface select code

Clear Lockout & Set Local

The LOCAL command, universal local, instructs all instruments on the specified interface to clear the local-lockout mode and return to local (front panel) operation. This command differs from the LOCAL command, addressed local, in that the LOCAL command addresses a specific device and does not clear the lockout mode.

BASIC example: LOCAL 7

Device Clear

The CLEAR command can affect a specific device (addressed clear) or all devices on a specified interface (universal clear). This command causes the HP 3563A to:

- Clear its HP-IB command buffer
- Reset the SRQ bus management line (if it had been activated by the instrument)
- Reset all status byte, instrument status and activity status masks
- Abort any data input or output

The CLEAR command unconditionally interrupts bus activity and gains control of the instrument. It does not, however, reset any HP 3563A parameters.

BASIC examples: CLEAR 720 (addressed clear)
CLEAR 7 (universal clear)

where: 7 is the interface select code
20 is the HP 3563A's address

Local

The LOCAL command returns local (front panel) control to the HP 3563A. When the instrument is under local control, the REMOTE front panel indicator is off and the keyboard is enabled. The HP-IB command buffer is not cleared by issuing the LOCAL command. Any load operation in progress continues but the HP 3563A aborts dump operations in progress.

BASIC example: LOCAL 720

Local Lockout

This command disables the **LOCAL** front panel key of the HP 3563A. It does not change the remote/local status of the instrument. However, it does prevent the operator from using the **LOCAL** hardkey to enable the front panel keyboard when the REMOTE command is in effect. When in remote control, LOCAL LOCKOUT secures the system from operator interference.

While the LOCAL LOCKOUT command is in effect and the instrument is in remote control, the only way to return to front panel operation is by issuing the LOCAL command on the bus. If an unaddressed (universal) LOCAL command is used (such as, LOCAL 7) LOCAL LOCKOUT is disabled and subsequent remote commands can be overridden from the front panel. If an addressed LOCAL command is used (such as, LOCAL 720), local lockout will still be in effect when the device is later returned to remote control.

BASIC example: LOCAL LOCKOUT 7

Parallel Poll

This command and its accompanying PARALLEL POLL CONFIGURE are ignored by the HP 3563A. See SERIAL POLL.

Parallel Poll Configure

This command and its accompanying PARALLEL POLL are ignored by the HP 3563A. See SERIAL POLL.

Pass Control

This command shifts control of the bus from one controller to another. The Controller Address command, CTAD, (default = 21) should be sent prior to Passing Control. Not all controllers have the ability to pass control — consult the operation manual of your controller.

If control is passed to the HP 3563A before it has a need for it, the analyzer immediately passes the control to the address specified by the Controller Address command. Refer to "Passing Control" in Chapter 6.

BASIC example: PASS CONTROL 720

Remote

When this command is issued, the front panel LED labeled "REMOTE" illuminates and the front panel keys are disabled (except the **LOCAL** key if local lockout is not active; if Local lockout is active, even the **LOCAL** hardkey is disabled). This command can be used to address the HP 3563A to listen.

BASIC examples: REMOTE 7 (universal)
REMOTE 720 (addressed)

Serial Poll

This command instructs the HP 3563A to send its status byte to the controller. If your program sends multiple serial polls, pause for at least 5 ms between them. Refer to chapter 6 for more information.

BASIC example: Status_byte = SPOLL(720)

Trigger

This command triggers measurements in the HP 3563A in the same manner as its other trigger modes. TRIGGER must first be enabled in the analyzer by sending the "HPT" command or pressing the HP-IB TRIG softkey to select HP-IB triggering.

BASIC examples: TRIGGER 7 (universal)
TRIGGER 720 (addressed)

Programming Hints

- See the beginning of Chapter 2 for hints on emulating front panel hardkeys/softkeys.
- Pause the controller for several seconds after sending resets or special presets if you want to send marker or math commands.
- When programming anything on the display—especially markers—make sure that there is a valid data display first.
- The HP 3563A can buffer up to 3 lines of 80 characters each.
- If you request information from the analyzer (query, data transfer, etc.), allow for the information to be input to the controller immediately.
- If AUTO CAL is ON, you will encounter long delays when the calibration routine is run. This could affect your program if it contains timeouts. To avoid this, you can turn auto calibration off and do a single calibration. Send the following commands:

```
AUTO 0
SNGC
```

- Wait 30 sec before sending another command (this deactivates auto cal, then runs a single cal routine). You should perform a single cal periodically.
- When activating external sampling (ESMP1), pause the program briefly to allow the HP 3563A to measure the external sample clock.

2

Front Panel Commands

About This Chapter

This chapter shows you the HP-IB commands for the HP 3563A's hardkeys and softkeys. In addition, it explains special considerations for some of the front panel commands. For syntax, entry ranges, and suffixes, refer to the *Quick Reference Guide* (Appendix A). For descriptions of the hardkeys and softkeys, see the HP 3563A *Operating Manual*. The remaining commands — the "bus-only" commands — are covered in Chapters 3 through 6.

This chapter is organized alphabetically by the hardkeys on the front panel. Under each hardkey, the softkeys accessed are displayed in order of appearance. The HP-IB mnemonic is shown beside every hardkey and softkey. This reinforces your knowledge of front panel operation and may help as you begin writing controller programs. If you know the particular functions you want to program, use the *Quick Reference Guide* (Appendix A), which lists all hardkey and softkey commands as well as bus-only commands in alphabetical order. The *Quick Reference Guide* also explains the syntax for all commands.

Another way to learn HP-IB mnemonics is to look in the lower left-hand corner of the display. This is the command echo field, and the letters that form the HP-IB mnemonic are underlined. The command COMD disables the command echo field and COME enables it. The default is to have command echo enabled. Also, when you press HELP for a particular hardkey or softkey, the appropriate HP-IB mnemonic appears at the top of the help display.

Front Panel versus HP-IB Control

Hardkey/Softkey Menu Structure Considerations

A major difference between operating the instrument from the front panel and programming it over the HP-IB is that you do not always have to follow the softkey menu structure with the HP-IB. For example, to select the FFT math function from the front panel, you press **MATH** followed by NEXT, followed by FFT. Over the bus, however, you simply send the FFT command. In a few cases, a particular menu must be displayed before a command can be used. An example is CLEAR TABLE in synthesis. You need to first display the POLE ZERO, POLE RESIDUE, or POLYNOMIAL menu before telling the instrument to clear the table. Refer to the HP 3563A *Operating Manual* for softkey structure information.

Hardkey/Softkey Name versus HP-IB Mnemonics

When programming over the HP-IB, several pairs of softkeys have the same name but different mnemonics. For example, to select the frequency response measurement and the frequency response display from the front panel, you press FREQ RESP in the **SELECT MEAS** menu and FREQ RESP in the **MEAS DISP** menu. Over the bus, however, you send FRSP and FRQR. Since this chapter is organized by key, this confusion is avoided.

Softkeys that Toggle

Softkeys that toggle between two states (for example, TIM AV ON OFF) can be toggled using the mnemonic. However, to guarantee the resulting state, these commands allow you to send 0 or 1 after the mnemonic to explicitly choose one state or the other. Sending "TIAV1" explicitly activates time averaging, while "TIAV" merely toggles the existing state. The results of sending 0 and 1 for each toggle softkey are explained in the *Quick Reference Guide.*

Softkeys that need to be Enabled

Some softkeys are not enabled and do not appear in the menu until the HP 3563A is set up in a particular measurement mode. For more information on activating softkeys see, the Operating Reference in the HP 3563A *Operating Manual* or see the comment section of the *Quick Reference Guide.*

Parameter Queries

You can learn the current value of any variable parameter in the analyzer by sending the appropriate command followed by a question mark. For example, to learn the current frequency span, you could send the following BASIC statements:

```
OUTPUT 720; "FRS?"
ENTER 720; Freq_span
```

where:	7 is the Interface select code and 20 is the analyzer's address
	FRS is the mnemonic for the FREQ SPAN softkey Freq_span is the variable the value is entered into

The Alpha Mode

Softkeys used in the alpha mode (SPACE FORWRD, SPACE BACKWD, INSERT ON OFF, DELETE CHAR, CLEAR LINE, AT POINTR, OVER WRITE, and CANCEL ALPHA) are not programmable over the HP-IB. When you need to send alpha characters, simply include them with the commands enclosed in single quotes. The *Quick Reference Guide* (Appendix A) shows the syntax for every command requiring alpha entries.

Front Panel Hardkey/Softkey to HP-IB Mnemonic Cross-Reference

A	(A)
A&B	(A B)
ARM	(ARM)
AUTO MATH	(AMTH)
EDIT MATH	(EDMA)
VIEW MATH	(VWMA)
START MATH	(STMA)
LABEL MATH	(LBLM)[1]
EDIT LINE #	(LINE)[1]
DELETE LINE	(DLTL)[1]
CHANGE LINE	(CHGL)[1,2]
ADD LINE	(ADDL)[1,2]
CLEAR MATH	(CLMA)[1]
END EDIT	(ENED)[1]

1 The EDIT MATH menu must be displayed before these commands can be used.

2 When ADDL or CHGL is sent, the analyzer stays in the add line or change line mode, respectively. All subsequent commands until ENED (END EDIT) are added or changed.

AUTO SEQ	(ASEQ)
START ASEQ 1	(ASQ1)[3]
START ASEQ 2	(ASQ2)[3]
START ASEQ 3	(ASQ3)[3]
START ASEQ 4	(ASQ4)[3]
START ASEQ 5	(ASQ5)[3]
PAUSE ASEQ	(PSAS)
CONT ASEQ	(CNAS)
SELECT ASEQ#	(SASQ)
EDIT	(EDIT)
VIEW	(VIEW)
LABEL ASEQ	(LBLA)[1]
EDIT LINE#	(LINE)[1]
DELETE LINE	(DLTL)[1]
CHANGE LINE	(CHGL)[1,2]
ADD LINE	(ADDL)[1,2]
CLEAR ASEQ	(CLAS)[1]
ASEQ FCTN	(ASFN)[1]
END EDIT	(ENED)[1]
LOOP TO	(LPTO)[4]
GO TO	(GOTO)[4]
ASEQ MESSGE	(ASMS)[4]
TIMED PAUSE	(TIPS)[4]
TIMED START	(TIST)[4]
DSPLAY ON OFF	(DSPL)[4]
LOAD ASEQ	(LASQ)[4]
RETURN	(RTN)4
LINRES FRF	(LFRF)[5]
SWEPT FRF	(SFRF)[5]
STEP RESPNS	(SRSP)[5]
DIGTAL MSMNTS	(DGMS)[5]
CURVE FIT	(CFIT)[5]
LOAD ALL	(LALL)[5]
RETURN	(RTN)[5]

1 The EDIT menu must be displayed before this command can be used.

2 When ADDL or CHGL is sent, the analyzer stays in the add line or change line mode, respectively. All subsequent commands until ENED (END EDIT) are added or changed.

3 The mnemonics for these are always ASQ1-5, even when the labels are replaced by user-defined labels.

4 The ASEQ FCTN menu must be displayed before this command can be used.

5 The LOAD ASEQ menu must be displayed before this command can be used.

AVG (AVG)

(linear res mode)

NUMBER AVGS	(NAVG)
AVG OFF	(AVOF)
STABLE (MEAN)	(STBL)
EXPON	(EXP)
PEAK HOLD	(PHLD)
CONT PEAK	(CNPK)
TIM AV ON OFF	(TIAV)
NEXT	(NX)
OVRLP%	(OVLP)[2]
OV REJ ON OFF	(OVRJ)
FST AV ON OFF	(FSAV)[2]
PRVIEW OFF	(PROF)[2]
MANUAL PRVIEW	(MAPR)[1,2]
TIMED PRVIEW	(TIPR)[1,2]
RETURN	(RTN)[2]

(log res mode)

NUMBER AVGS	(NAVG)
AVG OFF	(AVOF)
STABLE (MEAN)	(STBL)
EXPON	(EXP)
PEAK HOLD	(PHLD)
CONT PEAK	(CNPK)
NEXT	(NX)
OVRLP%	(OVLP)
OV REJ ON OFF	(OVRJ)
FST AV ON OFF	(FSAV)
RETURN	(RTN)

(swept sine mode)

NUMBER AVGS	(NAVG)
INTGRT TIME	(INTM)
AUTO INTGRT	(AUIN)
FIXED INTGRT	(FXIN)

(time capture mode)

NUMBER AVGS	(NAVG)
AVG OFF	(AVOF)
STABLE (MEAN)	(STBL)
EXPON	(EXP)
PEAK HOLD	(PHLD)
CONT PEAK	(CNPK)
TIM AV ON OFF	(TIAV)
OVRLP%	(OVLP)

[1]Use ACPT for YES and REJT for NO when previewing over the bus.
[2]Only valid when DEMOD ON OFF is off.

B (B)

CAL (CAL)

AUTO ON OFF	(AUTO)
SINGLE CAL	(SNGC)

COORD	(CORD)
MAG (dB)	(MGDB)
MAG (dBm)	(MDBM)
MAG (LOG)	(MGLG)
MAG (LIN)	(MAG)
PHASE	(PHSE)
REAL	(REAL)
IMAG	(IMAG)
NEXT	(NEXT)
NYQUST	(NYQT)
NICHOL	(NICL)
LOG X	(LOGX)
LIN X	(LINX)
RETURN	(RTN)
CURVE FIT	(CVFT)
CREATE FIT	(CRFI)
DOMAIN S Z	(DMNS)
NUMBER POLES	(NPOL)
NUMBER ZEROS	(NZER)
LAST MEAS	(LSMS)
A & B TRACES	(ABTR)
EDIT TABLE	(EDTB)
FIT FCTN	(FTFN)
START FIT	(STFT)
STOP FIT	(SPFT)
EDIT POLES	(EPOL)
EDIT ZEROS	(EZER)
FIX LINE#	(FXLN)
UNFIX LINE#	(UFLN)
ADD LINE	(ADLN)
DELETE LINE#	(DLLN)
TABLE FCTNS	(TBFN)
RETURN	(RTN)
TIME DELAY	(TMDL)
SCALE FREQ	(SCLF)[1]
SAMPLE FREQ	(SFRE)[2]
MEAS CLOCK	(MSCK)[2]
CLEAR TABLE	(CLTA)
RETURN	(RTN)

[1]Must be in S domain
[2]Must be in Z domain.

USER WEIGHT	(USWT)
AUTO WEIGHT	(AUWT)
USER ORDER	(USOR)
AUTO ORDER	(AUOR)
FIT → SYNTH	(FTSN)
SYNTH → FIT	(SNFT)
EDIT WEIGHT	(EDWT)
RETURN	(RTN)
VIEW WEIGHT	(VWWT)
WEIGHT REGION	(WTRG)
WEIGHT VALUE	(WTVL)
STORE WEIGHT	(STWT)
RETURN	(RTN)

DISC	(DISC)
SAVE FILE	(SAVF)[1]
RECALL FILE	(RCLF)[1]
DELETE FILE	(DLTF)
VIEW CATLOG	(CAT)
NEXT PAGE	(NXTP)
PREV PAGE	(PRVP)
CATLOG POINTR	(CTPT)
DISC FCTN	(DIFN)
SERVCE FCTNS	(SVFN)
DISC COPY	(DICO)
FORMAT	(FORM)
PACK DISC	(PKDI)
THRUPT SIZE	(THSZ)
CREATE THRUPT	(CRTH)
ABORT HP-IB	(ABIB)
RETURN	(RTN)
RESTOR CATLOG	(RSCT)
RO ERT TEST	(RERT)
OUTPUT LOG	(OULG)
NEXT PAGE	(NXPG)
CLEAR LOGS	(CLLG)
DISC STATUS	(DIST)
SPARE BLOCK	(SPBL)
RETURN	(RTN)
FAULT LOG	(FTLG)
ERT LOG	(ERLG)
RUN TM LOG	(RULG)
RETURN	(RTN)
DESTN ADDRES	(DEAD)
DESTN UNIT	(DEUN)
COPY FILES	(COFI)
OVERWR AU MAN	(OVAU)
RESUME OVERWR	(RSOV)
RESUME COPY	(RSCO)
IMAGE BACKUP	(IMBK)
RETURN	(RTN)
FORMAT OPTION	(FOOP)
INIT DISC	(INDI)
INIT CATLOG	(INCT)
RETURN	(RTN)

[1] Data must be displayed

ENGR UNITS (ENGR)

EU VAL CHAN1	(EUV1)
VOLTS CHAN1	(VLT1)
EU LBL CHAN1	(EUL1)
EU VAL CHAN2	(EUV2)
VOLTS CHAN2	(VLT2)
EU LBL CHAN2	(EUL2)

FREQ (FREQ)

(linear res & time capture modes)

FREQ SPAN	(FRS)
START FREQ	(SF)
CENTER FREQ	(CF)
ZERO START	(ZST)
MAX SPAN	(MAXS)
TIME LENGTH	(TLN)
E SMPL ON OFF	(ESMP)[3]
SAMPLE CLOCK	(SACL)[3]
SAMPLE FREQ	(SMPF)[2]
POD Q CLOCK	(PQCL)[3]
EXT SAMPLE	(EXTS)[3]
SAMPLE FREQ	(SMPF)[3]
RETURN	(RTN)

(log res mode)

FREQ SPAN	(FRS)
START FREQ	(SF)

(swept sine mode)

FREQ SPAN	(FRS)
START FREQ	(SF)
CENTER FREQ	(CF)
STOP FREQ	(SPF)
RESLTN	(RES)
RESLTN AU FIX	(RSAU)
SWEEP RATE	(SWRT)[1]

[1]Same as SWEEP RATE in **SOURCE** menu.
[2]Either or both channels are digital.
[3]Both channels are analog.

FRONT BACK (FRBK)

HP-IB FCTN (IBFN)

SYSTEM CNTRLR	(SYSC)
ADDRES ONLY	(ADRS)
SELECT ADDRES	(SADR)
USER SRQ	(USRQ)
OUTPUT STRING	(OUT)[1]
ABORT HP-IB	(ABIB)[2]
HP-IB ADDRES	(IBAD)
PLOT ADDRES	(PLAD)
DISC ADDRES	(DIAD)
DISC UNIT	(DIUN)
RETURN	(RTN)
User SRQ1	(SRQ1)
User SRQ2	(SRQ2)
User SRQ3	(SRQ3)
User SRQ4	(SRQ4)
User SRQ5	(SRQ5)
User SRQ6	(SRQ6)
User SRQ7	(SRQ7)
User SRQ8	(SRQ8)

[1] This is programmable over the bus only when entering output strings into an autosequence. OUTPUT STRING cannot be executed immediately because the HP 3563A must be the system controller to use this function.

[2] Same as ABORT HP-IB in the USER LIMITS, DISC FCTN, and START PLOT menus.

INPUT CONFIG (ICNF)

(Channel 1 analog Channel 2 analog)		(Channel 1 digital Channel 2 analog)	
CHAN 1 AN DIG	(C1AN)	CHAN 1 AN DIG	(C1AN)
CHAN 1 AC DC	(C1AC)	INTERFACE 1	(IFC1)
FLOAT CHAN 1	(FLT1)	FROM SOURCE	(FSRC)
GROUND CHAN 1	(GND1)	FROM POD 1	(FPOD)
CHAN 2 AN DIG	(C2AN)	CHAN 2 AN DIG	(C2AN)
CHAN 2 AC DC	(C2AC)	CHAN 2 AC DC	(C2AC)
FLOAT CHAN 2	(FLT2)	FLOAT CHAN 2	(FLT2)
GROUND CHAN 2	(GND2)	GROUND CHAN 2	(GND2)

(Channel 1 analog Channel 2 digital)		(Channel 1 digital Channel 2 digital)	
CHAN 1 AN DIG	(C1AN)	CHAN 1 AN DIG	(C1AN)
CHAN 1 AC DC	(C1AC)	INTERFACE 1	(IFC1)
FLOAT CHAN 1	(FLT1)	FROM SOURCE	(FSRC)
GROUND CHAN 1	(GND1)	FROM POD 1	(FPOD)
CHAN 2 AN DIG	(C2AN)	CHAN 2 AN DIG	(C2AN)
INTERFACE 2	(IFC2)	INTERFACE 2	(IFC2)
FROM POD 1	(FPD1)	FROM POD 1	(FPD1)
FROM POD 2	(FPD2)	FROM POD 2	(FPD2)

Note An interface must be selected before sending the following commands.

(interface 1 or 2)

TWOS COMPL	(TCOM)
OFFSET BINARY	(OBIN)
DATA SIZE	(DASZ)
DATA CLOCK	(DACL)
SAMPLE CLOCK	(SACL)
CHAN 1 CONFIG	(C1CN)
CHAN 2 CONFIG	(C2CN)
RETURN	(RTN)
# BITS 8 16	(BIT8)
UPR 13 BITS	(UBIT)
LOW 13 BITS	(LBIT)
BUS SZ 8 16	(BUS8)
X OVFL ON OFF	(XOVF)
RETURN	(RTN)
C EDGE + -	(CEDG)
CLOCK QUALFR	(CQUL)
LAST 1 0	(LAST)
COMPUT DELAY	(CMPD)
RETURN	(RTN)
POD Q CLOCK	(PQCL)
EXT SAMPLE	(EXTS)
CHAN 1 CLOCK	(C1CL)
CHAN 2 CLOCK	(C2CL)
MIXED RATIO	(MXRT)
C EDGE + -	(CEDG)
SAMPLE FREQ	(SMPF)
RETURN	(RTN)

LOCAL	(LCL)
MATH	(MATH)
ADD	(ADD)
SUB	(SUB)
MPY	(MPY)
DIV	(DIV)
SQUARE ROOT	(SQRT)
RECIP	(RCIP)
NEGATE	(NEG)
NEXT	(NXT)
DIFF	(DIFF)
jw	(JW)
INTGRT	(INGR)
INTGRT INIT=O	(INGI)
jw^{-1}	(JW1)
T/1 – T	(TT)
NEXT	(NEX)
RETURN	(RTN)
REAL PART	(RLPT)
COMPLX CONJ	(CMPC)
LN OF DATA	(LN)
LN^{-1} OF DATA	(LN1)
FFT	(FFT)
FFT^{-1}	(FFT1)
NEXT	(NT)
RETURN	(RTN)
EXPAND	(XPND)
COM-PRESS	(CMPR)
EX-TRACT	(XTRC)
RETURN	(RTN)

MEAS DISP (MDSP)

(linear res mode
freq resp measurement)

FREQ RESP	(FRQR)
COHER	(COHR)
POWER SPEC1	(PSP1)
POWER SPEC2	(PSP2)
CROSS SPEC	(CSPC)
IMPLS RESP	(IRSP)
AUTO MATH	(AUMT)
FILTRD INPUT	(FILT)

(linear res mode
power spec measurement)

POWER SPEC1	(PSP1)
POWER SPEC2	(PSP2)
AUTO MATH	(AUMT)
FILTRD INPUT	(FILT)

(linear res mode
cross corr measurement)

CROSS CORR	(CRCR)
AUTO CORR1	(AUC1)
AUTO CORR2	(AUC2)
AUTO MATH	(AUMT)
FILTRD INPUT	(FILT)

(linear res mode
auto corr measurement)

AUTO CORR1	(AUC1)
AUTO CORR2	(AUC2)
AUTO MATH	(AUMT)
FILTRD INPUT	(FILT)

(linear res mode
histogram measurement)

HIST1	(HIS1)
HIST2	(HIS2)
PDF1	(PDF1)
PDF2	(PDF2)
CDF1	(CDF1)
CDF2	(CDF2)
AUTO MATH	(AUMT)
FILTRD INPUT	(FILT)

TIME REC 1	(TMR1)
TIME REC2	(TMR2)
LINEAR SPEC1	(LSP1)
LINEAR SPEC2	(LSP2)
ORBITS T1vsT2	(ORBT)
DEMOD POLAR	(POLR)
INST	(INST)
INST WNDOWD	(IWND)
AVRG	(AVRG)
RETURN	(RTN)

(log res mode
freq resp measurement)

FREQ RESP	(FRQR)
COHER	(COHR)
POWER SPEC1	(PSP1)
POWER SPEC2	(PSP2)
CROSS SPEC	(CSPC)
AUTO MATH	(AUMT)

(log res mode
power spec measurement)

POWER SPEC1	(PSP1)
POWER SPEC2	(PSP2)
AUTO MATH	(AUMT)

(swept sine mode)

FREQ RESP	(FRQR)
COHER	(COHR)
POWER SPEC1	(PSP1)
POWER SPEC2	(PSP2)
CROSS SPEC	(CSPC)
AUTO MATH	(AUMT)

(time capture mode
power spec measurement)

POWER SPEC1	(PSP1)
POWER SPEC2	(PSP2)
FILTRD INPUT	(FILT)

(time capture mode
histogram measurement)

HIST1	(HIS1)
HIST2	(HIS2)
PDF1	(PDF1)
PDF2	(PDF2)
CDF1	(CDF1)
CDF2	(CDF2)
FILTRD INPUT	(FILT)

(time capture mode
auto corr measurement)

AUTO CORR1	(AUC1)
AUTO CORR2	(AUC2)
FILTRD INPUT	(FILT)

(all time capture)

TIME REC1	(TMR1)
TIME REC2	(TMR2)
LINEAR SPEC1	(LSP1)
LINEAR SPEC2	(LSP2)
INST	(INST)
AVRG	(AVRG)
RETURN	(RTN)

MEAS MODE (MSMD)

LINEAR RES	(LNRS)
LOG RES	(LGRS)
SWEPT SINE	(SSIN)
TIME CAPTUR	(CPTR)

(linear/log res mode)

THRUPT ON OFF	(THRU)
THRUPT SELECT	(THSE)
DEMOD ON OFF	(DMOD)[1]
DEMOD SELECT	(DMSE)[1]
START THRUPT	(STHR)
ABORT THRUPT	(ABTH)
ACTIVE FILE	(ACFL)
THRUPT LENGTH	(THLN)
THRUPT HEADER	(THED)
RETURN	(RTN)
DEMOD CHAN1	(DM1)[1]
DEMOD CHAN2	(DM2)[1]
DEMOD BOTH	(DMB)[1]
PRVIEW ON OFF	(PRON)[1]
PM/FM CRRIER	(PFCR)[1]
DELETE FREQ	(DLFR)[1]
DELETE ON OFF	(DLON)[1]
RETURN	(RTN)[1]
AM CHAN1	(AM1)[1]
FM CHAN1	(FM1)[1]
PM CHAN1	(PM1)[1]
AM CHAN2	(AM2)[1]
FM CHAN2	(FM2)[1]
PM CHAN2	(PM2)[1]
RETURN	(RTN)[1]
AUTO CRRIER	(ACRR)[1]
USER CRRIER	(UCRR)[1]
EDIT LINE#	(EDLN)[1,2]
DELETE REGION	(DLRG)[1,2]
CHANGE REGION	(CHRG)[1,2]
ADD REGION	(ADRG)[1,2]
CLEAR TABLE	(CLRT)[1,2]
RETURN	(RTN)

(swept sine mode)

LINEAR SWEEP	(LNSW)
LOG SWEEP	(LGSW)
A GAIN ON OFF	(AGON)
A GAIN SELECT	(AGSE)
REF CHAN 1	(RFC1)
REF CHAN 2	(RFC2)
REF LEVEL	(RFLV)
SOURCE LIMIT	(SRLM)
SOURCE CHANGE	(SRCH)
RETURN	(RTN)

(time capture mode)

CAPTUR SELECT	(CPSE)
START CAPTUR	(STCP)
ABORT CAPTUR	(ABCP)
CAPTUR POINTER	(CPNT)
POINTR INCRMT	(PTIN)
CAPTUR LENGTH	(CLEN)
CAPTUR HEADER	(CHED)
RETURN	(RTN)

[1] Active in linear res mode only
[2] The DELETE FREQ menu must be displayed before this command can be used.

PAUSE /CONT (PSCN)[1]

[1]PSCN switches back and forth between pause and continue. PAUS explicitly pauses, and CONT explicitly continues, regardless of the key's previous state. To be certain of the resulting state, use PAUS or CONT over the bus.

PLOT (PLOT)

START PLOT	(STPL)
SELECT DATA	(SDAT)
SELECT PENS	(SPEN)
SPEED F S	(SPED)
LINE TYPES	(LNTP)
PAGING CONTRL	(PCTL)
PLOT LIMITS	(PLIM)
PLOT PRESET	(PLPR)
ABORT HP-IB	(ABIB)
RETURN	(RTN)
DATA ONLY	(DATA)
DATA & ANNOT	(DAAN)
DFAULT GRIDS	(DFGR)
SOLID GRIDS	(SLGR)
TICK MARKS	(TKMK)
RETURN	(RTN)
GRID PEN	(GRDP)
TRACE A PEN	(TRAP)
TRACE B PEN	(TRBP)
ANNOT A PEN	(ANAP)
ANNOT B PEN	(ANBP)
MARKER PEN	(MKRP)
RETURN	(RTN)

SOLID LINES	(SLDL)
DASHED LINES	(DSHL)
DOTS	(DOTS)
SOLIDA DASH B	(SLDA)
USER LINES	(ULIN)
LINE A TYPE#	(LINA)[2]
LINE B TYPE#	(LINB)[2]
RETURN	(RTN)
PAGE FORWRD	(PGFW)
PAGE BACKWD	(PGBK)
NO PAGING	(NOPG)
CUT PG ON OFF	(CTPG)
RETURN	(RTN)
PLOT AREA	(PLAR)
GRID AREA	(GRAR)
DFAULT LIMITS	(DLIM)
USER LIMITS	(ULIM)
ROT 90 ON OFF	(ROT)
SET P1 LWR LF	(SEP1)
SET P2 UPR RT	(SEP2)
READ PEN→P1	(RDP1)
READ PEN→P2	(RDP2)
ABORT HP-IB	(ABIB)[1]
RETURN	(RTN)

[1]Same as ABORT HP-IB in the HP-IB FCTN, DISC FCTN, and START PLOT menus.
[2]USER LINES must be selected before this command can be used.

PRESET (PRST)

F RESP LINRES	(FRLN)
F RESP LOGRES	(FRLG)
F RESP SWEPT	(FRSW)
P SPEC LINRES	(PSLN)
TIME CAPTUR	(TMCP)
TIME THRUPT	(TMTH)
RESET	(RST)

RANGE (RNG)

(Channel 1 Analog Chanel 2 Analog)

CHAN 1 RANGE	(C1RG)
AUTO1 UP&DWN	(AU1)
AUTO1 RNG UP	(AU1U)
CHAN 2 RANGE	(C2RG)
AUTO 2 UP&DWN	(AU2)
AUTO 2 RNG UP	(AU2U)

(Channel 1 Digital Channel 2 Analog)

CHAN 1 RANGE	(CIRG)
SOURCE RNG 1	(SRR1)
AUTO 1 16 BIT	(AUT1)[1]
FIXD 1 13 BIT	(FIX1)[1]
CHAN 2 RANGE	(C2RG)
AUTO 2UP&DWN	(AU2)
AUTO 2 RNG UP	(AU2U)

(Channel 1 Analog Channel 2 Digital)

CHAN 1 RANGE	(C1RG)
AUTO 1 UP&DWN	(AU1)
AUTO 1 RNG UP	(AU1U)
CHAN 2 RANGE	(C2RG)
SOURCE RNG 2	(SRR2)
AUTO 2 16 BIT	(AUT2)[1]
FIXD 2 13 BIT	(FIX2)[1]

(Channel 1 Digital Channel 2 Digital)

CHAN 1 RAGE	(C1RG)
SOURCE RNG 1	(SRR1)
AUTO 1 16 BIT	(AUT1)[1]
FIXD 1 13 BIT	(FIX1)[1]
CHAN 2 RANGE	(C2RG)
SOURCE RNG 2	(SRR2)
AUTO 2 16 BIT	(AUT2)[1]
FIXD 2 13 BIT	(FIX2)[1]

[1]Must be in swept sine mode.

SAVE RECALL (SAVR)

RECALL PWR DN	(RCLP)
RECALL STATE#	(RCLS)
SAVE STATE#	(SAVS)
RECALL DATA#	(RCLD)
SAVE DATA#	(SAVD)

SCALE (SCAL)

X FIXD SCALE	(XSCL)
X MRKR SCALE	(XMKR)[1]
X AUTO SCALE	(XASC)[1]
Y FIXD SCALE	(YSCL)
Y MRKR SCALE	(YMKR)[2]
Y AUTO SCALE	(YASC)[2]
Y DFLT SCALE	(YDSC)[2]

[1]Same as corresponding softkey in the X menu.
[2]Same as corresponding softkey in the Y menu.

SELECT MEAS (SMES)

(linear res mode)		(log res mode)	
FREQ RESP	(FRSP)	FREQ RESP	(FRSP)
POWER SPEC	(PSPC)	POWER SPEC	(PSPC)
AUTO CORR	(AUCR)	CH 1&2 ACTIVE	(CH12)
CROSS CORR	(CCOR)	CH 1 ACTIVE	(CH1)
HIST	(HIST)	CH 2 ACTIVE	(CH2)
CH 1&2 ACTIVE	(CH12)		
CH 1 ACTIVE	(CH1)		
CH 2 ACTIVE	(CH2)		

(swept sine mode)		(time capture mode)	
FREQ RESP	(FRSP)	POWER SPEC	(PSPC)
		AUTO CORR	(AUCR)
		HIST	(HIST)
		CH 1 ACTIVE	(CH1)
		CH 2 ACTIVE	(CH2)

SELECT TRIG (SELT)

TRIG LEVEL	(TRLV)
ARM AU MAN	(ARMA)
FREE RUN	(FREE)
CHAN 1 INPUT	(C1IN)
CHAN 2 INPUT	(C2IN)
SOURCE TRIG	(STRG)
MORE TYPES	(MOTP)
SLOPE + −	(SLOP)
EXT TRIG	(EXT)
HP-IB TRIG	(HPT)
DIGTAL TRIG	(DIG)

SINGLE (SNGL)

SOURCE (SRCE)

SOURCE AN DIG	(SRAN)
SOURCE LEVEL	(SRLV)
SOURCE TYPE	(SRTP)
DC OFFSET	(DCOF)
INTERFACE	(IFC)
SOURCE OFF	(SROF)

(linear res & time capture modes)

SOURCE OFF	(SROF)
RANDOM NOISE	(RND)
BURST RANDOM	(BRND)
PRIODC CHIRP	(PCRP)
BURST CHIRP	(BCRP)
FIXED SINE	(FSIN)
MORE TYPES	(MTYP)
RETURN	(RTN)

PULSE	(PULS)
STEP	(STEP)
RAMP	(RMP)
USER SAVD 2	(USD2)
SINGLE	(SING)
REPEAT	(REPT)
RETURN	(RTN)

(Log res mode)

SOURCE OFF	(SROF)
RANDOM NOISE	(RND)
FIXED SINE	(FSIN)
RETURN	(RTN)

(Swept sine mode)

SOURCE ON OFF	(SRON)
SWEEP UP	(SWUP)
SWEEP DOWN	(SWDN)
SWEEP HOLD	(SWHD)
MANUAL SWEEP	(MNSW)
SWEEP RATE	(SWRT)
RETURN	(RTN)

TWOS COMPL	(TCOM)
OFFSET BINARY	(OBIN)
SOURCE RANGE	(SRRG)
SOURCE QUALFR	(SQUL)
RETURN	(RTN)

[1]Same as SWEEP RATE in the FREQ menu.

SPCL FCTN	(SPFN)
SELF TEST	(TST)
SERVIC TEST	(SVTS)
TIME H,M,S	(TIME)
DATE M,D,Y	(DATE)
BEEPER ON OFF	(BEEP)
SOURCE PROTCT	(SRPT)
VISUAL HELP	(VISU)
PwrSRQ ON OFF	(PSRQ)
PROTCT ON OFF	(PTON)
RAMP TIME	(RAMP)
RETURN	(RTN)
CHAN 1 CONFIG	(C1CN)
CHAN 2 CONFIG	(C2CN)
FLOW ON OFF	(FLOW)
MENU→ ON OFF	(MENU)
NMBR ? ON OFF	(NMBR)
RETURN	(RTN)
SPCL MARKER	(SPMK)
X FCTN OFF	(XFOF)
HMNC ON	(HMNC)
SBAND ON	(SBND)
SLOPE	(SLP)
MARKER CALC	(MKRC)
MRKR→ PEAK	(MKPK)
FNDMTL FREQ	(FNFR)
HMNC POWER	(HPWR)
THD	(THD)
CALC OFF	(CAOF)
RETURN	(RTN)
CRRIER FREQ	(CRFR)
SBAND INCRMT	(SBIN)
SBAND POWER	(SPWR)
CALC OFF	(CLOF)
RETURN	(RTN)
GAIN & PH MGN	(GAPH)
FREQ & DAMP	(FRDA)
POWER	(PWR)
AVG VALUE	(AVGV)
DATA EDIT	(DAED)
CUSTOM CHIRP	(CHRP)
RETURN	(RTN)
START	(STRT)
STATE/TRACE	(STTR)[1]

[1]STTR switches back and forth between state and trace. STAT explicitly displays the state and TRAC explicitly displays the trace(s). To be certain of the resulting condition, use STAT and TRAC over the bus. STDG displays the digital state.

SYNTH	**(SNTH)**
POLE ZERO	(PZRO)
POLE RESIDUE	(PRSD)
POLY NOMIAL	(POLY)
DOMAIN S Z	(DOMS)
CONVERT TO Z	((CVTZ)
CONVERT TO S	(CVTS)
CONVRT TABLE	(CVTB)
CREATE CONST	(CCON)
CREATE TRACE	(CTRA)
BI-LINEAR	(BILN)[4]
INVERS BI-LIN	(IBLN)[5]
P WARP ON OFF	(PWRP)
FREQ WARP	(WARP)[6]
STEP INVRNC	(STPI)[4]
INVERS STEP	(ISTP)[5]
INVERS IMPULS	(IIMP)[5]
IMPULS INVRNC	(IMPI)[4]
SAMPLE FREQ	(SFRQ)
MEAS CLOCK	(MSCL)
RETURN	(RTN)
TO→POL ZERO	(TOPZ)
TO→POL RESIDUE	(TOPR)
TO→POLY	(TOPY)
EDIT POLE#	(EDPL)[1,2]
EDIT ZERO#	(EDZR)[1]
EDIT RESDU#	(EDRS)[2]
EDIT NUMER#	(EDNM)[3]
EDIT DENOM#	(EDDN)[3]
DELETE VALUE	(DLTV)
CHANGE VALUE	(CHGV)
ADD VALUE	(ADDV)
SYNTH FCTN	(SNFN)
CLEAR TABLE	(CLTB)
RETURN	(RTN)
S DOMAIN	(SDOM)
Z DOMAIN	(ZDOM)
Z & S DOMAIN	(ZSDM)
0 HOLD ON OFF	(HOLD)
RETURN	(RTN)

[1]POLE ZERO menu must be displayed before these commands can be used.
[2]POLE RESIDUE menu must be displayed before these commands can be used.
[3]POLY NOMIAL menu must be displayed before these commands can be used.
[4]Must be in S-domain
[5]Must be in Z-domain
[6]P WARP ON OFF must be on

GAIN FACTOR	(GAIN)
TIME DELAY	(TDLY)
SCALE FREQ	(SCFR)[4]
SAMPLE FREQ	(SFRQ)[5]
MEAS CLOCK	(MSCL)[5]
Z PWR + -	(ZPWR)[5]
RETURN	(RTN)

[4]Must be in S-domain
[5]Must be in Z-domain

TRIG DELAY (TRGD)

CHAN1 DELAY	(C1DL)
CHAN2 DELAY	(C2DL)

UNITS (UNIT)

P SPEC UNITS	(PSUN)
L SPEC UNITS	(LSUN)
SWEPT UNITS	(SWUN)
Hz (Sec)	(HZS)
RPM (Sec)	(RPMS)
Orders (Revs)	(ORDR)
Orders CAL	(ORCL)
TRACE TITLE	(TITL)
RETURN	(RTN)
VOLTS PEAK	(VTPK)[1]
VOLTS RMS	(VTRM)[1]
VOLTS	(VLTS)[1]
VOLTS^2	(VT2)[1]
V/$\sqrt{Hz}$ ($\sqrt{PSD}$)	(VHZ)[1]
V^2/HZ (PSD)	(V2HZ)[1]
V^{2}S/Hz (ESD)	(V2SH)[1]
RETURN	(RTN)
VOLTS PEAK	(VTPK)[1]
VOLTS RMS	(VTRM)[1]
VOLTS	(VLTS)[1]
V/$\sqrt{Hz}$ ($\sqrt{PSD}$)	(VHZ)[1]
RETURN	(RTN)
VOLTS PEAK	(VTPK)[1]
VOLTS RMS	(VTRM)[1]
VOLTS	(VLTS)[1]
VOLTS^2	(VT2)[1]
RETURN	(RTN)

[1]Appropriate menu (L SPEC UNITS, P SPEC UNITS or SWEPT UNITS) must be displayed before these commands can be used. It is recommended that you always send the first level menu command before the second level command. For example, PSUN; VTPK for VOLTS PEAK in the P SPEC UNITS menu.

UPPER LOWER (UPLO)

VIEW INPUT (VWIN)

(linear res, log res & swept sine modes)

INPUT TIME 1	(ITM1)
INPUT TIME 2	(ITM2)
INPUT SPEC 1	(ISP1)
INPUT SPEC 2	(ISP2)
INPUT DIG 1	(DIG1)
INPUT DIG 2	(DIG2)
VIEW OFF	(VWOF)

(time capture mode)

INPUT TIME 1	(ITM1)
INPUT TIME 2	(ITM2)
INPUT SPEC 1	(ISP1)
INPUT SPEC 2	(ISP2)
INPUT DIG 1	(DIG1)
INPUT DIG 2	(DIG2)
CAPTUR DATA	(CADA)
VIEW OFF	(VWOF)
TIME RECORD	(TMRC)
LINEAR SPEC	(LSPC)
TIME BUFFER	(TMBF)
RETURN	(RTN)

(linear res & time throughput active)

INPUT TIME 1	(ITM1)
INPUT TIME 2	(ITM2)
INPUT SPEC 1	(ISP1)
INPUT SPEC 2	(ISP2)
INPUT DIG 1	(DIG1)
INPUT DIG 2	(DIG2)
THRUPT DATA	(THDA)
VIEW OFF	(VWOF)
THRUPT TIME 1	(THT1)
THRUPT TIME 2	(THT2)
NEXT RECORD	(NXRC)
RETURN	(RTN)

WINDOW (WNDO)

HANN	(HANN)
FLAT TOP	(FLAT)
UNIFRM (NONE)	(UNIF)
FORCE EXPON	(FOXP)
USER SAVD 1	(USD1)
LINES 801	(LIN8)
LINES 1024	(LIN1)
FORCE CHAN1	(FRC1)
EXPON CHAN1	(XPN1)
FORCE CHAN2	(FRC2)
EXPON CHAN2	(XPN2)

X	(X)
X VALUE	(XVAL)
X MRKR SCALE	(XMKR)[1]
X AUTO SCALE	(XASC)[1]
SCROLL ON OFF	(SCRL)
HOLD X CENTER	(HXCT)
HOLD X RIGHT	(HXRT)
HOLD X LEFT	(HXLF)
HOLD X OFF	(HXOF)

[1]Same as corresponding softkey in the SCALE menu.

X OFF	(XOFF)

Y	(Y)
Y VALUE	(YVAL)
Y MRKR SCALE	(YMKR)[1]
Y AUTO SCALE	(YASC)[1]
Y DFLT SCALE	(YDSC)[1]
HOLD Y CENTER	(HYCT)
HOLD Y UPPER	(HYUP)
HOLD Y LOWER	(HYLW)
HOLD Y OFF	(HYOF)

[1]Same as corresponding softkey in the SCALE menu.

Y OFF	(YOFF)

3

The Data Transfer Group

About This Chapter

This chapter shows you how to perform data block transfers between a controller and the HP 3563A. The following topics are addressed:

- Data formats offered by the HP 3563A
- Loading/dumping data traces
- Loading/dumping instrument states
- Dumping the coordinate transform block
- Loading/dumping the synthesis tables (s and z domain)
- Accessing capture and throughput files on disc

This chapter deals only with these data block transfers. For display buffer transfers, see Chapter 5. For signal processing primitive block transfers, see Chapter 4.

Three Data Formats

The HP 3563A offers three data formats for transferring data via HP-IB: ASCII, ANSI floating point binary, and a nonstandard binary used internally by the instrument. All three formats are provided to better address the needs of specific instrument/controller operations.

ASCII Format

The ASCII (American National Standard Code for Information Interchange) format is a common data communication code that uses seven bits to represent a single character (bit eight of the byte is ignored). Before the HP 3563A transfers ASCII encoded data, it must convert the internal binary numbers into a series of ASCII-encoded characters. Refer to "Internal Binary Format" for a description of how the internal binary data is converted to ASCII. When the computer receives ASCII encoded bytes it must decode the bytes and build real numbers out of them by linking the characters.

Note

ASCII transfers should not be performed when the HP 3563A is in autosequence edit mode.

ANSI Floating Point Binary Format

The ANSI binary format is the 64-bit floating point binary format specified by IEEE draft standard P754. ANSI binary is the format used by HP series 200/300 and many other computers/controllers to represent real numbers in floating point notation. Each real number is comprised of an 11-bit exponent and a 53-bit mantissa. The mantissa is in sign-and-magnitude notation. The sign bit of the mantissa is the most significant bit (MSB) of the entire eight bytes. The magnitude portion of the mantissa occupies the 13th through the 64th bit. The exponent occupies the 2nd through the 12th bit and is biased by 1023 to allow for positive and negative exponents. The following shows the bit arrangement of ANSI floating point:

```
byte1     byte2     byte3     byte4     ... byte7    byte8
SEEEEEEE EEEEMMMM MMMMMMMM MMMMMMMM... MMMMMMMM MMMMMMMM
```

where:
S is the sign bit of the mantissa
E is a bit of the biased exponent
M is a bit of the magnitude portion of the mantissa

Byte1 is transferred first with the sign bit being the MSB. The following equation will yield the value of the real number except when all bits are zero (the number is 0.0).

$$-1^{(\text{mantissa sign})} \times 2^{(\text{exponent-1023})} \times 1.\text{mantissa}$$

Before the HP 3563A transfers ANSI binary data it must convert the internal binary data to equivalent ANSI binary data. Refer to "Internal Binary Format" (later in this chapter) for a description of how the internal binary data is converted to ANSI binary. If the computer/controller uses 64-bit ANSI format to represent real numbers, eight successive bytes are transferred and stored as one number in a real array with no conversion required by the computer. This saves time if the data block is to be processed outside the HP 3563A.

Note

ANSI transfers should not be performed when the HP 3563A is in autosequence edit mode.

Internal Binary Format

The internal binary format is the nonstandard format used by the HP 3563A to represent five internal data types.

- 64 bit floating point ("Long real")
- 32 bit floating point ("Real")
- 32 bit integer ("long integer")
- 16 bit integer ("Integer")
- String

During an internal binary data transfer, the data types are sent with no change of data length or format. Since the internal binary format is a non-standard format, it is easiest to store the data in a 16-bit integer array — where two bytes transferred from the HP 3563A to the computer/controller become one element of the integer array. An internal binary transfer is the fastest way to get data in or out of the HP 3563A. However, once outside the HP 3563A, the data is not in a standard form (such as ANSI format) and that makes manipulation or interpretation difficult. The following paragraphs describe the five internal data types and how they are converted during an ANSI or ASCII transfer.

64-bit Floating Point Data Type ("Long Real")

```
byte1    byte2    byte3    byte4 ...byte7    byte8
SFFFFFFF FFFFFFFF FFFFFFFF FFFFF...FFFFFFFF SEEEEEEE
```

where:
S is the sign bit of each two's complement number
F is a bit in the mantissa
E is a bit in the exponent

The mantissa is in two's complement form, with a binary point between the sign bit and the next bit. Each bit represents a fractional portion of the mantissa. The 8-bit exponent is in two's complement integer form. The following equation will produce the value of the real number:

$$\text{mantissa} \times 2^{(\text{exponent})}$$

For ANSI transfers, each internal 64-bit value is converted to one 64-bit ANSI value. For ASCII transfers, each internal 64 bit value is converted to an ASCII encoded floating point real number.

32-bit Floating Point Data Type ("Real")

SFFFFFFF FFFFFFFF FFFFFFFF SEEEEEEE

Refer to the 64-bit description for explanations of S, E and F. For ANSI transfers, each internal 32-bit value is converted to one 64-bit value in ANSI format. For ASCII transfer each 32-bit value is converted to an ASCII-encoded real number.

32-Bit Integer Data Type ("Long Integer")

This data type consists of 32-bit integers in two's complement format. For ASCII and ANSI transfers, each 32-bit internal integer is converted to two 16-bit integers (high word, low word). For an ANSI transfer, each integer is then converted to two 64-bit ANSI format numbers. For ASCII transfers, each integer is converted to an ASCII encoded real number.

16-Bit Integer Data Type ("Integer")

This data type consists of 16-bit integers in two's complement format. For ANSI transfers, each internal 16-bit integer is converted to one 64-bit ANSI floating point value. For ASCII transfers, each internal 16-bit integer is converted to an ASCII-encoded number.

String Data Type

The string data type consists of ASCII-encoded bytes representing alphanumeric data. Each string is preceded by one byte, indicating the number of data bytes in the string. Each data byte represents one alphanumeric character. For ASCII and ANSI transfers, two successive string bytes are linked to form one 16-bit integer. For an ASCII transfer, the 16-bit integer is converted to an ASCII-encoded real number. For an ANSI transfer, the 16-bit integer is converted to 64-bit floating point ANSI format.

Note

Internal binary transfers should not be performed when the HP 3563A is in autosequence edit mode.

Elements of a Data Transfer

A typical data transfer between an HP 3563A and a computer can be divided into four parts:

- Format Specifier
- Length Specifier
- Header
- Data

The **format specifier** is the first element of data to be transferred. In both ASCII and binary transfers, the format specifier is an ASCII-encoded two character string that specifies the encoding format of the data to follow. "#I" is the format specifier for ASCII, and "#A" is the format specifier for binary (ANSI or Internal)

The **length Specifier** is the second element of data to be transferred. In an ASCII transfer the length specifier is ASCII-encoded. Its value equals the number of elements in the data transfer. Each element in the data transfer is made up of multiple ASCII-encoded characters, (one byte per character). In the case of a binary (ANSI or internal) transfer, the length specifier is a 16-bit integer and its value equals the number of 8-bit bytes to follow.

The **header** is a block of data which defines the conditions under which the data being asked for was collected or displayed. Not all data transfers include a header. For example, the instrument state and synthesis table dumps do not include a header.

The **data** is the final element of a data transfer. In most cases, the data is simply the item (trace data, coordinate transform data, instrument state, etc.) asked for in the "Dump" command. In the case of an ASCII transfer, the computer must decode the ASCII-encoded bytes into the numeric characters and build real numbers out of them by concatenating (linking) the characters until a delimiter is received. In a binary transfer, the computer simply takes the bytes it receives from the HP 3563A and places them in an array. If the array is a 16-bit integer array, the computer places two bytes into one array element. If the array is a 64-bit floating point real array, the computer places eight successive bytes into each array element.

In a binary transfer the format specifier and length specifier are not separated (delimited) in any way. The HP BASIC language system has output and enter image specifiers that can be used to separate the format specifier and length specifier from the data. The HP BASIC programming examples in this chapter use the enter image specifiers (%,2A,W), where the enter image:

"2A" instructs the system to enter two ASCII encoded characters;
"W" instructs the system to enter one 16-bit binary word;
"%" instructs the system to terminate the enter statement when the last enter item is satisfied.

The HP BASIC programming examples also use the output image specifiers (#,2A,W), where the output image:

"2A" instructs the system to output two ASCII encoded characters;
"W" instructs the system to output one 16-bit binary encoded number;
"#" instructs the system to suppress the End-of-Line sequence (carriage return and line feed) that would normally follow the last item in the statement to be output.

Loading/Dumping Data Traces

The active trace on the display can be dumped in any of the three data formats. When a trace is dumped, the data points are preceded by the format specifier, the length specifier, and a header. The contents of the data trace header are discussed in the next section.

Trace data is dumped either in real/imaginary pairs or real numbers. It represents the measured data, not what is necessarily on the screen. For example, if you attempt to dump a phase trace, you will get the entire frequency response function from which the phase trace was derived. If you want strictly what is on the display, refer to "Dumping the Coordinate Transform Block" later in this chapter.

Note: Time capture data should be dumped and loaded a record at a time using the CAPTUR POINTR (CPNT) and TIME RECORD (TMRC) HP-IB commands.

The Data Header

The header dumped with data traces is the same for all three data formats. The only difference is the variable count. Table 3-1 shows the contents of the data header. For data types listed in the table as "E-type" (enumerated type), the value of that variable can be decoded by referring to Table 3-2. The range of values for each E-type is shown in parentheses. The (+ 1) beside the byte count for strings is a reminder that each string is preceded by a length byte (binary encoded) that specifies the length of the data to follow.

Table 3-1 also shows the indexed position of each item for the three formats. The Internal binary index indicates the position of the header element in a 16-bit integer array. The ASCII/ANSI index indicates the position of the header element in a 64-bit real array.

Table 3-1. Contents of the Data Header

Item	Data Type	Size (bytes)	Internal Binary Index	ASCII/ANSI Index
Display function	Integer E-type (0-49)	2	1	1
Number of elements	Integer	2	2	2
Displayed elements	Integer	2	3	3
Number of averages	Integer	2	4	4
Channel selection	Integer E-type (0-3)	2	5	5
Overflow status	Integer E-type (0-3)	2	6	6
Overlap percentage	Integer	2	7	7
Domain	Integer E-type (0-2)	2	8	8
Volts peak/rms	Integer E-type (0-2)	2	9	9
Amplitude Units	Integer E-type (0-7)	2	10	10
X Axis Units	Integer E-type (0-35)	2	11	11
Auto Math Label	String	13(+1)	12-18	12-18
Trace Label	String	21(+1)	19-29	19-29
EU Label on active trace	String	5(+1)	30-32	30-32
EU Label on other trace	String	5(+1)	33-35	33-35
Float/integer	Integer (1/0)	2	36	36
Complex/Real	Integer (1/0)	2	37	37
Live/Recalled	Integer (1/0)	2	38	38
Math result	Integer (1/0)	2	39	39
Real/Complex input	Integer (1/0)	2	40	40
Log/Linear data	Integer (1/0)	2	41	41
Auto math	Integer (1/0)	2	42	42
Real time status	Integer (1/0)	2	43	43
Measurement Mode	Integer E-type (0-4)	2	44	44
Window	Integer E-type(0-8)	2	45	45
Demod type chan 1	Integer E-type (45-47)	2	46	46
Demod type chan 2	Integer E-type (45-47)	2	47	47
Demod active chan 1	Integer	2	48	48
Demod active chan 2	Integer	2	49	49
Average status	E-type (0-2)	2	50	50
Not used	Integers (2)	4	51-52	51-52
Samp freq/2(real)	Real	4	53-54	53
Samp freq/2(imag)	Real	4	55-56	54
Not used	Real	4	57-58	55
Delta X-axis[2]	Real	4	59-60	56
Max range (for scaling)	Real	4	61-62	57
Start time value	Real	4	63-64	58
Expon wind const 1	Real	4	65-66	59
Expon wind const 2	Real	4	67-68	60
EU value chan 1	Real	4	69-70	61
EU value chan 2	Real	4	71-72	62
Trig delay chan 1	Real	4	73-74	63
Trig delay chan 2	Real	4	75-76	64
Start freq value	Long Real	8	77-80	65
Start data value	Long Real	8	81-84	66

[2] $10^{\Delta X}$ for log x axis

(+1) indicated this string preceded by length specifier

Table 3-2. Enumerated Types for Data Header Variables

Display Function
0 No data
1 Frequency response
2 Power spectrum 1
3 Power spectrum 2
4 Coherence

5 Cross spectrum
6 Input time 1
7 Input time 2
8 Input linear spectrum 1
9 Input linear spectrum 2

10 impulse response

11 Cross correlation
12 Auto correlation 1
13 Auto correlation 2
14 Histogram 1

15 Histogram 2
16 Cumulative density function 1
17 Cumulative density function 2
18 Probability density function 1
19 Probability density function 2

20 Average linear spectrum 1
21 Average Linear spectrum 2
22 Average time record 1
23 Average time record 2
24 Synthesis pole-zero

25 Synthesis pole-residue
26 Synthesis polynomial
27 Synthesis constant
28 Windowed time record 1
29 Windowed time record 2

30 Windowed linear spectrum 1
31 Windowed linear spectrum 2
32 Filtered time record 1
33 Filtered time record 2
34 Filtered linear spectrum 1

35 Filtered linear spectrum 2
36 Time capture buffer
37 Captured linear spectrum
38 Captured time record
39 Throughput time record 1

40 Throughput time record 2
41 Curve fit
42 Weighting function
43 Not used
44 Orbits
45 Demodulation polar
46 Preview demod record 1
47 Preview demod record 2
48 Preview demod linear spectrum 1
49 Preview demod linear spectrum 2

Channel Selection (Analog or digital)
0 Channel 1
1 Channel 2
2 Channels 1&2
3 No channel

Average Status
0 No data (any data received are invalid)
1 Not averaged
2 Averaged

Overflow Status
0 Channel 1
1 Channel 2
2 Channels 1&2
3 No channel

Domain Type
0 Time
1 Frequency
2 Voltage (amplitude)

Volts Peak/RMS
0 Peak
1 RMS
2 Volts (indicates peak only)

Amplitude Units
0 Volts
1 Volts squared
2 PSD (V^2/Hz)

3 ESD (V^2s/Hz)
4 $\sqrt{PSD}$ ($V/\sqrt{Hz}$)
5 No amplitude units
6 Unit volts
7 Unit volts^2

X Axis Units
0 No units
1 Hertz
2 RPM
3 Orders

4 Seconds
5 Revs
6 Degrees
7 dB
8 dBV
9 Volts
10 $V\sqrt{HZ}$($\sqrt{PSD}$)
11 Hertz/second
12 Volts/EU
13 Vrms
14 V^2/HZ(PSD)
15 Percent
16 Points

17 Records
18 Ohms
19 Hertz/Octave
20 Pulses/Rev
21 Decades

22 Minutes

23 V^2s/Hz (ESD)
24 Octave
25 Seconds/Decade
26 Seconds/Octave

27 Hz/Point
28 Points/Sweep
29 Points/Decade
30 Points/Octave
31 V/Vrms

32 V^2
33 EU referenced to Chan 1
34 EU referenced to Chan 2
35 EU value

Measurement Mode
0 Linear resolution
1 Log resolution
2 Swept sine
3 Time capture
4 Linear resolution throughput

Demod Type Chan 1
45 AM
46 FM
47 PM

Demod Type Chan 2
45 AM
46 FM
47 PM

Window
0 Window not applicable
1 Hann
2 Flattop
3 Uniform
4 Exponential
5 Force
6 Force chan 1/expon chan 2
7 Expon chan 1/force chan 2
8 User

ASCII Format

To dump the active trace in ASCII, use the DDAS (Dump Data in AScii) command; to load data in ASCII, use LDAS (Load Data in AScii). The format specifier is #I, and the ASCII-encoded number that follows indicates the number of elements to be transferred. The following BASIC statements dump, then load a *frequency response* trace:

```
50    ASSIGN @Dsa TO 720
60    DIM Data_buffer(1:1668)          !   Create REAL array for data
70    OUTPUT @Dsa;"DDAS"               !   Dump data in ASCII format
80    ENTER @Dsa;Specifier$            !   Read format/length specifier
90    L=VAL (Specifier$[3])
110   PRINT "FORMAT SPECIFIER= ";Specifier$[1,2]
120   PRINT "LENGTH= ";L
130   ENTER @Dsa;Data_buffer(*)        !   Read header and trace data
140   !
150   OUTPUT @Dsa;"LDAS"               !   Load data in ASCII format
160   OUTPUT @Dsa;Specifier$           !   Output format and length
170   OUTPUT @Dsa;Data_buffer(*)       !   Output header and trace data
```

The following is printed on the display after running the program.

```
FORMAT SPECIFIER= #I
LENGTH = 1668
```

For the specific case of a frequency response trace, the length specifier (L) indicates 1668 variables (1602 data points (801 real/imaginary pairs) plus the 66 element header). Notice that the length specifier represents the number of elements to be transferred and not the number of bytes, as in a binary transfer. The data block size changes when a different type of data is to be transferred. Therefore, to make this a more general program, you should dimension Data_buffer to the largest size required, then redimension Data_buffer after reading the length specifier.

```
60    DIM Data_buffer(1:2114)          !   array size for a time record
125   REDIM Data_buffer(1:L)
```

ANSI Binary Format

To dump the active trace in ANSI binary, use the DDAN (Dump Data in ANsi) command. To load data in ANSI, use LDAN (Load Data in ANsi). The format specifier is #A, and the two bytes (one word) that follow indicate the number of bytes to be transferred. The following BASIC statements dump, then load a *frequency response* trace:

```
40    ASSIGN @Dsa TO 720
50    DIM Data_buffer(1:1668)           !   Create REAL array for data
60    OUTPUT @Dsa;"DDAN"                !   Dump Data in ANSI binary
70    ENTER @Dsa USING "%,2A,W";F$,L    !   %,2A- terminate enter item(F$) when
80                                      !     2 characters have been read.
90                                      !   %,W- terminate enter item(L) when
100                                     !     1 Word (2 bytes) has been read
110   PRINT "FORMAT SPECIFIER= ";F$
120   PRINT "LENGTH= ";L
130   ASSIGN @Dsa;FORMAT OFF            !   Allow binary data
140   ENTER @Dsa;Data_buffer(*)         !   Read data array
150   ASSIGN @Dsa;FORMAT ON             !   Allow ASCII data
160                                     !
170   OUTPUT @Dsa;"LDAN"                !   Load Data in ANSI binary
180   OUTPUT @Dsa USING "#,2A,W";F$,L   !   #- suppress the End of line sequence
190   ASSIGN @Dsa;FORMAT OFF            !   Allow binary data
200   OUTPUT @Dsa;Data_buffer(*)        !   Output data array
210   ASSIGN @Dsa;FORMAT ON             !   Allow ASCII data
```

The following is printed on the display after running the program.

```
FORMAT SPECIFIER= #A
LENGTH = 13344
```

Notice that length (L) is eight times larger then the dimensioned Data_buffer array (8× 1668=13344). This is due to each array element being a 64-bit floating point number (real) where every eight bytes transferred represent one element of the array. Since the HP 3563A is transferring data in the 64-bit ANSI format, and the HP Series 200/300 computers use this same format to represent real numbers, one element of the Data_buffer array represents one element of data from the HP 3563A. The number of bytes specified by the length specifier changes when a different type of data is selected. To make this a more general program, you should dimension Data_buffer to the largest size needed, then redimension Data_buffer after reading the length specifier (L).

```
50    DIM Data_buffer(1:2114)           !   array size for a time record
125   REDIM Data_buffer(1:L/8)
```

Refer to "Elements of a Data Transfer" for a description of ENTER and OUTPUT image specifiers used to transfer the format/length specifier.

Internal Binary Format

To dump the active trace in internal binary, use the DDBN (Dump Data in internal BiNary) command; to load data in internal binary use LDBN (Load Data in internal BiNary). The format specifier is #A, and the two bytes (one word) following that indicate the number of bytes to be transferred. The following BASIC statements dump then load a *frequency response* trace:

```
40    ASSIGN @Dsa TO 720
50    INTEGER Data_buffer(1:3288)      !   Set up 16 bit integer array for
60                                     !   freq. resp. (+header)  data
70    OUTPUT @Dsa;"DDBN"               !   Dump data in internal binary
80    ENTER @Dsa USING "%,2A,W";F$,L   !   %,2A- terminate enter item(F$) when
90                                     !     two characters have been read
100                                    !   %,W- terminate enter item(L) when 1
110   PRINT "FORMAT SPECIFIER= ";F$    !     word (two bytes) has been read
120   PRINT "LENGTH= ";L !
130   ASSIGN @Dsa;FORMAT OFF           !   Allow binary data
140   ENTER @Dsa;Data_buffer(*)        !   Read data
150   ASSIGN @Dsa;FORMAT ON            !   Allow ASCII data
160                                    !
170   OUTPUT @Dsa;"LDBN"               !   Load data in internal binary
180   OUTPUT @Dsa USING "#,2A,W";F$,L! #  - suppress the End-Of-Line sequence
190   ASSIGN @Dsa;FORMAT OFF
200   OUTPUT @Dsa;Data_buffer(*)       !   Output data array
210   ASSIGN @Dsa;FORMAT ON
```

The following is printed on the display after running the program:

```
FORMAT SPECIFIER= #A
LENGTH = 6576
```

Notice that LENGTH is two times larger then the dimensioned Data_buffer array (2× 3288=6576). This is due to each array element being a 16-bit integer where every two bytes transferred fills one element of the array. For the case of a frequency response trace, the length specifier (L) indicates 6576 bytes. Therefore, the Data_buffer is dimensioned to 3288 (6576/2) 16-bit elements. The number of bytes (L) is different for other types of displayed trace data. To make this a more general program, you should dimension Data_buffer to the largest size needed then, redimension Data_buffer after reading the length specifier (L).

```
50    INTEGER Data_buffer(1:4180)      !   array size for time rec
125   REDIM Data_buffer(1:L/2)
```

Loading/dumping the Instrument Setup State

The instrument setup state can be dumped and loaded in any of the three data formats. Examples of dumping the instrument state in each format follow the description of the instrument state's contents.

Contents of the Instrument State

Table 3-3 shows the contents of the instrument state. For those data types listed as "E-type" (enumerated type), refer to Table 3-4 to decode the value. The range of values for each E-type is shown in parentheses. The (+ 1) beside the byte count for strings is a reminder that each string is preceded by a binary encoded length byte that specifies the number of characters in the string.

Table 3-3 also shows the indexed position of each item for the three formats. The internal binary index indicates the position of the instrument state item in a 16-bit integer array. The ASCII/ANSI index indicates the position of the instrument state item in a 64-bit real array.

Table 3-3. Contents of the Instrument State

Item	Data Type	Size (bytes)	Internal Binary Index	ASCII/ANSI Index
Measurement mode	Integer E-type (0-3)	2	1	1
Measurement type 1	Integer E-type (0-5)	2	2	2
Measurement type 2	Integer E-type (0-5)	2	3	3
Window type	Integer E-type(11-15)	2	4	4
Force/Expon window 1	Integer E-type (0-1)	2	5	5
Force/Expon window 2	Integer E-type (0-1)	2	6	6
Average type	Integer E-type (6-10)	2	7	7
Overlap percentage	Integer	2	8	8
Number of averages	Integer	2	9	9
Sweep # of averages	Integer	2	10	10
Trigger type	Integer E-type (18-23)	2	11	11
Trigger slope	Integer E-type (16-17)	2	12	12
Preview type	Integer E-type (0-2)	2	13	13
Sample type	Integer E-type (24-25)	2	14	14
Range units chan 1	Integer E-type (8-35)	2	15	15
Range units chan 2	Integer E-type (8-35)	2	16	16
Range type 1	Integer E-type(26-28)	2	17	17
Range type 2	Integer E-type(26-28)	2	18	18
Input coupling 1	Integer E-type(29-30)	2	19	19
Input coupling 2	Integer E-type (29-30)	2	20	20
Analog Source type	Integer E-Type (31-37)	2	21	21
Chirp percent	Integer	2	22	22
Burst percent	Integer	2	23	23
Sweep direction	Integer E-type (41-44)	2	24	24
Sweep mode	Integer E-Type (39-40)	2	25	25
Ext sample freq units	Integer E-Type (1-20)	2	26	26
Bandwidth units	Integer E-Type (1-3)	2	27	27
Log span index	Integer	2	28	28
Log start index	Integer	2	29	29
Sweep rate units	Integer E-Type (11-26)	2	30	30
Auto gain ref chan	Integer E-Type (0-3)	2	31	31
Demod channels	Integer E-type (0-3)	2	32	32
Demod type chan 1	Integer E-type (45-47)	2	33	33
Demod type chan 2	Integer E-type (45-47)	2	34	34
Source level units	Integer E-type (8-13)	2	35	35
Source dc offset units	Integer E-type (9)	2	36	36
Trigger level units	Integer E-type (9-34)	2	37	37
Capt/thru length units	Integer E-type (4-17)	2	38	38
EU label 1	String	5(+1)	39-41	39-41
EU Label 2	String	5(+1)	42-44	42-44

Table 3-3. Contents of the Instrument State cont.

Item	Data Type	Size (bytes)	Internal Binary Index	ASCII/ANSI Index
Auto carrier on/off	Integer (1/0)	2	45	45
Time average on/off	Integer (1/0)	2	46	46
Auto/fixed resolution	Integer (1/0)	2	47	47
Auto gain on/off	Integer (1/0)	2	48	48
Auto/fixed integrate	Integer (1/0)	2	49	49
Fast average on/off	Integer (1/0)	2	50	50
Overload reject on/off	Integer (1/0)	2	51	51
Chan 1 float/ground	Integer (1/0)	2	52	52
Chan 2 float/ground	Integer (1/0)	2	53	53
Time throughput on/off	Integer (1/0)	2	54	54
Demodulation on/off	Integer (1/0)	2	55	55
EU volts chan 1	Integer (1/0)	2	56	56
EU volts chan 2	Integer (1/0)	2	57	57
Manual/auto arm	Integer (1/0)	2	58	58
Demod preview on/off	Integer (1/0)	2	59	59
Delete freq on/off	Integer (1/0)	2	60	60
Lin res Fstart pegged	Integer (1/0)	2	61	61
Swept Fstart pegged	Integer (1/0)	2	62	62
Force length chan 1	Real	4	63-64	63
Force length chan 2	Real	4	65-66	64
Expon time constant 1	Real	4	67-68	65
Expon time constant 2	Real	4	69-70	66
Sweep time	Real	4	71-72	67
Sweep rate	Real	4	73-74	68
Sweep resolution	Real	4	75-76	69
Sweep integrate time	Real	4	77-78	70
Auto gain level	Real	4	79-80	71
Auto gain Limit	Real	4	81-82	72
Source level	Real	4	83-84	73
EU value chan 1	Real	4	85-86	74
EU value chan 2	Real	4	87-88	75
Trigger delay chan 1	Real	4	89-90	76
Trigger delay chan 2	Real	4	91-92	77
Integrate var thresh	Real	4	93-94	78
Capt/thru length	Real	4	95-96	79
Frequency span	Real	4	97-98	80
Time record length	Real	4	99-100	81
Frequency resolution	Real	4	101-102	82
Time resolution	Real	4	103-104	83
External sample rate	Real	4	105-106	84

Table 3-3. Contents of the Instrument State cont.

Item	Data Type	Size (bytes)	Internal Binary Index	ASCII/ANSI Index
Sample rate (actual)	Real	4	107-108	85
Range channel 1	Real	4	109-110	86
Range channel 2	Real	4	111-112	87
Preview time	Real	4	113-114	88
Trigger level	Real	4	115-116	89
Source dc offset	Real	4	117-118	90
Fixed sine frequency	Long Real	8	119-122	91
Start frequency	Long Real	8	123-126	92
Center frequency	Long Real	8	127-130	93
Sweep start	Long Real	8	131-134	94
Sweep end	Long Real	8	135-138	95
Carrier frequency	Long Real	8	139-142	96
Input 1 Connection	Integer E-type (0-1)	2	143	97
Input 2 Connection	Integer E-type (1-2)	2	144	98
Input 1 # Format	Integer E-type (0-1)	2	145	99
Input 2 # Format	Integer E-type (0-1)	2	146	100
Clock 1 Qualifier	Integer	2	147	101
Clock 1 don't care[1]	Integer	2	148	102
Clock 2 Qualifier	Integer	2	149	103
Clock 2 don't care[1]	Integer	2	150	104
Sample clock	Integer E-type (0-4)	2	151	105
Source # Format	Integer E-type (0-1)	2	152	106
Mixed Ratio	Integer	2	153	107
Arb Source Type	Integer E-type (0-4)	2	154	108
Source Range Units	Integer E-type (9-13)	2	155	109
Digital Overload	Integer E-type (0-3)	2	156	110
Source don't care[1]	Integer	2	157	111
Source Qualifier	Integer	2	158	112
Not Used	Integer	2	159	113
Digital Trigger	Integer (1/0)	2	160	114
Source Digital	Integer (1/0)	2	161	115
Input 1 Data 8-bits	Integer (1/0)	2	162	116
Input 2 Data 8-bits	Integer (1/0)	2	163	117
Input 1 Bus 8-bits	Integer (1/0)	2	164	118
Input 2 Bus 8-bits	Integer (1/0)	2	165	119
Input 1 Digital	Integer (1/0)	2	166	120
Input 2 Digital	Integer (1/0)	2	167	121
Clock 1 pos Edge	Integer (1/0)	2	168	122
Clock 2 pos Edge	Integer (1/0)	2	169	123
Sample Clock pos Edge	Integer (1/0)	2	170	124
Source Repeat	Integer (1/0)	2	171	125

Table 3-3. Contents of the Instrument State cont.

Item	Data Type	Size (bytes)	Internal Binary Index	ASCII/ANSI Index
Input 1 rnd low bits	Integer (1/0)	2	172	126
Input 2 rnd low bits	Integer (1/0)	2	173	127
Input 1 auto 16-bits	Integer (1/0)	2	174	128
Input 1 last Q 0 is 1	Integer (1/0)	2	175	129
Input 2 last Q 0 is 1	Integer (1/0)	2	176	130
Input 2 auto 16-bits	Integer (1/0)	2	177	131
Input 1 Range is Source	Integer (1/0)	2	178	132
Input 2 Range is Source	Integer (1/0)	2	179	133
Not Used	Integer	2	180	134
1024 Spectrum Lines	Integer (1/0)	2	181	135
Input 1 Comput Delay	Real	4	182-183	136
Input 2 Comput Delay	Real	4	184-185	137
Source Range	Real	4	186-187	138
Source Change	Real	4	188-189	139

[1] A "0" in any bit position represents an "X" in the qualifier

Table 3-4. E-type for digital state Types for Instrument State Values

Measurement Mode
0 Linear resolution
1 Log resolution
2 Swept sine
3 Time capture

Measurement Type
0 Frequency response
1 Cross correlation
2 Power spectrum
3 Auto correlation
4 Histogram
5 No measurement

Window Type
11 Hanning
12 Flat top
13 Uniform
14 User window
15 Force/Exponential

Force/Exponential Window
0 Force
1 Exponential

Average Type
6 Stable
7 Exponential
8 Peak
9 Continuous Peak
10 Averaging Off

Trigger Type
18 Free Run
19 Channel 1
20 Channel 2
21 External
22 Source Trigger
23 HP-IB Trigger

Trigger Slope
16 Positive
17 Negative

Preview Type
0 Manual Preview
1 Timed Preview
2 Preview Off

Sample Type
24 Internal Sample
25 External Sample

Range Units
8 dBV
9 Volts
13 Vrms
35 EU

Auto Range Type
26 Auto Range On
27 Auto Range Off
28 Auto Range Set

Input Coupling
29 AC
30 DC

Analog Source type (see Arb Source)
31 Source off
32 Random noise
33 Burst random
34 Periodic chirp
35 Burst chirp
36 Swept sine
37 Fixed sine

Sweep Direction
41 Up
42 Sweep hold
43 Manual sweep
44 Down

Sweep Mode
39 Linear sweep
40 Log sweep

External Sample Frequency Units
1 Hertz
2 RPM
20 Pulses/rev

Bandwidth Units
1 Hertz
2 RPM
3 Orders

Sweep Rate Units
11 Hertz/second
25 Seconds/decade
26 Seconds/octave

Auto Gain Reference Channel
0 Channel 1
1 Channel 2
2 Not used
3 No channels

Demod Channels
0 Channel 1
1 Channel 2
2 Both channels
3 No channel

Demod Type Channel 1/2
45 AM
46 FM
47 PM

Source Level Units
8 dBV
9 Volts
13 Vrms

Source DC Offset Units
9 Volts

Trigger level units
9 Volts
33 EU channel 1
34 EU channel 2

Capt/Thru Length Units
4 Seconds
5 Revs
16 Points
17 Records

Input connection (1 or 2)
0 Source pod
1 Input 1 pod
2 Input 2 pod

Sample Clock
0 Source pod
1 Input 1 pod
2 Input 2 pod
3 Qualifier pod
4 External sample

Input # Format (1 or 2)
0 Two's complement
1 Offset binary

Source # Format
0 Two's complement
1 Offset binary

Arb Source Type
0 Arb off
1 Pulse
2 Step
3 Ramp
4 Arbitrary

Source Range Unit
8 dBV
9 Volts
13 Vrms

Digital Overload
0 Channel 1
1 Channel 2
2 Channels 1 & 2
3 No channel

ASCII Format

The Dump setup state in ASCII command (DGAS) and the Load setup state in ASCII command (LGAS) are used to transfer setup state data between the analyzer and computer.

All data transfers begin with the transfer of the format and length specifier. The format specifier for an ASCII data transfer is #I. The length specifier for the setup state transfer always indicates 139 elements. The following HP BASIC program dumps then loads the setup state.

```
24     Ana=720                        ! Analyzer address=20, interface=7
25     REAL Setup_buf(1:139)          ! Create array for setup state
26     OUTPUT Ana;"DGAS"              ! Dump analog/digital setup in ASCII
27     ENTER Ana;Specifier$           ! read format and length specifier
28     PRINT "FORMAT SPECIFIER= ";Specifier$[1,2]
29     PRINT "Length=";Specifier$[3]
30     ENTER Ana;Setup_buf(*)         ! Read state array
31     !
32     OUTPUT Ana;"LGAS"              ! Load analog/digital setup in ASCII
33     OUTPUT Ana;"#I 139"            ! Output format and length specifier
34     OUTPUT Ana;Setup_buf(*)        ! Output setup array
```

To transfer just the analog input setup data (elements 1 through 96) use the commands DSAS (dump analog setup in ASCII) and the command LSAS (load analog setup in ASCII).

ANSI Binary Format

The Dump setup state in ANSI command (DGAN) and the Load setup state in ANSI command (LGAN) are used to transfer analog/digital setup state data between the analyzer and computer in ANSI binary format. There are 139 setup state array elements. To transfer these elements in 64-bit (8-byte) ANSI binary format, eight bytes for each element must be transferred. The length specifier (which always equals the number of bytes in a binary transfer) equals 1112 (139× 8).

Before binary data can be transferred, the ASCII formatter must be turned off. This prevents the computer from trying to convert the binary data to the ASCII encoded equivalents. Each computer/language system might handle this differently. Refer to your system's manuals.

The following HP BASIC program dumps then loads the analog/digital setup state. The format and length specifier in this example are entered using the HP BASIC image specifiers %,2A,W and output using the HP BASIC image specifiers #,2A,W. Refer to "Elements of a data transfer" earlier in the chapter for a description of the HP BASIC image specifiers.

```
31    ASSIGN @Ana TO 720             ! Analyzer address=20, interface=7
32    REAL Setup_buf(1:139)          ! Create array for analog/digital data
33    OUTPUT @Ana;"DGAN"             ! Dump analog/digital setup in ANSI
34    ENTER @Ana USING "%,2A,W";F$,L ! read format and length specifier
35    PRINT "Format specifier= ";F$
36    PRINT "Length =";L;" bytes"
37    ASSIGN @Ana;FORMAT OFF         ! turn ASCII formatter off
38    ENTER @Ana;Setup_buf(*)        ! Read state data
39    ASSIGN @Ana;FORMAT ON          ! turn ASCII formatter on
40    !
41    OUTPUT @Ana;"LGAN"             ! Load analog/digital setup in ANSI
42    OUTPUT @Ana USING "#,2A,W";F$,L! Output format and length specifier
43    ASSIGN @Ana;FORMAT OFF         ! turn ASCII formatter off
44    OUTPUT @Ana;Setup_buf(*)       ! Output setup array
45    ASSIGN @Ana;FORMAT ON          ! turn ASCII formatter on
```

To transfer just the analog input setup data (elements 1 through 96), use the command DSAN (dump setup in ANSI) and the command LSAN (load setup in ANSI).

Internal Binary Format

The Dump setup state in Internal binary command (DGBN) and the Load setup state in Internal binary command (LGBN) are used to transfer analog/digital setup state data between the analyzer and computer. There are 139 setup state variables. Due to the different lengths of the variables (16 to 64-bit) when transferring data in the internal binary format, it is easiest to store the elements in an 16-bit integer array. It takes 189 16-bit integers to store the setup state variables. The format and length specifiers are entered using the HP BASIC image specifiers %,2A,W and output using the image specifiers #,2A,W. Refer to "Elements of a Data Transfer" for a description of the image specifiers.

Before binary data can be transferred, the ASCII formatter must be turned off to prevent the computer from trying to convert the binary data to ASCII encoded equivalents. Your computer/language system may handle this differently. Refer to you system's operating manual. The following HP BASIC program dumps then loads the analog/digital setup state data using the internal binary data format:

```
32    ASSIGN @Ana TO 720              ! Analyzer address=20, interface=7
33    INTEGER Setup_buf(1:189)        ! Create array for analog/digital data
34    OUTPUT @Ana;"DGBN"              ! Dump analog/digital setup
35    ENTER @Ana USING "%,2A,W";F$,L  ! read format and length specifier
36    PRINT "Format specifier = ";F$
37    PRINT "Length =";L;" bytes"
38    ASSIGN @Ana;FORMAT OFF          ! turn ASCII formatter off
39    ENTER @Ana;Setup_buf(*)         ! Read state data
40    ASSIGN @Ana;FORMAT ON           ! turn ASCII formatter on
41    !
42    OUTPUT @Ana;"LGBN"              ! Load analog/digital setup
43    OUTPUT @Ana USING "#,2A,W";F$,L ! Output format and length specifier
44    ASSIGN @Ana;FORMAT OFF          ! turn ASCII formatter off
45    OUTPUT @Ana;Setup_buf(*)        ! Output setup array
46    ASSIGN @Ana;FORMAT ON           ! turn ASCII formatter on
```

To transfer just the analog input setup data (elements 1 through 96), use the command DSBN to dump the data into a 142 element integer array and use the command LSBN to load the data.

Dumping the Coordinate Transform Block

Coordinate transform data is trace data that has been translated to the currently displayed coordinates and units. If you have a phase trace active, the coordinate data block contains phase data. Also, in some cases, the coordinate data block is modified to fit the display format of 801 or 1602 points. Keep in mind that the coordinate data is not as accurate as the trace data.

The coordinate transform block contains three groups of data: the display parameters (coordinate transform header), the data header for the active trace, and the displayed data in the active trace. As with other block transfers, the coordinate transfer block can be dumped in any of the three data formats. However, the coordinate transform block can only be dumped — there is no corresponding load command.

Table 3-5 shows the organization of the data received after dumping the coordinate transform block:

Table 3-5. The Coordinate Transform Block

Part 1:	Coordinate transform header
Part 2.	Data header for active trace
Part 3:	Displayed trace data

Contents of the Coordinate Transform Block Header

Table 3-6 shows the contents of the coordinate transform header. For data types listed in the table as "E-type" (enumerated type) the value of that variable can be decoded by referring to Table 3-7. The range of values for each E-type is shown in parentheses. The (+ 1) beside the byte count for strings is a reminder that each string is preceded by an 8-bit binary-encoded length specifier. Refer to Table 3-1 for interpretation of the data header.

Table 3-6 also shows the indexed position of each item for the three formats. The internal binary index indicates the position of the item in a 16 bit integer array. The ASCII/ANSI index indicates the position of the item in a REAL number array.

Calibrating the Coordinate Transform Data

To calibrate coordinate transform block data, you must multiply the data by the Y scale factor (an element in the coordinate transform header).

Calibrated point = (Y scale factor) × (data point)

Table 3-6. The Coordinate Transform Block

item	Data Type	Size (bytes)	Internal Binary index	ASCII/ANSI index
Y coordinates	Integer E-type (1-10)	2	1	1
# of disp elements	Integer	2	2	2
First element	Integer	2	3	3
Total elements	Integer	2	4	4
Display sampling	Integer E-type (0-2)	2	5	5
Scaling	Integer E-type(0-3)	2	6	6
Data Pointer	Long Integer	4	7-8	7
In Data	Long Integer	4	9-10	9
Log/Linear x-axis	Integers (1/0)	2	11	11
Sampled display data	Integers (1/0)	2	12	12
Plot/Graph mode	Integers (1/0)	2	13	13
Phase wrap	Integers (1/0)	2	14	14
Not used	Integers (18)	36	15-32	15-32
X scale factor	Real	4	33-34	33
Grid min Y scale	Real	4	35-36	34
Grid max Y scale	Real	4		35
/ Div	Real	4		36
Min value of data	Real	4		37
Max value of data	Real	4		38
Y cumulative Min	Real	4		39
Y cumulative Max	Real	4		40
Y scale factor[1]	Real	4		41
Not used	Reals (4)	16	57-58	42-45
Stop value	Long Real	8	59-62	46
Left grid	Long Real	8	63-66	47
Right grid	Long Real	8	67-70	48
Left data	Long Real	8	71-74	49
Right data	Long Real	8	75-78	50

[1] Multiply by data to calibrate trace data

Table 3-7. Enumerated (E-type) Values for Coordinate Transform Block

Y Coordinate		**Display Sampling**	
1	Real	0	not sampled (# of displayed elements = total elements)
2	Imaginary	1	half sampled (# of displayed elements = total elements/2)
3	Linear magnitude	2	sampled (# of displayed elements < total elements)
4	Log magnitude		
5	dB	**Scaling**	
		0	X and Y auto scale
6	Nyquist	1	X fixed scale, Y auto scale
7	Not used	2	X auto scale, Y fixed scale
8	Phase	3	X and Y fixed scale
9	Nichols		
10	dBm		

ASCII Format

To dump the coordinate transform block in ASCII, use the DCAS (Dump Coordinate block in AScii) command. The format specifier is #I, and the ASCII-encoded number that follows indicates the number of elements to be transferred. For example, the following BASIC statements dump the coordinate transform block when a *frequency response* is displayed:

```
60     DIM Coord_buf(1:917)           !   Create REAL array for data
70     OUTPUT @Dsa;"DCAS"             !   Dump coordinate block in ASCII
80     ENTER @Dsa;Specifier$          !   enter format and length specifier
100    L=VAL(Specifier$[3])
110    PRINT "FORMAT SPECIFIER= ";Specifier$[1,2]
120    PRINT "LENGTH= ";L
130    ENTER @Dsa;Coord_buf(*)        !   Read header and trace data
```

For the specific case of a frequency response trace with full X scale, the length specifier read into L indicates 917 elements (50 in coordinate transform header, 66 in the data header, and 801 from the display). To make this a general program, you should redimension the array Coord_buf to L:

```
125 REDIM Coord_buf (1:L)
```

ANSI Format

To dump the coordinate transform block in ANSI, use the DCAN (Dump Coordinate block in ANSI) command. The format specifier is #A, and the two bytes (one word) that follow indicate the number of bytes to be transferred. The following BASIC statements dump the coordinate transform block when a *frequency response* is displayed:

```
50      DIM Coord_buf(1:917)                !   Create REAL array for data
60      OUTPUT @Dsa;"DCAN"                  !   Dump Data in ANSI binary
70      ENTER @Dsa USING "%,2A,W";F$,L      !   %,2A- terminate enter item(F$) when
80                                          !     2 characters have been read
90                                          !   %,W- terminate enter item(L) when
100                                         !     1  Word (2 bytes) has been read
110     PRINT "FORMAT SPECIFIER= ";F$
120     PRINT "LENGTH= ";L
130     ASSIGN @Dsa;FORMAT OFF              !   Allow binary data
140     ENTER @Dsa;Coord_buf(*)             !   Read data array
150     ASSIGN @Dsa;FORMAT ON               !   Allow ASCII data
```

For the specific case of a frequency response trace with full X scale, the length specifier indicates 7336 bytes. There are 50 elements in the coordinate transform header, 66 in the data header, and 801 from the display for a total of 917 elements. Because Coord_buff is a 64-bit real array (where every eight bytes forms one element), the length specifier value is eight times that of the Coord_buf size ($8 \times 917 = 7336$). To make this a general program, you should redimension the array Coord__buf to $L \div 8$:

```
125 REDIM Coord_buf (1:L/8)
```

Refer to "Elements of a Data Transfer" for a description of ENTER and OUTPUT image specifiers used to transfer the format/length specifier.

Internal Binary Format

To dump the coordinate transform block in internal binary, use the DCBN (Dump Coordinate block in BiNary) command. The format specifier is #A, and the two bytes (one word) that follow indicate the number of bytes to be transferred. The following BASIC statements dump the coordinate transform block when a *frequency response* is displayed:

```
50    INTEGER Coord_buf(1:1764)        !   16 bit integer array
60    OUTPUT @Dsa;"DCBN"               !   Dump cord transform internal binary
70    ENTER @Dsa USING "%,2A,W";F$,L   !   %,2A- terminate enter item(F$) when
80                                     !    two characters have been read
90                                     !   %,W- terminate enter item(L) when 1
100   PRINT "FORMAT SPECIFIER= ";F$    !    word (two bytes) have been read
110   PRINT "LENGTH= ";L               !
120   ASSIGN @Dsa;FORMAT OFF           !   Allow binary data
130   ENTER @Dsa;Coord_buf(*)          !   Read data
140   ASSIGN @Dsa;FORMAT ON            !   Allow ASCII data
```

For the specific case of a frequency response trace with full X scale, the length specifier indicates 3528 bytes. Because every two bytes fills one element of the 16-bit integer array, Coord_buf is dimensioned to 1764 (3528 ÷ 2). To make this a general program, you should redimension the array Coord_buf to L ÷ 2.

Dumping/Loading the Synthesis and Curve Fit Tables

The synthesis table can be dumped/loaded in ASCII, ANSI binary, or Internal binary format. To dump the curve fit table, it must be converted to pole-zero synthesis format.

Contents of the Synthesis and Curve Fit Tables

Table 3-8 shows the contents of the synthesis and curve fit tables dumped via HP-IB. For enumerated (E-type) values, see Table 3-9.

Table 3-8. Contents of Synthesis & Curve Fit Tables

Item	Data Type	Size (bytes)	Internal Binary Index	ASCII/ANSI Index
Table type	E-type(0-4)	2	1	1
Number in left side	Integer	2	2	2
Number in right side	Integer	2	3	3
Left side values1	Long Real [1:44]	352	4-179	4-47
Right side values[1]	Long Real [1:44]	352	180-355	48-91
Left constraints[2]	Integer [1:22]	22	356-377	92
Right constraints[2]	Integer [1:22]	22	378-379	114
Time delay	Real	4	400-401	136
Gain factor	Real	4	402-403	137
Frequency[3]	Real	4	404-405	138
Current line	Integer	2	406	139
Current half	Integer E-type (0-1)	2	407	140
Zero total	Integer	2	408	141
Pole total	Integer	2	409	142
Z exponent sign	Integer E-type (0-1)	2	410	143
Measurement clock	Integer E-type (0-1)	2	411	144

[1] Every two values are a pair of 64-bit floating point values representing a complex conjugate pair arranged as real, imaginary. If a value is real only, the imaginary part is zero.

[2] This is an array of 22 boolean elements, one flag for each line in the table. A 1 indicates that the value in the table is constrained (user-created or fixed), and a 0 indicates that the value is unconstrained.

[3] For S domain this is the scale frequency and for Z domain this is the Sample frequency.

Table 3-9. E-types in Synthesis & Curve Fit Tables

Table Type		**Measurement clock**	
0	Pole zero synth	0	user defined
1	Pole residue synth	1	analyzer's sample freq.
2	Polynomial synth		
3	Constant trace	**Z exponent sign**	
4	Curve fit	0	positive
		1	negative
Current Half			
0	Left		
1	Right		

ASCII format

The Z-domain and S-domain synthesis table can be dumped in ASCII format by using the commands DZAS and DTAS, respectively. The tables can be loaded using the command LZAS and LTAS, respectively.

All data transfers begin with the transfer of the format and length specifier. The format specifier for an ASCII data transfer is #I. The length specifier always indicates 144 elements in a Z-domain synthesis table and 142 elements in an S-domain synthesis table data transfer. The following HP BASIC program dumps then loads the Z-domain synth table data using the ASCII data transfer format:

```
21    Ana=720                            ! Analyzer address=20, interface=7
22    REAL Table_buf(1:144)              ! Create array
23    OUTPUT Ana;"DZAS"                  ! Dump data to HP-IB in ASCII
24    ENTER Ana;Specifier$               ! read format and length specifier
25    PRINT "FORMAT SPECIFIER= ";Specifier$[1,2]
26    PRINT "Length=";Specifier$[3]
27    ENTER Ana;Table_buf(*)             ! Read  the array
28    !
29    OUTPUT Ana;"LZAS"                  ! Load data from HP-IB in ASCII
30    OUTPUT Ana;"#I 144"                ! Output format and length specifier
31    OUTPUT Ana;Table_buf(*)            ! Output the array
```

To transfer the S-domain synthesis table data using the ASCII data format, change the dimension of the array in line 22 to (1:142) and change the dump and load commands to DTAS and LTAS respectively.

ANSI binary format

The Z-domain and S-domain synthesis table can be dumped in ANSI binary format by using the commands DZAN and DTAN, respectively. The tables can be loaded by using the command LZAN and LTAN, respectively.

There are 144 Z-domain synthesis table variables and 142 S-domain synthesis table variables. To transfer these variables in ANSI 64-bit format, eight bytes for each variable must be transferred. Therefore, the length specifier (which always equals the number of bytes in a binary transfer) equals 1152 (144× 8) for Z-domain and 1136 (142× 8) for S-domain data. Before binary data can be transferred, the ASCII formatter must be turned off. This prevents the computer from trying to convert the binary data to the ASCII encoded equivalents. Your computer/language system might handle this differently. Refer to your system's manuals.

The following HP BASIC program dumps then loads the Z-domain synth table data in ANSI binary format. Refer to "Elements of a Data Transfer" in chapter 3 for a description of the HP BASIC image specifiers used to enter and load the format and length specifiers.

```
29     ASSIGN @Ana TO 720                ! Analyzer address=20, interface=7
30     REAL Table_buf(1:144)             ! Create REAL array for data storage
31     OUTPUT @Ana;"DZAN"                ! Dump Z-synth table in ANSI
32     ENTER @Ana USING "%,2A,W";F$,L    ! read format and length specifier
33     PRINT "Format specifier= ";F$
34     PRINT "Length =";L;" bytes"
35     ASSIGN @Ana;FORMAT OFF            ! turn ASCII formatter off
36     ENTER @Ana;Table_buf(*)           ! Read data
37     ASSIGN @Ana;FORMAT ON             ! turn ASCII formatter on
38     !
39     OUTPUT @Ana;"LZAN"                ! Load Z-synth table data in ANSI
40     OUTPUT @Ana USING "#,2A,W";F$,L !Output format and length specifier
41     ASSIGN @Ana;FORMAT OFF            ! turn ASCII formatter off
42     OUTPUT @Ana;Table_buf(*)          ! Output data array
43     ASSIGN @Ana;FORMAT ON             ! turn ASCII formatter on
```

To transfer the S-domain synthesis table data using the ANSI binary format, change the dimension of the array in line 30 to (1:142) and change the dump and load commands to DTAN and LTAN respectively.

Internal binary format

The Z-domain and S-domain synthesis table can be dumped in Internal binary format by using the commands DZBN and DTBN, respectively. The tables can be loaded by using the commands LZBN and LTBN, respectively.

There are 144 Z-domain synthesis table variables and 142 S-domain synthesis table variables. When dumping the synthesis table variables it is easiest to store the variables in a 16-bit integer array. It takes 411 16-bit integers to store the Z-domain variables and 409 16 bit integers to store the S-domain variables.

Before binary data can be transferred the ASCII formatter must be turned off to prevent the computer from trying to convert the binary data to the ASCII encoded equivalents. Your computer/language system may handle this differently. Refer to your system's operating manual. Refer to "Elements of a Data Transfer" in chapter 3 for a description of the HP BASIC image specifiers used to enter and output the format and length specifier.

The following HP BASIC program dumps then loads the Z-domain synthesis table data using the internal binary data format:

```
31     ASSIGN @Ana TO 720              ! Analyzer address=20, interface=7
32     INTEGER Table_buf(1:411)        ! Create INTEGER array for data
33     OUTPUT @Ana;"DZBN"              ! Dump Z-domain synth table
34     ENTER @Ana USING "%,2A,W";F$,L  ! read format/length specifier
35     PRINT "Format specifier = ";F$
36     PRINT "Length =";L;" bytes"
37     ASSIGN @Ana;FORMAT OFF          ! turn ASCII formatter off
38     ENTER @Ana;Table_buf(*)         ! Read data into INTEGER array
39     ASSIGN @Ana;FORMAT ON           ! turn ASCII formatter on
40     !
41     OUTPUT @Ana;"LZBN"              ! Load Z-domain synth table
42     OUTPUT @Ana USING "#,2A,W";F$,L ! Output format/length specifier
43     ASSIGN @Ana;FORMAT OFF          ! turn ASCII formatter off
44     OUTPUT @Ana;Table_buf(*)        ! Output INTEGER array
45     ASSIGN @Ana;FORMAT ON           ! turn ASCII formatter on
```

To transfer the S-domain synthesis table data using the internal binary data format change the dimension of the array in line 32 to (1:409) and the dump and load commands to DTBN and LTBN respectively.

Accessing Data Trace, Throughput, and Capture Disc Files

This section explains how to access data trace, throughput, and capture files, stored by the HP 3563A. All files stored on the disc by the HP 3563A are stored as binary data files (BDAT) using the HP 3563A internal binary format. Every BDAT file is preceded by a system sector that contains information about the file. The system sector cannot be directly accessed when using the HP Series 200/300 BASIC programming language, and will therefore not be included when discussing data record arrangement.

Accessing Data Trace Files on Disc

Reading data trace disc files is very similar to dumping traces directly out of the analyzer. The only difference being that the trace data portion of the file (following the header) always starts on a sector boundary. Since the HP 3563A uses only 256-byte sectors, you simply need to ignore the bytes between the end of the header and the beginning of the data. The data header is 168 bytes long, so ignore bytes 169 through 256 (words 85 through 128). See "Dumping /Loading Data Traces" earlier in this chapter for further information.

The following BASIC statements read a data trace file from a remote disc drive. Since the HP 3563A's internal binary format for all trace data is 32-bit floating point, two elements of a 16-bit integer array are required to store one element of trace data. If the length of the trace data file is unknown, then multiply the "number of elements" indicator in the data header by two to determine the required length of the 16-bit integer array:

```
50    INTEGER Trace_header(1:84)       !   set up data storage arrays
60    INTEGER Empty_space(1:44)
70    INTEGER Trace_data(1:4096)       !   space for a time record
80    ASSIGN @Path TO "DATRACE:,701"   !   open an I/O path to DATRACE which is
90                                     !      a trace stored on a disc
100   ENTER @Path;Trace_header(*)      !   read trace header
110   Length=Trace_header(2)*2         !   number of 16-bit words of trace data
120   REDIM Trace_data(1:Length)       !   redimension array
130   ENTER @Path;Empty_space(*)       !   read empty space
140   ENTER @Path;Trace_data(*)        !   read trace data
```

Notice that all data trace file names start with the prefix "DA" when accessed by the computer, but when the catalog is viewed by the HP 3563A this prefix does not appear. After dumping the header and trace data they can be loaded into the HP 3563A using the "load data in binary" (LDBN) command described earlier in this chapter. Recalling data files this way, as opposed to recalling the file from the HP 3563A, avoids having to pass control when another controller is on the bus.

Accessing Throughput and Capture Files

There are three types of throughput files:

- Single-channel
- Two-channel without delay
- Two-channel with delay

Capture files are treated as one-channel throughput files. "Delay" in this case indicates differential delay between the two channels.

The following sections explain:

- How data records are arranged for each of the three storage types
- How to scale data
- How to handle skipped tracks
- How to use the calibration table
- How to interpret the throughput/capture header

Data Record Arrangement

Throughput/capture files are stored by the HP 3563A as binary data files (BDAT) and are stored in the HP 3563A internal binary format. Each file contains a throughput/capture header of 666 bytes (333 words) followed by one or more time-domain data records. (See table 3-10 for a description of the header.)

The first data record always starts on a sector (256 byte) boundary. Therefore, the 102 bytes (51 words) between the end of the header and the start of the first data record should be ignored. Data are arranged on the disc by time records of 2048 16-bit words (4096 bytes).

For analog input data, each 16 bit word must be scaled to obtain the value in volts peak:

$$\text{Scaled data} = (-4/3)(\text{data})(\text{range})/26028.55$$

where: range and scaled data are in volts peak

For digital input data, each 16 bit word must be scaled to obtain the value in volts:

$$\text{Scaled data} = (-4/3)(\text{data})(\text{range})/32768.0$$

where: range and scaled data are in volts peak

Note

If the data is from a zoom measurement, then it is 1024 real/imaginary pairs of filtered data that has been mixed with a signal at the center frequency.

Record arrangement depends on the number of channels. Also, for throughput files, the presence of cross-channel trigger delay (used to start the session) will make a difference in file arrangement.

Single-Channel Throughput and Capture Files

In capture and single-channel throughput files, records are arranged sequentially, starting with the first record stored. See figure 3-1.

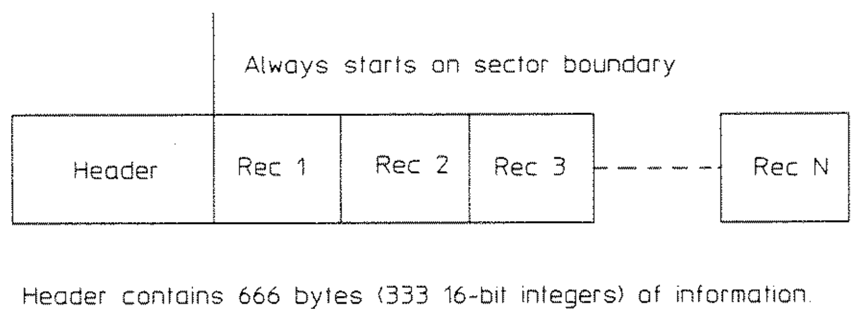


Figure 3-1. Disc Storage of One-channel Files

Two-Channel Throughput Without Trigger Delay

For two-channel throughput with no trigger delay, records alternate between channels. See figure 3-2.

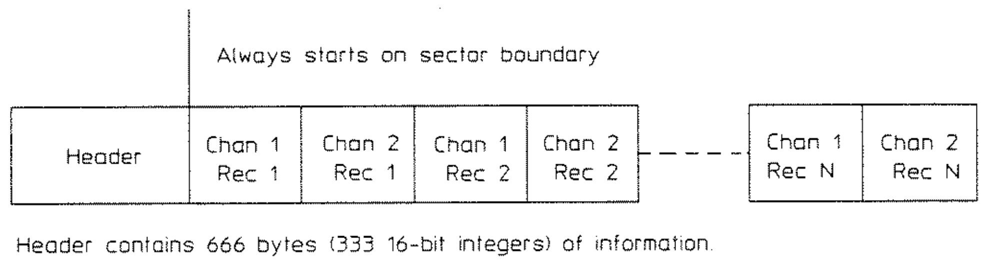


Figure 3-2. Disc Storage of Two-channel Files

The following BASIC commands read data from a two-channel throughput disc file. It then converts the channel one range indicator (range 1) from the two 16-bit integers (that represent a 32-bit floating point real number) into a number. Finally, it uses the data scaling formula to scale the channel one time record. If channel two data is to be scaled, then the range of channel two would also need to be calculated.

```
60    INTEGER Thrupt_head(1:333)      !   666 bytes(333 words) of header data
70    INTEGER Not_used(1:51)          !   Unused space between header and data
80    INTEGER Chan1_rec1(1:2048)      !   chan. 1 record 1 throughput data
90    REAL Scaled_chan1_r1(1:2048)    !   chan. 1 record 1 scaled data
100   INTEGER Chan2_rec1(1:2048)      !   chan. 2 record 1 throughput data
110   INTEGER Chan1_rec2(1:2048)      !   chan. 1 record 2 throughput data
120   INTEGER Chan2_rec2(1:2048)
130   REAL Chan1_range
140   ASSIGN @Path TO "TSTHRUPT_B:,701"
150                                   !   Assign I/O path to a file which was
160                                   !     stored during a throughput session
170   ENTER @Path;Thrupt_head(*)      !   Read throughput header
180   ENTER @Path;Not_used(*)         !   read the 120 unused bytes
190   ENTER @Path;Chan1_rec1(*)       !   read throughput data records
200   ENTER @Path;Chan2_rec1(*)
210   ENTER @Path;Chan1_rec2(*)
220   ENTER @Path;Chan2_rec2(*)
230                                   !
260   Calculate_range:                !   convert internal 32-bit binary format
265                                   !     into a number
270   Mantissa=(Thrupt_head(62)*2^(-15))+(SHIFT(Thrupt_head(63),8)*2^(-23))
280   Exponent=(SHIFT(Thrupt_head(63),-8))/2^8
290   Chan1_range=Mantissa*2^Exponent!    volts peak
300                                   !
310   Scale_data:                     !
320   FOR Point=1 TO 2048
330   Scaled_chan1_r1(Point)=(-4/3)*(Chan1_rec1(Point))*(Chan1_range)/26028.55
340   NEXT Point
```

Notice that all throughput file names, when accessed by the computer, start with the prefix TS and all capture file names start with the prefix CS. However, when the catalog is viewed by the HP 3563A, these prefixes do not appear.

Two-Channel Throughput with Trigger Delay

For two-channel throughput with trigger delay, records are interleaved according to the amount of differential delay between the two channels (see figure 3-3). Figure 3-3a defines the interleave, delay count, and delay channel indicator in the throughput header.

Interleave is the number of pairs of Channel 1/Channel 2 records between the Channel 1 records and the Channel 2 records. Delay count is the number of whole records of delay between the two channels (in this example, the delay is 2.5 records, but just two records are indicated by the delay count variable. The remaining partial record is defined by the "Partial record" indicator in the throughput header). The delay channel indicates the channel delayed beyond the other (channel 2 in this example).

Figure 3-3b explains how a remaining 1/2 record delay is handled. The partial record count shows the number of data points in the remaining partial delay record. If the data are real-only (baseband), the number of data points equals the number of words in the record. If the data are complex (zoom), the number of data points is 1/2 the number of words.

Figure 3-3c illustrates how the records are actually arranged on the disc and how they must be reassembled to get valid records for the delayed channel. In this example, the first half of Channel 2 Rec 1 and the last half of Chan 2 Rec 6 contains irrelevant data. Note that the partial record count shows both the number of invalid data points at the beginning of Rec 1 and the number of valid data points at the beginning of Rec 6.

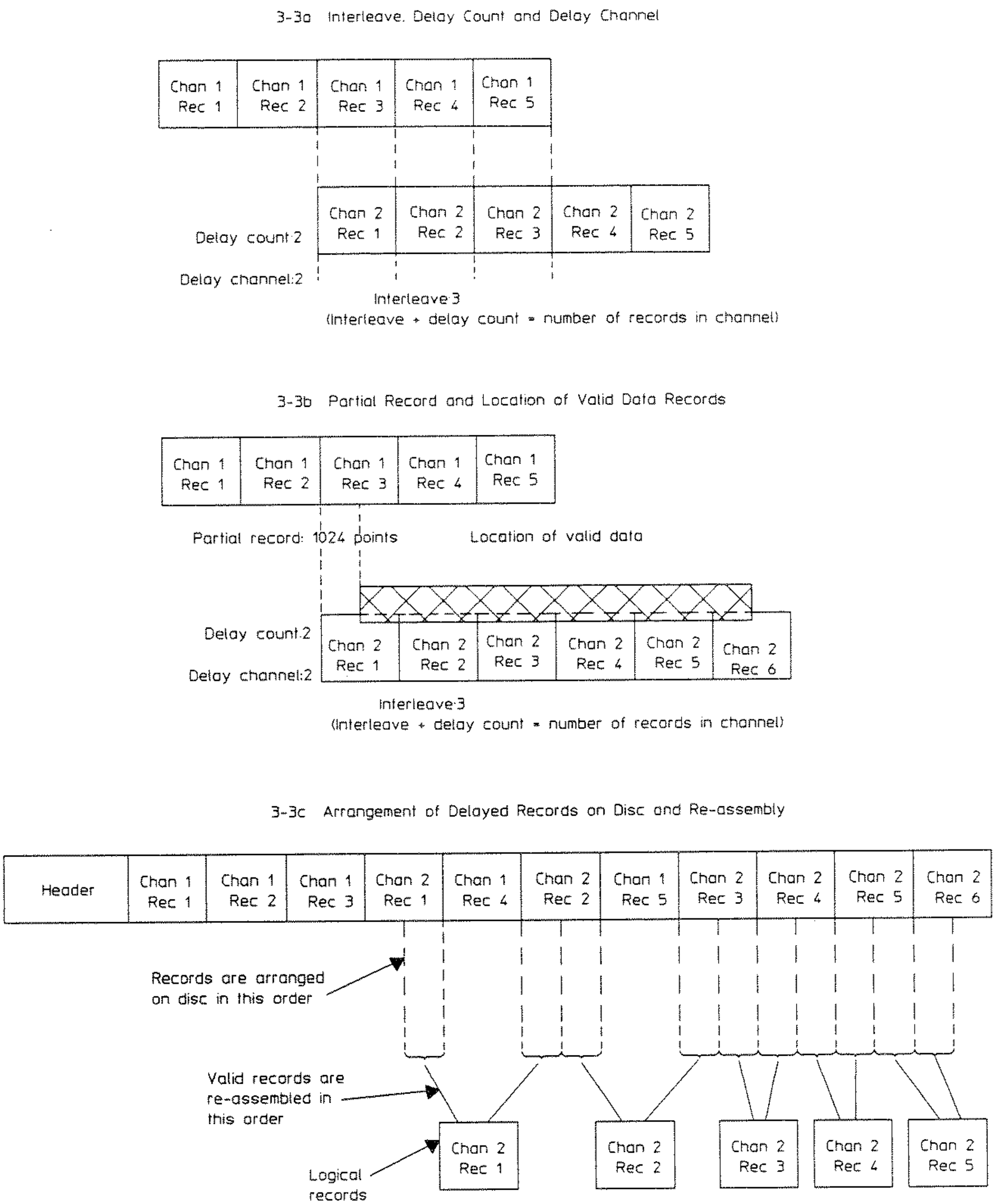


Figure 3-3. Disc Storage of Two-channel Files

Skipped Tracks

When the HP 3563A throughputs to Hewlett-Packard Command Set/80 (CS/80) disc drives, it skips over tracks that have been previously spared. (See the HP 3563A *Operating Manual* for information on sparing tracks.) Before reading data from a CS/80 disc file, you should read the skipped tracks indicator in the header and see if there are any spared tracks in the file area. If there are, you need to pass over these areas as you read the data.

The skipped track offset table shows the location of up to nine spared tracks. These are address offsets from the beginning of the entire file, including the BDAT system sector that precedes the header. You can use the sectors/ track indicator to determine how many sectors to skip. Remember that the HP 3563A always uses 256 byte sectors and there are 2048 data points per record (each point is one word, so there are 4096 bytes required to store one record). Consequently, each record requires 16 sectors (4096/256) of disc area.

Calibration Tables

Two calibration tables are stored in the throughput/capture header. Cal table 1 is used for Channel 1, and table 2 is used for Channel 2/Channel 1. The curves should be reconstructed over the desired frequency range using linear interpolation.

Each table is composed of 56 complex values. Each complex value is composed of two 16-bit integers representing a real/imaginary pair. The span from 0 to 90 kHz is covered in 2 kHz steps; the span from 91 to 100 kHz is covered in 1 kHz steps. The two cal tables are scaled by the "Mag cal scale factors" 1 and 2, respectively.

The Throughput/Capture Header

Table 3-10 shows the throughput/capture header. See Table 3-11 for enumerated (E-type) values.

Table 3-10. Throughput/Capture Header

Item	Data Type	Size (bytes)	Binary Index
Complex data flag	Integer (1 = yes 0=no)	2	1
Bytes per point	Integer	2	2
Points per record	Integer	2	3
Channel type	Integer E-type (0-3)	2	4
Bandwidth units	Integer E-type(1-3)	2	5
X units	Integer E-type(0-35)	2	6
Delay channel[1]	Integer E-type (0-3)	2	7
Delay count[1]	Integer	2	8
Partial record[1]	Integer	2	9
Interleave[1]	Integer	2	10
# of realtime records	Integer	2	11
Sectors/track	Integer	2	12
Skip track offsets[2]	Long Integers (9)	36	13-30
Digit Revision	Integer	2	31
Not used	Integer	2	32
# of skip tracks	Integer	2	33
Cal failure	Integer (1 = yes 0=no)	2	34
Start frequency	Long Real	8	35-38
Center frequency	Long Real	8	39-42
Frequency span	Real	4	43-44
Δt	Real	4	45-46
Mag cal cspc scale factor	Real	4	47-48
Mag cal fr sp scale factor	Real	4	49-50
Digitized pt len 1	Long Integer	4	51-52
Range units 1	Integer E-type (8-35)	2	53
Trig delay 1	Long Integer	4	54-55
Coupling 1	Integer E-type (29-30)	2	56
Input float 1	Boolean (1 = float)	2	57
Overflow status 1	Boolean (1 = overrange)	2	58

1 Relevant only in two-channel throughputs; refer to "Data Record Arrangement" earlier in this chapter.
2 Table contains 9 address offsets.
3 This is the correction factor at 100 kHz for the secondary attenuators.
4 These two variables are not used to calibrate throughput data, but they are available for your information.
5 Valid only if Digit Revision ≥ 1.

Table 3-10. Throughput/Capture Header cont.

Item	Data Type	Size (bytes)	Binary Index
EU Label 1	String	5(+1)	59-61
Range 1 (Vpk)	Real	4	62-63
Delay 1	Real	4	64-65
EU value 1	Real	4	66-67
Digitized pt len 2	Long Integer	4	68-69
Range units 2	Integer (E-type 8-35)	2	70
Trig delay 2	Long Integer	4	71-72
Coupling 2	Integer (E-type 29-30)	2	73
Input float 2	Boolean (1 = float)	2	74
Overflow status 2	Boolean (1 = overrange)	2	75
EU Label 2	String	5(+1)	76-78
Range 2 (Vpk)	Real	4	79-80
Delay 2	Real	4	81-82
EU value 2	Real	4	83-84
Cal table 1	Int array [2,56]	224	85-196
Cal table[2]	Int array [2,56]	224	197-308
Sec att corr 1[3]	Complex (2 reals)	8	309-312
Sec att corr 2[3]	Complex (2 reals)	8	313-316
Trigger phase corr[5]	Long Integer	4	317-318
Trigger path delay[5]	Real	4	319-320
Dig filter word 1[6]	Integer	2	321
Dig filter word 2[6]	Integer	2	322
Input 1 rnd low bits	Integer (1/0)	2	323
Input 2 rnd low bits	Integer (1/0)	2	324
Not used	Integer	2	325
Sample clock	Integer E-type (0-4)	2	326
Input 1 digital	Integer (1/0)	2	327
Input 2 digital	Integer (1/0)	2	328
Input 1 comput delay	Real	4	329-330
Input 2 comput delay	Real	4	331-332
Mixed ratio	Integer (1/0)	2	333

1 Relevant only in two-channel throughputs; refer to "Data Record Arrangement" earlier in this chapter.
2 Table contains 9 address offsets.
3 This is the correction factor at 100 kHz for the secondary attenuators.
5 These two variables are not used to calibrate throughput data, but they are available for your information.
6 Valid only if Digit Revision ≥ 1

Table 3-11. E-types in Throughput/Capture Header

Channel Type and Delay Channel		**Bandwidth Units and X Units Cont.**	
0	Channel 1	23	Oct/minute
1	Channel 2	24	Octave
2	Both channels	25	Sec/decade
3	No channels		
		26	Sec/octave
Bandwidth Units and X Units		27	Hz/point
0	Null	28	Points/sweep
1	Hz	29	Points/decade
2	RPM	30	Points/octave
3	Orders		
4	Seconds	31	V/Vrms
		32	Volts^2
5	Revs	33	Channel 1 EU
6	Degrees	34	Channel 2 EU
7	dB	35	EU
8	dBV		
9	Volts	**Range Units 1/2**	
		8	dBV
10	Volts/Hz	9	Volts
11	Hz/second	13	Volts rms
12	Volts/EU	35	EU
13	Volts rms		
14	Volts^2/Hz	**Coupling 1/2**	
		29	AC
15	Percent	30	DC
16	Points		
17	Records	**Sample clock**	
18	Ohms	0	Source pod
19	Hz/octave	1	Input 1 pod
20	Pulses/rev	2	Input 2 pod
		3	Qualifier pod
21	Decades	4	External sample
22	Minutes		

4

Signal Processing Group

About This Chapter

This chapter explains how to use the signal processing command group. These commands let you set up data blocks in the HP 3563A's memory, and then perform a number of signal processing operations on these blocks. The topics covered here include:

- Overview of signal processing steps
- General block operations
- Transferring blocks
- Math operations
- Averaging operations
- Measurement operations
- Plotting and graphing results

Note

Most of the signal-processing operations described in this chapter can be performed using waveform math. Review the chapter in the *Operating Manual* that describes the math operations before using the primitives in this chapter. If waveform math meets your needs, it presents a much simpler programming task than the signal processing primitives.

Note

The HP 3563A must be paused before you use signal processing primitives (otherwise the data blocks will be erased).

Overview of Signal Processing

There are five general steps to perform signal processing primitive operations in the HP 3563A:

1. Set up primitive blocks
2. Input data
3. Perform operations
4. Output results
5. Display results

The first step, setting up the blocks, is covered in the next section under "General Block Operations." There are four types of data that are used in the blocks:

- 32 bit floating point values
- Complex (real, imaginary) 32 bit floating point pairs
- 16 bit integers
- Complex (real, imaginary) 16 bit integer pairs

The default data type of a newly setup block is 32 bit floating point. The data type is changed when a block is loaded or when any subsequent block operation is performed.

For step two, there are two choices for input data: digital input via HP-IB or signals from the input channels. Transferring primitive data blocks via HP-IB is covered in "Transferring Blocks" later in this chapter. Collecting analog input data blocks is covered in "General Block Operations" (the ANIN command).

For step three, choose the desired operation from "Math Operations," "Averaging Operations" or "Measurement Operations" later in this chapter.

To output results via HP-IB, refer to "Transferring Blocks."

Finally, if you want to display the result on the analyzer's screen, refer to "Plotting and Graphing Results" at the end of this chapter.

Partial Memory Map

The following memory map shows the location of important data blocks. Signal processing blocks start overlaying RAM at "TRACE A DATA."

Partial Memory Map
(Typical Linear Resolution State)

DATA Block	Size (words)
TRACE A DATA	(4k)
TRACE B DATA	(4k)
DATA ON CHANNEL 1 (integer)	(4k)
DATA ON CHANNEL 2 (integer)	(4k)
FFT OUTPUT—CHANNEL 1 (integer)	(2k)
FFT OUTPUT—CHANNEL 2 (integer)	(2k)
MEASUREMENT WORKING BLOCK—CHANNEL 1	(4k)
MEASUREMENT WORKING BLOCK—CHANNEL 2	(4k)
MEASUREMENT AVERAGING BLOCKS	(8k)

Data in these blocks can be dumped or loaded by overlaying blocks and then using the commands in "Transferring blocks" later in this chapter.

General Block Operations

This section describes the commands used to create and handle data blocks. You should familiarize yourself with these commands first before attempting to use the other commands later in this chapter. The commands covered in this section are:

Block size	(BLSZ)
Point count	(PTCT)
Float block	(FLTB)
Unfloat block	(UFLB)
Move block	(MOVB)
Move complex constant	(MOVX)
Move real constant	(MOVC)
Partial block clear	(PCLR)
Analog input	(ANIN)

Block Size (BLSZ)

The block size command (BLSZ) allocates memory for signal processing operations by creating individual blocks. When creating blocks, you specify the size of the block(s), the number of the first block, and how many blocks you want to create. The syntax of BLSZ is:

BLSZs,n1[,n]

where:	s is the size of the block(s) in words
	n1 is the number of the first block
	n is the number of blocks to be created (optional)

There are approximately 37.9 Kwords of RAM reserved for signal processing blocks. Individual block size is limited to 32 Kwords. If you want to create multiple blocks, make sure that their combined size does not exceed 37.9 Kwords. The number of the first block, n1, must be between 0 and 15, inclusive. The number of blocks to be created, n, can be from 1 to 16, provided that the combination of n1 and n used does not attempt to create a block numbered higher than 15. Note that n is optional; if you do not specify it, one block is created.

As an example, the BASIC statement:

OUTPUT 720; "BLSZ100,0,2"

creates 2 100-word blocks, numbered 0 and 1.

The maximum number of points (elements of data) in the block depends on the type of data the block contains. For example, if the size of a data block is 4096 words, then that block can contain the following number of points:

2048 32 bit floating point values
1024 complex 32 bit floating point pairs
4096 16 bit integers
2048 complex 16 bit integer pairs

If more than one block exists in memory, changing the size of the lower-numbered blocks affects higher-numbered blocks. For example, if you have ten 500-word blocks, and you recreate block 1 only at 400 words, blocks 2 through ten will be shifted down 100 words. This can effectively erase data — so recreate blocks carefully.

If you intend to use the FFT operations, some constraints apply to primitive blocks. All blocks used in FFT operations must be 2048 words long (2 Kwords). Also, all blocks used for FFTs must reside on 2 Kwords boundaries (for example, the size of all lower-numbered blocks must be a multiple of 2 Kwords).

Point Count (PTCT)

The point count command (PTCT) allows you to specify the number of points of an existing block for use in subsequent operations. A point is an element of data in the block. Each point may be a 32 bit floating point value, a pair of 32 bit floating point values representing a complex number (real, imaginary), a 16 bit integer or a pair of 16 bit integers representing a complex number (real, imaginary). The syntax is:

PTCTn, p

where:
n is the block
p is the number of points

The block number, n, must be between 0 and 15 and must represent an active block. The number of points, p, specifies that the first p points in block n will be used. Of course, p cannot be greater than the size of the block. As an example, the BASIC statement:

```
OUTPUT 720;" PTCT1,50"
```

specifies that the first 50 points in block 1 will be used any time in the future that the block is used. To respecify the point count for a block, send PTCT again.

Float Block (FLTB)

The float block command (FLTB) is used to convert integer data to floating point format.
The syntax is:

FLTBn1,n2 [,count]

where:
n1 is the integer block (source)
n2 is the destination block
count is point count (optional)

FLTB floats n1 and puts the result in n2. Of course, n1 and n2 must be valid block numbers.

Also, n2 must be at least twice the size of n1 to accommodate the 32 bit floating point values.

Unfloat Block (UFLB)

The unfloat block command (UFLB) is used to convert floating-point data to integer format. To do this a block exponent is assigned (determined by the largest number in the block) and then the remaining mantissa is rounded to the nearest integer value.
The syntax is:

UFLBn1,n2 [,count]

where:
n1 is the floating point block (source)
n2 is the destination
count is point count (optional)

UFLB unfloats n1 and puts the result in n2. Of course, n1 and n2 must be valid block numbers.

Move Block (MOVB)

The move block command (MOVB) is used to move the contents of one block into another block. The syntax is:

MOVBn1,n2, [,p]

where:
n1 is the source block
n2 is the destination block
p is the number of points to be moved (optional)

The block numbers n1 and n2 specify the source and destination, respectively. The point count, p, specifies how many points from n1 are to be moved into n2. If p is not specified, all of n1 is moved. As an example, the BASIC statement:

OUTPUT 720;" MOVB9,3,50"

moves the first 50 points of block 9 into block 3.

Move Complex Constant (MOVX)

The move complex constant command (MOVX) moves a complex constant into a complex block. The syntax is:

MOVXn1,n2,n3[,count]

where:
n1 is the real part of source constant
n2 is the imaginary part of source constant
n3 is the complex destination block
count is destination point count (optional)

The block number n3 must represent a valid block. As an example, the BASIC statement:

OUTPUT 720;" MOVX1,2,3"

moves 1 + j2 into each element of block 3.

Move Real Constant (MOVC)

Moves a real constant into a block. The syntax is:

MOVCn1,n2[,count]

where: n1 is real source constant
n2 is real destination block
count is destination point count (optional)

Partial Block Clear (PCLR)

The partial block clear command (PCLR) allows you to clear points at the beginning of a block. The syntax is:

PCLRn1,p

where: n1 is the block to be partially cleared
p is the number of points to be cleared

For example, the BASIC statement:

OUTPUT 720; "PCLR1,5"

clears the first 5 points in block 1.

Analog Input (ANIN)

Note The measurement must be paused (PAUS) before using the ANIN command. Failure to pause the measurement will result in a total lockup of the analyzer.

The analog input command (ANIN) allows you to take raw 16 bit integer data from the input channels (analog or digital) for use in signal processing primitives. The syntax is:

ANINn1,n2,c1,c2

where:
- n1 is the destination block for Channel 1
- n2 is the destination block for Channel 2
- c1 is the number of points to take on Ch 1
- c2 is the number of points to take on Ch 2

For example, the BASIC statement:

```
OUTPUT 720;"ANIN1,2,1024,1024"
```

inputs 1024-point blocks on both channels into primitive blocks 1 (Channel 1) and 2 (Channel 2). If you don't want data from a channel, set the number of points for that channel to zero. The following formula can be used to scale the raw 16 bit analog input data.

Scaled data (Vpk) = (−4/3)(data)(range)/26028.55

where: range is input channel range in volts peak.

Digital data is scaled by using this formula:

Scaled data (Vpk) = (−4/3)(data)(range)/32768.0

where: range is input range in volts peak

Data collected by the ANIN command are digital filtered. If the frequency span is less the 100 Khz then the data has been decimated by the same amount as the frequency span reduction. If the measurement is not a baseband measurement (the start frequency is greater than 0), then the data collected are digitally mixed with a signal at the center frequency and are returned as real, imaginary number pairs. Each complex pair is considered one point.

See the example program at the end of this chapter for an example of how to use this command.

Transferring Blocks

This section explains how to transfer signal processing blocks between the controller and the HP 3563A. Each dump or load requires two steps — identifying the block to be transferred, and sending the dump or load command specifying the data transfer format. The topics covered in this section are:

- The primitive block header
- The block pointer (PBLK)
- Dumping blocks in ASCII (DBAS), in ANSI (DBAN), and in internal binary (DBBN)
- Loading blocks in ASCII (LBAS), in ANSI (LBAN), and in internal binary (LBBN)

The Primitive Block Header

Every primitive block has a three-word header located at the beginning of the block. These three words are transparent to any size specifications. If you dump a block, make sure to allow for these three non-data words.

Table 4-1 shows the primitive block header. Note that the header has this format regardless of the data format of the block.

Table 4-1 Primitive Block Header

Word	Description	Range
1	Block type	0 = floating point (32 bit values)
		1 = complex floating point (2 32 bit values)
		2 = integer (16 bit integers)
		3 = complex integer (2 16 bit values)
2	Block exponent	see text
3	Point count	equal to PTCT value

The value of word 1 depends on the data format in which the block was filled and any subsequent operations performed on it. The block exponent value in word 2 is used to calculate amplitude values for integer and complex integer data blocks (types two and three). The equation is:

$$\text{Amplitude} = \text{value}\ (2^{\text{block exponent}})$$

Finally, the value of word 3, the point count, is equal to the point count specified for the block. If you have previously specified this with the PTCT command, word 3 will be equal to the value of PTCT. If you have not used PTCT on this block, word 3 is equal to the dynamic length of the block in points.

Primitive Block Pointer (PBLK)

The primitive block pointer command (PBLK) specifies the active block for dumping and loading. The syntax is:

PBLKn

where: n is the number of the block

The number of the block, n, must be between 0 and 15 and must represent an existing block.

Dumping Blocks

Primitive data blocks can be dumped in each of the three data formats (see chapter 3 for descriptions of data types). When the HP 3563A receives the dump command, it outputs the following elements:

- Format specifier (#I for ASCII, #A for binary)
- Length specifier
- Header (described earlier in this section)

The length variable differs from the point count in the header — the length variable includes the three header elements while the point count does not.

Note The analyzer must be paused during a transfer of a primitive data block.

Dump Block in ASCII (DBAS)

The dump block in ASCII command (DBAS) dumps a block (specified by the block pointer) in ASCII format. (For a description of the ASCII format, see chapter 3.) The format specifier is #I, and the length specifier indicates the number of elements to be transferred. See the program at the end of this chapter for example of how to use this command.

Dump Block in ANSI (DBAN)

The dump block in ANSI format (DBAN) dumps a block (specified by the block pointer) in ANSI floating point format. (For a description of this format, see chapter 3.) The format specifier is #A, and the length specifier indicates the number of bytes to be transferred. Only blocks shorter than 32,768 bytes (including header) can be transferred this way.

When using the ANSI transfer, remember that these are 8-byte floating point values. Also, if your computer has an ASCII formatter, you must disable it for ANSI transfers.

Dump Block in Internal Binary (DBBN)

The dump block in internal binary (DBBN) dumps a block (specified by the block pointer) in the internal 32-bit floating point format. (For a description of this format, see chapter 3.) The format specifier is #A, and the length specifier indicates the number of bytes to be transferred. Only blocks shorter than 32,768 bytes (including header) can be transferred this way.

If your computer has an ASCII formatter, you must disable it for binary transfers.

Loading Blocks

When primitive data blocks are loaded into the HP 3563A, it expects the following elements:

- Format specifier (#I for ASCII, #A for binary)
- Length specifier
- Header (described earlier in this section)

Load Block in ASCII (LBAS)

The load block in ASCII command (LBAS) loads a block (specified by the block pointer) in ASCII format. (For a description of the ASCII format, see chapter 3.) The format specifier is #I, and the length specifier indicates the number of elements to be transferred. See the program at the end of this chapter for example of how to use this command.

Load Block in ANSI (LBAN)

The load block in ANSI format (LBAN) loads a block (specified by the block pointer) in ANSI floating point format. (For a description of this format, see chapter 3.) The format specifier is #A, and the length specifier indicates the number of bytes to be transferred. Only blocks shorter than 32,768 bytes (including header) can be transferred this way.

When using the ANSI transfer, remember that these are 8-byte floating point values. Also, if your computer has an ASCII formatter, you must disable it for ANSI transfers.

Load Block in Internal Binary (LBBN)

The load block in internal binary (LBBN) loads a block (specified by the block pointer) in the internal binary format. (For a description of this format, see chapter 3.)

The format specifier is #A, and the length specifier indicates the number of bytes to be transferred. Only blocks shorter than 32,768 bytes (including header) can be transferred this way.

If your computer has an ASCII formatter, you must disable it for binary transfers.

Math Operations

The commands in this section perform math operations on data blocks. If you have not yet created and filled the blocks needed for your math operation, refer to the previous section, "General Block Operations." The commands covered in this section are:

Add blocks	(ADDB)
Add complex constant	(ADDX)
Add real constant	(ADDC)
Subtract blocks	(SU BB)
Subtract complex constant	(SUBX)
Subtract real constant	(SUBC)
Multiply blocks	(MPYB)
Multiply by complex constant	(MPYX)
Multiply by real constant	(MPYC)
Multiply by $j\omega$	(MPJW)
Multiply by self-conjugate	(MPSC)
Multiply by magnitude squared	(MPMG)
Divide by block	(DIVB)
Divide by complex constant	(DIVX)
Divide by real constant	(DIVC)
Divide by $j\omega$	(DVJW)
Divide imaginary part	(DIVI)
Divide real part	(DIVR)
Divide into real constant	(DVIR)
Negate block	(NEGB)
Conjugate block	(CNJB)
Differentiate block	(DIFB)
Integrate block	(INGB)
Power spectrum summation	(PSPS)
Cross spectrum summation	(CSPS)

Add Blocks (ADDB)

The add blocks command (ADDB) allows you to add two data blocks. The syntax is:

ADDBn1,n2[,n3]

where:
- n1 is the first addend
- n2 is the second addend
- n3 is the destination of the result (optional)

ADDB adds n1 to n2 and puts the result in n3. The destination block n3 is an optional parameter—if it is not specified, the result is put in n2.

Add Complex Constant to Block (ADDX)

The add complex constant command (ADDX) allows you to add a complex constant to a complex block. The syntax is:

ADDXn1, n2, n3[,n4]

where:
- n1 is the real part of the source constant
- n2 is the imaginary part of the source constant
- n3 is the complex second addend block
- n4 is the optional destination for the result

If n4 is not specified, the result is put in n3. As an example, the BASIC statement:

```
OUTPUT 720;"ADDX1,2,3,4"
```

adds (1 + j2) to each element of block 3 and puts the result in block 4.

Add Real Constant to Block (ADDC)

The add real constant to block command (ADDC) adds a real constant to the contents of a second block. The syntax is:

ADDCn1,n2[,n3]

where:
- n1 is the source constant
- n2 is the real second addend block
- n3 is the optional destination for the result

If n3 is not specified, the result is put in n2.

Subtract Blocks (SUBB)

The subtract block command (SUBB) allows you to subtract one block from another. The syntax is:

SUBBn1,n2[,n3]

where:
- n1 is the minuend
- n2 is the subtrahend
- n3 is the optional destination block

SUBB subtracts n2 from n1 and puts the result in n3. If n3 is not specified, the result is put in n2.

Subtract Complex Block From Complex Constant (SUBX)

The subtract complex constant command (SUBX) allows you to subtract a complex block from a complex constant. The syntax is:

SUBXn1,n2,n3[,n4]

where:
n1 is the real part of the minuend
n2 is the imaginary part of the minuend
n3 is the complex subtrahend
n4 is the optional destination for the result

If n4 is not specified, the result is put in n3. As an example, the BASIC statement:

OUTPUT 720;"SUBX1,2,3,4"

subtracts each element in block 3 from (1 + j2) and puts the result in block 4.

Subtract Real Constant From Block (SUBC)

The subtract real constant from block command (SUBC) subtracts a block from a real constant. The syntax is:

SUBCn1,n2[,n3]

where:
n1 is the constant minuend
n2 is the subtrahend block
n3 is the optional destination for the result

If n3 is not specified, the result is put in n2.

Multiply Blocks (MPYB)

The multiply blocks command (MPYB) allows you to multiply two blocks. The syntax is:

MPYBn1,n2[,n3]

where:
n1 is the first factor
n2 is the second factor
n3 is the optional destination for the result

MPYB multiplies n1 by n2 and puts the result in n3. Since n3 is an optional parameter, if it is not specified the result is put in n2.

Multiply Block by Complex Constant (MPYX)

The multiply complex constant command (MPYX) allows you to multiply a complex constant by a complex block. The syntax is:

MPYXn1,n2,n3[,n4]

where:
n1 is the real part of the source constant
n2 is the imaginary part of the source constant
n3 is the complex block
n4 is the optional destination for the result

If n4 is not specified, the result is put in n3.

Multiply Block by Real Constant (MPYC)

The multiply real constant block command (MPYC) multiplies a real constant by a real block. The syntax is:

MPYCn1,n2[,n3]

where:
n1 is the source constant
n2 is the real block
n3 is the optional destination for the result

If n3 is not specified, the result is put in n2.

Multiply Block by jω (MPJW)

The multiply by jω command (MPJW) allows you to multiply a block by jω to perform artificial differentiation. The syntax is:

MPJW$\omega^{\underline{start}}$,$\Delta\omega$,n1[,n2]

where:
$\omega^{\underline{start}}$ is the starting value of ω
$\Delta\omega$ is the ω increment
n1 is the block to be differentiated
n2 is the optional destination block for the result

Multiply Block by Self-conjugate (MPSC)

The multiply by self-conjugate command (MPSC) allows you to multiply a complex block by its complex conjugate. The syntax is:

MPSCn1[,n2]

where:
n1 is the complex block
n2 is the optional destination for the result

Multiply Block by Magnitude Squared (MPMG)

The multiply by magnitude squared command allows you to multiply a real block by the magnitude squared of a complex block. The syntax is:

MPMGn1,n2[,n3]

where:

n1 is the real block
n2 is the complex block
n3 is the optional destination of the result

If n3 is not specified, the result is put in n1.

Divide Block by Block (DIVB)

The divide block command (DIVB) allows you to divide one block by another. The syntax is:

DIVBn1,n2[,n3]

where:

n1 is the dividend
n2 is the divisor
n3 is the optional destination for the result

DIVB divides n1 by n2 and puts the result in n3. Since n3 is an optional parameter, if it is not specified the result is put in n2.

Divide Block by Complex Constant (DIVX)

The divide block by complex constant command (DIVX) allows you to divide a block by a complex constant. The syntax is:

DIVXn1,n2,n3[,n4]

where:

n1 is the real part of the divisor
n2 is the imaginary part of the divisor
n3 is the complex dividend block
n4 is the optional destination for the result

If n4 is not specified, the result is put in n3. As an example, the BASIC statement:

OUTPUT 720; "DIVX1,2,3,4"

divides each element of block 3 by (1 + j2) and puts the result in block 4.

Divide Block by Real Constant (DIVC)

The divide block by real constant command (DIVC) divides a block by a real constant. The syntax is:

DIVCn1,n2[,n3]

where: n1 is the constant divisor
n2 is the dividend block
n3 is the optional destination for the result

If n3 is not specified, the result is put in n2.

Divide Block by $j\omega$ (DVJW)

The divide by $j\omega$ command (DVJW) allows you to divide a block by $j\omega$ to perform artificial integration. The syntax is:

DVJWω^{start},$\Delta\omega$,n1 [,n2]

where: ω^{start} is the starting value of ω
$\Delta\omega$ is the ω increment
n1 is the block to be integrated
n2 is the optional destination block for the result

If n2 is not specified, the result is put in n1.

Divide Imaginary Part of Block (DIVI)

The divide imaginary part of block command (DIVI) allows you to divide the imaginary part of a complex block by a real constant. The syntax is:

DIVIn1,n2[,n3]

where: n1 is the complex block
n2 contains the real value
n3 is the optional destination for the result

If n3 is not specified, the result is put in n1.

Divide Real Part of Block (DIVR)

The divide real part of block command (DIVR) allows you to divide the real part of a complex block by a real constant. The syntax is:

DIVRn1,n2[,n3]

where: n1 is the complex block
n2 contains the real value
n3 is the optional destination for the result

If n3 is not specified, the result is put in n1.

Divide Block into Real Constant (DVIC)

The divide block into real constant command (DVIC) allows you to divide a real block into a real constant. The syntax is:

DVICn1,n2[,n3]

where:	n1 is the real divisor block n2 is the real dividend constant n3 is the optional destination for the result

If n3 is not specified, the result is put in n1.

Negate Block (NEGB)

The negate block allows you to negate the contents of a block. The syntax is:

NEGBn1[,n2]

where:	n1 is the block to be negated n2 is the optional destination for the result

If n2 is not specified, the result is put in n1.

Conjugate Block (CNJB)

The conjugate block command (CNJB) computes the complex conjugate of a data block. The syntax is:

CNJBn1[,n2]

where:	n1 is the block to be conjugated n2 is the optional destination of the result

If n2 is not specified, the result is put in n1.

Differentiate Block (DIFB)

The differentiate block command (DIFB) computes the differential of a data block. The syntax is:

DIFB n1 [,n2]

where:	n1 is the block to be differentiated n2 is the optional destination for the result

If n2 is not specified, the result is put in n1.

Integrate Block (INGB)

The integrate block command (INGB) computes the integral of a data block. The syntax is:

INGBn1[,n2]

where:	n1 is the block to be integrated n2 is the optional destination for the result

If n2 is not specified, the result is put in n1.

Power Spectrum Summation (PSPS)

The power spectrum summation command (PSPS) computes the power spectrum of a complex floating point block and sums it with the contents of a second block. The syntax is:

PSPSn1,n2

where:	n1 is the block to be summed n2 contains the cumulative result

Cross Spectrum Summation (CSPS)

The cross spectrum summation command (CSPS) computes the cross spectrum of two complex floating point blocks and sums the result with the contents of a third block. The syntax is:

CSPSn1,n2,n3

where:	n1 is the first complex block n2 is the second complex block n3 contains the cumulative result

Averaging Operations

The HP 3563A offers the following averaging primitives:

Exponential averaging	(XAVG)
Power spectrum exponential averaging	(PXAV)
Cross spectrum exponential averaging	(CXAV)
Peak hold	(PKHD)
Power spectrum peak hold	(PPEK)
Cross spectrum peak hold	(CPEK)

Exponential Averaging (XAVG)

The exponential average command (XAVG) averages data blocks using an exponentially weighted averaging formula. The syntax is:

XAVG n1, n2,awf

where:
n1 is the block to be averaged
n2 is the cumulative average
awf is the exponential weighting factor

The weighting factor, awf, is interpreted as a power of 2. The formula used in exponential averaging is:

$$A_n = (1-2^{-n})a_n + 2^{-n}D_n$$

where:
A_n is cumulative average (in n2)
D_n is new block (in n1)
a_n is exponential weighting factor

Power Spectrum Exponential Averaging (PXAV)

The power spectrum exponential averaging command (PXAV) computes the power spectrum from a complex block; then exponentially averages this power spectrum with a cumulative average in another block. The syntax is:

PXAVn1, n2, awf

where:
n1 is the complex data block to be averaged
n2 is the cumulative average
awf is the exponential weighting factor

Refer to the Exponential Averaging Command (XAVG) for the formula used.

Cross Spectrum Exponential Averaging (CXAV)

The cross spectrum exponential averaging command (CXAV) computes the cross spectrum of two complex blocks, then exponentially averages this cross spectrum with a cumulative average in another block. The syntax is:

CXAVn1, n2, n3,awf

where:
n1 is the first complex block
n2 is the second complex block
n3 is the cumulative average
awf is the exponential weighting factor

Refer to the exponential averaging command (XAVG) for the formula used.

Peak Hold (PKHD)

The peak hold command (PKHD) compares the magnitudes of two blocks on a point-to-point basis and holds the larger values. The syntax is:

PKHDn1,n2

where:
n1 is the new block
n2 contains the peak values

Power Spectrum Peak Hold (PPEK)

The power spectrum peak hold command (PPEK) computes the power spectrum of a complex block, then compares its magnitudes to a second power spectrum block and holds the larger values. The syntax is:

PPEKn1,n2

where:
n1 is the new complex block
n2 contains the peak values

Cross Spectrum Peak Hold (CPEK)

The cross spectrum peak hold command (CPEK) computes the cross spectrum of two complex blocks, then compares those magnitudes to a third cross spectrum block and holds the larger values. The syntax is:

CPEKn1,n2,n3

where:
n1 is the first complex block
n2 is the second complex block
n3 contains the peak values

Measurement Operations

The HP 3563A offers the following measurement primitives:

Histogram	(HST)
Real FFT	(RFFT)
Complex FFT	(CFFT)
Real inverse FFT	(RFT1)
Complex inverse FFT	(CFT1)

Histogram (HST)

The histogram command (HST) computes the histogram of a block and records the histogram count in a second block. The syntax is:

HSTn1, n2,vmax

where:
n1 is the block to be computed (cannot be complex)
n2 is the destination block
vmax is the maximum absolute amplitude range for block n1

The number of histogram bins equals the number of points in the destination block (must be greater than zero). Vmax should be greater than the magnitude of any element in n1 to allow for rounding.

Real FFT (RFFT)

The real FFT command (RFFT) computes the FFT of a integer data block and stores the result in a second block. The syntax is :

RFFTn1, n2

where: n1 is the block to be transformed
n2 is the destination for the result

The result is a 1K point complex block. RFFT can be performed only on block sizes of 2048 words that reside on 2K boundaries in memory. The imaginary part of the DC bin contains the Fs/2 point (used by the inverse FFT). To place a block on a 2K boundary, make sure that all data blocks up to the block to be transformed are multiples of 2 Kwords long. Also, blocks for FFT and inverse FFT operations must reside in the first 32 Kwords of the 37.9 Kwords available for signal processing primitives. The FFT commands use the window currently selected, unless the force or exponential window is active (in this case, the uniform window is used).

Note

To obtain the correct two-sided linear spectra from the FFT commands, multiply by the appropriate window correction factor:

Uniform→ 1.414242555
Hann → 2.828485107
Flat top→ 7.403524615

Note

Forward FFTs use a coefficient of $\frac{1}{\sqrt{2N}}$
where: N= number of points

Complex FFT (CFFT)

The complex FFT command (CFFT) computes the FFT of a complex integer data block and stores the result in a second block. The syntax is:

CFFTn1,n2

where: n1 is the block to be transformed
n2 is the destination for the result

The result is a 1K point complex block. CFFT can be performed only on block sizes of 1024 complex points that reside on 2K boundaries in memory. To place a block on a 2K boundary, make sure that all data blocks up to the block to be transformed are multiples of 2 Kwords long. Also, blocks for FFT and inverse FFT operations must reside in the first 32 Kwords of the 37.9 Kwords available for signal processing primitives. The FFT commands use the window currently selected, unless the force or exponential window is active (in this case, the uniform window is used).

Real Inverse FFT (RFT1)

The real inverse FFT command (RFT1) computes the inverse FFT of a complex integer data block and stores the result in a second block. The syntax is:

RFT1n1,n2

where: n1 is the block to be transformed
n2 is the destination for the result

The result is a 2K point real block. RFT1 can be performed only on block sizes of 1024 complex points that reside on 2K boundaries in memory. To place a block on a 2K boundary, make sure that all data blocks up to the block to be transformed are multiples of 2 Kwords long. Also, blocks for FFT and inverse FFT operations must reside in the first 32 Kwords of the 37.9 Kwords available for signal processing primitives. The FFT commands use the window currently selected, unless the force or exponential window is active (in this case, the uniform window is used).

Note The variable n1 is destroyed by the inverse FFT operation

---

Complex Inverse FFT (CFT1)

The complex inverse FFT command (CFT1) computes the inverse FFT of a complex integer data block and stores the result in a second block. The syntax is:

CFT1n1,n2

where: n1 is the block to be transformed
n2 is the destination for the result

The result is a 1K point complex block. CFT1 can be performed only on block sizes of 1024 complex points that reside on 2K boundaries in memory. To place a block on a 2K boundary, make sure that all data blocks up to the block to be transformed are multiples of 2 Kwords long. Also, blocks for FFT and inverse FFT operations must reside in the first 32 Kwords of the 37.9 Kwords available for signal processing primitives. The FFT commands use the currently selected window.

Plotting and Graphing Data Blocks

The HP 3563A's plotting and graphing primitives allow you to display data blocks on the analyzer's screen. The plotting operations plot data versus data to create traces. The graphing operations create displays given a data block and an X-axis increment. The commands covered in this section are:

Plot complex block	(PCBL)
Plot real block	(PRBL)
Graph block	(GRBL)
Graph imaginary part	(GRIM)
Graph real part	(GRRE)

Plotting Complex Blocks (PCBL)

The plot complex block command (PCBL) plots the real part of a complex block versus the imaginary part of that block. The syntax is:

PCBLn1

where: n1 is the complex block to be plotted.

Plotting Real Blocks (PRBL)

The plot real block command (PRBL) allows you to create a display by plotting one real floating point data block against another. The syntax is:

PRBLn1,n2

where: n1 is the first real block
n2 is the second real block

Both blocks must be real and their point counts must be set the same.

Graphing Real Blocks (GRBL)

The graph real block command (GRBL) creates a trace from a real block and an X-axis increment. The syntax is:

GRBLn1, x,Δ x

where: n1 is the block to be graphed
x is the X-axis starting point
Δx is the X-axis increment

Before using this command, you need to create and activate a display buffer that is at least as big as the primitive block you want to graph. Refer to Chapter 5 for handling display buffers. The primitive block n1 is transferred to the active display buffer when GRBL is executed. See the program on the next page for an example of how to use this command.

Graphing Imaginary Part of Blocks (GRIM)

The graph imaginary part command (GRIM) is similar to the graph block command (GRBL)—however, GRIM uses just the imaginary part of a complex block to create the trace. The syntax is:

GRIMn1, x,Δx

where: n1 is the block to be graphed
x is the X-axis starting point
Δx is the X-axis increment

Refer to GRBL if you need more information.

Graphing Real Parts of Blocks (GRRE)

The graph real part command (GRRE) is similar to GRIM, but GRRE graphs only the real part of a complex block. The syntax is:

GRREn1,x,Δx

where: n1 is the block to be graphed
x is the X-axis starting point
Δx is the X-axis increment

Refer to GRIM and GRBL if you need more information.

The following program demonstrates the use of some of the primitive block commands.

```
13    Ana=720                     ! Analyzer address=20, interface=7
14    INTEGER Block1(1:2048)      ! Set up arrays for data block transfers
15    REAL Scaled_data1(1:2048)
16    OUTPUT Ana;"FSIN 500 HZ; SRLV 1.0V"! Set up source to 1 volt sign
17    OUTPUT Ana;"C1RG 1.26 V"    ! set input range at 1.26 volt peak
18    OUTPUT Ana;"PAUS"           ! The analyzer MUST BE PAUSED!!!!!!!!!!!!!
19  ! CREATE BLOCKS========================================================
20    OUTPUT Ana;"BLSZ4096,1,2"   ! Create 2 Blocks(#1&#2) of size 4096 words
21    ! Note, the default data type is 32 bit real
22  ! FILL BLOCK WITH ANALOG INPUT DATE ==================================
23    OUTPUT Ana;"ANIN1,2,2048,2048"!Load block#1 and #2 with 2048 points
24  ! DUMP BLOCK===========================================================
25    OUTPUT Ana;"PBLK1"          ! set primitive block pointer to 1
26    OUTPUT Ana;"DBAS"           ! dump block specified by PBLK in ASCII
27    ENTER Ana;Specifier$        ! enter format & length specifier
29    ENTER Ana;Blk_type,Blk_exp,Point_count ! enter 3 element header
31    ENTER Ana;Block1(*)         ! enter block data
32  ! SCALE ANALOG INPUT DATA==============================================
33    OUTPUT Ana;"C1RG?"          ! ask analyzer for value of channel 1 range
34    ENTER Ana;Range             ! enter range in VOLTS PEAK
35    FOR X=1 TO Point_count
36    Scaled_data1(X)=(-4/3)*Block1(X)*Range/26028.55!scale analog data
37    NEXT X
39  ! LOAD BLOCK===========================================================
40    OUTPUT Ana;"PBLK1"          ! set primitive block pointer to 1
41    OUTPUT Ana;"LBAS"           ! load block specified by PBLK in ASCII
42    OUTPUT Ana;"#I 2051"        ! send format and length specifier
43    OUTPUT Ana;"0,0,2048"       ! Send header"type, exponent, # of points"
44    OUTPUT Ana;Scaled_data1(*)
45  ! SCALE BLOCK FOR DISPLAY==============================================
46    OUTPUT Ana;"MPYC256,1"      ! Multiply each element in block1 by 256 to
47                                ! to set 1 volt pk equal to 256 pixels.
48    OUTPUT Ana;"ADDC1024,1"     ! Add 1024 to each element of block #1 to
49                                ! shift time record to center of display.
50    PRINT "DATA SCALED FOR 4 VOLT peak to peak MAX"
51  ! GRAPH BLOCK==========================================================
52    OUTPUT Ana;"PTCT1,2000"     ! Set point count of block 1 to 2000 points
53    OUTPUT Ana;"DBSZ4096,1,1"   ! create 1 display buffer #1 of size 4096
54    OUTPUT Ana;"DBAC1"          ! activate display buffer #1
55    OUTPUT Ana;"GRBL1,18 ,1"    ! fill display buffer with graph of block
56    OUTPUT Ana;"DBUP1"          ! display user buffer #1
57    LOCAL Ana
58    END
```

Display Control Group

About This Chapter

This chapter explains the use of the display control group of bus-only commands. There are three approaches to programming the display: the Hewlett-Packard Graphics Language (HP-GL), the binary language used by the display, or defining the display as a plotter for HP BASIC graphics commands. This chapter addresses the following topics:

- Description of the vector display (including display programming methods)
- Handling buffers
- Programming with HP-GL
  - Moving the pen
  - Writing into buffers
  - Drawing into buffers
- Direct binary programming
  - The programming language
  - Loading binary display buffers
- Defining the display as an HP BASIC plotter
- Dumping display buffers

To get started, read the description of the display and the instructions on handling buffers. Then select the method best suited to your application.

Description of the Vector Display

The HP 3563A's display produces images by combining vectors and text characters. There are 2048 points on each axis, for a total of over 4 million addressable points on the display. The lower left corner of the display is address 0,0 and the upper right corner is 2047,2047. The display's aspect ratio is 4.7:3.9 (X,Y).

How to Program the Display

As stated at the beginning of the chapter, there are three methods you can use to program images on the display:

- HP-GL commands
- Direct binary programming
- Defining the display as an HP BASIC plotter

HP-GL is the language used by Hewlett-Packard plotters. The HP 3563A implements a subset of this language. Using HP-GL is a simple way to create custom graphics. You can create up to 16 display buffers, which you then fill with commands and put on the display as needed. The commands are straightforward; each performs just one function. For example, to select line type 1 (solid lines), you simply send the Line Type command "LT1" to the appropriate buffer.

Direct binary programming, instead of having many simple commands, has just four commands, (each performing multiple tasks). Each command is a 16-bit word, and you configure each bit in the command. For example, the Set Conditions command selects line types, but it can also select brightness and writing speed. While the direct binary commands are more complicated, they provide faster display control because fewer individual commands are required. (In fact, the HP-GL commands are used internally to select the binary commands; HP-GL isolates you from the bit-by-bit programming.)

Defining the display as an HP BASIC plotter lets you program the display in a high level language. This method is the easiest for BASIC users, but it is the slowest.

In summary, use the display as a BASIC plotter when you want programming that is easy to learn and easy to use, and when speed is not a concern. Use the direct binary method for more serious graphics work when both program size and execution time are critical. Finally, use HP-GL when you need faster execution than BASIC, and "friendlier programming" than direct binary.

A two-step procedure that gives you the ease of HP-GL and the speed of direct binary is to load a buffer with HP-GL commands, dump it back to the analyzer, then reload it as a binary command buffer. Once you convert a set of HP-GL commands to binary (which is done automatically as you fill the buffer) you can then take advantage of direct binary's speed. "Dumping Display Buffers," later in this chapter, explains how to do this.

Overview of Display Programming Steps

Regardless of the method you use, there are four general steps to programming user displays:

1. Create display buffers in the analyzer's memory
2. Activate a particular buffer
3. Load the buffer (with HP-GL, binary or BASIC)
4. Display the buffer

You must follow this sequence to get anything on the display. Steps 1, 2, and 4 are independent of the method used and are covered in the next section, "Handling Display Buffers." Step 3 is dependent on the method used (this is discussed later in this chapter).

Handling Display Buffers

A display buffer is simply an area you reserve in the HP 3563A's memory for display programming. You can create up to 16 display buffers. There are six commands for handling buffers:

1. DBSZ (display buffer size): creates and sizes buffers
2. DBAC (display buffer activate & clear): clears and activates a particular buffer
3. DBAA (display buffer activate & append): activates a buffer and allows commands to be added to it
4. DBUP (display buffer up): puts a buffer up on the display
5. DBDN (display buffer down): takes a buffer down off the display
6. DBSW (display buffer switch): replaces the buffer on the display with another buffer.

These commands are discussed in the following paragraphs. Keep in mind that the general sequence used with buffers is to create a buffer, activate it, fill it with commands, then put it up on the display.

Creating Buffers

Buffers are created with the DBSZ (display buffer size) command. This sets the size, identifies each buffer with a unique number, and determines how many buffers are created. The syntax is:

DBSZs, n1, n

where: s is size of buffer in words (4960 max)
n1 is number of first buffer
n is number of buffers created

For example, the BASIC statement:

OUTPUT 720;" DBSZ100,0,4"

creates four buffers, numbered 0, 1, 2 and 3, each 100 words long. There are approximately 11 Kwords of memory available for all display buffers, and the combined size of all buffers you create cannot exceed this. The number of the first buffer, n1, must be between 0 and 15, inclusive. The number of buffers, n, cannot cause buffers to be numbered higher than 15. For example, if n1 is 10, n cannot be greater than 6.

Clearing and Activating a Buffer

Before a buffer can be filled, it must be activated. You have two choices: clear and activate, or append and activate (discussed next). One buffer can be active at any time. It is the active buffer that receives the graphics commands sent to the analyzer. The syntax for clearing and activating is:

DBACn

where: n is the number of the buffer

The buffer specified must already exist, and n must be between 0 and 15, inclusive. For example, the command:

DBAC1

clears buffer number 1 and then activates it. If the specified buffer is already on the display, DBAC takes it down and clears it.

Clearing Buffers

To clear a buffer without activating it, use the clear buffer command (CLBFn, where n is the buffer to be cleared).

Appending and Activating a Buffer

If you need to add commands to a buffer that has some commands already in it but is not currently active, you need to append and activate, rather than clear and activate. The syntax is:

DBAAn

where: n is the buffer to be activated

For example, the BASIC statement:

OUTPUT 720;"DBAA5"

activates buffer number 5 without clearing it. As with DBAC, the buffer must already exist, and n must be between 0 and 15, inclusive. If buffer n is already on the display, it is taken down and activated.

Putting Buffers Up and Down

After you have filled a buffer (user buffers) with the desired commands, the next step is to put it up on the display. This is done with DBUPn, where n is the buffer to be displayed. The command is ignored if n is already up.

To take a buffer down, use DBDNn, were n is the buffer to be taken down. For both DBUP and DBDN, the buffer must already exist, and n must be between 0 and 15, inclusive.

Display Buffer Switch

For fast buffer switching, the DBSW (display buffer switch) command is provided. The syntax is:

DBSWn1, n2

where: n1 is the buffer to go up
n2 is the buffer to come down

If n1 is already on the display, the command has no effect. Both buffers must already exist, and n1 and n2 must be between 0 and 15, inclusive.

Programming with HP-GL

The Hewlett-Packard Graphics Language (HP-GL) provides a simple method of programming the analyzer's display. Here is the general sequence of steps used with HP-GL:

1. Set up necessary buffer(s)
2. Activate one buffer
3. Move pen to desired location
4. Write text or draw vector
5. Repeat steps 3 and 4 as needed
6. Put the buffer up on the display

Modify this sequence as needed to produce your display. Steps 1, 2, and 6 are discussed earlier in this chapter under "Handling Display Buffers." Remember, the screen does not change until the buffer is put up on the display. The following sections show you how to move the pen, write text, and draw vectors.

Moving the Pen

The "pen" is the beam used to produce images on the display. The nomenclature is carried over from the original use of HP-GL, where the pen is an actual pen in a plotter. This section explains how to control and move the pen. Pen control is used for two purposes: positioning the pen to start writing or drawing, and actually drawing vectors.

Turning the "Pen" On and Off

Two commands determine whether the pen is up or down. PU (pen up) lifts the pen (turns off the beam). PD (pen down) sets the pen down (turns on the beam). To move from one point to another without drawing on the display (as if positioning the pen to start drawing) turn off the beam again. To move while drawing, (when drawing a vector or writing text), turn on the beam again. In many cases you cannot be certain of the beam's current status, so it is a good idea to explicitly turn it on or off before moving it. Note that, unlike a plotter, dropping the pen on a display does not produce a dot; you need to move it a short distance to produce a mark.

Absolute and Relative Plotting

There are two ways of moving the pen: Absolute plotting moves to an address relative to the origin (0,0—the lower left corner). The command is PA (Plot Absolute). For example, the BASIC statement:

```
OUTPUT 720; "PA1000,1000"
```

moves the beam to approximately the center of the display. The first number is the X-axis location, and the second is the Y-axis location. Remember, moving the pen may or may not draw to the specified address—it depends if the beam is on or off.

Relative plotting moves to an address relative to the current position of the beam. The command is PR (Plot Relative). For example, if the pen had not been moved since the PA1000,1000 command, sending the basic statement:

```
OUTPUT 720;"PR0,-500"
```

moves the beam 500 Y-axis units down from the previous y-address of 1000. The X-axis location is not changed because its relative address was specified as 0. Note that negative X values move the beam to the left, and negative Y values move the beam down.

Writing into Buffers

Once you have the pen positioned, you can write text into the buffer. You can control character size, brightness, and rotation when writing text.

Setting Character Size

Character size is set with CHSZn, where n is 0-3:

0 = 24 x 36 points(default)
1 = 36 x 54 points
2 = 48 x 72 points
3 = 60 x 90 points

Setting Brightness

There are four levels of brightness you can select, using BRITn, where n is 0-3:

0 = off
1 = dim
2 = half bright
3 = full bright (default)

Rotating Characters

Characters can be rotated at four angles, using CHROn, where n is 0-3:

0 = 0° (default)
1 = 90°
2 = 180°
3 = 270°

Writing on the Display

When you have positioned the beam, set size, brightness and rotation, you are ready to write text. The command is WRIT, and the alpha string must be enclosed either in single quote marks or a pair of double quote marks. For example, the BASIC statements:

```
OUTPUT 720;"WRIT'MESSAGE'"
    and
OUTPUT 720;"WRIT""MESSAGE"""
```

both write MESSAGE at the current beam position. Because of the obvious complexity of the second format, the first is recommended.

As an example of combining the four text commands, the BASIC statements:

```
10    OUTPUT 720;"DBSZ20,1,1"
11    OUTPUT 720;"PU"
12    OUTPUT 720;"PA800,1000"
13    OUTPUT 720;"PD"
14    OUTPUT 720;"CHSZ2"
15    OUTPUT 720;"BRIT3"
16    OUTPUT 720;"CHRO1"
17    OUTPUT 720;"WRIT'XXXXXXXXXX'"
18    OUTPUT 720;"DPUP1"
```

write XXXXXXXXXX on the display at a 90 degree angle, with character size 2(48× 72 points) and brightness 3(full bright).

Drawing into Buffers

Drawing vectors is merely a special application of moving the beam. Send the PD command to turn the beam on, then PA (Plot Absolute) and PR (Plot Relative) can draw vectors. For example, the following BASIC statements draw a vector from the center of the screen to 800 units down the Y-axis to 1000,200.

```
10      OUTPUT 720;"DBSZ20,1,1
11      OUTPUT 720;"DBAC1"
12      OUTPUT 720;"PU"
13      OUTPUT 720;"PA1000,1000"
14      OUTPUT 720;"PD"
15      OUTPUT 720;"PR0,-800"
16      OUTPUT 720;"DBUP1"
```

The brightness selection (BRITn) explained in the last section applies to vectors as well. There is one more selection for vectors only — selecting the line type.

Selecting Line Types

Lines types can be selected with LTn, where n is 0-4:

0 = solid lines (default)
1 = solid lines with intensified endpoints
2 = long dashed lines
3 = short dashed lines
4 = endpoints only

If an optional second parameter is sent, it is ignored (for HP-GL compatibility).

Figure 5-1 shows the five line types available.

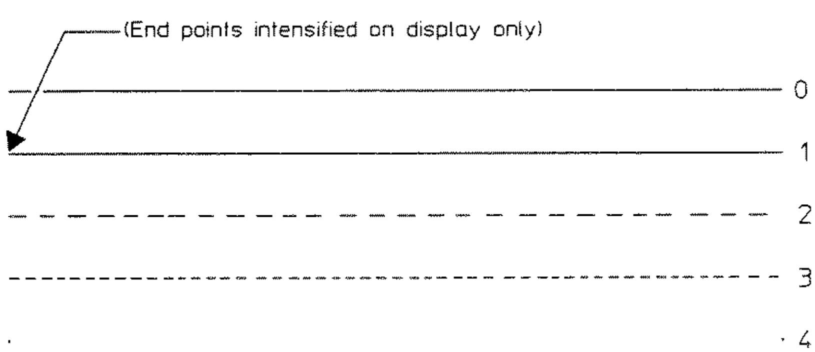


Figure 5-1. Display Line Types

The following HP BASIC program demonstrates how HP-GL commands can be used to control the display. The program gives two examples: first writing a message, then drawing some boxes on the display.

```
20   Ana=720                      ! Analyzer address=20, interface=7
25                                !
30   OUTPUT Ana;"COMD"            ! disable command echo
35   OUTPUT Ana;"DBSZ500,1,2"     ! set up 2 500-word display buffers #1 & #2
40   OUTPUT Ana;"DBAC1"           ! activate buffer #1
45   OUTPUT Ana;"CHSZ2"           ! character size 2
50   OUTPUT Ana;"PU"              ! pen up (beam off)
55   OUTPUT Ana;"PA100,1000"      ! move pen to left center of display
60   OUTPUT Ana;"PD"              ! pen down (beam on)
65   OUTPUT Ana;"WRIT'Custom Graphics Display'"   ! write text to buffer
70   OUTPUT Ana;"DBUP1"           ! put up buffer #1
75   WAIT 1                       ! wait one second
80                                !
85   ! The rest of the program draws fifteen boxes on the display.
90   !
95   OUTPUT Ana;"DBAC2"           ! activate buffer #2
100  !
105  T=1900                       ! top of box
110  B=300                        ! bottom of box
115  L=100                        ! left side
120  R=1700                       ! right side
125  FOR Box=1 TO 15              ! draw 15 boxes
130  OUTPUT Ana;"PU"              ! pen up
135  OUTPUT Ana;"PA";L,B          ! Move pen to bottom left corner.
140  OUTPUT Ana;"PD"              ! pen down (beam on)
145  OUTPUT Ana;"PA";L,T          ! These 4 lines form the box.
150  OUTPUT Ana;"PA";R,T
155  OUTPUT Ana;"PA";R,B
160  OUTPUT Ana;"PA";L,B
165                               !
170  OUTPUT Ana;"DBSW2,1"         !  switch buffers
175  OUTPUT Ana;"DBAA2"           !  add to buffer #2
180  T=T-50                       !  These 4 lines decrease the
185  B=B+50                       !    size of the next box.
190  L=L+50
195  R=R-50
200  NEXT Box
205  END
```

Direct Binary Programming

Direct binary programming is the fastest way to control the user display. The overall programming method is the same as the HP-GL method: create a buffer, activate it, load it, then display it. The only difference being that the buffer is directly loaded with binary display commands instead of sending HP-GL commands that the analyzer must convert to binary display commands. The buffer can be loaded via the HP-IB bus using the ASCII, ANSI binary or the analyzer's internal binary data format. Refer to Chapter 3 for descriptions of the three data formats.

The Display's Binary Language

The display module is controlled with four commands: PLOT, GRAPH, SET CONDITION, and TEXT. These four commands provide complete programmable vector and text generation. Most vector and text operations can be handled with only one 15-bit command word.

These 15-bit data words are decoded by the display module into one of four distinct commands. Each 15-bit data word can be separated into two distinct data fields. The display module 15-bit data word is shown in Figure 5-2.

Each of the commands the display module can recognize is selected by the state of data bits D14 and D13. The lower 13 data bits D0-D12 are used as command modifiers.

MSB														LSB
D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
COMMAND		COMMAND MODIFIERS												

	Bit 14	Bit 13
Set Condition	1	1
Plot Command	0	0
Graph Command	0	1
Text Command	1	0

Figure 5-2. Display Module 15 Bit Data Word

These modifiers allow each command to have several selectable attributes. Vector drawing operations are directly dependent on the status of these data bits in every command. Each of these commands and their modifiers will be discussed using programming examples. The 15-bit data for the examples will be in HEXADECIMAL or HEX format. This format is easier to follow than 15-bit binary data words.

The HP BASIC programming examples send 16-bit data words in decimal form. One way to convert the binary words to decimal form is to add the decimal value of each bit that is high (1). For example, if bit 12 and bit 14 are high, then the decimal value is $2^{12} + 2^{14}$. Another way to convert the binary words to decimal form is to use the HP BASIC command DVAL, which will convert binary or hexadecimal strings to decimal.

The next section explains how to transfer binary display data to the analyzer and the following sections describe how to program the display using the four commands.

Loading User Buffers

The LUAS command is used to load the active display buffer with 16 bit words using the ASCII format data transfer. Here is a sample listing:

```
55    OUTPUT @Ana;"DBSZ100,1,1"    ! create a buffer of 100 elements
60    OUTPUT @Ana;"DBAC1"          ! activate display buffer
65    OUTPUT @Ana;"LUAS"           ! Load User buffer in ASCII
70    OUTPUT @Ana;"#I  8"          ! Send format and length specifier
75    OUTPUT @Ana;Array(*)         ! send INTEGER array to analyzer
80    OUTPUT @Ana;"DBUP1"          ! put up buffer #1
```

After receiving LUAS, the analyzer first expects #I to specify ASCII data, then expects a variable containing the number of ASCII variables to be sent. The analyzer is now ready for data, which is "Array" in this example. After the active buffer (1) is loaded with the contents of "Array", the buffer is put up on the display with DBUP.

The LUAN command is used to load the active display buffer with 16 bit words using the 64-bit ANSI floating point format data transfer. Here is a sample listing:

```
65    OUTPUT @Ana;"LUAN"            ! Load User buffer in ANSI
70    OUTPUT @Ana USING  #,2A,W ;"#A",64 ! Send format and length specifier
75                                  !  #,2A- send 2-characters, suppress EOI
80                                  !  #,W-  send a 16 bit integer,suppress EOI
85    ASSIGN @Ana;FORMAT OFF        ! turn off ASCII formatter
90    OUTPUT @Ana;Array(*)          ! send REAL array to analyzer
95    ASSIGN @Ana;FORMAT ON         ! turn on ASCII formatter
```

After receiving LUAN, the analyzer expects to receive #A to specify binary data followed by a 16 bit word specifying the number of bytes to be output (64 in this example). The analyzer is now ready for data, which is in "Array", in this example. The ASCII formatter must be deactivated to prevent it from converting ANSI to ASCII. Your computer/language may handle this differently; if it automatically formats output data to ASCII, you need to disable this feature before sending ANSI data. The HP 3563A converts 64-bit ANSI numbers to 16-bit words before putting them in the display buffer. Buffer #1 is then put up on the display.

The LUBN command is used to load the active display buffer with 16 bit words using a the internal binary data transfer format. Here is a sample listing:

```
65    OUTPUT @Ana;"LUBN"            ! Load User buffer in Internal binary
70    OUTPUT @Ana USING  #,2A,W ;"#A",16 ! Send format and length specifier
75                                  ! #,2A- send 2-characters, suppress EOI
80                                  ! #,W-  send a 16 bit integer,suppress EOI
85    ASSIGN @Ana;FORMAT OFF        ! turn ASCII formatter off
90    OUTPUT @Ana;Array(*)          ! send INTEGER array
95    ASSIGN @Ana;FORMAT ON         ! turn ASCII formatter on
```

After receiving LUBN, the analyzer expects to receive #A to specify binary data followed by a 16-bit word specifying the number of bytes to be output (16 in this example). The analyzer is now ready for data, which is in "Array" in this example. The ASCII formatter must be deactivated to prevent it from converting binary to ASCII. Your computer/language may handle this differently; if it automatically formats output data to ASCII, you need to disable this feature before sending binary data. Finally, buffer #1 is put up on the display.

Display Module Commands

Set Condition Command

When D14 and D13 are both High (1), the display module will interpret the display buffer data word as a SET CONDITION command. This command is used to set vector attributes. The attributes affected are line type, speed, and intensity. The required bit patterns for this command and its command modifiers are contained in Figure 5-3.

By combining line intensity and writing speed parameters, up to twelve levels of discernible intensities can be generated. Figure 5-4 contains several example combinations. This allows the user to create displays with background graticules and intensify important trace data. The beam will be brightest with the intensity set at full bright at the slowest writing speed. The beam will be dimmest with the intensity set at dim and at the fastest writing speed. The SET CONDITION command may be executed at any time and the vector attributes will remain in effect until another SET CONDITION command is executed. Data bit 6 in this command is defined to be low. This MUST occur when the Set Condition command is executed or the display may respond in an undefined fashion.

Set Condition Command:

MSB D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	LSB D0
1	1	I_1	I_0	X	X	L_1	L_0	0	X	W_1	W_0	X	X	X

Note: Bit 6 (D6) must be zero.

Command Modifiers:

a. To Set Line Intensity:

I_1	I_0	Intensity
0	0	Blank
0	1	Dim
1	0	Half Brightness
1	1	Full Brightness

b. To Set Line Type:

L_1	L_0	Type
0	0	Solid Line
0	1	Intensified End Points on Solid Line
1	0	Long Dashes
1	1	Short Dashes

c. To Set Writing Speed:

W_1	W_0	Speed
1	1	0.05 in. per μs
1	0	0.10 in. per μs
0	1	0.15 in. per μs
0	0	0.20 in. per μs

Figure 5-3. Set Condition Command

6998h	Dim, Short Dash, Speed 0.05
7800h	Bright, Solid, Speed 0.2
7000h	Half Bright, Solid, Speed 0.2
7100h	Half Bright, Long Dash, Speed 0.2

Figure 5-4. Set Condition Examples in HEX

Plot Command

When the two most significant bits of the data word, D14 and D13 are low (0), the display will recognize the display buffer data word to be a PLOT command. Figure 5-5 contains the correct bit pattern for this command.

Plot Command:

MSB D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	LSB D0
0	0	XY	PC	D_{10}	D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
				DATA										

Command Modifiers:

a. XY Information (D12)

0 = X coordinate (0-2047), specified by D_0-D_{10}
1 = Y coordinate (0-2047), specified by D_0-D_{10}

b. PC Beam Control Information (D11)

0 = Beam OFF (move)
1 = Beam ON (draw)

Figure 5-5. Plot Command Bit Pattern

This command moves the beam to a specific X-Y location in the defined cartesian coordinate plane each time an X-Y coordinate pair is received. The values of the X and Y coordinates range from 0 to 2047. The origin of the cartesian plane is located in the lower left corner and has an X-Y value of (0,0). This command also turns the beam on or off for each vector. The beam may be moved in either mode. The vector is drawn from the previous beam location to the current location specified by the last two X,Y coordinate values in the PLOT commands. The vector is drawn in accordance with the last SET CONDITION command.

The diagram in Figure 5-6 is a single vector defined by its endpoints in the vector drawing area. To draw this line the display module would need to receive two sets of X and Y coordinates. The display module receives the coordinates in the specified order X1, Y1, X2, Y2. The beam is moved only when the Y coordinate is received. The status of the beam is only affected by the beam status bit in the Y coordinate command.

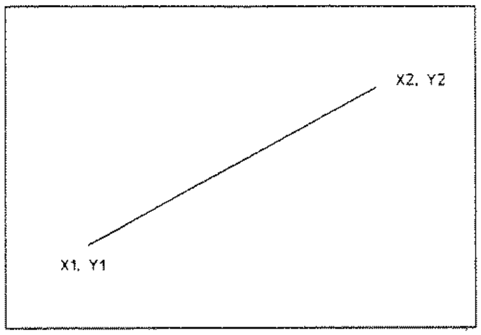


Figure 5-6. Vector Defined by Endpoints

An example of vector plotting is contained in Figure 5-7. This example contains vectors drawn with the beam on and with the beam off. The steps to draw these figures are given in the required sequence with equivalent HEX and decimal code for the 16-bit data words.

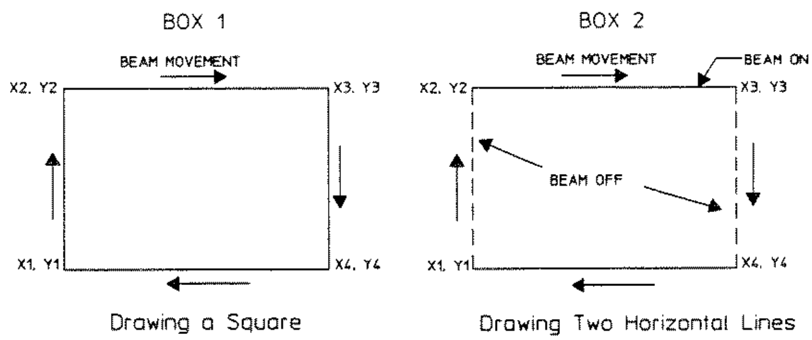


Figure 5-7. Plot Vector Example

To draw the figures, send a display buffer array containing the following data words in sequence to the analyzer.

	Command Step	Box 1 data	Box 2 data	Command
1.	Set Condition	7818h (30744d)	7818h (30744d)	Sets Vector type (Solid Full Bright,.05)
2.	Plot X1	0200h (512d)	0200h (512d)	X1=512
3.	Plot Y1 (beam off)	1200h (4608d)	1200h (4608d)	move to Y1=512
4.	Plot Y2	1F00h (7936d)	1700h (5888d)	move to Y2=1792
5.	Plot X3	0F00h (3840d)	0F00h (3840d)	X3=1792
6.	Plot Y3 (beam on)	1F00h (7936)	1F00h (7936d)	move to Y3=1792
7.	Plot Y4	1A00h (6656d)	1200h (4608d)	move to Y4=512
8.	Plot X1	0200h (512d)	0200h (512d)	X1=512
9.	Plot Y1 (beam on)	1A00h (6656d)	1A00h (6656d)	move to Y1=512

A description of these two examples will help the user understand the vector plotting process. Step 1 defines the vector attributes for the vectors to be plotted. Definition of a starting point is crucial when plotting vectors. Steps 2 and 3 initialize the starting point of the box. Next a new Y value indicates that the beam may be turned on. Since the X value did not change, only a new Y value need be sent. The beam will move to the location specified by the X-Y location when the Y value is received. The vector is drawn according to the status of the last SET CONDITION command.

When a new horizontal location is required, both the X and Y coordinates need to be sent. The beam is only moved and the vector drawn when a Y coordinate is received. The Y value does not change going from Step 4 to Step 5, but the X value does. This requires that a new X-Y coordinate pair be sent as in Steps 5 and 6. In Step 7, the X value doesn't require a change so only a new Y value is sent in Step 7. The beam is turned on to draw the vector. In Steps 8 and 9 a new X-Y pair is required so both values must be sent. To draw box 2, only Steps 4 and 7 need to be changed. The beam status bit tells the display module to turn the beam off during the movement. A vector is still drawn, but with the beam turned off.

The user should notice that when a vector is to be drawn vertically, only a Y value is sent for the second vector endpoint. The display module has a "last X" register that stores the value of the last X location. This feature allows vertical vectors with the same X values to be drawn with one less endpoint requirement.

When plotting vectors in the vector drawing area the user should take into account the difference in CRT screen height and width. The display module vector drawing area is 9.5 cm high by 12.5 cm wide and has 2048 addressable points in either direction. If this difference is not taken into account, boxes will appear as rectangles. To plot vectors correctly, the user may need to apply a scaling factor to vector endpoint calculations. The scaling factors are approximately 215.58 addressable points/cm in the Y direction and 163.84 addressable points/cm in the X direction. These figures are used when calculating the actual length of vectors in cm.

The following HP BASIC program creates the graphics in Figure 5-7 (box 1) using the above commands.

```
25    ASSIGN @Ana TO 720           ! Analyzer address=20, select code=7
30    OUTPUT @Ana;"PAUS"           ! PAUSE the measurement
35    WAIT 1
40    INTEGER Array(1:9)
45    DATA 30744                   ! define vector attributes
50    DATA 512,4608                ! initialize the box starting point
55    DATA 7936,3840,7936,6656,512,6656! define box coordinates
70    READ Array(*)
75    OUTPUT @Ana;"DBSZ100,1,1"    ! create a Display Buffer of SiZe 100
80    OUTPUT @Ana;"DBAC1"          ! Display Buffer ACtivate
85    OUTPUT @Ana;"LUAS"           ! Load User buffer in ASCII
90    OUTPUT @Ana;"#I  9"          ! Send format specifier and length
95    OUTPUT @Ana;Array(*)         ! send array
100   OUTPUT @Ana;"DBUP1"          ! put up user buffer on display
120   END
```

Graph Command

The GRAPH command is very similar to the PLOT command. The GRAPH command allows plotting of vectors that have equal incremental X coordinates. When data word bits D14 and D13 are low and high respectively the display module interprets the display buffer data word to be a GRAPH command as shown in Figure 5-8. In the GRAPH mode, the display module automatically increments the X coordinate after each Y coordinate is received. This allows single valued functions to be plotted in graph form with fewer endpoints than would be possible using X,Y coordinates for each data point.

Graph Command:

MSB														LSB
D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0	1	XY	PC	D_{10}	D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
				DATA										

Command Modifiers:

a. XY Information (D12)

0 = Set Delta-X increment, specified by D_0-D_{10} for all subsequent Y coordinates
1 = Set Y coordinate, specified by D_0-D_{10}. The beam is to be moved to this Y in conjunction with the Delta X increment.

b. PC Beam Control Information (D11)

0 = Beam OFF (move)
1 = Beam ON (draw)

Figure 5-9. Bit Definition For Graph Commands

There are three command modifiers in the GRAPH command. These modifiers control the X increment, Y coordinate data value, and the beam status. When D12 is 0, the data in bits D0-D10 define the value of the X increment. This is the amount the X coordinate will increase after each Y coordinate is plotted. The range of the X increment is 0 to 2047. It should be noted that X increases relative to the present X,Y coordinate values on the screen. Figure 5-9 contains an example of the graph mode commands. The beam moves when the Y coordinate value is received.

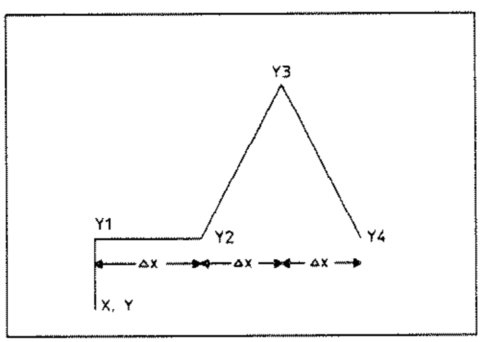


Figure 5-8. Graph Mode Example

To create the output in Figure 5-9, a display buffer containing the following steps was executed.

Command Step	16 Bit Data	Command
1. Set Condition	7818h (30744d)	Set Vector Attributes (Solid Full Bright,.05)
2. Plot X	0200h (512d)	X=512
3. Plot Y (beam off)	1200h (4608d)	move to Y=512
4. Graph command Set Delta X	2040h (8256d)	set X increment to 64
5. Graph command Y1 (beam on)	3A80 (14976d)	Y=640
6. Graph command Y2	3A80 (14976d)	Y=640
7. Graph command Y3	3B00 (15104d)	Y=768
8. Graph command Y4	3A80 (14976d)	Y=640

Step 1 defines the line type, speed, and intensity. Steps 2 and 3 determine the starting point of the graph. The delta X increment is established in Step 4. The four Y values are set in Steps 5-8. The value of X is incremented AFTER each Y value.

If the graph is to start at the axis origin, execute a graph command with a first Y value set to zero. This will not plot anything, but will increment the X value by delta X. The next vector will be drawn from the origin to the Y value for the first X increment. If the graph is to start at the Y axis, execute a Y value command. The next vector will be drawn from the Y value on the Y axis to the Y value of the first X increment.

The following HP BASIC program creates the graphics in Figure 5-9, using the commands shown above.

```
50    ASSIGN @Ana TO 720           ! Analyzer address=20, select code=7
60    OUTPUT @Ana;"PAUS"           ! PAUSE the measurement
70    WAIT 1
80    INTEGER Array(1:8)
90    DATA 30744                   ! define vector attributes
100   DATA 512,4608                ! initialize the plot starting point
101   DATA 8256                    ! set delta X increment value for graph mode
110   DATA 14976,14976,15104,14976 ! set Y values for graph mode
140   READ Array(*)
150   OUTPUT @Ana;"DBSZ99,1,1"     ! create a Display Buffer of SiZe 99
160   OUTPUT @Ana;"DBAC1"          ! Display Buffer ACtivate
170   OUTPUT @Ana;"LUAS"           ! Load User buffer in ASCII
180   OUTPUT @Ana;"#I  8"          ! Send format specifier and length
190   OUTPUT @Ana;Array(*)         ! send array
200   OUTPUT @Ana;"DBUP1"          ! put up user buffer on display
240   END
```

Text Command

The display module has an internal character generator. This internal character data is an ASCII character set modified for graphics use. Figure 5-10 shows the command used to switch the display from graphics to text mode. Data bits D14 must be high and D13 must be low. When this command is executed, the display module will interpret the lower eight data bits, D0-D7, as an equivalence for an ASCII or special character. Each vector of the character is drawn on the CRT screen according to the vector characteristics of the SET CONDITION command. The characters are always drawn at the slowest writing speed. The line type has no visible effect except on the largest character size (2.5x). The position of the character is defined by the last X and Y coordinates.

When generating characters, the display module automatically provides character spacing to the right of each character. The TEXT command has command modifiers for size and rotation information. New size and rotation information is controlled by the status of data word bit D8. To initiate new character attributes, bit D8 must be set high as a new information indicator. If this data bit is low, the size and rotation bits are ignored.

There are four character sizes. These four sizes are defined by the status of bits D11 and D12. The amount of space needed to draw the characters is shown in Figure 5-10 (this is the required space needed out of 2048 x 2048 possible points). The number of characters that can be drawn across the screen at the different sizes is shown in Figure 5-11. An example of 1x character spacing is shown in Figure 5-12.

Text Command:

MSB D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	LSB D0
1	0	S_1	S_0	R_1	R_0	Es	C_7	C_6	C_5	C_4	C_3	C_2	C_1	C_0
							CHARACTER →							

Command Modifiers:

For C0-C7, see Figure 5-13

a. ES Establish Size of Character

0 = Use previous size and rotation
1 = Establish new size and rotation according to S_1, S_0, R_1 and R_0

b. Rotate Character counter-clockwise

R_1	R_0	Rotation
0	0	0 degrees
0	1	90 degrees
1	0	180 degrees
1	1	270 degrees

c. Character Size

S_1	S_0	Size	W X H (in addressable points)
0	0	1X	24 X 36
0	1	1.5X	36 X 54
1	0	2X	48 X 72
1	1	2.5X	60 X 90

Figure 5-10. Text Command Bit Pattern

FOUR PROGRAMMABLE CHARACTER SIZES:

1.0 X 56 characters per line, 29 horizontal lines possible.
1.5 X 37 characters per line, 19 horizontal lines possible.
2.0 X 28 characters per line, 14 horizontal lines possible.
2.5 X 22 characters per line, 11 horizontal lines possible.

Figure 5-11. Character Display Capabilities

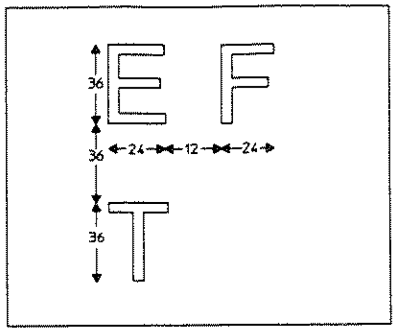


Figure 5-12. Example of 1x Character Spacing

The starting position of each character is the lower left corner of the defined character cell. After drawing a character, the display module advances to the starting point of the next character (like a typewriter). The display module also contains many special characters for graphics and display annotation. Figure 5-13 contains the modified ASCII character set in HEX format.

MODIFIED ASCII CODE CONVERSION TABLE

LEAST SIGNIFICANT CHARACTER	MOST SIGNIFICANT CHARACTER 0	1	2	3	4	5	6	7
0		center *	SP	0	@	P	'	p
1	HP logo	centered o	!	1	A	Q	a	q
2	β	↑	"	2	B	R	b	r
3		←	#	3	C	S	c	s
4	upper-half tic	↓	$	4	D	T	d	t
5	lower-half tic	→	%	5	E	U	e	u
6	left-half tic	√	&	6	F	V	f	v
7	right-half tic	π	'	7	G	W	g	w
8	back space	Δ	(	8	H	X	h	x
9	1/2 shift down	μ	)	9	I	Y	i	y
A	line feed	°(degree)	*	:	J	Z	j	z
B	inv. line feed	Ω	+	;	K	[	k	{
C	1/2 shift up	ρ	,	<	L	\	l	|
D	carriage return	Γ	-	=	M	]	m	}
E	horizontal tic	θ	.	>	N	∧	n	□
F	vertical tic	λ	/	?	O	_	o	▶

EXAMPLES:

HP logo	=	01
A	=	41
i	=	69
√	=	16
▶	=	7F
line feed	=	09

Figure 5-13. Modified ASCII Character Set

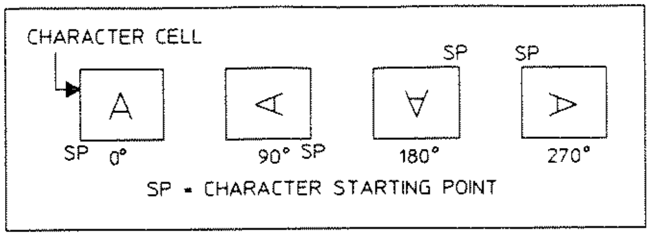


Figure 5-14. Character Rotation

The display module can be programmed to rotate any character 0, 90, 180, or 270 degrees rotation, measured counter clockwise from horizontal. This can be done for any character, at any size. The starting point of the character is always the lower left corner relative to any rotation. For character rotation, the entire character area is rotated the specified number of degrees. The starting point moves counter-clockwise. For example, the starting point of a character rotated 180 degrees would be the upper-right corner. This technique is shown in Figure 5-14.

Since the starting point of the character changes with rotation, so does the direction of character spacing. If the rotation is 180 degrees, the characters are written upside down from right to left. If the rotation mode is 270 degrees, the characters advance from top to bottom. Rotation spacing examples are shown in Figure 5-15.

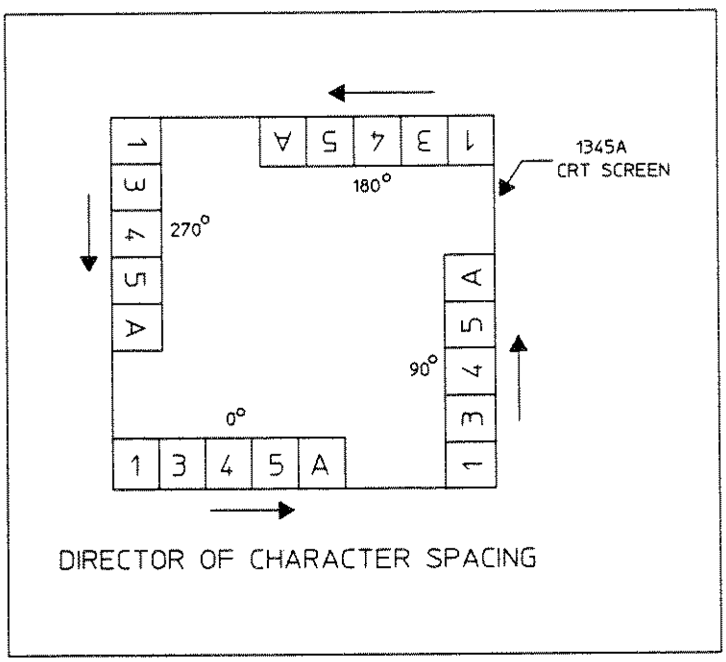


Figure 5-15. Character Rotation Spacing

There are a few constraints that you should know about. Some characters cannot be written within certain distances of certain CRT screen boundaries. These characters are listed in Figure 5-16. The characters are referenced to the screen boundary at which the limitation occurs.

It is important to observe the recommended character boundary specifications, to avoid problems that may be encountered by writing at the screen edges. Figure 5-17 shows recommended limits for each character size at each screen edge. Failure to observe these limits may result in undefined results, particularly when writing characters listed in Figure 5-16. You should plot all characters within these specified borders.

You should not attempt to write any character along a screen edge. The character spacing guidelines in Figure 5-17 allow ample spacing for characters of all specified sizes. Characters NOT specified in Figure 5-16 may be written closer to the screen borders, but this is not recommended.

BOUNDARY CHARACTERS

Left Boundary:	07 "right-half tic"; 08 "back space"; 0E "horizontal tic"; 0F "vertical tic"; 10 "centered *"; 11 "centered o"; 41 "A"; 57 "W"; 5F "_"; 77 "w"
Bottom Boundary:	02 "β"; 05 "lower-half tic"; 09 "1/2 shift down"; 0A "line feed"; 0F "vertical tic"; 10 "centered *"; 11 "centered o"; 19 "μ"; 1C "ρ"; 24 "$"; 28 "("; 29 ")"; 2C ","; 3B ";"; 51 "Q"; 5B "["; 5D "]"; 5F "_"; 67 "g"; 6A "j"; 70 "p"; 71 "q"; 79 "y"; 7B "{"; 7D "}"
Top Boundary:	01 "HP logo"; 0B "inv. line feed"; 0C "1/2 shift up"; 16 "√‾"; 1A "° (degree)"; 24 "$"; 28 "("; 29 ")"; 38 "8"; 5B "["; 5D "]"; 7B "{"; 7D "}"; 7E "□"
Right Boundary:	01 "HP logo"; 16 "√‾"; 41 "A"; 51 "Q"; 57 "W"; 61 "a"; 71 "q"; 77 "w"; 7E "□"

NOTE: HEX character equivalents appear in quotation marks.

Figure 5-16. Boundary Characters

Wrap Around

You should be aware of a phenomenon called "wrap around." If one or more vectors are drawn outside the vector drawing area, the display will draw vectors on opposite sides of the CRT. One part of the vector appears at one side of the screen, while the other part of the vector appears on the opposite side of the CRT. The picture appears distorted (visible vectors connecting ends of vectors). This can be corrected by plotting inside the vector drawing area.

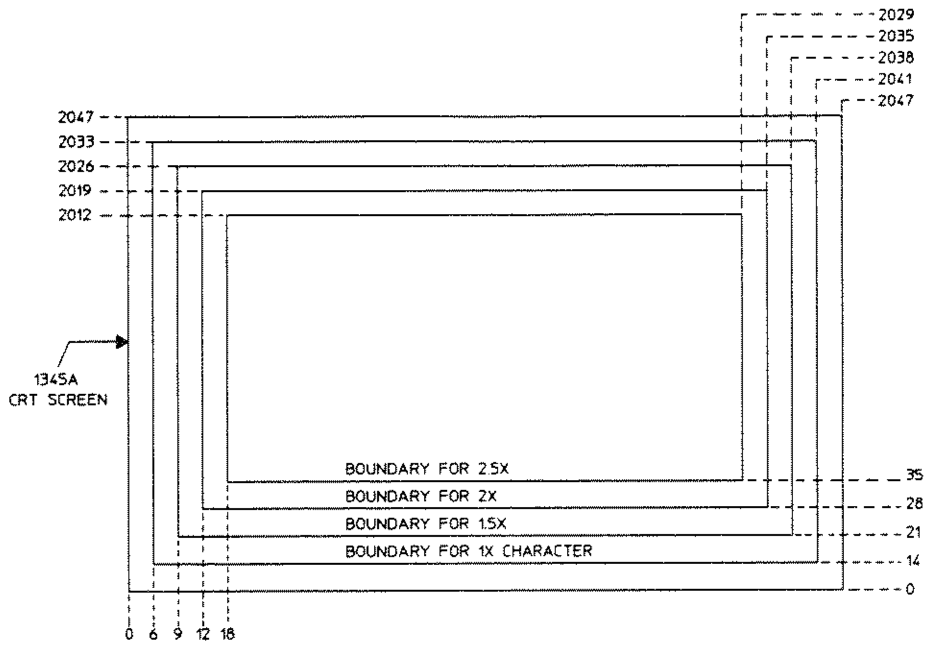


Figure 5-17. Character Borders

The following HP BASIC program demonstrates how text can be put up on the display. The characters used in the example are the same characters used in the examples in figure 5-13. In this example, codes used to generate the characters are calculated by adding the decimal values of the bits that are set high (1) to the decimal value of the ASCII encoded character.

```
60    ASSIGN @Ana TO 720              ! Analyzer address=20, select code=7
70    OUTPUT @Ana;"PAUS"
80    WAIT 1
90    INTEGER Array(1:9)
100   Array(1)=10                     ! plot at X location 10
110   Array(2)=2^12+1024              ! plot at Y location 1024
120   Array(3)=2^14+2^12+2^11+2^8 !   text character size set to large
130   Array(4)=2^14+1                 ! text ""HP logo""
140   Array(5)=2^14+9                 ! text ""line feed""
150   Array(6)=2^14+65                ! text "A"
160   Array(7)=2^14+105               ! text "i"
170   Array(8)=2^14+22                ! text ""SQRT""
180   Array(9)=2^14+127               ! text ""indicator""
190   OUTPUT @Ana;"DBSZ99,1,1"        ! create a Display Buffer of SiZe 99
200   OUTPUT @Ana;"DBAC1"             ! Display Buffer ACtivate
210   OUTPUT @Ana;"LUAS"              ! Load User buffer in ASCII
220   OUTPUT @Ana;"#I  9"             ! Send format specifier and length specifier
230   OUTPUT @Ana;Array(*)            ! send array
240   OUTPUT @Ana;"DBUP1"             ! put up display buffer
280   END
```

Display Programming with HP Basic

As explained at the beginning of this chapter, the third method of display programming is defining the analyzer's display as the plotter for HP BASIC graphics commands. The use of these commands is described in "BASIC Graphics Techniques" manual for HP series 200/300 computers. The command used to specify the display is:

PLOTTER IS 720,"HPGL"

where: 20 is the analyzer's address, 7 is the interface

For example, the following BASIC statements plot a box and some large text on the display:

```
11      PLOTTER IS 720,"HPGL"
12      OUTPUT 720;"DBSZ250,1"
13      OUTPUT 720;"DBAC1"
14      VIEWPORT 0,88,5,99
15      WINDOW   0, 130,130,-100,100
16      FRAME
17      LORG 5
18      CSIZE17
19      MOVE 66,0
20      LABEL"BIG TEXT"
21      OUTPUT 720;"DBUP1"
```

This example also demonstrates the ability of this technique to draw larger text than is possible with the HP-GL technique.

Dumping Display Buffers

Every display buffer in the HP 3563A, including both user buffers and the analyzer's own internal buffers, can be dumped via HP-IB. The internal buffers hold data traces, marker readouts, etc. This section shows you how to select the buffer to be dumped, describes the internal display buffers, and shows how to dump the selected buffer.

Dumping buffers takes two steps. First, use the vector block pointer (VBLK) to identify the buffer to be dumped. Second, select the data format in which you want the data dumped and then send the appropriate command. Buffers can be dumped in ASCII, ANSI floating point, and the internal binary formats. (For general information on these formats, see Chapter 3.)

Display buffers contain display module binary commands (see "Direct Binary Programming" earlier in this chapter). When HP-GL or BASIC commands are loaded, they are converted to display module commands by the analyzer. Because of this conversion, you can program a display initially with HP-GL or BASIC, load it into the analyzer, then dump out the direct binary equivalent. If you then store these binary commands, you can have the speed advantage of direct binary any time in the future when this display is needed.

The Vector Display Buffer Pointer (VBLK)

The buffer to be dumped is selected with the vector buffer pointer command (VBLK). The syntax is:

VBLKn

where: n is the buffer number

The number you specify with n depends on whether or not user buffers are being used. Table 5-1 shows the value of n to be used for dumping all user and internal display buffers. Note that to dump user buffers, their numbers are offset by +4 from the number used to identify them for other graphics commands.

Table 5-1. Identifying Buffer Pointer Values

Value of n (VBLKn)	User buffer	Internal Buffer
0	−4	Softkey underlining
1	−3	Softkey menu
2	−2	Command echo
3	−1	Message
4	0	Special markers, trace A
5	1	Special markers, trace B
6	2	X marker readout
7	3	Y marker readout
8	4	Trace A
9	5	Trace B
10	6	Grid
11	7	—
12	8	Ya readout
13	9	Yb readout
14	10	Xa readout
15	11	Xb readout
16	12	A label
17	13	B label
18	14	—
19	15	—

If any user buffer has been created, the user buffer corresponding to n is dumped. Otherwise, the internal buffer corresponding to n is dumped. For example, if you set up a user buffer with the DBSZ command then send VBLK10, you will get user buffer 6 if you send a dump command. However, if you had not created a user buffer and you sent VBLK10, you would get the internal grid buffer in response to a dump command.

Dumping Buffers in ASCII (DVAS)

The buffer identified with the vector buffer pointer (VBLK) can be dumped in ASCII format with the DVAS command. There is no header with this transfer, just #I and the length specifier. The following BASIC statements dump the buffer specified by VBLK:

```
25    Ana=720                    ! Analyzer address=20, select code=7
30    INTEGER Buffer(1:99)
35    OUTPUT Ana;"VBLK01"        ! set vector display buffer pointer (0-17)
40    OUTPUT Ana;"DVAS"          ! dump buffer values in ASCII
45    ENTER Ana ;Specifier$      !Enter format and length specifier
50    Length= VAL (Specifier$[3])
55
60    PRINT "LENGTH=";Length
65    REDIM Buffer(1:Length)
70    ENTER Ana;Buffer(*)        ! enter INTEGERs from buffer
```

The analyzer dumps the format specifier (#I) and the length specifier into "Specifier$" and the ASCII variables into integer array "Buffer."

Dumping Buffers in ANSI Floating Point (DVAN)

The buffer identified with the vector buffer pointer (VBLK) can be dumped in ANSI binary format with the DVAN command. There is no header with this transfer, just #A and the length specifier indicating the number of bytes to be transferred. The following BASIC statements dump the buffer specified by VBLK:

```
25    ASSIGN @Ana TO 720         ! Analyzer address=20, select code=7
30    REAL Buffer(1:99)
35    OUTPUT @Ana;"VBLK 1"       ! set vector display buffer pointer (0-17)
40    OUTPUT @Ana;"DVAN"         ! dump buffer values in ANSI
45    ENTER @Ana USING "%,2A,W";A$,Length! Enter format and length specifier
50                               !  %,2A- terminate A$ when two characters
55                               !        have been read
60                               !  %,W-  terminate "Length" when a word
65                               !        has been read
70    PRINT "LENGTH=";Length;" bytes  ";Length/8;" "elements"
75    REDIM Buffer(1:Length/8)   !
80    ASSIGN @Ana;FORMAT OFF     ! turn ASCII formatter off
85    ENTER @Ana;Buffer(*)       ! enter REAL data
90    ASSIGN @Ana;FORMAT ON      ! turn ASCII formatter on
```

The analyzer dumps the format specifier (#A) into A$ and the length specifier into "Length." The program then redimensions the array to Length/8 because there are eight bytes per element.

Dumping Buffers in Internal Binary (DVBN)

The display buffer identified with the vector buffer pointer (VBLK) can be dumped in the analyzer's internal binary format with the DVBN command. There is no header with this transfer, just #A and the length specifier indicating the number of bytes to be transferred. The following BASIC statements dump the internal buffer specified by VBLK:

```
25    ASSIGN @Ana TO 720          ! Analyzer address=20, select code=7
30    INTEGER Buffer(1:99)        !
35    OUTPUT @Ana;"VBLK 1"        ! set vector display buffer pointer (0-17)
40    OUTPUT @Ana;"DVBN"          ! DUMP buffer values in internal binary
45    ENTER @Ana USING "%,2A,W";A$,Length ! Enter format and length specifier
50    PRINT "LENGTH=";Length;" bytes ";Length/2;"elements"
55    REDIM Buffer(1:Length/2)    ! redimension array
60    ASSIGN @Ana;FORMAT OFF      ! turn ASCII formatter off
65    ENTER @Ana;Buffer(*)        ! enter INTEGER data
70    ASSIGN @Ana;FORMAT ON       ! turn ASCII formatter on
```

The analyzer dumps the format specifier (#A) into A$, and the length specifier into "Length," then redimensions the array to Length/2 because there are 2 bytes per array element.

6

Command/Communication Group

About This Chapter

This chapter explains the bus-only control and communication commands. The topics covered are:

Instrument Status & Service Requests	The status byte The instrument status register The activity status register Service requests Labeling user SRQs Power-on SRQ
Interactive Swept sine	Reading sweep points
General Status information	Ready status Source fault status Reference locked status Measurement done status Missed sample status Trace limits Overflow status Identify query Revision query Serial number query Setup state transfer HP-IB trigger enable
Control of the bus	Passing control
Reading Error Codes	Error code query
Reading Marker Values	X marker Individual special markers Grouped special markers
Communicating with the Front Panel	Key presses Reading Entry knob movement Reading Markers knob movement Writing to the message field Controlling display updating Reading auto carrier values Controlling HP logo for plotting

Most of the topics in this chapter are also discussed in condensed format in the "Quick Reference Guide" (Appendix A).

Table 6-1 shows a list of status checks and the registers or other query commands used to access the status information. The three main status registers are: the status byte, the instrument status (IS) register, and the activity status (AS) register.

Table 6-1. Summary of Status Checks in the HP 3563A

Condition/Event	Where/How to read it			
	Status Byte	IS	AS	Command
Requested service	*			
Error generated	*			ERR?
Ready for HP-IB commands	*			
User SRQs	*			
End of disc action	*			
End of plot action	*			
Power up	*			
Key pressed	*			KEY?
Various plotter & disc requests	*			
Instrument status change	*			IS?
Measurement pause		*		
Auto sequence pause		*		
End of measurement, capture or throughput		*		SMSD
Sweep point ready		*		SSWP
Channel 1 over range		*		SOV1
Channel 2 over range		*		SOV2
Channel 1 half range		*		
Channel 2 half range		*		
Source fault		*		SFLT
Reference locked		*		RLOK
Marker knob turned		*		
Entry knob turned		*		
Activity status change		*		AS?
Power on test failure		*		
Fault log check			*	
Filling time record			*	
Filters settling			*	
Curve fit in progress			*	
Missed external sample			*	SMSP
Paused for Timed preview			*	
Paused for Manual preview			*	
Waiting for trigger			*	
Waiting for arm			*	
Ramping source			*	
Diagnostic in progress			*	
Marker calc in progress			*	
Identify				ID?
Revision				REV?
Send analog setup state				SET?

The Status Byte

The status byte contains information about the analyzer's interaction with the HP-IB. The status byte is read by serial polling the HP 3563A. Refer to "Bus-management commands" (chapter 1) for more information on the serial poll. Five of the bits in the status byte are encoded and indicate one of 32 status conditions. The other three bits indicate individual status conditions. Table 6-2 shows the contents of the status byte. Table 6-3 shows the conditions represented by the encoded bits.

The status byte can be used to cause the analyzer to activate the service request (SRQ) bus management line. See "Service Requests" (later in this chapter) for information on disabling conditions indicated by the status byte.

Table 6-2. The HP 3563A's Status Byte

Bit	Value	Description
7	128	see Table 6-3
6	64	RQS (HP 3563A requested service)
5	32	ERR (HP-IB error)
4	16	RDY (ready to accept HP-IB)
3	8	see Table 6-3
2	4	see Table 6-3
1	2	see Table 6-3
0	1	see Table 6-3

RDY (bit 4) is set when the instrument is ready to receive commands over the bus. This occurs when the command buffer is empty. The HP-IB command buffer has a capacity of three 80-byte command lines. A byte represents one character. A line is terminated by a line-feed or activation of the EOI (End Or Identify) bus management line (carriage returns are ignored).

ERR (bit 5) is set when the instrument encounters an error condition. It is cleared only when the error register is read by the controller with the ERR? query command. Refer to "Error Codes" (later in this section) to decode the number returned with ERR?.

RQS (bit 6) is set when the analyzer activates the SRQ bus management line. It is cleared when the status byte is read.

The following HP BASIC program reads the status byte, extracts the value of the encoded bits using the HP BASIC binary AND command, and extracts the value of the three non-encoded bits using the BIT command. It then prints the results.

```
20     Status_byte=SPOLL(720)             !  Read Status byte.
25     !
30     Decoded_val=BINAND(Status_byte,143) !AND with 10001111
35     PRINT "Val of encoded bits(0-3,7)   = ";Deded_val
40     PRINT "Ready for HP-IB commands(4)  = ";BIT(Status_byte,4);" 1(TRUE)"
45     PRINT "HP-IB error(5)               = ";BIT(Status_byte,5);" 0(FALSE)"
50     PRINT "Request service (6)          = ";BIT(Status_byte,6)
55     END
```

Table 6-3 shows the condition codes represented by bits 7, 3, 2, 1 and 0 in the status byte.

Table 6-3. Status Byte Condition Codes

Status BIT 7 3 2 1 0	Status Byte Value	Description
0 0 0 0 0	0	No service requested
0 0 0 0 1	1	User SRQ #1
0 0 0 1 0	2	User SRQ #2
0 0 0 1 1	3	User SRQ #3
0 0 1 0 0	4	User SRQ #4
0 0 1 0 1	5	User SRQ #5
0 0 1 1 0	6	User SRQ #6
0 0 1 1 1	7	User SRQ #7
0 1 0 0 0	8	User SRQ #8
0 1 0 0 1	9	End of disc action SRQ
0 1 0 1 0	10	End of plot action SRQ
0 1 0 1 1	11	Instrument status change
0 1 1 0 0	12	Power on SRQ
0 1 1 0 1	13	Key pressed
0 1 1 1 0	14	Device Clear Plotter
0 1 1 1 1	15	Unaddress Bus, Listen HP 3563A
1 0 0 0 0	128	Talk plotter, Listen HP 3563A
1 0 0 0 1	129	Talk disc execution, Listen HP 3563A
1 0 0 1 0	130	Talk disc report, Listen HP 3563A
1 0 0 1 1	131	Talk Amigo disc command, Listen HP 3563A
1 0 1 0 0	132	Talk Amigo disc data, Listen HP 3563A
1 0 1 0 1	133	Talk Amigo short status, Listen HP 3563A
1 0 1 1 0	134	Talk disc identify, Listen HP 3563A
1 0 1 1 1	135	Talk Amigo parallel poll, Listen HP 3563A
1 1 0 0 0	136	Listen Plotter, Talk HP 3563A
1 1 0 0 1	137	Listen disc command, Talk HP 3563A
1 1 0 1 0	138	Listen disc execution, Talk HP 3563A
1 1 0 1 1	139	Listen Amigo disc command, Talk HP 3563A
1 1 1 0 0	140	Listen Amigo disc data, Talk HP 3563A
1 1 1 0 1	141	Listen Amigo disc read, Talk HP 3563A
1 1 1 1 0	142	Listen Amigo disc write, Talk HP 3563A
1 1 1 1 1	143	Listen Amigo disc format, Talk HP 3563A

Condition 0 indicates that no service was requested. Conditions 1-8 are the eight USER SRQ softkeys (see "User SRQs" later in this chapter). Condition 9 indicates that disc action under the analyzer's control is finished; 10 shows the same thing for a plotter. Condition 11 is the "window" into the instrument status register; any change in an unmasked bit of the instrument status register sets this condition. Condition 12 is set if the PwrSRQ ON OFF softkey (in the SPCL FCTN menu) is ON and power is applied to the analyzer. Condition 13 is set if key-code monitoring is enabled and a key on the analyzer's front panel is pressed. Conditions 14, 15 and 128-143 are provided for controllers incapable of passing control (see "Passing Control," later in this chapter).

The status byte can indicate up to three conditions simultaneously:

- Occurrence of an error with ERR (bit 5)
- Readiness to accept more commands with RDY (bit 4)
- One of 32 encoded conditions (bits 7,3,2,1,0)

For status conditions 1 through 13, the analyzer "remembers" more than just the first status condition returned. The status byte is loaded with the first condition that occurs. Any conditions occurring after that are stored and sorted by numerical order. The one with the lowest value is loaded into the status byte when the first condition is read.

For example, assume that the power on SRQ is on. The analyzer is powered on, keycode monitoring is enabled (and one or more keys are pressed), keycode monitoring is disabled, and all the user SRQ softkeys are pressed in random order. Then the status byte is read 10 times. The first time the status byte is read, the analyzer returns condition code 12, power on SRQ (which occurred first). The second through the ninth status byte returned indicates condition 1 through 8, and last is condition code 13, key pressed. If a new status condition occurs before all the buffered status conditions are returned, the analyzer returns the status condition that was to be returned next. It then returns the rest of the conditions in numerical order.

When the status byte is used to determine when a condition is true, it is important to read the status byte until all conditions are cleared. The following HP BASIC program lines will clear all conditions:

```
55    REPEAT                       ! read the status byte until it is clear
60    Status_byte=SPOLL(720)
65    PRINT "status byte = ";Status_byte
70    IF BIT(Status_byte,5) THEN ! check for HP-IB error
75     OUTPUT 720;"ERR?"          ! read/clear the error register
80     ENTER 720;Err
85     PRINT "analyzer error = ";Err
90    END IF
95    UNTIL BIT(Status_byte,6)=0 ! until not RQS (require service)
```

Status Registers

The Instrument Status Register

The instrument status register shows various analyzer conditions that have occurred since the register was last cleared. The instrument status register is cleared when it is read. The contents of the instrument status register can be read/cleared by sending the IS? command. When using the instrument status register to determine when a task is complete, it is important to clear the register before starting the task. For example, when checking for the completion of a measurement, clear the register just before starting the measurement. Table 6-5 shows the contents of the instrument status register.

Table 6-5. Instrument Status Register

Bit	value	Condition/Event
0	1	Measurement pause
1	2	Autosequence pause
2	4	End of measurement, capture or throughput
3	8	End of autosequence
4	16	Sweep point ready
5	32	Channel 1 over range
6	64	Channel 2 over range
7	128	Channel 1 half range
8	256	Channel 2 half range
9	512	Source fault
10	1024	Reference unlocked
11	2048	Remote marker knob turn
12	4096	Remote entry knob turn
13	8192	Activity status register change
14	16384	Power-on test failed

Each bit in the instrument status register represents a single condition or event: Bit 0 is set when the measurement has been paused, either from the front panel or over the HP-IB bus. Bit 1 is set when an autosequence has been paused. Bit 2 is set when a measurement, capture, or throughput ends (for non-averaged measurements, this is only at the end of the first measurement; for averaged measurements, this is at the end of the last average).

Bit 3 is set when an autosequence is finished. Bit 4 is set when the analyzer is in the swept sine mode, sweep point ready service request is enabled (ESWQ), and a sweep point is ready. Bits 5 through 8 indicate that an over or half range condition has occurred. (For a non-averaged measurement, a bit is set when the range condition occurs; for averaged measurement, a bit is set at the end of each average in which a range condition occurred. For a time capture measurement, a bit is set at the end of the entire capture in which a range condition occurred.)

Bit 9 is set if the analog source is forced above 12 volts peak. Bit 10 is set if the analyzer has lost phase lock with either the external reference input signal (rear panel) or its own internal reference. Bit 11 is set if the remote marker knob is enabled (RMKE) and the remote marker knob has been turned. This bit can not be set again until the remote marker knob value has been read using the RMKV? command or until the remote marker knob is re-enabled.

Bit 12 is set if the remote entry knob is enabled (RENE) and the remote entry knob has been moved. This bit can not be set again until the remote entry knob value has been read using the RENV? command or until the remote entry knob is re-enabled. Bit 13 is set if there is any change in an "unmasked" bit of the activity status register. (See "Masking the Activity Status Register" for more information). Bit 14 is set if the power-on self test fails. Bit 15 is not used.

The instrument status register can cause the analyzer to activate the service request (SRQ) bus management line. See "Service Requests" for more information.

The Status Query (STA?)

The status query command (STA?) provides information from both the status byte and the instrument status register. Sending STA? causes the HP 3563A to return the 16-bit word shown in Table 6-6. Note that STA? does not clear the information shown in these bits.

Table 6-6. The STA? Word

Bit	Value	Condition/Event
0	1	Not used
1	2	Not used
2	4	Key pressed
3	8	Not used
4	16	RDY (ready to accept HP-IB)
5	32	ERR (HP-IB ERROR)
6	64	RQS (Request service)
7	128	Message on screen
8	256	Measurement pause
9	512	Auto sequence pause
10	1024	End of measurement
11	2048	End of auto sequence
12	4096	Sweep point ready
13	8192	Channel 1 over range
14	16384	Channel 2 over range
15	32768	Math overflow

The only unique information provided by STA? is the message on screen (bit 7) and the math overflow (bit 15) indicator. Message on screen is set when a message is displayed in the message field on the screen. This field is the second line from the bottom on the right side. Messages appear in half-bright upper and lower case. To read the message, send the display message query command (DSP?). This returns up to 24 characters (see "Writing/Reading the Message Field" later in this chapter). Math overflow indicates a math overflow error has occurred. Here is a sample listing:

```
OUTPUT 720;"STA?"
ENTER 720;Status
```

The Activity Status Register

The activity status register indicates several aspects of the HP 3563A's current activity. Unlike the status byte and instrument status register, reading the activity status register does not clear it. The contents of the register can be read by sending the AS? command:

```
OUTPUT 720;"AS?"
ENTER 720;As
```

Table 6-7 shows the contents of the activity status register.

Table 6-7. Activity Status Register Bit

Bit	Value	Condition/Event
0	1	Check fault log
1	2	Filling time record
2	4	Filters settling
3	8	Curve fit in progress
4	16	Missed sample
5	32	Paused for timed preview
6	64	Paused for manual preview
7	128	Waiting for trigger
8	256	Waiting for arm
9	512	Not used
10	1024	Ramping source
11	2048	Diagnostic in progress
12	4096	Marker calc in progress

Each bit in the activity status register indicates a single condition. Bit 0 indicates that a system error in the HP 3563A has been entered in the fault log. The fault log is intended for use by trained service people only; refer to the HP 3563A Service Manual for details. Bit 1 indicates the time record is being filled—this becomes more noticeable as the frequency span decreases. Bit 3 indicates that a curve fit is in progress. Bit 4 indicates that a sample was missed while in external sampling mode because the external sample frequency was too high.

Bit 5 indicates that the analyzer is paused for a "timed preview" of a time record while in the linear resolution mode with average on. Bit 6 indicates that the analyzer is paused for a "manual preview" of a time record while in linear resolution mode with average on. Bit 7 indicates that the analyzer is waiting for a trigger signal. Bit 8 indicates that the analyzer is waiting for manual arming. Bit 9 is not used.

Bit 10 indicates that the source is being ramped. Bit 11 indicates that a service diagnostic is in progress. Bit 12 indicates that a special marker calculation is in progress. Bits 13 through 15 are not used.

The activity status register can be used in one of three ways:

- Check its contents after assigning a task to the analyzer.
- Check bit 13 (activity status register change) of the instrument status register after assigning a task to the analyzer. Bit 13 of the instrument status register is set if any "unmasked" condition in the activity status register is true. See "Masking the Activity Status Register" for information on setting up the activity status register masks.
- The activity status register can cause the analyzer to activate the service request (SRQ) bus-management line by unmasking bit 13 of the instrument status register and unmasking a bit in the activity status register. See "Service Requests" for more information.

Service Requests

The analyzer activates the service request (SRQ) bus-management line to gain the attention of the system controller. When the HP 3563A issues an SRQ, it also sets the request service (RQS) bit in the status byte. The controller can then read the status byte of all instruments on the bus and, by checking the RQS bit, determine the instrument that activated the SRQ line.

The HP 3563A activates the SRQ line due to true conditions in the status byte, the instrument status register, or the activity status register. Only conditions in the status byte can directly cause an SRQ. Conditions in the instrument status register can indirectly cause an SRQ by setting condition code 11 (instrument status register change) in the status byte. Conditions in the activity status register can indirectly cause an SRQ by setting bit 13 (activity status register change) in the instrument status register, which in turn, sets condition code 11 in the status byte.

Service requests (SRQs) can be controlled by disabling conditions in the status byte, or by masking bits in the instrument and activity status registers.

Disabling Conditions in the Status Byte to Prevent SRQs

When a condition is disabled, it will not cause an SRQ. In some cases the condition code representing the condition will be set, but the analyzer will not generate an SRQ. In other cases the condition code is simply prevented from being set. At power-on, all conditions that can be disabled except power-on SRQ are disabled. DEVICE CLEAR or RESET sets all conditions to the power-on state. Table 6-4 summarizes how to disable/enable conditions in the status byte.

Table 6-4. Disabling/Enabling Status Byte Conditions

Condition Code	Condition	How to Disable/Enable a Condition
1-8	User SRQs	Cannot be disabled
9-10	end of disc action end of plot action	These are the optional SRQ conditions which can be disabled by sending SRQD and enabled by sending SRQE. Disabling the optional service requests prevent the condition codes from being set.
11	Instrument status register change	Can not be disabled, but can be indirectly disabled by masking the Instrument status register. Refer to the following section.
12	Power on SRQ	Can be disabled by sending PSRQ0 and enabled by sending PSRQ1. Turning off "Power on SRQ" prevents the condition code from being set. Power on SRQ is recalled from nonvolatile memory when power is applied to the analyzer.
13	Keycode reporting	Disabled by sending KEYD and enabled by sending KEYE to the analyzer. Disabling the keycode reporting prevents the condition code from being set.
14-15		Cannot be disabled
16 (bit 4)	Ready for HP-IB (RDY)	Disabled by sending RDYD and enabled by sending RDYE to the analyzer. Disabling the RDY bit prevents generation of an SRQ caused by the set bit; it does not prevent the bit from being set or read.
32 (bit 5)	HP-IB error (ERR)	Disabled by sending ERRD and enabled by sending ERRE. Disabling the ERR bit prevents the generation of an SRQ caused by the set bit, it does not prevent the ERR bit from being set or read.
64 (bit 6)	Requires service (RQS)	Cannot cause an SRQ
128-143		Cannot be disabled

Masking the Instrument Status Register

The instrument status register mask is used to control SRQs caused by true conditions in the instrument status register. As mentioned before, true conditions in the instrument status register cause condition code 11 in the status byte to be set, which in turn causes the analyzer to generate an SRQ. Instrument status register bits are prevented from causing an SRQ (masked) by placing a 0 (low) in the corresponding bit position of the mask. Bits are allowed to cause an SRQ (unmasked) by placing a 1 (high) in the corresponding bit position of the mask. The instrument status register mask is set with the ISMn command, where n is the decimal value of the binary mask. For example, the HP BASIC statement

```
OUTPUT 720;"ISM20"
```

unmasks bit 2 (value = 4) and bit 4 (value = 16) and masks all other bits. Any unmasked bit (masked with a 1) in the instrument status register will cause an SRQ when the bit is set to 1. At power-on, device CLEAR, or RESET the instrument status mask is set to 0 (all bits masked). You can read the current value of the mask by sending the ISM? command:

```
OUTPUT 720;"ISM?"
ENTER 720;Is_mask
```

Masking a bit neither prevents the bit from being read, nor does it prevent the bit from being set.

Some of the conditions in the instrument status register can be "disabled" or "enabled." When a condition is disabled, the bit that indicates the condition will not be set even if the condition is true. When a condition is enabled, the bit indicating the condition can be set. The following table list the conditions and how they can be disabled/enabled.

Bit	Condition	How to disable/enable the condition.
4	Sweep point ready	Disabled by sending DSWQ (disable sweep SRQ) and enabled by sending ESWQ (enable sweep SRQ).
11	Remote marker knob turn	Disabled by sending RMKD (remote marker knob disable) and enabled by sending RMKE (remote marker knob enable).
12	Remote entry knob turn	Disabled by sending REND (remote entry knob disable) and enabled by sending RENE (remote entry knob enable).

Note

A condition that is enabled must also be unmasked if it is to cause an SRQ.

Masking the Activity Status Register

When there is a change in an unmasked bit of the activity status register, bit 13 in the instrument status register is set. If bit 13 in the instrument status register is not masked then condition code 11 in the status byte will be set and the analyzer will activate the SRQ bus management line. The activity status masks are used to control the setting of bit 13 in the instrument status register. Because it is useful to know if an activity has started (as well as if it has completed), the activity status register is masked to detect the occurrence of a bit going high or going low. The command ASMHn sets up a mask, where n is the decimal value of the mask, for bits going high (0 to 1). The command ASMLn sets up a mask for bits going low (1 to 0). For example the BASIC statement

OUTPUT 720;"ASML18"

unmasks bit 1 (value = 2) and bit 4 (value = 16) going low, and masks all other bits. The current value of the activity status masks can be read by sending the commands ASML? and ASMH?. For example, the following HP BASIC program segment reads and displays the value of both masks:

```
15    OUTPUT 720;"ASML?"          ! mask of bits going low
20    ENTER 720;Asm_low
25    PRINT "Mask of bits going low = ";Asm_low
30    OUTPUT 720;"ASMH?"          ! mask of bits going high
35    ENTER 720;Asm_high
40    PRINT "Mask of bits going high= ";Asm_high
```

As an example of activity status masking, the following HP BASIC program sets up the activity status register mask so that bit 13 (activity status register change) of the instrument status register will be set when bit 1 (filling time record) goes low.

```
70    Ana=720                     ! Analyzer address = 20, interface code = 7
80    OUTPUT Ana;"RST"            ! reset status/activity register mask and
85                                !  put analyzer in lin. res. mode
90    OUTPUT Ana;"ASML2"          ! unmask bit 1 going low  filling time record
100   OUTPUT Ana;"FRS100HZ"       ! set up narrow band measurement.
110   OUTPUT Ana;"IS?"            ! clear instrument status register
120   ENTER Ana;Is                !
130   OUTPUT Ana;"STRT"           ! START measurement.
140   REPEAT                      !
150     DISP "waiting for time record to fill"
160     OUTPUT Ana;"IS?"          ! check the instrument status register.
170     ENTER Ana;Is
180   UNTIL BIT(Is,13)=1          ! Bit 13 set by activity status register.
200   DISP "TIME RECORD FULL   program complete"
210   CLEAR Ana                   ! reset masks
220   LOCAL Ana
230   END
```

Programming for Service Requests

You can write a program so that when an interrupt occurs (such as activation of the SRQ line), the program will branch to an interrupt service routine. This routine can then serial poll all instruments on the bus and check the RQS bit (#6) of each status byte to identify the instrument that requires service. When the instrument requesting service is identified, you can find the reason for the SRQ by decoding the status byte. The following HP BASIC program lines are a simplified example of an interrupt driven program. Press one of the USER SRQ softkeys found in the **HP-IB FCTN** menu to cause an SRQ.

```
50    ON INTR 7 GOTO Poll          ! Branch on interrupt
60    ENABLE INTR 7;2              ! Enable SRQ interrupts
70    LOCAL 720
80    LOOP
90      DISP "waiting for SRQ"
100   END LOOP                     ! Endless loop while waiting for SRQ
110 Poll:                          !
120   DISP
130   Status_byte=SPOLL(720)       ! Read and clear status byte
140   PRINT "status byte = ";Status_byte
150   END
```

Programs do not have to be interrupt-driven; every condition/event can be periodically checked. The scheme you use depends on your application.

Note

The status byte must be cleared before using it to generate an SRQ. See "The Status Byte" for a description of how to clear the status byte.

User SRQs

The HP 3563A offers a special class of interrupts called user SRQs. Under the **HP-IB FCTN** key, there is a softkey labeled USER SRQ. This softkey displays a menu containing the User SRQ1 through User SRQ8 softkeys. You can label each of these softkeys and individually detect the eight user SRQs. This feature has many potential uses—by utilizing the user SRQ softkeys, you can run the controller in the "background" while operating the analyzer from its front panel softkeys. You can also create an entire menu structure by redefining the USER SRQ menu with the controller program. These newly-created labels are saved in nonvolatile memory and are not affected by power-down or preset.

To label the USER SRQ softkeys, use the LBS1 through LBS8 commands. Labels can be one or two lines, with a maximum of six characters per line. The label must be enclosed in single quote marks. If two lines are labeled, they must be separated by a comma. For example, the BASIC statement:

```
OUTPUT 720;"LBS4'TWO, LINES'"
```

labels the User SRQ4 softkey as

```
TWO
LINES
```

Labels can contain letters, numbers, and any punctuation that does not affect command syntax. Lines with fewer than six characters are automatically centered. Refer to "The Status Byte" (earlier in this chapter) for handling the SRQs generated by user SRQs.

The following HP BASIC program labels and handles all eight user SRQ's:

```
60    Ana=720  !  select code 7, analyzer address = 20
70    !
80    OUTPUT Ana;"LBS1'Soft ,key 1'"      !  Label softkeys
90    OUTPUT Ana;"LBS2'Soft ,key 2'"
100   OUTPUT Ana;"LBS3'Soft ,key 3'"
110   OUTPUT Ana;"LBS4'Soft ,key 4'"
120   OUTPUT Ana;"LBS5'Soft ,key 5'"
130   OUTPUT Ana;"LBS6'Soft ,key 6'"
140   OUTPUT Ana;"LBS7'Soft ,key 7'"
150   OUTPUT Ana;"LBS8'Soft ,key 8'"
160   !
170   OUTPUT Ana;"USRQ"             ! Display User SRQ menu on HP3562
180   LOCAL Ana                     ! put analyzer in local
190   ON INTR 7 GOSUB Poll          ! Branch to subroutine on interrupt
200   ENABLE INTR 7;2               ! Enable SRQ interrupts
210   LOOP                          !
220     DISP "WAITING FOR KEY PUSH"
230   END LOOP                      ! Endless loop while waiting for SRQ
240 Poll:                           !
250   DISP "PROCESSING THE INTERRUPT"
260   Status_byte=SPOLL(Ana)        ! Read and clear status byte
270   Code=BINAND(Status_byte,15) ! AND status with 15 (00001111)
280                                 !   to pass bits 1-4
290   !
300   FOR X=1 TO 8
310      IF Code=X THEN PRINT  "Soft Key ";X
320   NEXT X
330   IF Code >8 THEN PRINT "status byte condition code = ";X
340   ENABLE INTR 7;2               ! re-enable SRQ interrupts
350   RETURN                        ! back to endless loop
360   END
```

The Power-on SRQ

By setting the PwrSRQ ON OFF softkey in the **SPCL FCTN** menu to ON, the HP 3563A activates the SRQ bus-management line during subsequent power-ups. The state of PwrSRQ ON OFF is saved in nonvolatile memory in the analyzer, so it is not affected by power-down or reset. The power on SRQ is detected as condition 12 in the status byte. See "The Status Byte" (earlier in the chapter) for information on decoding the status byte.

Reading Sweep Points (ESWQ,SSWP)

When the HP 3563A is measuring in the swept sine mode, each sweep point can be read as it becomes available by using the commands ESWQ (enable sweep point ready reporting) and SSWP (send sweep point data). The general procedure for reading the sweep point is:

1. Enable sweep point ready reporting (ESWQ).
2. Clear the instrument status register by reading it.
3. Start a swept sine measurement.
4. Check the instrument status register until bit 4 (sweep point ready) is set to a 1 (high).
5. Read the sweep point data (SSWP)
6. Go back to step 4 until bit 2 (measurement complete) of the instrument status register is set to a 1 (high).
7. Disable sweep point ready reporting (DSWQ).

The command SSWP instructs the analyzer to return five variables, in the following order:

1. Input power (Volts peak)^2
2. Output power (Volts peak)^2
3. Cross spectrum real part (Volts peak)^2
4. Cross spectrum imaginary part (Volts peak)^2
5. Frequency (Hz)

The following HP BASIC program reads sweep points:

```
40   Ana=720                      ! Analyzer address=20, interface=7
50   OUTPUT Ana;"ISMO"            ! mask all bits in the instrument status reg.
60   OUTPUT Ana;"ESWQ"            ! Enable sweep point ready status reporting
70   OUTPUT Ana;"SSIN"            ! put analyzer in swept sine meas. mode.
80   OUTPUT Ana;"IS?"             ! clear the instrument status register
90   ENTER Ana;Is                 !   before starting
100  OUTPUT Ana;"STRT"            ! start measurement
110  REPEAT
120    OUTPUT Ana;"IS?"           ! read/clear the instrument status register
130    ENTER Ana;Is
140    IF BIT(Is,4) THEN          ! if sweep point ready then
150     OUTPUT Ana;"SSWP"         !   ask analyzer to Send SWeep Point info
160     ENTER Ana;I_power,O_power,Xspec_real,Xspec_imag,Freq
170     PRINT I_power,O_power,Xspec_real,Xspec_imag,Freq
180    END IF
190  Meas_done=BIT(Is,2)          ! is measurement done
200  UNTIL Meas_done              ! repeat until the measurement is done
205  OUTPUT Ana;"DSWQ"            ! Disable sweep point ready reporting
210  END
```

General Status Information

Source Fault Status (SFLT)

The source fault status query (SFLT) returns a 1 if the analog source is being forced over 12 volts. A 0 is returned when the analog source level is in the normal operating range.

Reference Locked Status (RLOK)

The reference locked status command (RLOK) indicates whether or not the analyzer is locked to a reference. The reference could be the analyzer's own internal reference or the external reference signal (applied to the EXT REF IN rear panel connector). A 1 is returned if it is locked, a 0 if not. This command provides information similar to bit 10 in the instrument status register.

Measurement Done Status (SMSD)

The SMSD command asks the analyzer to send the measurement done status flag. This status flag is set to 0 at the beginning of a measurement. It is set to 1 when a measurement is complete or paused.

The response to the SMSD command depends on the type of measurement. For an averaged measurement, a 1 is returned after the last average is complete. For a non-averaged measurement, a 1 is returned when the measurement is paused. If you need to know when the first measurement of a series of non-averaged measurements is complete, use the instrument status register bit 2. For time capture measurements, a 1 is returned when the capture is complete. Unlike the instrument status register bit 2, this measurement done status information is not cleared when it is read and it does not have to cleared before it is used. To learn other differences between the measurement done status flag and bit 2 of the instrument status register, see "The Instrument Status Register."

Missed Sample Status (SMSP)

The missed sample status command (SMSP) is set if the analyzer missed a sample while in external sampling mode. This is caused by an external sample rate greater than 256 kHz.

Trace Limits (STRL)

The send trace limits command (STRL) returns two ASCII encoded integers representing the start and stop bin for calibrated data in the active trace. There are 2048 sampled points (bins) in a time record and there are 801 points (bins) in a power spectrum.

```
OUTPUT 720 ;"STRL"
ENTER 720 ;Start,End
```

Overflow Status (SOV1, SOV2)

These commands instruct the analyzer to send the overrange status flag for channel 1 (SOV1) and channel 2 (SOV2). The analyzer reports a non-overrange condition by returning a 0 and reports an overrange condition by returning a non-zero number. These status flags are only set when a measurement is in progress. The flags are set to 0 at the beginning of a measurement. When a flag is set to indicate an overrange condition, it is not reset to 0 until a new measurement is started. For averaged measurements, an overrange status flag is set at the end of the average in which the overrange condition occurred and remains set throughout the rest of the averages. For non-averaged measurements, an overrange flag is set when an overrange condition occurs. It is reset at the beginning of the next measurement. For time capture measurements, an overrange flag is set at the end of a capture during which an overrange occurred.

Unlike the instrument status register bit 5 and 6, the overrange status flags are not cleared when read, and do not have to be cleared before used.

Identify Query (ID?)

This query (ID?) is used to identify devices on the bus. The HP 3563A responds to ID? by returning the 7-character string "HP3563A".

Revision Query (REV?)

This query (REV?) causes the analyzer to return the revision code and the format of the software. For example, the following BASIC statements read the software revision and format code:

```
OUTPUT 720;"REV?"
ENTER 720; Software, Format
```

Serial Number Query (SER?)

This command is a partial implementation of the serial number query. The HP 3563A responds by returning a 10-character string: prefix (four numbers indicating the date of the analyzer's introduction), country of manufacture (A for USA), and five zeros. Individual instrument serial numbers are not provided (the five zeros are returned instead).

Setup State Transfer (SET, SET?)

The SET? command dumps the current analog input instrument setup state in the ANSI floating point format. To transfer the digital input setup data in ANSI format, use the DGAN and LGAN commands described in chapter 3.

The SET command loads a setup state that has been previously dumped with SET? back into the analyzer. SET? is interchangeable with the DSAN (Dump State in ANsi) command, and SET is interchangeable with the LSAN (Load State in ANsi) command. See Chapter 3 for information on using DSAN and LSAN.

HP-IB Trigger Enable (HPT)

The analyzer can be triggered via HP-IB. Select HP-IB triggering by sending the HPT command. Once HPT is sent, the analyzer can respond to the HP-IB bus management command "TRIGGER."

Passing Control

The HP 3563A can control the bus to control plotters, disc drives, and output command strings. The general procedure for passing control to the analyzer is:

1. Clear the analyzers status byte of all conditions that can cause activation of the service request line (SRQ).
2. Send the CTAD command (control address) to the analyzer so that it knows where to return control when it is finished.
3. If the analyzer is to return control to the computer after the completion of a disc/plotter action (status byte conditions code 9 and 10), enable SRQ generation for these conditions by sending the command SRQE.
4. Send a command (or press a key) that requires the analyzer to have control of the bus. The analyzer responds by setting a condition code in the status byte that causes the analyzer to activate the SRQ line.
5. Clear the analyzer's status byte.
6. Pass control to the analyzer. This is a controller dependent operation; HP BASIC provides the PASS CONTROL command for this purpose.
7. When the analyzer is finished, it passes control back to the computer. If SRQE has been sent, the analyzer will activate the SRQ line after control has been passed back.

If your controller is unable to pass control, use status byte condition codes 14, 15, and 128-143 to detect when each device on the bus needs to talk and listen. Then, explicitly address and unaddress each device as needed to complete the data transfer. Your controller's HP-IB documentation should explain the HP-IB secondary commands needed to do this.

A troubleshooting hint—if your controller grabs control of the bus before the HP 3563A is finished, see if some other device on the bus is sending an SRQ. Such an SRQ causes the controller to prematurely resume control of the bus.

The following HP BASIC program passes control of the bus to the analyzer. The computer then resumes control of the bus when the analyzer activates the SRQ line.

```
50    Ana=720                     ! Analyzer address = 20, interface = 7
60    CLEAR Ana                   ! Clear HP-IB buffer, reset masks
80    REPEAT
90      Status_byte=SPOLL(Ana)    ! Clear status byte of all SRQ conditions
100     DISP "Status byte = ";Status_byte
110   UNTIL BIT(Status_byte,6)=0
120   OUTPUT Ana;"CTAD21"         ! Controller address = 21
130   ON INTR 7 GOTO Got_srq      ! When SRQ received goto Got_srq
140   ENABLE INTR 7;2             ! Enable interrupts caused by SRQ
150   OUTPUT Ana;"SRQE"           ! Enable an SRQ to be generated
160                               ! by the analyzer when bus action is complete.
170   LOCAL Ana
180   DISP "Do something that requires the analyzer to have control of the
bus"
190 Idle1:GOTO Idle1              ! Wait for an SRQ caused by the analyzer
200                               !    needing the bus.
210 Got_srq:                      !
220   Status=SPOLL(Ana)           ! Clear status byte
240   DISP "Control passed to analyzer"
250   PASS CONTROL Ana            !       pass control
260   !
270   ON INTR 7 GOTO Done         ! wait until finished with bus
280   ENABLE INTR 7;2             ! enable SRQ interrupts
290   Idle2:GOTO Idle2            ! wait for an SRQ
300                               !
310 Done:                         !       finished with bus
320   Status=SPOLL(Ana)           ! clear status byte
340   DISP "Control passed back to computer"
350   LOCAL Ana                   ! put analyzer in LOCAL
360   END
```

Line 180 in the above program could be replaced with a command that requires the analyzer to have control of the bus—for example, start plot (STPL), save file (SAVF), and so forth.

Time-out Control

To enable time-out control, send TMOE. This causes the HP 3563A to abort bus activity if two conditions are met—the analyzer has control, and a device under its control does not respond to a command after five seconds.

To disable time-out, send TMOD. TMOE is the default.

Table 6-9. Error Codes

Code	Error	Code	Error
100	No Peak Avg in HIST Meas	157	Do manually for pole-res
101	No Peak Avg n CORR Meas	158	Can't cnvrt empty table
102	Freq Resp No 1 Ch Demod	159	Gain too large
103	Cross Corr No 1 Ch Demod	160	Gain too small
104	No fundamental	161	No from Src in mix ratio
105	X Marker Must Be Active	162	Auto ordr bounds invalid
106	Buffer Overflow	200	Not Active Softkey
107	No Coord Change Allowed	201	Unknown Mnemonic
108	Not In Frequency Domain	202	Line Too Long
109	No Data	203	Command Too Long
110	Measurement In Progress	204	Alpha Delimiter Expected
111	Trace Not Compatible	205	Not a Valid Terminator
112	Data Type incompatible	206	Extra Chars In Command
113	Data Blocks Incompatible	207	Function Inactive
114	Source Block Empty	300	Missing Input
115	User Display Not Enabled	301	Not Valid Units
116	No Active Display Buffer	302	Not A Valid Number
117	Recursive Call	303	Alpha Too Long
118	Not A Valid Auto Math	304	Number Too Long
119	Bad Setup State	305	Out Of Range
		306	Unable To Curve Fit
120	Bad Auto Sequence Table	307	Bad # Of Parameters
121	Bad Synth Table	308	Auto Carrier Selected
122	Bad Non-volatile State	309	ENTRY Not Enabled
123	Bad Data Block	350	Table Overflow
124	Bad Data Header		
125	Marker Not On	400	Not A Valid Block Length
126	No Valid Marker Units	401	Not A Valid Block Mode
127	No Capture Data	402	Not HP-IB Controller
128	No Thruput Data	403	HP-IB Timeout Abort
129	Thruput Data Too Long	500	Bad Plotter Data Read
130	Bad Curve Fit Table	600	Cannot Recall Thruput
131	Bad Capture	601	Not A Valid Catalog
132	Bad Thruput	602	Unformatted Disc
133	Not A Valid User Window	603	Catalog Full
134	Bad Primitive Block	604	Not A Valid Name
135	View Input Disabled	605	Not A Valid Display
136	Cannot Use Zoom Data	606	File Not Found
137	Already Running	607	Disc Full
138	May Be Inaccurate	608	Disc Reject
139	Cannot Be Complex	609	Recall Active Auto Seq
140	Bad Delete Freq Table	610	Unknown Disc Command Set
141	Loops Nested Too Deep	611	No Disc In Drive
142	Demod In Zoom Only	612	Disc Write Protected
143	Numeric Overflow	613	Disc Fault
144	Invalid: Nyquist/Nichols	614	Disc Transfer Error
145	Invalid: Log Data	615	No Spares Or Fault Areas
146	No Carrier	616	No Thruput File
147	No Peak Hold In Time Avg	617	Catalog Not In Memory
148	Calibration In Progress	618	File Size Not Specified
149	No Avg for Demod Hist	619	Select Capture To Recall
		620	Source = Destination
151	No Src Protct in Dig Src	621	Sector Size < >256 Bytes
152	No dig src in src protct	622	Not Valid Format Option
153	User 2 not valid for src	623	Not Valid For This Disc
154	No input trg on dig chan	624	Destination Too Small
155	Not valid in src protct		
156	No protct in more src		

Error Codes

Sending the error query command (ERR?) causes the analyzer to return the code of the most recent error. Reading the error status clears the error code. The error could have occurred during front-panel or HP-IB operation. Each error code has a corresponding description in table 6-9. Note that these are the same errors as those encountered in front-panel operation. For complete descriptions, (with suggested corrective actions), see the HP 3563A Operating Manual.

Reading Markers Values

The X marker and the special marker can be read only when they are active. This section explains the commands used for reading the markers and describes the data they provide. Note that the coordinates and units should be explicitly set so the values returned can be interpreted. The HP BASIC program at the end of this section reads all three marker values.

Reading the X Marker (RDMK)

The read marker command (RDMK) causes the analyzer to return the x-axis and y-axis values of the X marker. The x-axis value is the number after "X=" in the upper left corner of the analyzer's display. The y-axis value is the number after "Ya=" or "Yb=."

Reading the Special Marker (RSMO)

The read special marker once command (RSMO) causes the analyzer to return the value of the special marker function that was last activated (POWER, AVG VALUE, or FREQ & DAMP). This value is scaled in the current display coordinates and units.

Reading the Special Marker Group (RSMG)

The read special marker group command (RSMG) causes the analyzer to return the value of the active special marker function (SLOPE, HMNC POWER, THD or SBAND POWER). This value is scaled in the current units and coordinates.

Note

The marker values should only be read when the instrument is paused. If you need to read a coordinate value while a measurement is in progress, dump the coordinate transform block as described in Chapter 3.

The following HP BASIC program reads all three marker values

```
40    Ana=720                    ! Analyzer address=20, interface=7
50    CLEAR Ana                  !
60    OUTPUT Ana;"X"             ! activate X marker
70    OUTPUT Ana;"RDMK"          ! read marker command
80    ENTER Ana;X,Y              ! read x- & y-axis values
90    PRINT "X=";X,"Y=";Y        ! display the x and y
100   !
110   OUTPUT Ana;"PWR"           ! special power marker
130   OUTPUT Ana;"RSMO"          ! read special marker once
140   ENTER Ana;Power            ! read power value
150   PRINT "POWER= ";Power      ! display the power value
160   !
170   OUTPUT Ana;"SLP"           ! special slope marker
190   OUTPUT Ana;"RSMG"          ! read slope command
200   ENTER Ana;Slope            ! read slope value
210   PRINT "SLOPE= ";Slope      ! display the slope value
220   OUTPUT Ana;"XFOF"          ! X function off
250   END
```

Communicating with the Front Panel

The rest of this chapter shows you how to communicate with the analyzer's front panel by mimicking the hardkeys, eight softkeys, and two RPG knobs. The end of this section shows you how to write messages to the message field and control display updating.

Key Codes

Each hardkey and the eight generic softkeys are assigned a key code. You can use these codes in two ways: monitoring key presses by interpreting key codes, and simulating key presses by sending key codes to the analyzer.

There are four commands used with this feature. KEY? is a query that returns the key code of the last key pressed since power-up or reset (if KEYE has been sent previously). KEYn sends a key code to the analyzer, where n is the code from 1 to 70. There are two commands used to enable/disable the key pressed condition in the status byte—KEYD disables the condition and KEYE enables it.

Table 6-10 lists the HP 3563A's key codes. Note that the eight softkeys have unique codes, but individual softkey labels do not. The code of the last key pressed (since power-up or reset) is returned by the KEY? command. Key presses are simulated by sending the analyzer the KEYn command, where n is the code of the key to be simulated. The key buffer holds the last three key presses. COM? returns the key code and HP-IB command of the last key pressed (this is especially useful for detecting softkeys).

Table 6-10. Key Codes

Key Name	Code	Key Name	Code
NO KEY PRESSED	0		
ENGR UNITS	1	Softkey 4	36
INPUT CONFIG	2	Softkey 5	37
TRIG DELAY	3	Softkey 2	38
HP-IB FCTN	4	Softkey 1 (top)	39
DISC	5	Softkey 3	40
SELECT TRIG	6	5	41
CAL	7	6	42
RANGE	8	4	43
AVG	9	Softkey 7	44
SELECT MEAS	10	Softkey 6	45
WINDOW	11	1	46
LOCAL	12	3	47
PLOT	13	2	48
SOURCE	14	MARKER VALUE	49
FREQ	15	- (negative sign)	50
MEAS MODE	16	BACKSPACE	51
START	17	Softkey 8 (bottom)	52
SPCL FCTN	18	VIEW INPUT	53
PRESET	19	0	54
MATH	20	,(comma)	55
SYNTH	21	.(decimal point)	56
AUTO SEQ	22	A	57
PAUSE CONT	23	B	58
SAVE RECALL	24	A&B	59
Y	25	COORD	60
SPCL MARKER	26	MEAS DISP	61
HELP	27	ARM	62
AUTO MATH	28	SINGLE	63
CURVE FIT	29	UPPER LOWER	64
X OFF	30	STATE/TRACE	65
X	31	UNITS	66
Y OFF	32	FRONT BACK	67
8	33	SCALE	68
9	34	UP arrow	69
7	35	DOWN arrow	70

Accessing the Remote Knobs

The rotary pulse generator (RPG) knobs, on the front panel of the analyzer, can be set up to act as a remote interface between the computer and user. The remote marker enable command (RMKE) assigns the marker knob as a remote knob. To return the marker knob to normal use, send the remote marker knob disable command (RMKD). The remote entry knob enable command RENE assigns the entry knob as a remote knob. To return the entry knob to its normal use, send the remote entry knob disable command (REND).

Once the knobs are assigned (enabled) as remote knobs, their current value can be read by the computer. The value of the a knob is set to zero when it is first enabled. It can be varied negative to –32768 by turning the knob counter-clockwise and varied positive to 32767 by turning it clockwise. To read the value of the remote marker knob, send the command RMKV?. To read the value of the remote entry knob, send the command RENV?.

The value of the remote knobs can be set to a number by the computer. To set the value of the remote marker knob, send the command RMKVn; to set the value of the remote entry knob, send the command RENVn, where n is the value of the knob. The remote entry knob has variable acceleration. This is set with the remote entry knob speed command RENS. RENS 0,32767 specifies fixed acceleration, and RENS 1,32767 specifies variable acceleration. Each of the remote knobs set a bit in the instrument status register when turned (see "The Instrument Status Register" for more information).

The following HP BASIC program demonstrates the use of the remote knob commands:

```
40    Ana=720                   ! Analyzer address=20, interface=7
50    CLEAR Ana                 ! clear buffer and all masks
60    OUTPUT Ana;"ISM0          ! set instrument status mask, unmasking
70                              !   will allow condition to cause an SRQ
80    OUTPUT Ana;"RMKE"         ! enable the remote marker knob
90    OUTPUT Ana;"RENE"         ! enable  remote entry knob
100   OUTPUT Ana;"RMKV 0"       ! set remote marker knob value to 0
110   OUTPUT Ana;"RENV 0"       ! set remote entry knob value to 0
120   OUTPUT Ana;"RENS 0,32767" ! Set entry knob acceleration
130 Start_loop:                 !
140   OUTPUT Ana;"IS?"          ! Read/clear the instrument status register
150   ENTER Ana;Is
160   OUTPUT Ana;"RMKV?"        ! ask for current value of remote marker knob
170   ENTER Ana;Mark_knob
180   OUTPUT Ana;"RENV?"        ! ask for current value of remote entry knob
190   ENTER Ana;Ent_knob
200   CLEAR SCREEN
210   PRINT "  Remote marker knob turn ";BIT(Is,11),"""RMKV?"" ";Mark_knob
220   PRINT " Remote entry knob turn  ";BIT(Is,12),"""RENV?"" ";Ent_knob
230   GOTO Start_loop
240   END
```

Writing/Reading the Message Field

You can write messages up to 24 characters long to the display's message field. Use the DSP command and put the message string in single quotes. For example, the BASIC statement:

```
OUTPUT 720; "DSP'Test Message'"
```

displays "Test Message" (without quotes) in the message field. To read the message currently in the field, use the DSP? query, which returns an alphanumeric string up to 24 characters long. For example:

```
OUTPUT 720; "DSP?"
ENTER 720; Message$
```

reads the current message. When a measurement is started a "blank" message is displayed, which sets bit 7 of the STA? word.

Controlling Display Updating

Two commands are provided to enable/disable updating on the display. To disable updating, send the DSPD (display disable) command. To enable it, send DSPE (display enable). Note that once you send DSPD, updating is disabled until you re-enable it by sending DSPE or resetting the analyzer.

Reading Auto Carrier Values

The values calculated by the demodulation algorithm's auto carrier feature can be read via HP-IB. The command SACR (Send Auto Carrier) returns four values:

Auto carrier calculated for Channel 1
Auto carrier calculated for Channel 2
Phase offset removed from Channel 1
Phase offset removed from Channel 2

For example, the BASIC statements:

```
OUTPUT 720; "SACR"
ENTER 720; Carrier1, Carrier2, Phase1, Phase2
```

read the four values.

Controlling the HP Logo for Plotting

The HP logo that appears at the top of table display is not normally plotted, but you can specify it to be plotted if desired. Send the command "LOGO0" to disable it or "LOGO1" to enable it.

A

Quick Reference Guide

Introduction

This appendix provides condensed HP-IB programming information for the HP 3563A Control Systems Analyzer. It contains the following information in quick reference format:

- General command syntax
- Response to bus management commands
- Command mnemonics, including syntax, range & terminators

The mnemonic table provides a complete list of HP 3563A HP-IB programming commands (both front panel and bus-only), listed alphabetically by mnemonic.

This appendix is a reference for programmers familiar with both the HP 3563A and the computer/controller being used. For additional information about the bus-only commands, see chapters 1 through 6. For additional information about the front panel commands, refer to the HP 3563A *Operators Manual.*

General Command Syntax

The general syntax for sending commands to the HP 3563A is:

<mnem><opt sp><para><sep><para><opt sp><suff><term>

where:

<mnem> is the command mnemonic
<opt sp> is ignored optional space
<para> is first command-dependent parameter
<sep> is required comma (,) for multi-parameter commands
<para> is second command-dependent parameter
<opt sp> is ignored optional space
<suff> is command-dependent suffix
<term> is command terminator (semicolon)

For example, to set up a frequency span from 10 to 60 kHz, you would send the command:

FRS 10,60 KHZ;

where:

FRS is the mnemonic
10 is the first command-dependent parameter
, is the parameter separator
60 is the second command-dependent parameter
KHZ is the command-dependent suffix
; is the command terminator

Note that front-panel mnemonics usually emulate their respective hardkey or softkey. In some cases, suffixes (terminators, delimiters) are not required. The syntax required for every command is described in the mnemonic table. You should consult this table if you have a question about a particular command's syntax.

Parameter Queries

To query the current value of any variable parameter, send the appropriate mnemonic followed by a question mark. For example, to learn the current frequency span, send FRS?.

Response to Bus-Management Commands

Table 1 summarizes the HP 3563A's response to the HP-IB primary bus management commands.

Table 1. Response to Bus-Management Commands

Command	Response
ABORT I/O	Aborts data input or output and unaddresses the analyzer. Does not clear the HP-IB command buffer.
CLEAR LOCKOUT & SET LOCAL	Clears local lockout and returns to local control.
DEVICE CLEAR	Unconditionally interrupts bus activity: clears the HP-IB command buffer
LOCAL	Returns to local (front panel) control and aborts load operations in progress
LOCAL LOCKOUT	Disables the front panel LOCAL key
PARALLEL POLL	Does not respond.
PARALLEL POLL CONFIGURE	Does not respond.
PASS CONTROL	Accepts control if needed; passes control back when finished to address specified by the CTAD command. Immediately passes control back if it doesn't need control.
REMOTE	Forces the HP 3563A into REMOTE mode.
SERIAL POLL	Responds by sending its status byte.
TRIGGER	Accepts HP-IB triggering if it is first enabled by sending the analyzer the HPT command.

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
A	A		
	Activates trace A		
AB	A&B		
	Activates both traces A and B		
ABCP	ABORT CAPTUR		
	Aborts time capture		
ABIB	ABORT HP-IB		
	Aborts HP-IB activity		
ABTH	ABORT THRUPT		
	Aborts time throughput operation		
ABTR	A & B TRACES		
	Selects current A and B traces as input to the curve fitter		
ACFL'aaaaaaaa'	ACTIVE FILE		
	Defines name (up to 8 characters) of active file for throughput operation		
ACPT (Bus-only)	ACCEPT		
	HP-IB "yes" response		
ACRR	AUTO CRRIER		
	Calculates the carrier frequency for FM and PM modulation		
ADC	ADC		
	Displays the ADC internal diagnostics test menu		
ADDrrr	ADD	$10^{\pm 38}$	
	Adds numeric constant to active trace		
ADDssss	ADD		TRCA,TRCB SAV1,SAV2, 1SAV, 2SAV, 3SAV, 4SAV, 5SAV
	Adds Trace A, Trace B or measurement data stored in SAVED 1, SAVED 2, or SAVED # to the active trace		
ADDBn1,n2[,n3] (Bus-only)	ADD BLOCK		
	Adds block n1 to n2 and puts result in n2 or optional block n3		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
ADDCn1,n2[,n3] (Bus-only)	ADD CONSTANT		
	Adds constant n1 to real block n2 and puts result in n2 or optional block n3		
ADDL	ADD LINE		
	Adds lines to an autosequence or auto math table (all subsequent commands will be placed in the table)		
ADDVrr,rrsss	ADD VALUE	$10^{\pm 38}$	MHZ,HZ,KHZ
	Adds a line to a synthesis table		
ADDXn1,n2,n3[,n4] (Bus-only)	ADD COMPLEX		
	Adds complex constant n1,n2 (n1 is real part and n2 is imaginary part) to real block n3 and puts result in n3 or optional block n4		
ADLNrr,rrss	ADD LINE	$10^{\pm 38}$	MHZ,HZ,KHZ
	Adds a line to a curve fit table		
ADRGrr,rr,sss	ADD REGION	0– 100kHz	MHZ,HZ,KHZ
	Adds a delete frequency band to the demodulation table		
ADRS	ADDRES ONLY		
	Puts instrument in non-active controller mode		
AF	ADVANCE FEED		
	Loads a new sheet of paper on plotters so equipped		
AGONr	A GAIN ON OFF	0 or 1	
	Turns auto gain function in swept sine mode on (1) or off (0)		
AGSE	A GAIN SELECT		
	Displays the auto gain select menu		
ALTA	ALU TEST A		
	Runs an internal diagnostic test (see Service Manual)		
ALTB	ALU TEST B		
	Runs an internal diagnostic test (see Service Manual)		
AM1	AM CHAN1		
	Selects amplitude modulation of Channel 1 when in demodulation mode		
AM2	AM CHAN2		
	Selects amplitude modulation of Channel 2 when in demodulation mode		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
AMTH	AUTO MATH		
	Displays the auto math menu		
ANAPrr	ANNOT A PEN	see comment	
	Designates the first of two pen numbers to be used to label a plot (range depends on number of pens in plotter)		
ANBPrr	ANNOT B PEN	see comment	
	Designates the second of two pen numbers to be used to label a plot (range depends on number of pens in plotter)		
ANINn1,n2,c1,c2 (Bus-only)	ANALOG INPUT		
	Allows data to be taken from inputs for use in signal processing		
ARM	ARM		
	Triggers a measurement when trigger arming is in manual arm mode		
ARMAr	ARM AU MAN	0 or 1	
	Selects manual (0) or auto (1) trigger arming mode		
AS? (Bus-only)	ACTIVITY STATUS		
	Requests the contents of the Activity Status Register from the instrument		
ASEQ	AUTO SEQ		
	Displays the autosequence menu		
ASFN	ASEQ FCTN		
	Displays the autosequence function menu		
ASMHn (Bus-only)	ACTIVE ST MASK HIGH		
	Masks the Activity Status Register (unmasks bits equal to n as they change from low to high)		
ASMLn (Bus-only)	ACTIVE ST MASK LOW		
	Masks the Activity Status Register (unmasks bits equal to n as they change from high to low)		
ASMS'aaa..a'	ASEQ MESSGE		
	Allows a text message (1 to 24 characters) to be added to an autosequence		
ASQ1	START ASEQ 1		
	Invokes autosequence number 1		
ASQ2	START ASEQ 2		
	Invokes autosequence number 2		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
ASQ3	START ASEQ 3		
	Invokes autosequence number 3		
ASQ4	START ASEQ 4		
	Invokes autosequence number 4		
ASQ5	START ASEQ 5		
	Invokes autosequence number 5		
ATPT	AT POINTR		
	Enters the file name identified by the catalog pointer		
AU1	AUTO 1 UP&DWN		
	Allows Channel 1 to auto range up and down		
AU1U	AUTO 1 RNG UP		
	Allows Channel 1 to auto range up only		
AU2	AUTO 2 UP&DWN		
	Allows Channel 2 to auto range up and down		
AU2U	AUTO 2 RNG UP		
	Allows Channel 2 to auto range up only		
AUC1	AUTO CORR1		
	Selects the auto correlation display for Channel 1		
AUC2	AUTO CORR2		
	Selects the auto correlation display for Channel 2		
AUCR	AUTO CORR		
	Selects the auto correlation measurement		
AUIN	AUTO INTGRT		
	Selects auto integration when in swept sine measurement mode		
AUMT	AUTO MATH		
	Selects the display calculated in the auto math table		
AUOR	AUTO ORDER		
	Estimates optimum number of poles and zeros during curve fit		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
AUT1	AUTO 1 16 BIT		
	Specifies 16 bit dynamic range in the swept sine measurement mode for the Channel 1 input		
AUT2	AUTO 2 16 BIT		
	Specifies 16 bit dynamic range in the swept sine measurement mode for the Channel 2 input		
AUTOr	AUTO ON OFF	0 or 1	
	Turns the auto calibration feature on(1) or off(0)		
AUWT	AUTO WEIGHT		
	Generates a weighting function during curve fit		
AVGrrr	AVG	1 – 32767	
	Displays the averaging menu and allows number of averages to be specified		
AVGV	AVG VALUE		
	Shows average value of trace area in X marker band		
AVOF	AVG OFF		
	Turns all averaging functions off		
AVRG	AVRG		
	Selects the cumulative average of all time records acquired with the current measurement for the filtered input displays (time averaging must be on)		
B	B		
	Activates trace B		
BCRPrr	BURST CHIRP	1 – 99	
	Selects the burst chirp source output and sets burst percentage		
BEEPr	BEEPER ON OFF	0 or 1	
	Turns the internal warning beeper on (1) or off (0)		
BILN	BI- LINEAR		
	Converts current s-domain synthesis table into the z-domain using the bilinear transformation		
BIT8r	# BITS 8 16	0 OR 1	
	Selects digital input data bit quantity to be 16 bits(BIT80) or 8 bits(BIT81)		
BLSE	BLOCK SET		
	Runs an internal diagnostic test (see Service Manual)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
BLSZsize,n[,count] (Bus-only)	BLOCK SIZE		
	Allocates memory for signal processing operations (block size range is 1 to 37900,n is number of first block 0 to 15,count is number of blocks)		
BRITr (Bus-only)	BRIGHTNESS	0– 3	
	Specifies brightness of a trace in a user display (0=off, 1=dim, 2=med, 3=full)		
BRNDrr	BURST RANDOM	1– 99	
	Selects the burst random noise source output and sets burst percentage		
BSPC	BACK SPACE		
	Moves cursor back one space during alphanumeric entries		
BUS8r	BUS SZ 8 16	0 or 1	
	Selects digital input bus size to be 16 bits(BUS80) or 8 bits(BUS81)		
C1ACr	CHAN1 AC DC	0 or 1	
	Selects Channel 1 input coupling mode (0=dc,1=ac)		
C1ANr	CHAN 1 AN DIG	0 or 1	
	Configures Channel 1 for either digital data(C1AN0) or analog data(C1AN1)		
C1CL	CHAN 1 CLOCK		
	Selects the Channel 1 clock line to be the sample clock		
C1CN	CHAN 1 CONFIG		
	Displays a graphic representation of the current configuration for the inputs to Channel 1		
C1DL	CHAN 1 DELAY	see comment	USEC,MSEC SEC,MIN REVS,REC
	Specifies Channel 1 trigger delay interval (range depends on suffix)		
C1IN	CHAN 1 INPUT		
	Specifies trigger source as the Channel 1 input signal		
C1RGrrsss	CHAN 1 RANGE	see comment	V,MV,VRMS MVRM,DBV,EU
	Sets input range for Channel 1 (range depends on suffix)		
C2AC	CHAN2 AC DC		
	Selects Channel 2 input coupling mode (0=dc,1=ac)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
C2ANr	CHAN 2 AN DIG	0 or 1	
	Configures Channel 2 for either digital data(C2AN0) or analog data(C2AN1)		
C2CL	CHAN 2 CLOCK		
	Selects the Channel 2 clock line to be the sample clock		
C2CN	CHAN 2 CONFIG		
	Displays a graphic representation of the current configuration for the inputs to Channel 2		
C2DL	CHAN 2 DELAY	see comment	USEC,MSEC SEC,MIN REVS,REC
	Specifies Channel 2 trigger delay interval (range depends on suffix)		
C2IN	CHAN 2 INPUT		
	Specifies trigger source as the Channel 2 input signal		
C2RGrrsss	CHAN 2 RANGE	see comment	V,MV,VRMS MVRM,DBV,EU
	Sets input range for Channel 2 (range depends on suffix)		
CADA	CAPTUR DATA		
	Displays the Time Capture Data menu		
CAL	CAL		
	Displays the calibration menu		
CAOF	CALC OFF		
	Disables all harmonic marker calculations		
CAT	VIEW CATLOG		
	Catalogs and displays the contents of the disc		
CCONrr CCONrr,rr	CREATE CONST	$10^{\pm 38}$	
	Creates a constant for use in a waveform synthesis		
CCOR	CROSS CORR		
	Selects the cross correlation measurement		
CDF1	CDF 1		
	Selects cumulative density function for histogram measured on Channel 1		
CDF2	CDF 2		
	Selects cumulative density function for histogram measured on Channel 2		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND COMMENTS	RANGE	SUFFIXES
CEDGr	C EDGE + -	0 or 1	
	Selects active edge for the selected clock to be high-to-low(CEDG0) or low-to-high(CEDG1)		
CFrrsss	CENTER FREQ	see comment	MHZ,HZ,KHZ ORD,RMP
	Enables center of freq span entry (range limited to 100kHz-(10.24mHz/2))		
CFFTn1,n2 (Bus-only)	COMPLEX FFT		
	Performs FFT on complex block n1 and puts result in n2		
CFIT	CURVE FIT		
	Loads an autosequence to synthesize and fit a frequency response function using sample data		
CFT1n1,n2 (Bus-only)	COMPLEX FFT^{-1}		
	Performs inverse FFT on complex block n1 and puts result in n2		
CH1	CH 1 ACTIVE		
	Activates Channel 1 for the selected measurement		
CH12	CH 1&2 ACTIVE		
	Activates both Channels for the selected measurement		
CH2	CH 2 ACTIVE		
	Activates Channel 2 for the selected measurement		
CHED	CAPTUR HEADER		
	Displays the time capture header		
CHGL	CHANGE LINE		
	Allows changing of a line in an autosequence or auto math table		
CHGVrr,rrsss	CHANGE VALUE	$10^{\pm 38}$	MHZ,HZ,KHZ
	Allows changing of a value in a synthesis table		
CHRGrr,rrsss	CHANGE REGION	0– 100kHz	MHZ,HZ,KHZ
	Allows changing of a frequency region in the demodulation table		
CHROr (Bus-only)	CHARACTER ROTATION	0– 3	
	Defines rotation for display characters (0=0 degrees(default), 1=90 degrees, 2=180 degrees, 3=270 degrees)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
CHRP	CUSTOM CHIRP		
	Allows you to edit a time record to include a 1 V sine or sine chirp signal		
CHSZr (Bus-only)	CHARACTER SIZE	0–3	
	Defines size for display characters. (0=24x36 points(default), 1=36x54 points, 2=48x72 points, 3=60x90 points)		
CLAD	CALIBR ADJUST		
	Runs an internal diagnostic test (see Service Manual)		
CLAS	CLEAR ASEQ		
	Clears the displayed autosequence table		
CLBFr (Bus-only)	CLEAR BUFFER	–4–15	
	Clears the specified display buffer without activating it		
CLENrrsss	CAPTUR LENGTH	see comment	USEC,MSEC,SEC MIN,REVS,PNTS REC
	Set time capture length (range depends on suffix)		
CLF	CLEAR FAULT		
	Clears the fault log		
CLLG	CLEAR LOGS		
	Clears all disc logs		
CLMA	CLEAR MATH		
	Clears the auto math table		
CLOF	CALC OFF		
	Disables the sideband power calculations of the sideband markers		
CLRL	CLEAR LINE		
	Clears entry line during alpha entry operations		
CLRT	CLEAR TABLE		
	Clears the delete frequency demodulation table		
CLT	CLEAR TEST		
	Clears the test log		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
CLTA	CLEAR TABLE		
	Clears the curve fit table		
CLTB	CLEAR TABLE		
	Clears the displayed synthesis table		
CMPC	COMPLX CONJ		
	Calculates the complex conjugate of the active trace		
CMPDrrss	COMPUT DELAY	10± 38	USEC,MSEC,SEC
	Sets a computational delay to correct time lags associated with the computations in your system (delays greater than 1 sample may not be as accurate)		
CMPRr	COM- PRESS	2 – 512	
	Compresses the active trace into a segement and replicates the segment in successive segments to fill entire data block (range must be a power of 2)		
CMPT	COMAND POINTR		
	Runs an internal diagnostic test (see Service Manual)		
CNAS	CONT ASEQ		
	Continues a paused autosequence		
CNCA	CANCEL ALPHA		
	Cancels input when in alpha entry mode		
CNCL	CANCEL		
	Cancels numeric entry		
CNJBn1 [,n2] (Bus-only)	CONJUGATE BLOCK		
	Computes complex conjugate of data block n1 and puts result in n1 or optional block n2		
CNPK	CONT PEAK		
	Selects the continuous peak averaging function		
COFI	COPY FILES		
	Copies all files on the source disc to the destination disc		
COFI'aaaaaaaa'	COPY FILES		
	Copies a file on the source disc to the destination disc		
COFI;ATPT	COPY FILES		
	Copies file at catalog pointer		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
COFI'aaa,aaa'	COPY FILES		
	Copies contiguous files from the source disc to the destination disc		
COFI'<,aaaaa'	COPY FILES		
	Copies contiguous files starting with the first file in the catalog and ending with the specified file		
COFI'aaaaa,>'	COPY FILES		
	Copies contiguous files starting with the specified file and ending with the last file in the catalog		
COHR	COHER		
	Selects the coherence display		
COM? (Bus-only)	COM		
	Returns keycode and HPIB command of last key pressed		
COMD (Bus-only)	COMMAND DISABLE		
	Disables the command echo on the instrument's display screen		
COME (Bus-only)	COMMAND ENABLE		
	Enables the command echo on the instrument's display screen		
CONT	CONTINUE		
	Continues a paused measurement		
COPY	COPY SCREEN		
	Instructs the instrument to plot the current display screen to a plotter		
CORD	COORD		
	Displays the coordinates menu		
CPEKn1,n2,n3 (Bus-only)	CROSS SPEC PEAK HOLD		
	Computes cross spectrum of complex blocks n1 and n2 and compares magnitudes of the result with complex block n3 (larger values are put in n3)		
CPNTrrss	CAPTUR POINTR	see comment	USEC,MSEC,SEC MIN,REVS,PNTS REC
	Enables the Entry group to move the capture pointer (range depends on suffix)		
CPSE	CAPTUR SELECT		
	Displays the capture select menu		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND COMMENTS	RANGE	SUFFIXES
CPTR	TIME CAPTUR Selects the time capture measurement mode		
CQUL'rrrrrrr'	CLOCK QUALFR Selects bits on qualifier pod to qualify the clock (each bit can be 0, 1 or X(don't care))	0,1,0R X	
CRCR	CROSS CORR Selects cross correlation measurement mode (only available in Linear Res mode)		
CRFI	CREATE FIT Displays the create fit start/stop menu		
CRFRrrsss	CRRIER FREQ Enters the carrier frequency for the sideband marker	0– 100kHz	MHZ,HZ,KHZ RPM,ORD
CRFT	CREATE FIT Starts the curve fitting process using the current setup		
CRTH'aaaaaaaa'	CREATE THRUPT Enters a filename and creates a throughput file on disc		
CSPC	CROSS SPEC Selects the cross spectrum display		
CSPSn1,n2,n3 (Bus-only)	CROSS SPEC SUMMATION Computes the cross spectrum of complex floating point blocks n1 and n2 and adds it to complex floating point block n3 putting the results in n3		
CTADr (Bus-only)	CONTROLLER ADDR Tells the instrument the address of the controller	0– 31	
CTPG[r]	CUT PG ON OFF Enables (1) or disables (0) the page cutting feature on plotters equipped with this feature	0 or 1	
CTPTrr	CATLOG POINTR Enables the Entry group to move the catalog pointer	1– 20	
CTRA	CREATE TRACE Displays the synthesis create trace menu		
CTRC	CREATE TRACE Starts the synthesis process using the currently selected synthesis table		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
CVFT	CURVE FIT		
	Displays the curve fit menu		
CVTB	CONVRT TABLE		
	Displays the convert table menu which is used to convert synthesis tables from one type to another		
CVTS	CONVRT TO S		
	Displays menu of z-domain to s-domain conversion methods		
CVTZ	CONVRT TO Z		
	Displays menu of s-domain to z-domain conversion methods		
CXAVn1,n2,n3,awf (Bus-only)	CROSS SPEC EXPON AVG		
	Exponentially averages cross spectrum of complex blocks n1 and n2 with complex block n3 and puts result into n3 (awf=average weighting factor)		
D (Bus-only)	D		
	Indicates an exponent in scientific notation (same as "E" and "L")		
DAAN	DATA & ANNOT		
	Selects both data and alphanumeric annotation to be plotted		
DACL	DATA CLOCK		
	Displays the data clock menu		
DAEDrr[,rr]	DATA EDIT	± 10 E 38	Same as display
	Specifies a new value for the trace at or between the marker(s)		
DASZ	DATA SIZE		
	Displays the data size menu		
DATA	DATA ONLY		
	Selects data trace(s) only to be plotted (no grid or annotation)		
DATEmm,dd,yy	DATE M,D,Y		
	Enters the date of the non-real time clock		
DB	dB		
	Specifies numeric entry in decibel units and terminates entry		
DBAArr (Bus-only)	DISPLAY BUFFER ACTV APPEND	0 – 15	
	Activates the specified display buffer		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
DBACrr (Bus-only)	DISPLAY BUFFER ACTIVE CLR	0–15	
	Clears and activates the specified display buffer		
DBAN (Bus-only)	DUMP BLOCK ANSI BINARY		
	Dumps block specified with PBLKn in ANSI floating point format		
DBAS (Bus-only)	DUMP BLOCK ASCII		
	Dumps block specified with PBLKn in ASCII format		
DBBN (Bus-only)	DUMP BLOCK BINARY		
	Dumps block specified with PBLKn in internal 32 bit floating point format		
DBDNrr (Bus-only)	DISPLAY BUFFER DOWN	0–15	
	Removes the specified display buffer from the display screen		
DBSWn1,n2 (Bus-only)	DISPLAY BUFFER SWITCH		
	Switches the current display from buffer n1 to buffer n2		
DBSZs,rr[,c] (Bus-only)	DISPLAY BUFFER SIZE	0–15	
	Creates display buffer(s) where s is the size of the buffer(s) in words and rr is the first buffer and c is the number of buffers to create		
DBUPrr (Bus-only)	DISPLAY BUFFER UP	0–15	
	Sends the specified display buffer to the display screen		
DBV	dBVrms		
	Specifies numeric entry in dBVrms units and terminates entry		
DCAN (Bus-only)	DUMP COORD BLK ANSI BINARY		
	Dumps coordinate transform block in ANSI floating point format		
DCAS (Bus-only)	DUMP COORD BLK ASCII		
	Dumps coordinate transform block in ASCII format		
DCBN (Bus-only)	DUMP COORD BLK BINARY		
	Dumps coordinate transform block in internal binary format		
DCOFrrss	DC OFFSET	see comment	MV,V,VRMS MVRM,DBV
	Adds positive or negative dc offsets to the source output (offset is limited to 10V minus the source level)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
DDAN (Bus-only)	DUMP DATA ANSI BINARY		
	Dumps active data trace in ANSI floating point format		
DDAS (Bus-only)	DUMP DATA ASCII		
	Dumps active data trace in ASCII format		
DDBN (Bus-only)	DUMP DATA BINARY		
	Dumps active data trace in internal binary format		
DEADr	DESTN ADDRES	0–7	
	Enters the address of the destination disc drive for file copying		
DEC	Decade		
	Specifies numeric entry in decade units and terminates entry		
DEG	Degree		
	Specifies numeric entry in degree units and terminates entry		
DELC	DELETE CHAR		
	Deletes a character when in alpha entry mode		
DEUNrr	DESTN UNIT	0–15	
	Enters the unit # of the destination disc drive for file copying		
DF	DUMP FAULT LOG		
	Dumps the fault log in internal binary format		
DFAF	DFA FUNCTN		
	Runs the internal diagnostic test (see Service Manual)		
DFGR	DFAULT GRIDS		
	Causes the grids to be plotted exactly as they appear on the display		
DFP1	DFA PATT 1		
	Runs an internal diagnostic test (see Service Manual)		
DFP2	DFA PATT 2		
	Runs an internal diagnostic test (see Service Manual)		
DGAN (Bus-only)	DUMP ANL/DIG SETUP IN ANSI		
	Dumps analog/digital setup in ANSI binary format		
DGAS (Bus-only)	DUMP ANL/DIG SETUP IN ASCII		
	Dumps analog/digital setup in ASCII format		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
DGBN (Bus-only)	DUMP ANL/DIG SETUP IN BINARY		
	Dumps analog/digital setup in internal binary format		
DGMS	DIGTAL MSMNTS		
	Loads an autosequence with a digital measurement using the digital input and internal digital source		
DGTR	DIGITAL TRACE		
	Runs an internal diagnostic test (see Service Manual)		
DGTS	DIGITAL TEST		
	Runs an internal diagnostic test.(see Service Manaul)		
DIADr	DISC ADDRES	0–7	
	Enters the HP-IB bus address of a disc drive		
DICO	DISC COPY		
	Displays the disc copy menu		
DIFBn1[,n2] (Bus-only)	DIFFERENTIATE BLOCK		
	Computes differential of block n1 and puts result in n1 or optional block n2		
DIFF	DIFF		
	Differentiates the active trace		
DIFN	DISC FCTN		
	Displays the disc functions menu		
DIG	DIGITAL TRIG		
	Specifies the digital trigger on POD Q		
DIG1	INPUT DIG 1		
	Displays data from Channel 1 in a digital format		
DIG2	INPUT DIG 2		
	Displays data from Channel 2 in a digital format		
DIGT	DIGTAL		
	Displays the digital input service test menu		
DIN1	DSPINT TEST 1		
	Runs an internal diagnostic test (see Service Manual)		
DIN2	DSPINT TEST 2		
	Runs an internal diagnostic test (see Service Manual)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
DIN3	DSPINT TEST 3		
	Runs an internal diagnostic test (see Service Manual)		
DIN4	DSPINT TEST 4		
	Runs an internal diagnostic test (see Service Manual)		
DISC	DISC		
	Displays the disc menu		
DIST	DISC STATUS		
	Retrieves the disc status display which shows the last disc access error		
DIUNrr	DISC UNIT	0 – 15	
	Selects the desired disc unit in multiple-disc set ups		
DIVrrr	DIV	$10^{\pm 38}$	
	Divides active trace data by a numeric constant		
DIVssss	DIV		TRCA,TRCB SAV1,SAV2, 1SAV,2SAV,3SAV 4SAV,5SAV
	Divides active trace by Trace A, Trace B or measurement data stored in SAVED 1, SAVED 2, or SAVED#		
DIVBn1,n2[,n3] (Bus-only)	DIVIDE BLOCK		
	Divides block n1 by block n2 and puts result in block n2 or optional block n3		
DIVCn1,n2[,n3] (Bus-only)	DIVIDE CONSTANT		
	Divides n2 by real constant n1 and puts result in n2 or optional block n3		
DIVIn1,n2[,n3] (Bus-only)	DIV IMAG PART		
	Divides imaginary part of complex block n1 by real constant n2 and puts result in n1 or optional block n3		
DIVRn1,n2[,n3] (Bus-only)	DIV REAL PART		
	Divides real part of complex block n1 by real constant n2 and puts result in n1 or optional block n3		
DIVXn1,n2,n3[,n4] (Bus-only)	DIVIDE COMPLEX		
	Divides block n3 by a complex constant (n1 is the real part and n2 is the imaginary part) and puts result in n3 or optional block n4		
DLFR	DELETE FREQ		
	Displays the demodulation delete frequency menu and table		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
DLIM	DFAULT LIMITS		
	Specifies use of plotter's P1 and P2 locations to define plot boundaries		
DLLNrr	DELETE LINE#	1 – 20	
	Deletes a line in the curve fit table		
DLONr	DELETE ON OFF	0 or 1	
	Enables (1) or disables (0) the demodulation delete frequency table		
DLRGrr	DELETE REGION	1 – 20	
	Deletes the edit line in the demodulation delete frequency table		
DLTF'aaaaaaaa'	DELETE FILE		
	Deletes specified file from disc		
DLTF;ATPT	DELETE FILE		
	Deletes file at catalog pointer		
DLTL	DELETE LINE		
	Deletes edit line in an autosequence table or auto math table		
DLTV	DELETE VALUE		
	Deletes the edit value in the synthesis table		
DM1	DEMOD CHAN1		
	Selects a demodulation measurement on Channel 1		
DM2	DEMOD CHAN2		
	Selects a demodulation measurement on Channel 2		
DMB	DEMOD BOTH		
	Selects a demodulation measurement on both channels		
DMBS	DMA BUS		
	Runs an internal diagnostic test (see Service Manual)		
DMNSr	DOMAIN S Z	0 or 1	
	Specifies curve fit table domain to be s(DMNS1) or z(DMNS0)		
DMODr	DEMOD ON OFF	0 or 1	
	Turns demodulation on (1) or off (0)		
DMSE	DEMOD SELECT		
	Displays the demodulation softkey menu		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
DOMSr	DOMAIN S Z	0 or 1	
	Specifies synthesis table domain to be s(DOMS1) or z(DOMS0)		
DOTS	DOTS		
	Selects dotted lines for use in plotting the display		
DOWN	DOWN ARROW		
	Decrements active numeric entry		
DSAN (Bus-only)	DUMP STATE ANSI BINARY		
	Dumps instrument state in ANSI format		
DSAS (Bus-only)	DUMP STATE ASCII		
	Dumps instrument state in ASCII format		
DSBN (Bus-only)	DUMP STATE BINARY		
	Dumps instrument state in internal binary format		
DSHL	DASHED LINES		
	Selects dashed lines for use in plotting the display		
DSP'aaa..a' (Bus-only)	DISPLAY		
	Writes a message up to 24 characters long to the instrument display line (DSP? lets you read the current message on the display line)		
DSPD (Bus-only)	DISPLAY DISABLE		
	Disables display updating		
DSPE (Bus-only)	DISPLAY ENABLE		
	Enables display updating		
DSPLr	DSPLAY ON OFF	0 or 1	
	Turns the display of intermediate trace updating on (1) or off (0) during autosequence execution		
DSWQ (Bus-only)	DISABLE SWEEP SRQ		
	Disables sweep point ready condition from generating a service request		
DT	DUMP TEST LOG		
	Dumps the test log in internal binary format		
DTAN (Bus-only)	DUMP SYNTH ANSI BINARY		
	Dumps an S-domain synthesis table in ANSI format		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
DTAS (Bus-only)	DUMP S SYNTH IN ASCII		
	Dumps an S-domain sythesis table in ASCII format		
DTBN (Bus-only)	DUMP S SYNTH IN BINARY		
	Dumps an S-domain systhesis table in internal binary format		
DVAN (Bus-only)	DUMP VECTOR DISP ANSI		
	Dumps display buffer identified with the vector buffer pointer (VBLK) in ANSI format		
DVAS (Bus-only)	DUMP VECTOR DISP ASCII		
	Dumps display buffer identified with the vector buffer pointer (VBLK) in ASCII format		
DVBN (Bus-only)	DUMP VECTOR DISP BINARY		
	Dumps display buffer identified with the vector buffer pointer (VBLK) in the instruments internal binary format		
DVICn1,n2[,n3] (Bus-only)	DIVIDE INTO CONSTANT		
	Divides real block n1 into real constant n2 and puts result into n1 or optional block n3		
DVJWωstart,$\Delta\omega$,n1[,n2] (Bus-only)	DIVIDE BY jω		
	Divides block n1 by jω to perform artificial integration		
DZAN (Bus-only)	DUMP Z SYNTH TABLE IN ANSI		
	Dumps a Z-domain synthesis table in ANSI binary format		
DZAS (Bus-only)	DUMP Z SYNTH TABLE IN ASCII		
	Dumps a Z-domain synthesis table in ASCII format		
DZBN (Bus-only)	DUMP Z SYNTH TABLE IN BINARY		
	Dumps a Z-domain synthesis table in internal binary format		
rrErr	E		
	Indicates exponential notation for numeric entries (i.e. 10E4 etc.)		
E2/H	EU^2/Hz		
	Terminates numeric entry		
EDDNrr	EDIT DENOM#	1 – 20	
	Specifies denominator to be edited in the polynomial synthesis table		
EDIT	EDIT		
	Displays the selected autosequence and enables its editing menu (measurement in progress must be paused)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
EDLNrr	EDIT LINE#	1 – 20	
	Specifies a line in the demodulation delete frequency table to edit		
EDMA	EDIT MATH		
	Displays the auto math table and enables its editing menu (measurement in progress must be paused.)		
EDNMrr	EDIT NUMBER#	1 – 20	
	Specifies numerator to be edited in the polynomial synthesis table		
EDPLrr	EDIT POLE#	1 – 20	
	Specifies the pole to be edited in the pole/zero synthesis table		
EDRSrr	EDIT RESDU#	1 – 20	
	Specifies the residue to be edited in the pole/residue synthesis table		
EDTB	EDIT TABLE		
	Displays the curve fit table and its editing menu		
EDWT	EDIT WEIGHT		
	Displays edit weight menu to modify weighting function used by curve fitter		
EDZRrr	EDIT ZERO#	1 – 20	
	Specifies the zero to be edited in the pole zero synthesis table		
ENED	END EDIT		
	Ends the edit session of an autosequence or auto math table		
ENGR	ENGR UNITS		
	Displays the engineering units menu		
ENT	ENTER		
	Terminates entry when in alpha entry mode		
EPOL	EDIT POLES		
	Activates curve fit editing functions for the poles in the table		
ERLG	ERT LOG		
	Displays the ERT log showing the results of the read-only error rate test		
ERR? (Bus-only)	ERROR		
	Returns error code of most recent instrument error		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
ERRD (Bus-only)	ERROR SRQs DISABLE		
	Disables instrument errors from generating a service request		
ERRE (Bus-only)	ERROR SRQs ENABLE		
	Enables instrument errors to generate a service request		
ES/H	EU^2 S/H		
	Terminates numeric entry		
ESMPr	E SMPL ON OFF	0 or 1	
	Turns external sampling on (1) or off (0)		
ESWQ (Bus-only)	ENABLE SWEEP SRQ		
	Enables sweep point ready condition to generate a service request		
EU	EU		
	Specifies numeric entry in previously defined engineering units		
EU/H	EU/√ Hz		
	Terminates numeric entry		
EU2	EU^2		
	Specifies numeric entry engineering units squared		
EUC1	EU CHAN 1		
	Specifies numeric entry in Channel 1 engineering units		
EUC2	EU CHAN 2		
	Specifies numeric entry in Channel 2 engineering units		
EUL1'aaaaa'	EU LBL CHAN1		
	Labels the engineering units defined for Channel 1		
EUL2'aaaaa'	EU LBL CHAN2		
	Labels the engineering units defined for Channel 2		
EUV1rrss	EU VAL CHAN1	±10nV – ±1000V	VEU,MVEU,DB
	Selects engineering units for Channel 1		
EUV2rrss	EU VAL CHAN2	±10nV – ±1000V	VEU,MVEU,DB
	Selects engineering units for Channel 2		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
EXP	EXPON		
	Selects exponential averaging		
EXT	EXT		
	Selects external triggering		
EXTS	EXT SAMPLE		
	Specifies the sample clock as the external sample signal (BNC connector on back panel)		
EZER	EDIT ZEROS		
	Activates curve fit editing functions for the zeros in the table		
FEAD	FR END ADJUST		
	Displays the front end internal diagnostics test menu (see Service Manual)		
FEFN	FR END FUNCTN		
	Runs an internal diagnostic test (see Service Manual)		
FEIN	FR END INTFCE		
	Runs an internal diagnostic test (see Service Manual)		
FFGL	FFT GL INTFCE		
	Runs an internal diagnostic test (see Service Manual)		
FFIN	FFT INTRPT		
	Runs an internal diagnostic test (see Service Manual)		
FFRA	FFT RAM		
	Runs an internal diagnostic test (see Service Manual)		
FFRO	FFT ROM		
	Runs an internal diagnostic test (see Service Manual)		
FFST	FFT STATUS		
	Runs an internal diagnostic test (see Service Manual)		
FFT	FFT		
	Performs a fast Fourier transform (FFT) on the active trace		
FFT1	FFT^{-1}		
	Performs an inverse fast Fourier transform (FFT) on the active trace		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
FFTF	FFT FUNCTN		
	Runs an internal diagnostic test (see Service Manual)		
FILT	FILTRD INPUT		
	Displays the filtered input measurement display menu		
FIPA	FIRST PASS		
	Runs an internal diagnostic test (see Service Manual)		
FIX1	FIXD 1 13 BIT		
	Specifies Channel 1 input data to be determined by the UPR 13 BITS or the LOW 13 BITS (softkey in the data size menu)		
FIX2	FIXD 2 13 BIT		
	Specifies Channel 2 input data to be determined by the UPR 13 BITS or the LOW 13 BITS (softkey in the data size menu)		
FL	FAULT LOG		
	Displays the fault log		
FLAT	FLAT TOP		
	Selects the flat top window		
FLBS	FILTER BUS		
	Runs an internal diagnostic test (see Service Manual)		
FLOWr	FLOW ON OFF	0 or 1	
	Illustrates the data path from input to display (0=off, 1=on)		
FLT1	FLOAT CHAN1		
	Causes Channel 1 to float both sides of the differential input		
FLT2	FLOAT CHAN2		
	Causes Channel 2 to float both sides of the differential input		
FLTBn1,n2[,count] (Bus-only)	FLOAT BLOCK		
	Converts integer data in n1 to floating point data and put results in n2 (count is number of points in n1 to convert)		
FLTS	FILTER TEST		
	Runs an internal diagnostic test (see Service Manual)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
FM1	FM CHAN1		
	Selects FM demodulation for Channel 1		
FM2	FM CHAN2		
	Selects FM demodulation for Channel 2		
FNFRrrrsss	FNDMTL FREQ	0– 100kHz	MHZ,HZ,KHZ RPM,ORDS
	Enters the fundamental frequency for the harmonic markers		
FOOPrrr	FORMAT OPTION	0– 239	
	Allows specification of the format option for HP Subset/80 disc media		
FORM	FORMAT		
	Displays the disc formatting menu		
FOXP	FORCE/EXPON		
	Displays the menu for selecting force and exponential windows		
FPD1	FROM POD 1		
	Connects Channel 2 to receive digital data from Pod 1		
FPD2	FROM POD 2		
	Connects Channel 2 to receive digital data from Pod 2		
FPOD	FROM POD 1		
	Connects Channel 1 to receive digital data from Pod 1		
FPPF	FPP FUNCTN		
	Runs an internal diagnostic test (see Service Manual)		
FRBK	FRONT BACK		
	Superimposes trace A and trace B on a full size display		
FRC1rrrsss	FORCE CHAN1	$10^{\pm 38}$	USEC,MSEC,SEC MIN,REVS
	Selects the force window for Channel 1		
FRC2rrrsss	FORCE CHAN2	$10^{\pm 38}$	USEC,MSEC,SEC MIN,REVS
	Selects the force window for Channel 2		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
FRDA	FREQ & DAMP		
	Shows resonant frequency and damping		
FREE	FREE RUN		
	Selects the free run triggering mode		
FREQ	FREQ		
	Displays the frequency selection menu		
FRLG	F RESP LOGRES		
	Presets instrument to frequency response measurement in log resolution mode		
FRLN	F RESP LINRES		
	Presets instrument to frequency response measurement in linear resolution		
FRQR	FREQ RESP		
	Selects the frequency response display		
FRSrrrsss FRSrrr,rrrsss (lin,capture)	FREQ SPAN	10.24 mHz– 100 kHz	MHZ,HZ,KHZ RPM,ORD
	Specifies frequency span in linear and time capture measurement modes		
FRSrsss (log)	FREQ SPAN	1–5	DEC
	Specifies frequency span in log measurement mode		
FRSrrrsss FRSrrr,rrrsss (swept sine)	FREQ SPAN	2 mHz– 100 kHz	MHZ,HZ,KHZ DEC,OCT
	Specifies frequency span in swept sine measurement mode		
FRSP	FREQ RESP		
	Selects the frequency response measurement		
FRSW	F RESP SWEPT		
	Presets instrument to frequency response measurement in the swept sine mode		
FSAVr	FST AV ON OFF	0 or 1	
	Turns fast averaging on (1) or off (0)		
FSINrrsss	FIXED SINE	64 μHz– 100kHz	MHZ,HZ,KHZ RPM,ORD
	Selects a constant frequency sine wave source		
FSRC	FROM SOURCE		
	Connects Channel 1 to receive digital data directly from the internal source		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
FTFN	FIT FCTN		
	Displays the fit functions menu		
FTLG	FAULT LOG		
	Displays the disc fault log		
FTSN	FIT SYNTH		
	Transfers the curve fit table to the pole/zero synthesis table		
FXIN	FIXED INTGRT		
	Selects fixed integration when in swept sine measurement mode		
FXLNrr	FIX LINE#	1 – 20	
	Fixes a pole or zero while editing the curve fit table		
GAINrr	GAIN FACTOR	$10^{\pm 38}$	
	Sets desired gain of synthesized frequency response function		
GAPH	GAIN & PH MGN		
	Calculates gain and phase margins for the active trace		
GND1	GROUND CHAN1		
	Grounds the Channel 1 input		
GND2	GROUND CHAN2		
	Grounds the Channel 2 input		
GOTOrr	GO TO	1 – 20	
	Instructs the analyzer to jump to indicated line within an autosequence table		
GRAM	GLOBAL RAM		
	Runs an internal diagnostic test (see Service Manual)		
GRAR	GRID AREA		
	Expands grid area to plot as if grid area filled the screen		
GRBLn1,x,Δx (Bus-only)	GRAPH BLOCK		
	Creates a trace from real data block n1, x-axis starting point x and x-axis increment Δx		
GRDPrr	GRID PEN	see comment	
	Selects pen number for plotting the grid (range depends on number of pens in plotter)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
GRIMn1,x,Δx (Bus-only)	GRAPH IMAG PART		
	Creates a trace from the imaginary part of complex data block n1, x-axis starting point x and x-axis increment Δx		
GRREn1,x,Δx (Bus-only)	GRAPH REAL PART		
	Creates a trace from the real part of complex data block n1, x-axis starting point x and x-axis increment Δx		
H/MS	Hz/mSec		
	Specifies numeric entry in Hertz per millisecond		
HANN	HANN		
	Selects the Hanning window		
HBCN	HP-IB CONNEC		
	Runs an internal diagnostic test (see Service Manual)		
HBD	HP-IB DIAG		
	Runs an internal diagnostic test (not valid via HP-IB) (see Service Manual)		
HBFN	HP-IB FUNCTN		
	Runs an internal diagnostic test (not valid via HP-IB) (see Service Manual)		
HELP	HELP		
	Accesses descriptions of other keys		
HIS1	HIST1		
	Selects histogram display of measurement on Channel 1		
HIS2	HIST2		
	Selects histogram display of measurement on Channel 2		
HIST	HIST		
	Selects the histogram measurement type		
HMNC	HMNC ON		
	Activates the harmonic marker and displays its menu		
HOLDr	0 HOLD ON OFF	0 or 1	
	Turns the synthesis zero-order hold function on(HOLD1) or off(HOLD0)		
HMNC	HMNC ON		
	Activates the harmonic marker and displays its menu		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
HPT	HP-IB TRIG		
	Selects HP-IB triggering		
HPWR	HMNC POWER		
	Shows harmonic power contained in the X marker band		
HSTn1,n2,vmax (Bus-only)	HISTOGRAM		
	Computes histogram of data block n1 and puts result in n2 (vmax is maximum absolute amplitude range for n1)		
HXCT	HOLD X CENTER		
	Marks off a band on the horizontal axis of the display. Marker splits and expands/contracts symetrically around its origin.		
HXLF	HOLD X LEFT		
	Marks off a band on the horizontal axis of the display. Marker splits and expands/contracts to the right of its origin.		
HXOF	HOLD X OFF		
	Disables further expansion/contraction of band defined with HOLD X modes		
HXRT	HOLD X RIGHT		
	Marks off a band on the horizontal axis of the display. Marker splits and expands/contracts to the left of its origin.		
HYCT	HOLD Y CENTER		
	Marks off a band on the vertical axis of the display. Marker splits and expands/contracts symetrically around its origin.		
HYLW	HOLD Y LOWER		
	Marks off a band on the vertical axis of the display. Marker splits and expands/contracts upward from the point of its origin.		
HYOF	HOLD Y OFF		
	Disables further expansion/contraction of band defined with HOLD Y modes		
HYUP	HOLD Y UPPER		
	Marks off a band on the vertical axis of the display. Marker splits and expands/contracts downward from the point of its origin.		
HZ	Hz		
	Specifies numeric entry in Hertz units		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
HZ/M	Hz/Min		
	Specifies numeric entry for sweep rate in Hertz per minute		
HZ/O	Hz/Order		
	Specifies numeric entry for Orders Cal in Hertz per order		
HZ/P	Hz/Point		
	Specifies numeric entry for resolution of sine sweep in Hertz per point		
HZ/S	Hz/Sec		
	Specifies numeric entry for sweep rate in Hertz per second		
HZS	Hz(Sec)		
	Specifies Hertz as the basic horizontal unit for frequency domain displays		
IBADrr	HP-IB ADDRESS	0 – 31	
	Enters the HP-IB bus address of the HP 3563A		
IBFN	HP-IB FCTN		
	Displays the HP-IB functions menu		
IBLN	INVERS BI-LIN		
	Converts current z-domain synthesis table into the s-domain using the inverse bilinear transformation		
ICNF	INPUT CONFIG		
	Displays the input configuration menu		
ID? (Bus-only)	IDENTIFY		
	Returns the 7-character string "HP3563A"		
IFC	INTER- FACE		
	Displays the digital source interface menu		
IFC1	INTER- FACE 1		
	Displays the Channel 1 digital input interface menu		
IFC2	INTER- FACE 2		
	Displays the Channel 2 digital input interface menu		
IIMP	INVERS IMPULS		
	Converts current z-domain synthesis table into the s-domain using the inverse impulse invariance transformation		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
IMrrr (Bus-only)	INPUT MASK	0– 256	
	HP-GL Status Register Mask (not implemented)		
IMAG	IMAG		
	Displays the imaginary portion of complex measurement data		
IMBK	IMAGE BACKUP		
	Backs up source disc on destination disc (erases destination disc first)		
IMPI	IMPULS INVRNC		
	Converts current s-domain synthesis table into the z-domain using the impulse invariance transformation		
INCT'aaaaaa'	INIT CATLOG		
	Initializes a media catalog and enables alpha entry mode to enter disc name		
INDI'aaaaaa'	INIT DISC		
	Initializes a disc and enables alpha entry mode to enter disc name		
INGBn1[,n2] (Bus-only)	INTEGRATE BLOCK		
	Integrates block n1 and puts result in n1 or optional block n2		
INGI	INTGRT INIT=0		
	Sets the first point to zero and integrates the active trace		
INGR	INTGRT		
	Integrates the active trace		
INSr	INSERT ON OFF	0 or 1	
	Enables insert character editing feature when in alpha entry mode		
INST	INST		
	Selects the most recent time record for the filtered input displays		
INTMrrss	INTGRT TIME	10^{-3} – 10^{+38}	USEC,MSEC,SEC
	Selects integration time in swept sine mode		
IRSP	IMPLS RESP		
	Selects the impulse response display		
IS? (Bus-only)	INSTR STATUS		
	Returns the contents of the Instrument Status Register		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
ISMrrr (Bus-only)	INSTR STATUS MASK	0–32767	
	Specifies SRQ conditions by unmasking bits in Instrument Status Register		
ISP1	INPUT SPEC 1		
	Displays the frequency spectrum of the signal on Channel 1		
ISP2	INPUT SPEC 2		
	Displays the frequency spectrum of the signal on Channel 2		
ISTP	INVERS STEP		
	Converts current z-domain synthesis table into the s-domain using the inverse step invariance transformation		
ITM1	INPUT TIME 1		
	Displays the time domain signal on Channel 1		
ITM2	INPUT TIME 2		
	Displays the time domain signal on Channel 2		
IWND	INST WNDOWD		
	Displays filtered time record after exponential windowing		
JECH	JUMPER ECHO		
	Runs an internal diagnostic test (see Service Manual)		
JW	$j\omega$		
	Performs artificial differentiation on the active trace		
JW1	$j\omega^{-1}$		
	Performs artificial integration on the active trace		
KEYrr (Bus-only)	KEY	0–70	
	Simulates pressing a front panel key by sending the code for the key		
KEY? (Bus-only)	KEY		
	Returns the key code of the last key pressed		
KEYD (Bus-only)	KEY CODE DISABLE		
	Disables key press from generating a service request		
KEYE (Bus-only)	KEY CODE ENABLE		
	Enables key press to generate a service request		
KH/O	kHz/Order		
	Specifies numeric entry for Orders Cal in kilohertz per order		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
KHZ	kHz		
	Specifies numeric entry in kilohertz		
L (Bus-only)	L		
	Indicates an exponent in scientific notation (see "D" and "E")		
LALL	LOAD ALL		
	Loads all five of the measurement autosequence programs into the autosequence softkeys		
LASTr	LAST 1 0	0 or 1	
	Specifies which of two bytes from an eight bit bus is the last one to be read when using 16 bit data		
LASQ	LOAD ASEQ		
	Displays the LOAD ASEQ menu		
LBAN (Bus-only)	LOAD BLOCK ANSI BINARY		
	Loads block specified by PBLKn in ANSI floating point format		
LBAS (Bus-only)	LOAD BLOCK ASCII		
	Loads block specified by PBLKn in ASCII format		
LBBN (Bus-only)	LOAD BLOCK BINARY		
	Loads block specified by PBLKn in internal 32 bit floating point format		
LBIT	LOW 13 BITS		
	Selects the lower 13 bits of the 16 bit digital data to be used as input		
LBLA'aaa,aaa'	LABEL ASEQ		
	Labels autosequence programs (must be editing table)		
LBLM'aaa,aaa'	LABEL MATH		
	Labels the auto math table. (Must be editing the table)		
LBS1'aaaaaa[,bbbbbb]' (Bus-only)	LABEL USER SRQ1		
	Labels user SRQ softkey 1 where aaaaaa is the top line and bbbbbb is the bottom line		
LBS2'aaaaaa[,bbbbbb]' (Bus-only)	LABEL USER SRQ2		
	Labels user SRQ softkey 2 where aaaaaa is the top line and bbbbbb is the bottom line		
LBS3'aaaaaa[,bbbbbb]' (Bus-only)	LABEL USER SRQ3		
	Labels user SRQ softkey 3 where aaaaaa is the top line and bbbbbb is the bottom line		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
LBS4'aaaaaa[,bbbbbb]' (Bus-only)	LABEL USER SRQ4		
	Labels user SRQ softkey 4 where aaaaaa is the top line and bbbbbb is the bottom line		
LBS5'aaaaaa[,bbbbbb]' (Bus-only)	LABEL USER SRQ5		
	Labels user SRQ softkey 5 where aaaaaa is the top line and bbbbbb is the bottom line		
LBS6'aaaaaa[,bbbbbb]' (Bus-only)	LABEL USER SRQ6		
	Labels user SRQ softkey 6 where aaaaaa is the top line and bbbbbb is the bottom line		
LBS7'aaaaaa[,bbbbbb]' (Bus-only)	LABEL USER SRQ7		
	Labels user SRQ softkey 7 where aaaaaa is the top line and bbbbbb is the bottom line		
LBS8'aaaaaa[,bbbbbb]' (Bus-only)	LABEL USER SRQ8		
	Labels user SRQ softkey 8 where aaaaaa is the top line and bbbbbb is the bottom line		
LCBS	LOCAL BUS		
	Runs an internal diagnostic test (see Service Manual)		
LCL	LOCAL		
	Overrides remote HP-IB control		
LDAN (Bus-only)	LOAD DATA ANSI BINARY		
	Loads a data trace into the HP3563A in ANSI floating point format		
LDAS (Bus-only)	LOAD DATA ASCII		
	Loads a data trace into the HP3563A in ASCII format		
LDBN (Bus-only)	LOAD DATA BINARY		
	Loads a data trace into the HP3563A in internal binary format		
LFRF	LINRES FRF		
	Loads an autosequence with measurement and display parameters for a frequency response measurement in linear resolution mode		
LGAN (Bus-only)	LOAD ANL/DIG SETUP IN ANSI		
	Loads analog/digital setup in ASNI binary format		
LGAS (Bus-only)	LOAD ANL/DIG SETUP IN ASCII		
	Loads analog/digital setup in ASCII format		
LGBN (Bus-only)	LOAD ANL/DIG SETUP IN BINARY		
	Loads analog/digital setup in internal binary format		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
LGRS	LOG RES		
	Configures the instrument to the log resolution mode		
LGSW	LOG SWEEP		
	Selects logarithmic frequency sweeps in swept sine mode		
LIN1	LINES 1024		
	Sets the instrument to use and display 1024 frequency lines		
LIN8	LINES 801		
	Sets the instrument to use and display 801 frequency lines		
LINArrr[,rrr]	LINE A TYPE#	see comment	
	Enters a line type for plotting trace A (refer to your plotter's manual for more information)		
LINBrrr[,rrr]	LINE B TYPE#	see comment	
	Enters a line type for plotting trace B (refer to your plotter's manual for more information)		
LINErr	EDIT LINE#	1 – 20	
	Selects an edit line in a table (must be editing a table)		
LINX	LIN X		
	Converts the horizontal axis of the displayed trace to a linear scale		
LN	LN OF DATA		
	Calculates the natural logarithm of the active trace		
LN1	LN^{-1} OF DATA		
	Calculates the antilog of the active trace		
LNRS	LINEAR RES		
	Configures the instrument to the linear resolution mode		
LNSW	LINEAR SWEEP		
	Selects linear frequency sweeps in the swept sine mode		
LNTP	LINE TYPES		
	Displays the line types menu for setting up plot parameters		
LOFN	LO FUNCTN		
	Runs an internal diagnostic test (see Service Manual)		
LOGOr (Bus-only)	HP LOGO	0 or 1	
	Turns the HP logo on (1) or off (0) on plots		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
LOGX	LOG X		
	Converts the horizontal axis of the displayed trace to a logarithmic scale		
LOOPr	LOOP ON OFF	0 or 1	
	Turns the loop mode on (1) or off (0)		
LOP1	LO DSA PATT 1		
	Runs an internal diagnostic test (see Service Manual)		
LOP2	LO DSA PATT 2		
	Runs an internal diagnostic test (see Service Manual)		
LPTOr1,r2	LOOP TO	see comment	
	Allows programming of loops in autosequence programs where r1 is the first line of loop (1—20) and r2 is the cycle count (1—32767)		
LSAN (Bus-only)	LOAD STATE ANSI BINARY		
	Loads the analog setup state in ANSI floating point format		
LSAS (Bus-only)	LOAD STATE ASCII		
	Loads the analog setup state in ASCII format		
LSBN (Bus-only)	LOAD STATE BINARY		
	Loads the analog setup state in internal binary format		
LSMS	LAST MEAS		
	Selects last frequency response measurement as input to curve fitter		
LSP1	LINEAR SPEC 1		
	Selects the filtered input linear spectrum display on Channel 1		
LSP2	LINEAR SPEC 2		
	Selects the filtered input linear spectrum display on Channel 2		
LSPC	LINEAR SPEC		
	Displays linear spectrum of the time record outlined by the capture pointer		
LSUN	L SPEC UNITS		
	Displays the linear spectrum units menu		
LTr (Bus-only)	LINE TYPE	0 – 4	
	Specifies line type for direct display programming		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
LTAN (Bus-only)	LOAD S SYNTH IN ANSI BINARY		
	Loads the S-domain synthesis table in ANSI floating point format		
LTAS (Bus-only)	LOAD S SYNTH IN ASCII		
	Loads the S-domain synthesis table in ASCII format		
LTBN (Bus-only)	LOAD S SYNTH IN BINARY		
	Loads the S-domain synthesis table in internal binary format		
LUAN (Bus-only)	LOAD U DISPL ANSI BINARY		
	Loads the active display buffer in ANSI floating point format		
LUAS (Bus-only)	LOAD U DISPL ASCII		
	Loads the active display buffer in ASCII format		
LUBN (Bus-only)	LOAD U DISPL BINARY		
	Loads the active display buffer in internal format		
LZAN (Bus-only)	LOAD Z SYNTH IN ANSI		
	Loads the Z-domain synthesis table in ANSI binary format		
LZAS (Bus-only)	LOAD Z SYNTH IN ASCII		
	Loads the Z-domain synthesis table in ASCII format		
LZBN (Bus-only)	LOAD Z SYNTH IN BINARY		
	Loads the Z-domain synthesis table in internal binary format		
M/DC	Min/Dec		
	Specifies numeric entry for sweep rate in minutes per decade		
M/OC	Min/Oct		
	Specifies numeric entry for sweep rate in minutes per octave		
MAG	MAG LIN		
	Defines the Y-axis as magnitude displayed linearly		
MAPR	MANUAL PRVIEW		
	Displays time records for approval before they are included in the measurement		
MATH	MATH		
	Displays the math menu		
MAXS	MAX SPAN		
	Sets frequency span at 0 to 100 kHz		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
MDBM	MAG(dBm)		
	Defines the Y-axis as magnitude displayed in dBm (dB referenced to 1mW)		
MDSP	MEAS DISP		
	Displays the measurement display menus		
ME2H	mEU^2/Hz		
	Terminates numeric entry		
MENUr	MENU→ ON OFF	0 or 1	
	Turns an arrow on(MENU1) or off(MENU0) above softkeys that have lower level menus		
MESH	mEU^2 S/H		
	Terminates numeric entry		
MEU	mEU		
	Specifies numeric entry in milli-engineering units		
MEU2	$m(EU^2)$		
	Specifies numeric entry in milli-engineering units squared		
ME/H	mEU/√ Hz		
	Terminates numeric entry		
MGDB	MAG(dB)		
	Defines the Y-axis as magnitude displayed in dB		
MGLG	MAG(LOG)		
	Defines the Y-axis as magnitude displayed logarithmically		
MH/O	mHz/Order		
	Specifies numeric entry for Orders Cal in millihertz per order		
MHZ	mHz		
	Specifies numeric entry in millihertz units		
MIN	Min		
	Specifies numeric entry in minutes units		
MKPK	MRKR→ PEAK		
	Causes the X marker to automatically track the waveform peak		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
MKRC	MARKER CALC		
	Displays the one-time marker value calculations menu		
MKRPrr	MARKER PEN	see comment	
	Selects the pen number for plotting markers (range depends on number of pens in plotter)		
MKVL	MARKER VALUE		
	Enters the marker value when numeric entry mode is active		
MNSW	MANUAL SWEEP		
	Selects manual sweep and activates the Entry group to move the sweep marker		
MOTP	MORE TYPES		
	Displays a menu of additional trigger types		
MOVBn1,n2[,count] (Bus-only)	MOVE BLOCK		
	Moves contents of block n1 to block n2 (count specifies number of points to move)		
MOVCn1,n2[,count] (Bus-only)	MOVE CONSTANT		
	Moves real constant n1 into n2 (count specifies points in n2 affected by move)		
MOVXn1,n2,n3[,count] (Bus-only)	MOVE COMPLEX		
	Moves complex constant n1,n2 (n1 is real part and n2 is imaginary part) into n3 (count specifies points in n3 affected by move)		
MPJWωstart,Δw,n1[,n2] (Bus-only)	MULTIPLY BY jω		
	Multiplies n1 by jω and puts result in n1 or optional block n2 (ωstart is the floating point starting value and Δω is incremental value)		
MPMGn1,n2[,n3] (Bus-only)	MULTIPLY BY MAG SQUARED		
	Multiplies real floating point block n1 by the magnitude squared of the complex floating point block n2 and puts the result in n1 or optional block n3		
MPSCn1[,n2] (Bus-only)	MULTIPLY SELF-CONJUGATE		
	Multiplies n1 by complex conjugate of n1 and puts result in n1 or optional block n2		
MPYrrr	MPY	$10^{\pm 38}$	
	Multiplies active trace by numeric constant		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
MPYssss	MPY		TRCA,TRCB SAV1,SAV2, 1SAV,2SAV,3SAV, 4SAV,5SAV
	Multiplies active trace by Trace A, Trace B or measurement data stored in SAVED 1, SAVED2, or SAVED#		
MPYBn1,n2[,n3] (Bus-only)	MULTIPLY BLOCK		
	Multiplies block n1 by n2 and put results in n2 or optional block n3		
MPYCn1,n2[,n3] (Bus-only)	MULTIPLY CONSTANT		
	Multiplies n2 by constant n1 and puts result in n2 or optional block n3		
MPYXn1,n2,n3[,n4] (Bus-only)	MULTIPLY COMPLEX		
	Multiplies complex block n3 by complex constant n1,n2 (real,imag) and puts result in n3 or optional block n4		
MSCK	MEAS CLOCK		
	Selects the measurement clock frequency as the curve fit sample frequency		
MSCL	MEAS CLOCK		
	Selects the measurement clock frequency as the synthesis sample frequency in the z-domain		
MSEC	mSec		
	Specifies numeric entry in milliseconds		
MSMD	MEAS MODE		
	Displays the measurement mode menu		
MTYP	MORE TYPES		
	Displays a menu of additional source types and modes		
MV	mV		
	Specifies numeric entry in millivolts		
MV2	m(V^2)		
	Specifies numeric entry in millivolts squared		
MV2H	m(V^2/Hz)		
	Terminates numeric entry		
MVEU	mV/EU		
	Specifies numeric entry in millivolts per defined engineering unit		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
MVRM	mVrms		
	Specifies numeric entry in millivolts rms		
MVSH	m(V^2S/Hz)		
	Terminates numeric entry		
MV/H	mV/√ Hz		
	Terminates numeric entry		
MXRTr	MIXED RATIO	1 – 512	
	Specifies the ratio between the analog and digital sample rates		
NAVGrrr	NUMBER AVGS	1 – 32767	
	Selects number of averaged measurements		
NEG	NEGATE		
	Negates the active trace		
NEGBn1[,n2] (Bus-only)	NEGATE BLOCK		
	Negates block n1 and puts result in n1 or optional block n2		
NEX	NEXT		
	Displays the third level of the math menu		
NEXT	NEXT		
	Displays the second level of the coordinate menu		
NICL	NICHOL		
	Configures the display as a Nichols diagram		
NMBRr	NMBR ? ON OFF	0 or 1	
	Turns a question mark on(NMBR1) or off(NMBR0) beside softkeys used to enter numeric or alphanumeric values		
NOPG	NO PAGING		
	Disables the paging feature on plotters so equipped		
NPOLrr	NUMBER POLES	1 – 40	
	Specifies the number of poles for a curve fit operation		
NT	NEXT		
	Displays the fourth level of the math menu		
NX	NEXT		
	Displays the second level of the average menu		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
NXPG	NEXT PAGE		
	Displays the second page of the disc service logs		
NXRC	NEXT RECORD		
	Displays the next record of time data in the active throughput file		
NXT	NEXT		
	Displays the second level of the math menu		
NXTP	NEXT PAGE		
	Displays the next page of a disc catalog containing more than one page		
NYQT	NYQUST		
	Configures the display as a Nyquist diagram		
NZERrr	NUMBER ZEROS	1 – 40	
	Specifies the number of zeros for a curve fit operation		
OBIN	OFFSET BINARY		
	Sets the specified digital input channel to treat data as an offset binary number		
OCT	Octave		
	Specifies numeric entry in octaves		
OFAD	OFFSET ADJUST		
	Runs an internal diagnostic test (see Service Manual)		
OHM	Ohm		
	Specifies numeric entry in ohms		
OP[p1x,p1y,p2x,p2y]	OUTPUT PARAMETERS		
	Returns pen location (x, y coordinates)		
ORBT	ORBITS T1vsT2		
	Displays the orbits diagram showing Channel 1 time domain signal on vertical axis and Channel 2 time domain signal on horizontal axis.		
ORCLrrrsss	Orders CAL	$10^{\pm 38}$	HZ/O,KH/O,MH/O
	Enters orders calibration in Hz/Ord		
ORD	Orders		
	Specifies numeric entry for carrier frequency for sideband marker as orders		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
ORDR	Orders (Revs)		
	Selects orders as the basic horizontal unit for frequency domain displays and revolutions for time domain displays		
OULG	OUTPUT LOG		
	Displays the disc output log menu		
OUTrr,'aaa...'	OUTPUT STRING		
	Instrument must be the system controller to use this feature. Not programmable in HP-IB. (see Operating Manual)		
OVAUr	OVERWR AU MAN	0 or 1	
	Selects automatic (1) or manual (0) overwriting for file copying		
OVLPrr	OVRLP%	0 – 90	
	Determines percentage of time record used for overlap processing		
OVRJr	OV REJ ON OFF	0 or 1	
	Turns overload rejection on (1) or off (0)		
OVWR	OVER WRITE		
	Terminates disc entry when used in conjunction with SAVE FILE (SAVF)		
P/DC	Points/Dec		
	Specifies numeric entry in points per decade (Swept Sine mode; Log sweep)		
P/OC	Points/Oct		
	Specifies numeric entry in points per octave (Swept Sine mode; Log sweep)		
P/RV	Pulse/Rev		
	Specifies numeric entry in pulses per revolution (External Sample Frequency)		
P/SW	Points/Sweep		
	Specifies numeric entry in points per sweep (Swept Sine mode; Log sweep)		
PA (Bus-only)	PLOT ABSOLUTE		
	Moves pen to a location on the screen relative to the 0,0 origin		
PATH	PASS THRU		
	Runs an internal diagnostic test (see Service Manual)		
PAUS (Bus-only)	PAUSE		
	Pauses measurement in progress		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
PBLKr (Bus-only)	PRIMITIVE BLOCK	0– 15	
	Specifies active block for dumping and loading		
PCBLn1 (Bus-only)	PLOT COMPLEX BLOCK		
	Plots the real part of complex block n1 versus imaginary part of block n1		
PCLRn1,p (Bus-only)	PARTIAL BLK CLR		
	Clears points starting a beginning of block (n1 is block to clear,p is number of points to clear)		
PCRP	PRIODC CHIRP		
	Selects periodic chirp source		
PCT	PERCNT		
	Specifies numeric entry as a percentage		
PCTL	PAGING CONTRL		
	Displays the plot paging control menu		
PD (Bus-only)	PEN DOWN		
	Puts the pen down when plotting to the analyzer's display (turns the beam on)		
PDF1	PDF 1		
	Selects the probability density function display for histogram measurement on Channel 1		
PDF2	PDF 2		
	Selects the probability density function display for histogram measurement on Channel 2		
PFCR	PM/FM CRRIER		
	Displays the demodulation PM/FM carrier menu		
PGBK	PAGE BACK		
	Enables plotter page back feature		
PGFW	PAGE FORWARD		
	Enables plotter page forward feature		
PHLD	PEAK HOLD		
	Selects the peak hold averaging function		
PHSErrrsss	PHASE(CENTER)	± 180	DEG
	Selects phase coordinates for the display		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
PKDI	PACK DISC		
	Packs disc		
PKHDn1,n2 (Bus-only)	PEAK HOLD		
	Compares values in blocks n1 and n2 and puts larger values into block n2		
PLADrr	PLOT ADDRES	0 – 31	
	Defines HP-IB plotter address		
PLAR	PLOT AREA		
	Plots entire screen		
PLIM	PLOT LIMITS		
	Displays the plot limits menu		
PLOT	PLOT		
	Displays the plot menu		
PLPR	PLOT PRESET		
	Presets plot variables to defaults		
PM1	PM CHAN1		
	Selects PM demodulation for Channel 1		
PM2	PM CHAN2		
	Selects PM demodulation for Channel 2		
PNTS	Points		
	Specifies numeric entry in points		
POD1	EXTERN POD 1		
	Runs an internal diagnostic test (see Service Manual)		
POD2	EXTERN POD 2		
	Runs an internal diagnostic test (see Service Manual)		
POLR	DEMOD POLAR		
	Displays a plot of the tip of the carrier vector as it is being modulated by AM and PM		
POLY	POLY-NOMIAL		
	Specifies ratio-of-polynomials synthesis, displays the polynomial table, and enables its editing menu		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
PPEKn1,n2 (Bus-only)	PWR SPEC PEAK HOLD		
	Computes power spectrum of block n1, compares its magnitudes with block n2 and then holds the larger values in block n1		
PQCL	POD Q CLOCK		
	Selects the clock line on the digital input Pod Q to be the sample clock		
PR (Bus-only)	PLOT RELATIVE		
	Moves pen to a location on the screen relative to the current pen position		
PRBLn1,n2 (Bus-only)	PLOT REAL BLOCK		
	Plots one real floating data block n1 against another real floating block n2		
PROF	PRVIEW OFF		
	Deactivates manual or timed previewing		
PROM	PROG ROM		
	Runs an internal diagnostic test (see Service Manual)		
PRONr	PRVIEW ON OFF	0 or 1	
	Turns demodulation previewing on (1) or off (0)		
PRSD	POLE RESIDU		
	Specifies pole/residue synthesis, displays the pole/residue table and enables its editing menu		
PRST	PRESET		
	Presets the HP 3563A to the current operating mode and displays the special preset menu		
PRVP	PREV PAGE		
	Displays previous page of disc catalogs containing more than one page		
PSAS	PAUSE ASEQ		
	Pauses a running autosequence		
PSCN	PAUSE/CONT		
	Toggles measurement activity on/off (see also PAUS and CONT)		
PSLN	P SPEC LINRES		
	Presets the instrument to the power spectrum measurement in the linear resolution mode		
PSP1	POWER SPEC 1		
	Selects power spectrum display for measurement on Channel 1		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
PSP2	POWER SPEC 2		
	Selects power spectrum display for measurement on Channel 2		
PSPC	POWER SPEC		
	Selects power spectrum measurement		
PSPSn1,n2 (Bus-only)	PWR SPEC SUMMATION		
	Computes power spectrum of complex floating point block n1, sums it with contents of second block n2 and places final result in block n2		
PSRQr	PwrSRQ ON OFF	0 or 1	
	Turns power on SRQ generation on (1) or off (0)		
PSUN	P SPEC UNITS		
	Displays the power spectrum units menu		
PTCTn,p (Bus-only)	POINT COUNT		
	Specifies a portion of a block for use in subsequent operations (n specifies block (0-15), p=number of points)		
PTINrrrssss	POINTR INCRMT	see comment	USEC,MSEC,SEC MIN,REVS,PNTS REC
	Defines increment in which capture pointer moves through time buffer (range depends on suffix)		
PTONr	PROTCT ON OFF	0 or 1	
	Turns source protection on (1) or off (0)		
PU (Bus-only)	PEN UP		
	Lifts the pen when plotting to the analyzer's display (turns the beam off)		
PULS	PULSE		
	Sets the source type to be the unit sample		
PWR	POWER		
	Shows power in the area defined by the X marker		
PWRPr	P WARP ON OFF	0 or 1	
	Turns the frequency pre-warping feature on(PWRP1) or off(PWRP0) for the bi-linear transformation		
PXAVn1,n2,awf (Bus-only)	PWR SPEC EXPON AVG		
	Computes power spectrum from a complex block n1 and averages it with a cumulative average in another block n2 using weighting factor awf		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
PZRO	POLE ZERO		
	Specifies pole/zero synthesis, displays the pole/zero table and enables its editing menu		
QPOD	QUALFR POD		
	Runs an internal diagnostic test (see Service Manual)		
RAMPrrrsss	RAMP TIME	$10^{\pm 38}$	USEC,MSEC,SEC MIN,REC
	Defines ramp time for source protection		
RCIP	RECIP		
	Calculates recipricol of active trace		
RCLDr	RECALL DATA#	1–5	
	Recalls a saved data trace from internal memory		
RCLF'aaaaaaaa'	RECALL FILE		
	Recalls a file from disc		
RCLP	RECALL PWR DN		
	Recalls the instrument's last power down state		
RCLSr	RECALL STATE#	1–5	
	Recalls an instrument state saved in internal memory		
RDMK (Bus-only)	READ MARKER		
	Returns X-axis and Y-axis marker values		
RDP1	READ PEN P→1		
	Reads plotter's current pen position into P1		
RDP2	READ PEN P→2		
	Reads plotter's current pen position into P2		
RDY? (Bus-only)	READY		
	Returns 1 if instrument is on the bus		
RDYD (Bus-only)	READY DISABLE		
	Disables ready bit in status byte		
RDYE (Bus-only)	READY ENABLE		
	Enables ready bit in status byte		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
REAL	REAL		
	Displays real portion of complex measurement data		
REC	Record		
	Specifies numeric entry in records		
REJT (Bus-only)	REJECT		
	HP-IB "no" response		
REND (Bus-only)	REMOTE ENTRY DISABLE		
	Disables entry group RPG knob		
RENE (Bus-only)	REMOTE ENTRY ENABLE		
	Enables entry group RPG knob		
RENSn,max (Bus-only)	REMOTE ENTRY SPEED		
	Sets RPG acceleration and speed where n specifies acceleration type (0=fixed and >1=variable) and max specifies maximum velocity		
RENVrrr (Bus-only)	REMOTE ENTRY VAL	−32,768−32,767	
	Sets the entry RPG value (RENV? returns current entry RPG value)		
REPT	REPEAT		
	Sets the source type to repeat continuously		
RERT	RO ERT TEST		
	Runs the disc read-only error rate test		
RESrrrsss	RESLTN	64 μHz − 99.99994 kHz 5 − 1^{10} pts/dec	HZ/P,P/SW P/DC,P/OC,P/SW
	Specifies resolution of the sine sweep		
REV (Bus-only)	REVISION		
	Displays software revision information (REV? returns revision information)		
REVS	Revs		
	Specifies numeric entry in revolutions		
RFC1	REF CHAN 1		
	Designates Channel 1 as the reference channel for the auto gain function		
RFC2	REF CHAN 2		
	Designates Channel 2 as the reference channel for the auto gain function		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
RFFTn1,n2	REAL FFT		
	Performs real FFT on block n1 and puts result in block n2		
RFLVrrrsss	REF LEVEL	5mV – 31.5 Vpk	V,MV,VRMS,MVRM DBVR,EU
	Specifies the reference level for the auto gain function		
RFPP	RESET FPP		
	Runs an internal diagnostic test (see Service Manual)		
RFT1n1,n2 (Bus-only)	REAL FFT $^{-1}$		
	Performs real inverse FFT on block n1 and puts result in block n2		
RLOK (Bus-only)	SEND REFERENCE LOCKED		
	Returns a 1 if clock is phase-locked to a reference		
RLPT	REAL PART		
	Displays the real part of complex measurement data		
RMKD (Bus-only)	REMOTE MARKER DISABLE		
	Disables marker knob turn from generating a service request		
RMKE (Bus-only)	REMOTE MARKER ENABLE		
	Enables marker knob turn to generate a service request		
RMKVr (Bus-only)	REMOTE MARKER VAL		
	Sets and returns (RMKV?) the marker RPG value		
RND	RANDOM NOISE		
	Selects random noise source		
RNG	RANGE		
	Displays the range menu		
ROTr	ROT 90 ON OFF	0 or 1	
	Rotates the plot 0 degrees (0) or 90 degrees (1)		
RPM	RPM		
	Specifies numeric entry in rotations per minute		
RPMS	RPM (Sec)		
	Selects rotations per minute as the basic horizontal unit for frequency domain displays and seconds for time domain displays		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
RSAUr	RESLTN AU FIX	0 or 1	
	Selects auto (1) or fixed (0) sweep resolution		
RSCO	RESUME COPY		
	Proceeds copying next file if duplicate filename is encountered or copying resumes when new disc is installed after previous disc ran out of space		
RSCT'aaaaa'	RESTOR CATLOG		
	Reads the disc directory and restores the file catalog		
RSMG (Bus-only)	READ SPCL MARKER GROUP		
	Returns the value of the SLOPE,HMNC POWER,THD,or SBAND POWER (whichever one is active) for each trace		
RSMO (Bus-only)	READ SPCL MARKER ONCE		
	Returns the value of the POWER,FREQ & DAMP,or AVG VALUE special marker (whichever one was pressed last)		
RSOV	RESUME OVERWR		
	Causes destination file to be overwritten if duplicate filename is encountered while copying files		
RST	RESET		
	Resets instrument to its default power-on conditions		
RTN	RETURN		
	Displays previous menu		
RULG	RUN TM LOG		
	Shows the cumulative run time errors of the disc		
S/DC	Sec /Dec		
	Specifies numeric entry for sweep rate in seconds per decade		
S/OC	Sec /Oct		
	Specifies numeric entry for sweep rate in seconds per octave		
SA20	SIDE A 20 dB		
	Runs an internal diagnostic test (see Service Manual)		
SA40	SIDE A 40 dB		
	Runs an internal diagnostic test (see Service Manual)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
SACL	SAMPLE CLOCK		
	Displays the sample clock menu		
SACR (Bus-only)	SEND AUTO CARRIER	Auto carrier 2	Phase offset 1
	Returns values from the demodulation algorithm's auto carrier feature (Auto carrier 1, Auto carrier 2, phase offset 1, phase offset 2)		
SADR	SELECT ADDRES		
	Displays the HP-IB bus select address menu		
SASQrssss	SELECT ASEQ#	1 – 5	EDIT,VIEW
	Selects an autosequence program for viewing or editing		
SAV1	SAVED 1		
	Enters trace stored in SAVE DATA #1 memory location for math operations		
SAV2	SAVED 2		
	Enters trace stored in SAVE DATA #2 memory location for math operations		
rSAV	SAVED #	1 – 5	
	Enters trace stored in SAVE DATA memory location 1,2,3,4 or 5 for math operations		
SAVDr	SAVE DATA#	1 – 5	
	Saves a data trace in internal non-volatile memory		
SAVF'aaaaaaaa'	SAVE FILE		
	Saves the current display on disc		
SAVR	SAVE RECALL		
	Displays the menu for saving and recalling states and traces		
SAVSr	SAVE STATE#	1 – 5	
	Saves an instrument state in internal memory		
SB20	SIDE B 20 dB		
	Runs an internal diagnostic test (see Service Manual)		
SB40	SIDE B 40 dB		
	Runs an internal diagnostic test (see Service Manual)		
SBINrrrsss	SBAND INCRMT	12.8 μHz – 100 kHz	KHZ,HZ,MHZ RPM,ORD
	Enters the sideband increment for the sideband markers		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
SBND	SBAND ON		
	Activates the sideband marker and displays its menu		
SCAL	SCALE		
	Displays the scale menu		
SCFRrrrsss	SCALE FREQ	$10^{\pm 6}$	KHZ,HZ,MHZ
	Adds a scaling frequency to the synthesis table		
SCLFrrrsss	SCALE FREQ	$10^{\pm 6}$	KHZ,HZ,MHZ
	Enters scaling frequency for the curve fitter		
SCRLr	SCROLL ON OFF	0 or 1	
	Turns the X marker scrolling feature on (1) of off (0)		
SDAT	SELECT DATA		
	Displays the select data plot menu		
SDOM	S DOMAIN		
	Creates a trace using the current s-domain synthesis table		
SEC	Sec		
	Specifies numeric entry in seconds		
SELT	SELECT TRIG		
	Displays the trigger selection menu		
SEP1rrr,rrr	SET P1 LWR LF	± 32767	
	Specifies the x,y coordinates of the P1 location for the plot		
SEP2rrr,rrr	SET P2 UPR RT	± 32767	
	Specifies the x,y coordinates of the P2 location for the plot		
SEPA	SECOND PASS		
	Runs an internal diagnostic test (see Service Manual)		
SER? (Bus-only)	SERIAL		
	Returns 10 character string representing serial # of instrument		
SET (Bus-only)	SETUP		
	Loads the analog input instrument setup state (same as LSAN command)		
SET? (Bus-only)	SETUP		
	Dumps the analog input instrument setup state (same as DSAN command)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
SFrrrsss (lin,capture)	START FREQ	0– 99999.97952 Hz	MHZ,HZ,KHZ RPM,ORD
	Specifies start frequency for linear and time capture measurement modes		
SFrrrsss (log)	START FREQ	0.1Hz– 100 kHz	MHZ,HZ,KHZ
	Specifies start frequency for log measurement mode		
SFrrrsss (swept sine)	START FREQ	64 μHz– 99999.99988Hz	MHZ,HZ,KHZ RPM,ORD
	Specifies start frequency for swept sine measurement mode		
SFLT (Bus-only)	SEND SOURCE FAULT		
	Returns 0 if source is operating normally or 1 if not		
SFREtrss	SAMPLE FREQ	1E– 6 to 1E6	KHZ,HZ,MHZ
	Specifies sample frequency for z-domain curve fit		
SFRF	SWEPT FRF		
	Programs an autosequence with measurement and display parameters for a frequency response measurement in linear resolution mode		
SFRQrrss	SAMPLE FREQ	1E– 6 to 1E6	KHZ,HZ,MHZ
	Specifies sample frequency for z-domain synthesis		
SING	SINGLE		
	Sets the source type to output only during data collection of the measurement		
SLDA	SOLIDA DASH B		
	Selects a solid line for plotting trace A and a dashed line for trace B		
SLDL	SOLID LINES		
	Selects solid lines for use in plotting the display		
SLGR	SOLID GRIDS		
	Causes the grid to be plotted with solid lines		
SLOPr	SLOPE±	0 or 1	
	Selects positive (1) or negative (0) trigger slope		
SLP	SLOPE		
	Shows the slope of the active trace at the current X marker position		
SMES	SELECT MEAS		
	Displays the measurement selection menus		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND COMMENTS	RANGE	SUFFIXES
SMPFrrrsss	SAMPLE FREQ	≤ 256 kHz	MHZ,HZ,KHZ RPM,P/RV
	Enters the external sampling frequency		
SMSD (Bus-only)	SEND MEAS DONE		
	Returns 1 if measurement is done		
SMSP (Bus-only)	SEND MISSED SAMPLE		
	Returns 1 if a sample was missed		
SNFN	SYNTH FCTN		
	Displays the synthesis functions menu		
SNFT	SYNTH FIT		
	Transfers the pole/zero synthesis table to the curve fit table		
SNGC	SINGLE CAL		
	Performs a single calibration		
SNGL	SINGLE		
	Puts CRT into single display format. If both traces are active trace A is displayed.		
SNTH	SYNTH		
	Displays the synthesis menu		
SOV1 (Bus-only)	SEND OVFL CH 1		
	Returns 1 if Channel 1 is overloaded		
SOV2 (Bus-only)	SEND OVFL CH 2		
	Returns 1 if Channel 2 is overloaded		
SP[r]	SELECT PEN	see comment	
	Selects HPGL plotter pen (range depends on number of plotter pen)		
SPBK	SPACE BACKWD		
	Moves the cursor one space backward when in alpha entry mode		
SPBL	SPARE BLOCK		
	Allows a block on a disc to be spared if it is determined to be defective (drive dependent)		
SPEDr	SPEED F S	0 or 1	
	Selects fast (1) or slow (0) plotting		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
SPEN	SELECT PENS		
	Displays the select pens menu		
SPFrrrsss	STOP FREQ	120 μHz – 100 kHz	MHZ,HZ,KHZ RPM,ORD
	Specifies the measurement stop frequency		
SPFN	SPCL FCTN		
	Displays the special functions menu		
SPFT	STOP FIT		
	Stops the curve fit process		
SPFW	SPACE FORWRD		
	Moves the cursor one space forward when in alpha entry mode		
SPMK	SPCL MARKER		
	Displays the special markers menu		
SPWR	SBAND POWER		
	Shows the sideband power in the area contained by the sideband markers		
SQRT	SQUARE ROOT		
	Calculates the square root of the active trace		
SQUL'rrrrrrr'	SOURCE QUALFR	0,1 or X	
	Selects bits on the qualifier pod(Pod Q) which will be used to qualify the source(Pod X) to clock the output (each bit can be 0, 1, or X)		
SRANr	SOURCE AN DIG	0 or 1	
	Sets the source to the analog(SRAN1) or digital(SRAN0) mode		
SRCE	SOURCE		
	Displays the source menu		
SRCF	SOURCE FUNCTN		
	Runs an internal diagnostic test (see Service Manual)		
SRCHrrss	SOURCE CHANGE	0.2 – 10dB	DB
	Sets the tracking resolution of the source to changes in level measured by the reference input		
SRCM	SOURCE MAIN		
	Runs an internal diagnostic test (see Service Manual)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
SRLMrrrsss	SOURCE LIMIT	5 mV – 5V	V,MV,VRMS MVRM,DBV
	Sets a limit on the source level for auto gain		
SRLVrrsss	SOURCE LEVEL	0 – 5V	V,MV,VRMS MVRM,DBV
	Sets amplitude level of source		
SROF	SOURCE OFF		
	Turns the source off		
SRONr	SOURCE ON OFF	0 or 1	
	Turns the source on (1) or off (0) in swept sine mode		
SRPT	SOURCE PROTCT		
	Displays the source protection menu		
SRQ1	USER SRQ 1		
	Sends a user SRQ to an HP-IB controller		
SRQ2	USER SRQ 2		
	Sends a user SRQ to an HP-IB controller		
SRQ3	USER SRQ 3		
	Sends a user SRQ to an HP-IB controller		
SRQ4	USER SRQ 4		
	Sends a user SRQ to an HP-IB controller		
SRQ5	USER SRQ 5		
	Sends a user SRQ to an HP-IB controller		
SRQ6	USER SRQ 6		
	Sends a user SRQ to an HP-IB controller		
SRQ7	USER SRQ 7		
	Sends a user SRQ to an HP-IB controller		
SRQ8	USER SRQ 8		
	Sends a user SRQ to an HP-IB controller		
SRQD (Bus-only)	SRQ DISABLE		
	Disables end-of-plot-action and end-of-disc-action conditions from generating a service request		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
SRQE (Bus-only)	SRQ ENABLE		
	Enables end-of-plot-action and end-of-disc-action conditions to generate a service request		
SRR1	SOURCE RNG 1		
	Sets the Channel 1 range to be the same as the source range		
SRR2	SOURCE RNG 2		
	Sets the Channel 2 range to be the same as the source range		
SRRGrrss	SOURCE RANGE	5.12– 0.128V -0.128– -5.12V	V,MV,VRMS,MVRMS DBVRMS,.,.
	Sets the full scale range (relates the source level to the actual digital source scaling)		
SRSP	STEP RESPNS		
	Loads an autosequence with measurement and display parameters for a step response measurement in linear resolution mode		
SRTP	SOURCE TYPE		
	Displays the source type menu		
SSIN	SWEPT SINE		
	Selects the swept sine measurement mode		
SSWP (Bus-only)	SEND SWEEP POINT		
	Returns value of input power, output power, cross spectrum real, cross spectrum imaginary, and frequency in swept sine mode		
STA? (Bus-only)	STATUS/EVENT		
	Returns a 16-bit word comprised of selected bits from the Status Byte and Instrument Status Register		
STAT (Bus-only)	STATE		
	Displays the analog setup STATE display		
STBL	STABLE (MEAN)		
	Selects stable averaging		
STCP	START CAPTUR		
	Starts the time capture using the current capture setup		
STDG (Bus-only)	DIGITAL SETUP		
	Displays the digital setup STATE display		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
STEP	STEP		
	Sets the source type to be a step		
STFT	START FIT		
	Starts the curve fit process		
STHR	START THRUPT		
	Start the time throughput session using the current throughput setup		
STMA	START MATH		
	Starts the auto math program		
STPI	STEP INVRNC		
	Converts current s-domain synthesis table into the z-domain using the step invariance transformation		
STPL	START PLOT		
	Plots the information on the display		
STRG	SOURCE TRIG		
	Selects triggering on an internal signal synchronized to the source		
STRL (Bus-only)	SEND TRACE LIMITS		
	Returns two integers representing the start and stop bin for the active trace		
STRS (Bus-only)	SEND TEST RESULT		
	Returns 1 if most recent internal diagnostic test passed		
STRT	START		
	Initiates a measurement based on current setup		
STTR	STATE TRACE		
	Toggles display between the instrument state and the data trace		
STWT	STORE WEIGHT		
	Stores the user defined weighting function for use by the curve fitter		
SUBrrr	SUB	$10^{\pm 38}$	
	Subtracts numeric constant from active trace		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
SUBssss	SUB		TRCA,TRCB SAV1,SAV2, 1SAV,2SAV,3SAV 4SAV,5SAV
	Subtracts Trace A, Trace B or measurement data stored in SAVED 1, SAVED 2, or SAVED# from active trace		
SUBBn1,n2[,n3] (Bus-only)	SUBTRACT BLOCK		
	Subtracts block n2 from n1 and puts result in n2 or optional block n3		
SUBCn1,n2[,n3] (Bus-only)	SUBTRACT CONSTANT		
	Subtracts n1 from n2 and puts result in n2 or optional block n3		
SUBXn1,n2,n3[,n4] (Bus-only)	SUBTRACT COMPLEX		
	Subtracts complex constant n1,n2 (real,imag) from n3 and stores it in n3 or optional n4		
SVFN	SERVCE FCTNS		
	Displays the disc service functions menu		
SVTS	SERVIC TEST		
	Displays the service diagnostic menu		
SWDN	SWEEP DOWN		
	Causes the measurement to sweep down from current point to start frequency		
SWHD	SWEEP HOLD		
	Halts the sweep without stopping the measurement		
SWRTrrrsss	SWEEP RATE	$10^{\pm 38}$	S/DC,M/DC,S/OC M/OC,H/MS,HZ/S HZ/M
	Specifies sweep rate in swept sine mode		
SWUN	SWEPT UNITS		
	Displays the swept units menu		
SWUP	SWEEP UP		
	Causes the measurement to start sweeping up from the current point to the specified stop frequency		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
SYSC	SYSTEM CNTRLR		
	Identifies the instrument as the system controller		
TALL	TEST ALL		
	Runs an internal diagnostic test (see Service Manual)		
TBFN	TABLE FCTNS		
	Displays the curve fit table functions menu		
TCOM	TWOS COMPL		
	Sets the specified digital input channel to treat input data as a Two's Complement number		
TCPU	TEST CPU		
	Displays the CPU internal diagnostics test menu		
TDFA	TEST DFA		
	Displays the DFA assembly internal diagnostics test menu		
TDLYrrrsss	TIME DELAY	$10^{\pm 38}$	USEC,MSEC,SEC
	Adds time delays to synthesized functions		
TFFT	TEST FFT		
	Displays the FFT assembly internal diagnostics test menu		
TFPP	TEST FPP		
	Displays the FPP assembly internal diagnostics test menu		
THD	THD		
	Shows the total harmonic distortion generated by the harmonics identified with the harmonic markers		
THDA	THRUPT DATA		
	Displays the Thruput Data menu		
THED	THRUPT HEADER		
	Displays the throughput header showing the throughput setup		
THLN	THRUPT LENGTH		
	Sets the length of the next throughtput session		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
THRUr	THRUPT ON OFF	0 or 1	
	Turns the throughput mode on (1) or off (0)		
THSE	THRUPT SELECT		
	Displays the throughput selection menu		
THSZrrrsss	THRUPT SIZE	1 – 32767 REC	USEC,MSEC,SEC MIN,REVS,REC
	Specifies the throughput file size (range depends on suffix)		
THT1	THRUPT TIME 1		
	Recalls first Channel 1 time record from throughput file into the active trace		
THT2	THRUPT TIME 2		
	Recalls first Channel 2 time record from throughput file into the active trace		
TIAVr	TIM AV ON OFF	0 or 1	
	Switches between linear(0) and power spectrum(1) averaging		
TIMEhh,mm,ss	TIME H,M,S		
	Enters the time of the non-real time clock		
TINP	TEST INPUT		
	Displays the input internal diagnostics test menu		
TIPRrrsss	TIMED PRVIEW	$0 - 10^{\pm 38}$	SEC
	Displays time records for approval before they are included in the measurement and waits specified time interval for your response		
TIPSrrr	TIMED PAUSE	0 – 32767 SEC	SEC,MIN
	Inserts a timed pause into an autosequence (range depends on suffix)		
TISTrr,rr,rr	TIMED START	00,00,00 – 23,59,59	
	Specifies the starting time of an autosequence (24,00,00 deactivates)		
TITL'aaa..a'	TRACE TITLE		
	Enables alpha entry mode to add titles to the active trace (20 characters maximum)		
TKEY	TEST KEYBD		
	Runs an internal diagnostic test (see Service Manual)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
TKMK	TICK MARKS		
	Plots short tick marks on the grid axes in place of grid lines		
TL	TEST LOG		
	Displays the contents of the test log		
TLNrrrsss	TIME LENGTH	8ms– 78125s	USEC,MSEC,SEC MIN,REVS
	Enters length of time record		
TMBF	TIME BUFFER		
	Displays the time capture buffer		
TMCP	TIME CAPTUR		
	Presets instrument to the time capture mode		
TMDLrrrssss	TIME DELAY	$10^{\pm 38}$	USEC,MSEC,SEC
	Enters time delay of system to be fitted before curve fit is started		
TMEM	TEST MEMORY		
	Displays the memory internal diagnostics test menu		
TMOD (Bus-only)	TIMEOUT DISABLE		
	Disables ability to abort bus activity if instrument is the controller and a device under its control fails to respond within 5 seconds		
TMOE (Bus-only)	TIMEOUT ENABLE		
	Enables ability to abort bus activity if instrument is the controller and a device under its control fails to respond within 5 seconds		
TMR1	TIME REC 1		
	Displays the current time record for Channel 1		
TMR2	TIME REC 2		
	Displays the current time record for Channel 2		
TMRC	TIME RECORD		
	Displays the time record outlined in the time buffer by the capture pointer		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
TMTH	TIME THRUPT		
	Presets instrument to the linear resolution mode and activates time throughput		
TOPR	TO→ POL RESIDU		
	Converts a pole/zero or polynomial table to the pole/residue format		
TOPY	TO→ POLY		
	Converts a pole/zero or pole/residue table to ratio-of-polynomials format		
TOPZ	TO→ POL ZERO		
	Converts a pole/residue or polynomial table to the pole/zero format		
TPRC	TEST PROC		
	Displays the processing assemblies internal diagnostics test menu		
TRAC	TRACE		
	Displays the instrument data trace		
TRAPrr	TRACE A PEN	see comment	
	Defines pen number for plotting Trace A (range depends on number of pens in plotter)		
TRBPrr	TRACE B PEN	see comment	
	Defines pen number for plotting Trace B (range depends on number of pens in plotter)		
TRCA	TRACE A		
	Enters trace currently displayed in Trace A for math operations		
TRCB	TRACE B		
	Enters trace currently displayed in Trace B for math operations		
TRES	TEST RESULT		
	Displays the test and fault logs menu		
TRGD	TRIG DELAY		
	Displays the trigger delay menu		
TRLVrrrsss	TRIG LEVEL	$10^{\pm 38}$	V,MV,EUC1,EUC2
	Defines trigger level amplitude (limit is 10V for ext trigger)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
TSRC	TEST SOURCE		
	Displays the source internal diagnostics test menu		
TST	SELF TEST		
	Runs an internal diagnostic test (see Service Manual)		
TT	T/1—T		
	Calculates the open-loop response from a measured closed-loop response		
UBIT	UPR 13 BITS		
	Selects the upper 13 bits of the 16 bit digital data to be used as input		
UCRRrrrsss	USER CRRIER	current span	MHZ,HZ,KHZ RPM,ORD
	Defines the carrier frequency to be used in PM and FM demodulation		
UFLBn1,n2,[,count] (Bus-only)	UNFLOAT BLOCK		
	Converts floating point data in block n1 into integer data and puts result in n2 (count is number of points from n1 to convert)		
UFLNrr	UNFIX LINE#	1–20	
	Unfixes a previously fixed line in the curve fit table		
ULIM	USER LIMITS		
	Displays the plot user limits menu		
ULIN	USER LINES		
	Displays the user line type softkeys used to define custom line types		
UNIF	UNIFRM (NONE)		
	Selects the uniform window		
UNIT	UNITS		
	Displays the units menu		
UNLB (Bus-only)	UNIT LABEL		
	Displays units associated with most recent numeric entry (UNLB? allows unit labels to be read over HP-IB)		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
UP	UP ARROW		
	Increments active numeric entry		
UPLO	UPPER LOWER		
	Places display in upper/lower format with trace A in upper half and trace B in lower half		
USD1	USER SAVD 1		
	Selects the time waveform stored in SAVED 1 as the window		
USD2	USER SAVD 2		
	Sets the source type to be arbitrary data from the User Save Data Block 2		
USEC	μSec		
	Specifies numeric entry in microseconds		
USOR	USER ORDER		
	Causes curve fitter to find number of poles and zeros specified with NUMBER POLES (NPOL) and NUMBER ZEROS (NZER)		
USRQ	USER SRQ		
	Displays the user SRQ menu		
USWT	USER WEIGHT		
	Causes curve fitter to use the user-defined weighting function during fits		
V	V		
	Specifies numeric entry in volts		
V/HZ	$V/\sqrt{}$ Hz		
	Terminates numeric entry		
V2	V^2		
	Specifies numeric entry in volts squared		
V2/H	V^2/Hz		
	Terminates numeric entry		
V2HZ	V^2/Hz(PSD)		
	Displays trace in volts squared divided by frequency		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
V2SH	V^2s/Hz(ESD)		
	Displays trace in volts squared times record length divided by frequency		
VBLKn (Bus-only)	VECTOR DISP BLOCK	0–19	
	Selects the vector display buffer to be dumped with DVAN,DVAS,or DVBN		
VEU	V/EU		
	Specifies numeric entry in volts per defined engineering unit		
VHZ	$V/\sqrt{Hz}$ ($\sqrt{PSD}$)		
	Displays trace in volts divided by the square root of the frequency		
VIEW	VIEW		
	Displays first page of an autosequence program (measurement must be paused)		
VISU	VISUAL HELP		
	Displays the visual help menu		
VLT1	VOLTS CHAN1		
	Selects volts on the Y-axis for Channel 1 and disables engineering units		
VLT2	VOLTS CHAN2		
	Selects volts on the Y-axis for Channel 2 and disables engineering units		
VLTS	VOLTS		
	Displays trace in volts		
VRMS	Vrms		
	Defines source level in volts rms		
VS/H	V^2S/Hz		
	Terminates numeric entry		
VT2	$VOLTS^2$		
	Displays trace in volts squared		
VTPK	VOLTS PEAK		
	Defines volts peak as the basic voltage unit		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
VTRM	VOLTS RMS		
	Defines volts rms as the basic voltage unit		
VWIN	VIEW INPUT		
	Displays the view input menu		
VWMA	VIEW MATH		
	Displays the auto math table (measurement in progress must be paused)		
VWOF	VIEW OFF		
	Pauses the view input displays		
VWWT	VIEW WEIGHT		
	Displays the weighting function in the bottom trace		
WARPrrss	FREQ WARP	1E−6 to 1E6	KHZ,HZ,MHZ
	Specifies frequency at which the amplitude of the z-domain frequency response will match that of the s-domain frequency response		
WNDO	WINDOW		
	Displays the window menu		
WRIT'aaaaaaaa' (Bus-only)	WRITE TEXT		
	Writes text to a display buffer		
WTRGrrrsss	WEIGHT REGION	0−100 kHz	MHZ,HZ,KHZ
	Specifies portion of weighting function to be modified by WEIGHT VALUE		
WTVLrrr	WEIGHT VALUE	$10^{\pm 38}$	
	Defines value of weighting function inside region identified by WEIGHT REGION		
X	X		
	Activates the X marker and displays its menu		
XASC	X AUTO SCALE		
	Automatically adjusts horizontal scale to display all measured data		
XAVGn1,n2,awf (Bus-only)	EXPON AVG		
	Averages data blocks using an exponentially weighted averaging formula		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
XFOF	X FCTN OFF		
	Turns off the special marker functions		
XMKR	X MRKR SCALE		
	Changes horizontal scale to range currently defined by the X marker in one of the HOLD X modes		
XOFF	X OFF		
	Deactivates the X marker		
XOVFr	X OVFL ON OFF	0 or 1	
	Enables the OVF input line on POD Q to monitor overflow conditions in the system under test		
XPN1rrsss	EXPON CHAN1	$10^{\pm 38}$	USEC,MSEC,SEC MIN,REVS
	Selects exponential window for Channel 1		
XPN2rrsss	EXPON CHAN2	$10^{\pm 38}$	USEC,MSEC,SEC MIN,REVS
	Selects exponential window for Channel 2		
XPNDr	EXPAND	2,4 or 8	
	Expands a segment of data in the active trace to fit the entire trace		
XSCLrrrsss	X FIXD SCALE		
	Specifies the horizontal scale (range and suffix depend on display)		
XTRCrrr,rrrsss	EX- TRACT	see comment	MHZ,HZ,KHZ
	Extracts data in the active trace between specified start and stop values(range depends on suffix and setup)		
XVALrrrsss	X VALUE		
	Moves the X marker to a specific point on the trace (range and suffix depend on display)		
Y	Y		
	Activates the Y marker and displays its menu		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

MNEM/SYNTAX	KEYNAME/COMMAND	RANGE	SUFFIXES
	COMMENTS		
YASC	Y AUTO SCALE		
	Automatically adjusts vertical axis to obtain optimum display		
YDSC	Y DFLT SCALE		
	Adjusts vertical scale to default value		
YMKR	Y MRKR SCALE		
	Changes vertical scale to the scale currently defined by the Y marker in one of the HOLD Y modes		
YOFF	Y OFF		
	Deactivates the Y marker		
YSCLrrrsss YSCLrrr,rrrsss	Y FIXD SCALE		
	Specifies the vertical scale (range and suffix depend on display)		
YVALrrrsss YVALrrr,rrrsss	Y VALUE		
	Moves the Y marker to a specific point on the trace (range and suffix depend on display)		
ZDOM	Z DOMAIN		
	Creates a trace using the current z-domain synthesis table		
ZOOM	ZOOM		
	Runs an internal diagnostic test (see Service Manual)		
ZPWRr	Z PWR + −	0 or 1	
	Changes the power of z for the current table (0 – negative, 1 = positive)		
ZSDM	Z & S DOMAIN		
	Creates a trace in the active trace using the current z-domain synthesis table followed by the current s-domain synthesis table		
ZST	ZERO START		
	Selects zero Hz as the start frequency		

r = values within the range specified in the RANGE column
s = one of the suffixes from the SUFFIX column
a = alphanumeric character
[ ] = optional parameter

Example Programs

Purpose of This Appendix

This appendix contains example HP BASIC 5.0 programs written for the HP 3563A. These programs were written to provide you with ideas for controlling the HP 3563A via HP-IB. They are not intended to be final solutions to any particular programming problems, but rather to demonstrate the analyzer's power and flexibility.

Note These programs are not warranted, guaranteed, or supported by Hewlett-Packard or any of its representatives in any manner whatsoever.

Description of Demo Programs

Example Program #1 – Pass Control

Demonstrates the HP 3563A's ability to be a controller on the HP-IB bus and communicate directly with a plotter. The key elements of the program are setting up an interrupt branch, enabling interrupts on the HP-IB interface, enabling the HP 3563A to generate service request interrupts, processing service requests, and passing active control of the system to the HP 3563A allowing it to control the plotter directly.

Example Program #2 – Dump Data Trace

Demonstrates the HP 3563A's ability to dump an active data trace to a controller. The key elements of the program are setting up appropriate size arrays to hold the data header and data trace, dumping the data to the controller, determining data limits by reading elements of the data header, and using the data limits to scale and plot the data on the controller's CRT display.

Example Program #3 – 1/3 Rd Octave Analysis

This program does a simulated 1/3 octave analysis of a data trace dumped from the HP 3563A. The key elements of this program are dumping a trace from the HP 3563A, processing the data (in this case reformatting the data in 1/3 octaves), and loading the processed data trace back into the HP 3563A for display.

Example Program #4 – Dump Coordinate Transform Block

This program plots a spectral map on the controller display by doing 25 successive reads of the coordinate transform block from the HP 3563A. The key elements of the program are setting up and starting a measurement, enabling the HP 3563A to generate a service request when the measurement is completed, servicing the interrupt request, reading the data from the HP3563A, and plotting it on the controller display.

Example Program #5 – HP 3563A Plot

This program demonstrates the ability of the HP 3563A to dump its display buffers to a controller. The key elements of the program are reading the display buffers, translating the information in the buffers to HP-GL commands, and plotting the information on the controller display. This program also illustrates the use of the HP BASIC "SYMBOL" command which is used to specify and draw the nonstandard characters that are available on the HP 3563A display screen.

Example Program #6 – Composite Demo Program

This is a softkey driven program that illustrates most of the concepts used in the previous example programs. In addition it contains several demonstration routines that illustrate direct programming of the HP 3563A's display, as well as other useful routines. From a programming standpoint, this program illustrates using subprograms to perform specific tasks, resulting in more readable and supportable code. Many of the subprograms included are generic enough that they could be used in a wide variety of HP 3563A control programs with little or no modification.

APPENDIX B - EXAMPLE PROGRAM 1

```
!//////////////////////////////////////////////////////////////////////
!
!             DEMO PROGRAM PASS CONTROL
!
!    (c)    COPYRIGHT 1985, Hewlett-Packard Co.
!              last update 8-14-89
!                  BASIC 5.0
!
!  PURPOSE:
!
!  This program responds to a request for service by the HP3563A so that
!  it can make a direct digital plot of its display screen to a plotter.
!  The contents of the HP 3563A's fault log are displayed and the user
!  initiates the plot by pressing the 'PLOT' and 'START PLOT' keys.
!  The Series 200 controller passes control of the HPIB bus to the
!  HP 3563A, which then talks directly to the plotter.  When the plot is
!  finished the HP 3563A generates another request which gives control of
!  the bus back to the Series 200 controller.  If this program is run
!  without a plotter on the bus, the HP 3563A generates its own internal
!  HPIB timeout after a few seconds and passes control back to the
!  Series 200 controller.
!
!  SETUP:
!
!  Connect a HP-GL plotter set to address 5 to the bus.  (Note: the
!  essential elements of the program will work even without a plotter
!  connected to the system).
!
!  DATA DICTIONARY:
!
!  Spoll             Masked serial poll byte
!
!  Dsa               HP-IB code assignment of the HP 3563A
!
!  Hpib_intr         HP-IB interupt service routine
!
!  Ready_to_plot     Flag set when HP 3563A needs control of the bus.
!
!  Plot_done         Flag set when HP 3563A is finished with the bus.
!
!//////////////////////////////////////////////////////////////////////
!
Dsa=720                               ! Specify HP-IB address of HP 3563A.
DISP "Setting up HP 3563A....."
OUTPUT Dsa;"RST;PAUS;FL"              ! Tell HP 3563A to show the Fault Log.
OUTPUT Dsa;"CTAD21"                   ! Tell HP 3563A the address of the
                                      !   system controller.
WAIT 5
ON INTR 7,1 GOSUB Hpib_intr           ! Setup interrupt branch and
ENABLE INTR 7;2                       !   enable interrupts.
```

```
52     OUTPUT Dsa;"SRQE"                    ! Enable HP 3563A to generate
                                              interrupts.
53     LOCAL Dsa                            ! Put HP 3563A in local mode.
54     !
55     DISP "Press 'PLOT' hardkey and 'START PLOT' softkey to plot screen."
56     !
57     REPEAT                               ! Wait for user to initiate plot by
58     UNTIL Ready_to_plot                  !   pressing the 'START PLOT' softkey.
59     PASS CONTROL Dsa                     ! HP 3563A becomes active controller.
60     ENABLE INTR 7;2                      ! Re-enable interrupts.
61     !
62     DISP "Control passed.  HP 3563A talking directly to plotter."
63     !
64     REPEAT                               ! Wait for interrupt signifying that
65     UNTIL Plot_done                      !   the plot is done.
66     DISP "Program complete."
67     STOP
68     !
69 Hpib_intr:                               ! Interrupt service routine.
70     Spoll=BINAND(SPOLL(Dsa),143)         ! Mask out bits 4,5,6.
71     IF Spoll>=14 AND Spoll<=143 THEN     ! Determine if plot is ready to
72       Ready_to_plot=1                    !   start or....
73     ELSE                                 !
74       IF Spoll=10 THEN                   !   if plot is done.
75         Plot_done=1
76       END IF
77     END IF
78     RETURN
79     !
80     END
```

APPENDIX B - EXAMPLE PROGRAM 2

```
81    !///////////////////////////////////////////////////////////////////////////
82    !
83    !///////////////////////////////////////////////////////////////////////////
84    !
85    !            DEMO PROGRAM DUMP DATA TRACE
86    !
87    !      (c)  COPYRIGHT 1985, Hewlett-Packard Co.
88    !               last updated 8-14-89
89    !                    BASIC 5.0
90    !
91    !
92    !  PURPOSE:
93    !
94    !  This program will read data directly from the HP 3563A over the
95    !  HP-IB bus using the capability of the Series 200 controller.
96    !  The data is assumed to be linear resolution data and is plotted
97    !  in real and, if complex, imaginary format.
98    !
99    !  SETUP:
100   !
101   !  User sets up any type of measurement using linear resolution before
102   !  executing the program.
103   !
104   !  DATA DICTIONARY:
105   !
106   !   Max_val(*)       The data array max and/or min value used
107   !   Min_val(*)       in determining the plotting limit.
108   !
109   !   Header_len       Data header length (constant)
110   !
111   !   Data_len         Data buffer length (bytes)
112   !
113   !   N_points         Number of data points
114   !
115   !   Start_f          Start frequency
116   !
117   !   Delta_f          Frequency or time spacing
118   !
119   !   Hbuf(*)          Real buffer containing data header
120   !
121   !   Fbuf(*)          Real buffer containing data trace
122   !
123   !///////////////////////////////////////////////////////////////////////////
124   !
125   DIM A$[2],Max_val(1:2),Min_val(1:2)
126   INTEGER I,Real,Imag,Mag,Phase
127   Real=1                                    ! Initialize variables
128   Imag=2
129   GINIT                                     ! Initialize graphics raster
130   !
```

```
131    Header_len=66                              ! Data header length
132    ASSIGN @Io TO 720                          ! Setup I/O path to HP 3563A.
133    REMOTE @Io
134    !
135    !   GET DATA
136    !
137    DISP "Getting trace data from 3563A....."
138    OUTPUT @Io;"DDAN"                          ! Dump data ANSI format
139    ENTER @Io USING "#,2A,W";A$,Data_len
140    ASSIGN @Io;FORMAT OFF                      ! Turn ASCII formatter off
141    ALLOCATE REAL Hbuf(1:Header_len)
142    ENTER @Io;Hbuf(*)                          ! Read data header
143    !
144    !  EXTRACT HEADER INFORMATION
145    !
146    N_points=Hbuf(2)                           ! Number of data points
147    Cmplx_flg=Hbuf(37)                         ! Complex data flag
148    Start_f=Hbuf(66)                           ! Data start frequency
149    Delta_f=Hbuf(56)                           ! Delta frequency or time
150    IF Cmplx_flg=1 THEN
151      ALLOCATE Fbuf(1:N_points,1:2)
152    ELSE
153      ALLOCATE Fbuf(1:N_points,1:1)
154    END IF
155    ENTER @Io;Fbuf(*)                          ! Read data trace
156    ASSIGN @Io;FORMAT ON
157    !
158    ! FIND MAX VALUE
159    !
160    DISP "Finding minimum and maximum values for plot...."
161 Max_min:  ! Calculates the MAX and MIN for plotting
162           ! Initialize Variables
163    Max_val(Real)=0                            ! Real trace Max
164    Max_val(Imag)=0                            ! Imag trace Max
165    Min_val(Real)=0                            ! Real trace Min
166    Min_val(Imag)=0                            ! Imag trace Min
167    FOR I=1 TO N_points                        ! Find Max's and Min's
168      FOR J=1 TO Cmplx_flg+1
169        IF Fbuf(I,J)>Max_val(J) THEN Max_val(J)=Fbuf(I,J)
170        IF Fbuf(I,J)<Min_val(J) THEN Min_val(J)=Fbuf(I,J)
171      NEXT J
172    NEXT I
173 Plot_out:                                     ! Plots data
174    GCLEAR                                     ! Clear graphics raster.
175    GRAPHICS ON                                ! Graphics raster on.
176    X_min=Start_f                              ! Set x-axis limits.
177    X_max=(N_points-1)*Delta_f+Start_f
178    ALPHA OFF                                  ! Alpha raster off.
179    FOR K_funct=1 TO Cmplx_flg+1
180      IF Cmplx_flg=0 THEN                      ! Specify plotting area, if
181        VIEWPORT 10,110,15,85                  ! imaginary data available
182      ELSE                                     ! then each plot gets half
183        VIEWPORT 10,110,15,48                  ! of the controller display.
184      END IF
```

```
185     IF K_funct=2 THEN VIEWPORT 10,110,53,85
186     WINDOW X_min,X_max,Min_val(K_funct),Max_val(K_funct)
187     MOVE Start_f,Fbuf(1,K_funct)
188     FOR I=2 TO N_points                       ! Plot the data.
189       PLOT Start_f+(I*Delta_f),Fbuf(I,K_funct)
190     NEXT I
191   NEXT K_funct
192  !
193 Border:                                       ! Plots borders around data.
194   VIEWPORT 10,110,10,90
195   WINDOW 0,1000,0,1000
196   MOVE 0,500
197   DRAW 0,1000
198   PLOT 1000,1000
199   PLOT 1000,500
200   PLOT 0,500
201   PLOT 0,0
202   PLOT 1000,0
203   PLOT 1000,500
204  !
205   DISP
206   LOCAL @Io
207   END
```

APPENDIX B - EXAMPLE PROGRAM 3

```
208   !//////////////////////////////////////////////////////////////////////
209   !
210   !//////////////////////////////////////////////////////////////////////
211   !
212   !          DEMO PROGRAM 1/3 RD OCTAVE
213   !
214   !    (c) COPYWRIGHT 1985, Hewlett-Packard Co.
215   !             last update 8-14-89
216   !                 BASIC 5.0
217   !
218   !  PURPOSE:
219   !
220   !  This program will read data directly from a HP 3563A analyzer over the
221   !  HPIB bus using a Series 200 controller.  The HP 3563A must be in LOG
222   !  RESOLUTION mode and the measurement units must be volts^2.  The data
223   !  trace is dumped to the Series 200 controller and is converted to a
224   !  psuedo 1/3 octave format and dumped back to the HP 3563A.
225   !
226   !
227   !  SUB PROGRAMS REQUIRED:
228   !
229   !    F_shape            Computes the ANSI class III filter shape.
230   !
231   !
232   !  DATA DICTIONARY:
233   !
234   !
235   !    Header_len         Data header length (constant).
236   !
237   !    Data_len           Data buffer length (bytes).
238   !
239   !    N_points           Number of data points.
240   !
241   !    Start_f            Start frequency.
242   !
243   !    Delta_f            Frequency spacing in decades per point.
244   !
245   !    Pt_dec             Points per decade.
246   !
247   !    Hbuf(*)            Real buffer containing data header.
248   !
249   !    Fbuf(*)            Real buffer containing log res data.
250   !
251   !    Oct buf(*)         Buffer with synthesized 1/3 octave data.
252   !
253   !//////////////////////////////////////////////////////////////////////
254   !
255   INTEGER I,N_points,Header_len,Pt_dec,N_fact,Flag
256   N_fact=32                             ! +- Number of lines in 1/3
257   ALLOCATE Trans(-N_fact:N_fact)        !    octave filter.
```

```
258    !
259    CLEAR SCREEN
260    Header_len=66
261    GOSUB Get_data                            ! Gets data from the HP 3563A.
262    GOSUB Get_pwr                             ! Reads total power using markers
263    GOSUB Oct_1_3                             ! Calculates 1/3d Octave Spectrum
264    GOSUB Restore_dat                         ! Restores data to analyzer.
265    LOCAL @Io                                 ! Puts analyzer in local mode.
266    DISP "Program finished."
267    STOP
268    !
269 Get_pwr:    !                                ! Reads power using power marker.
270    OUTPUT @Io;"XOFF;PWR;RSMO"                ! Read special marker for power.
271    ENTER @Io;Pwr_a,Pwr_b
272    RETURN
273    !
274 Get_data:                                    ! Reads data block
275    ASSIGN @Io TO 720                         ! Setup I/O path to HP 3563A.
276    ASSIGN @Io;FORMAT ON                      ! Turn ascii formatter on.
277    REMOTE @Io                                ! Put HP 3563A in remote mode.
278    OUTPUT @Io;"COME"                         ! Command echo enable.
279    OUTPUT @Io;"DDAN"                         ! Dump data trace in ANSI binary.
280    DISP "Getting data trace from HP 3563A...."
281    ENTER @Io USING "#,2A,W";A$,Data_len      ! Enter format specifier.
282    ASSIGN @Io;FORMAT OFF                     ! Turn ascii formatter off.
283    ALLOCATE REAL Hbuf(1:Header_len)          ! Allocate array for data header.
284    ENTER @Io;Hbuf(*)                         ! Enter the header.
285    CALL Fshape(Trans(*),N_fact)              ! Calculates 1/3d Octave filter.
286    N_points=Hbuf(2)                          ! Read pertinent header info...
287    Cmplx_flg=Hbuf(37)                        !   "     "         "      "
288    Log_data=Hbuf(41)                         !   "     "         "      "
289    Amp_units=Hbuf(10)                        !   "     "         "      "
290    IF Log_data=0 OR Amp_units<>1 THEN        ! If data not in LOG RES mode or
291      BEEP                                    !   incorrect units selected....
292      CLEAR SCREEN
293      PRINT "Data not in proper Measurement Mode for 1/3 octave analysis."
294      PRINT "Measurement must be made in LOG RESOLUTION mode and in amplitude"
295      PRINT "units of Volts^2."
296      PRINT
297      PRINT "Press 'RUN' to start program again."
298      CLEAR @Io
299      LOCAL @Io
300      STOP
301    END IF
302    Hbuf(10)=0
303    Start_f=Hbuf(66)                          ! Reads start frequency.
304    Pt_dec=1/Hbuf(56)                         ! Reads points per decade.
305    Delta_f=1/Pt_dec                          ! Calculates decades per point.
306    ALLOCATE Fbuf(1:N_points)                 ! Allocate array for data trace.
307    ALLOCATE Oct_buf(1:N_points)              ! Allocate array for converted
308                                              !   1/3 octave data.
309    ENTER @Io;Fbuf(*)                         ! Enter the data trace.
310    ASSIGN @Io;FORMAT ON                      ! Turn ascii formatter on.
311    DISP "Data transfer complete."
```

```
312    RETURN
313    !
314  Oct_1_3:                                  ! Reformats data in 1/3 Octaves
315    FOR I=1 TO N_points STEP 8
316      Oct_buf(I)=0
317      FOR J=-(N_fact-1) TO (N_fact-1)
318        IF (I-J)<1 OR (I-J)>N_points THEN
319          IF (I-J) <1 THEN Oct_dum=Fbuf(2)
320          IF (I-J)>N_points THEN Oct_dum=Fbuf(N_points)
321        ELSE
322          Oct_dum=Fbuf(I-J)
323        END IF
324        Oct_buf(I)=Oct_dum*Trans(J)+Oct_buf(I)
325      NEXT J
326      Oct_dum=Oct_buf(I)
327      FOR J=-3 TO 4
328        IF (I+J)>=1 AND (I+J)<=N_points THEN
329          Oct_buf(I+J)=SQR(Oct_dum)
330        END IF
331      NEXT J
332    NEXT I
333    PRINT "Total Power is = ";Pwr_a;" dB"
334    RETURN
335    !
336  Restore_dat:   !
337    DISP "Sending simulated 1/3 octave response to HP 3563A."
338    OUTPUT @Io;"LDAN"                        ! Tell HP3563A to load data trace
339    OUTPUT @Io USING "#,2A,W";"#A",Data_len  !   from the controller.
340    ASSIGN @Io;FORMAT OFF                    ! Turn ascii formatter off.
341    OUTPUT @Io;Hbuf(*);Oct_buf(*);END        ! Send data to the HP 3563A.
342    RETURN
343    !
344    END
345    !///////////////////////////////////////////////////////////////////////////
346    SUB Fshape(Trans(*),INTEGER N_fact)
347    !
348    !  Subprogram to calculate the filter shape of a 1/3rd octave class III
349    !
350    !
351      INTEGER N
352      FOR N=-N_fact TO N_fact
353        IF N<=4 AND N>=-4 THEN
354          Atten=1
355        ELSE
356          Atten=(8/13+2500*(10^(N/80)-10^(-N/80))^6)
357        END IF
358        Trans(N)=1/Atten
359      NEXT N
360    SUBEND
```

APPENDIX B - EXAMPLE PROGRAM 4

```
361   !//////////////////////////////////////////////////////////////////////////
362   !
363   !//////////////////////////////////////////////////////////////////////////
364   !
365   !   DEMO PROGRAM DUMP COORDINATE TRANSFORM BLOCK
366   !
367   !     (c) COPYRIGHT 1985, Hewlett-Packard Co.
368   !             last update  8-14-89
369   !                  BASIC 5.0
370   !
371   !
372   !  PURPOSE:
373   !
374   !  This program will read the coordinate transform block from
375   !  the HP 3563A analyzer over the HP-IB bus using the
376   !  capability of the Series 200 controller.  The data is assumed
377   !  to be dB magnitude data and Hz frequency domain power
378   !  spectrum data.  The data is repeatedly read and displayed
379   !  in a spectral map format.  Only the data actually displayed
380   !  is read and plotted.
381   !
382   !  DATA DICTIONARY:
383   !
384   !   Header_len       Data header length (constant)
385   !
386   !   Chead_len        Coordinate transform header length
387   !
388   !   Data len         Data buffer length (bytes)
389   !
390   !   N_points         Number of data points
391   !
392   !   Cbuf(*)          Real buffer for coord transform header
393   !
394   !   Hbuf(*)          Real buffer containing data header
395   !
396   !   Buff(*)          Real buffer containing coord trans data
397   !
398   !   Mask(*)          Data buffer containing max values; used
399   !                    for hidden line calculations
400   !
401   !   Penc(*)          Pen control buffer for hidden lines
402   !
403   !//////////////////////////////////////////////////////////////////////////
404   !
405     CLEAR SCREEN
406     KEY LABELS OFF                         ! Turn softkeys off.
407     INTEGER I,Done_flg
408     DIM A$[3]
409   !
410   !  Initialize constants....
```

```
411   !
412     First_read=1                      ! Flag denotes first data read.
413     Header_len=66                     ! Data header length.
414     Chead_len=50                      ! Coord transform header length.
415     Done_flg=0                        ! Measurement done flag.
416     N_spect=25                        ! # of spectral lines to plot.
417   !
418     ALLOCATE REAL Hbuf(1:Header_len),Cbuf(1:Chead_len)
419   !
420 Control:  !
421     ASSIGN @Io TO 720                 ! Setup I/O path to HP 3563A.
422     OUTPUT @Io;"COMD"                 ! Disable command echo.
423     OUTPUT @Io;"UNIT; HZS"            ! Sets X axis units to Hertz.
424     OUTPUT @Io;"MGDB"                 ! Sets Y axis to Mag dB.
425     GOSUB Hpib_init                   ! Enable HP-IB interrupts.
426     GOSUB Start_meas                  ! Start the measurement.
427     GOSUB Get_block                   ! Read coordinate xform block.
428     GOSUB Plot_init                   ! Setup plot parameters.
429     GOSUB Hpib_init                   ! Re-enable interrupts.
430     GOSUB Start_meas                  ! Get another measurement started.
431   !
432   ! In the following loop the current measurement is taking place while the
433   !   data from the previous measurement is being plotted.  Each time a
434   !   measurement finishes it causes a branch to the interrupt routine
435   !   which sets 'done_flg' true.
436   !
437     FOR K=0 TO N_spect-1
438       WHILE NOT Done_flg
439       END WHILE
440       GOSUB Get_data
441       GOSUB Start_meas
442       GOSUB Plot_out
443     NEXT K
444   !
445     Junk=SPOLL(@Io)                   ! Clear last service request.
446     OUTPUT @Io;"SRQD"                 ! Disable further interrupts.
447     LOCAL @Io                         ! Put HP 3563A in local mode.
448     DISP RPT$(" ",60)&CHR$(129)&"Program finished."&CHR$(128)
449     STOP                              ! End program.
450   !
451 Hpib_init:   !
452     OUTPUT @Io;"SRQE "                ! Enable SRQ's.
453     OUTPUT @Io;"ISM 4"                ! End of Measurement status mask
454     ON INTR 7 GOSUB Hpib_intr         ! Specify interrupt branch.
455     ENABLE INTR 7;2                   ! Enable HP-IB interrupts.
456     RETURN
457   !
458 Start_meas:   !
459     Done_flg=0                        ! Reset flag.
460     OUTPUT @Io;"STRT"                 ! Start measurement.
461     RETURN
462   !
463 Hpib_intr: ! Processes End of Measurement interrupts.
464     Spoll_byte=SPOLL(@Io)             ! Read HP 3563A status byte.
```

```
465      Stest_byte=BINAND(Spoll_byte,143)     ! Mask out bits 4,5,6.
466      IF Stest_byte=11 THEN                 ! If 'end-of-measurement' then...
467        ASSIGN @Io;FORMAT ON                ! Turn on ascii formatter.
468        OUTPUT @Io;"IS?"                    ! Read the HP3563A's Instrument
469        ENTER @Io;Stat                      !   status register and check for
470        IF BINAND(Stat,4)=4 THEN Done_flg=1!   'end-of-measurement' bit set.
471      END IF
472 End_intr:ENABLE INTR 7                    ! Re-enable interrupts.
473      RETURN
474    !
475 Get_block: !
476      OUTPUT @Io;"DCAN"                     ! Tell HP 3563A to dump block.
477      ENTER @Io USING  #,2A,W ;A$,Data_len ! Enter format specifier.
478      ASSIGN @Io;FORMAT OFF                 ! Turn ascii formatter off.
479      ENTER @Io;Cbuf(*);Hbuf(*)             ! Read Transform and Data headers.
480      IF First_read THEN                    ! For first read of program...
481        N_points=Cbuf(2)                    ! ...determine # of datapoints and
482        ALLOCATE Buff(0:N_points-1)         !    allocate an array of the
483        First_read=0                        !    appropriate size.
484        ALLOCATE Penc(0:N_points-1),Mask(0:N_points-1)
485      END IF
486      ENTER @Io;Buff(*)                     ! Read Coordinate Transform block.
487      ASSIGN @Io;FORMAT ON                  ! Turn ascii formatter on.
488      RETURN
489    !
490 Get_data:  ! Gets data and calculates hidden lines
491    !
492      FOR I=N_points-N_delta_x TO N_points-1! This loop does mask updating.
493        Mask(I)=(Y_min-Delta_y)
494      NEXT I
495    !
496      FOR I=N_delta_x TO N_points-1         ! This loop does X and Y axis
497        Buff(I)=Buff(I)-Delta_y             !   shifting.
498        Mask(I-N_delta_x)=MAX(Mask(I)-Delta_y,Buff(I))
499      NEXT I
500    !
501      GOSUB Get_block                       ! Go get the block of data.
502    !
503      FOR I=0 TO N_points-1                 ! This loop sets plot boundaries.
504        IF Buff(I)<Y_min1 THEN Buff(I)=Y_min1
505        IF Buff(I)>Y_max1 THEN Buff(I)=Y_max1
506      NEXT I
507    !
508      MAT Penc= Buff-Mask                   ! This loop sets of the pen
509      FOR I=0 TO N_points-1                 !   control for plotting.
510        Penc(I)=SGN(Penc(I))
511      NEXT I
512      Done_flg=1                            ! Set flag.
513      RETURN
514    !
515 Plot_out:   !
516    ! Set viewport boundaries to match spec'd min/max's.
517      X1=X_min_view+X_inc*K                 ! Calculate x-axis lower boundary.
518      X2=X_min_view+X_delta_view+X_inc*K    ! Calculate x-axis upper boundary.
```

```
519       Y1=Y_min_view+Y_inc*K                    ! Calculate y-axis lower boundary.
520       Y2=Y_min_view+Y_delta_view+Y_inc*K       ! Calculate y-axis upper boundary.
521       VIEWPORT X1,X2,Y1,Y2                     ! Set hard-clip limits.
522       WINDOW 0,N_points-1,Y_min,Y_max          ! Setup user units for plot.
523       MOVE 0,Buff(0)
524       FOR I=1 TO N_points-1                    ! This loop puts the pen control
525         Pnt_cnt=Penc(I)-1                      !   parameter in the proper format
526         IF Pnt_cnt=0 THEN Pnt_cnt=1            !   and plots the data on the
527         PLOT I,Buff(I),Pnt_cnt                 !   controller's crt display.
528       NEXT I
529       RETURN
530     !
531   Plot_init:    ! Initialize plot
532       PLOTTER IS CRT,"INTERNAL"                ! Specify crt as plotting device.
533       GINIT                                    ! Initialize graphics raster.
534       GCLEAR                                   ! Clear graphics raster.
535       GRAPHICS ON                              ! Turn graphics raster on.
536       Y_min1=Cbuf(34)                          ! Read Y min from header.
537       Y_max1=Cbuf(35)                          ! Read Y max from header.
538       Y_scale_f=Cbuf(41)                       ! Amplitude scale factor.
539       X_min=Cbuf(49)                           ! Read X min from header.
540       X_max=Cbuf(50)                           ! Read X max from header.
541       Y_off=ABS(.05*(Y_min1-Y_max1))           ! Cal offset= 5% full scale.
542       Y_min=Y_min1-Y_off                       ! Adjust Y min
543       Y_max=Y_max1+Y_off                       !    and Y max.
544       Y_delta=Y_max-Y_min                      ! Calculate Y span.
545     !
546     ! VIEWPORT VALUES FOR INDIVIDUAL SPECTRA
547     !        IN % OF FULL SCALE
548     !
549       INTEGER N_delta_x
550       Y_min_view=10                           ! Y min for single spectrum (in %).
551       X_min_view=10                           ! X min for single spectrum (in %).
552       Y_delta_view=45                         ! Single spectrum height (in %).
553       X_delta_view=80                         ! Single spectrum width (in %).
554       Y_delta_bound=85                        ! Entire map height (in %).
555       X_delta_bound=100                       ! Entire map width (in %).
556       Y_inc=(Y_delta_bound-Y_delta_view)/(N_spect-1)
557     !  Y_inc is incremental vertical movement (in %)
558       X_inc=(X_delta_bound-X_delta_view)/(N_spect-1)
559     !  X_inc is incremental horizontal movement (in %)
560       Delta_y=Y_inc*(Y_max-Y_min)/Y_delta_view
561     !  Delta_y is incremental vert Movement in plot units
562       N_delta_x=X_inc*(N_points-1)/X_delta_view
563     !  N_delta_x is incremental horizonal movement in number
564     !  of data points (rounded integer)
565     !
566     ! RECALULATE X_INC FOR INTEGER N_DELTA
567     !
568       X_inc=N_delta_x/(N_points-1)*X_delta_view
569       X_delta_bound=X_inc*(N_spect-1)+X_delta_view
570     !
571   Init_hidden:  ! Initial for hidden lines
572       MAT Buff= (Y_min1)                       ! Set to Min Y value
```

```
573     MAT Mask= (Y_min1)
574     ALPHA OFF
575   !
576 Plot_axis: !  DRAW THE AXIS AND BOUNDARIES OF THE PLOT
577     X1=X_min_view
578     X2=X_min_view+X_delta_bound
579     Y1=Y_min_view
580     Y2=Y_min_view+Y_delta_bound
581     VIEWPORT X1,X2,Y1,Y2
582     WINDOW X1,X2,Y1,Y2
583     Offset_y=.05*Y_delta_view
584     MOVE X1,Y1+Offset_y
585     DRAW X1,Y1
586     DRAW X1+X_delta_view,Y1
587     DRAW X1+X_delta_view,Y1+Offset_y
588     DRAW X2,Y2-Y_delta_view+Offset_y
589     DRAW X2,Y2
590     DRAW X2-X_delta_view,Y2
591     DRAW X2-X_delta_view,Y2-Y_delta_view+Offset_y
592     DRAW X1,Y1+Offset_y
593   !
594 Right_tics:  !  DOES VERTICAL TICK MARKS
595   ! Reset viewport and window
596     X1=X_min_view+X_inc*(N_spect-1)
597     X2=X_min_view+X_delta_view*1.2+X_inc*(N_spect-1)
598     Y1=Y_min_view+Y_inc*(N_spect-1)
599     Y2=Y_min_view+Y_delta_view+Y_inc*(N_spect-1)
600     VIEWPORT X1,X2,Y1,Y2
601     WINDOW 0,(N_points-1)*1.20,Y_min,Y_max
602     MOVE N_points-1,Y_min1
603     DRAW (N_points-1)*1.03,Y_min1         ! Draw lower tick mark
604     CSIZE (3)
605     LORG (2)
606     Y_label$="dB"
607     Y_fmt$="X,SDDD.D"
608     LABEL USING Y_fmt$;(Y_min1*Y_scale_f)
609     MOVE N_points-1,Y_max1
610     DRAW (N_points-1)*1.03,Y_max1         ! Draw upper tick mark
611     LABEL USING Y_fmt$;(Y_max1*Y_scale_f)
612     MOVE (N_points-1)*1.05,Y_min1+(Y_max1-Y_min1)*.5
613     LABEL Y_label$
614   !
615 Lower_tics:  ! DOES FREOUENCY AXIS
616   ! Reset viewport and window
617     X1=X_min_view-X_delta_view*.10
618     X2=X_min_view+X_delta_view*1.15
619     Y1=Y_min_view-Y_delta_view*.15
620     Y2=Y_min_view+Y_delta_view
621     VIEWPORT X1,X2,Y1,Y2
622   !
623     X1=0-(N_points-1)*.10
624     X2=(N_points-1)*1.15
625     Y1=Y_min-Y_delta*.15*1.1
626     Y2=Y_max
```

```
627      WINDOW X1,X2,Y1,Y2
628      MOVE 0,Y_min
629      DRAW 0,Y_min-Y_off
630      LORG (6)
631      X_fmt$="SDD.D"
632      X_label$="HZ"
633      LABEL (X_min)
634      MOVE N_points-1,Y_min
635      DRAW N_points-1,Y_min-Y_off
636      LABEL (X_max)
637      MOVE (N_points-1)*.5,Y_min-Y_off
638      LABEL X_label$
639      RETURN
640    !
641      END
```

APPENDIX B - EXAMPLE PROGRAM 5

```
642   !/////////////////////////////////////////////////////////////////////////
643   !
644   !/////////////////////////////////////////////////////////////////////////
645   !
646   !           DEMO PROGRAM HP 3563A PLOT
647   !
648   !     (c) COPYRIGHT 1985, Hewlett-Packard Co.
649   !              last update 8-14-89
650   !     BASIC 5.0  (with GRAPH binary extension)
651   !
652   !
653   !  PURPOSE:
654   !
655   !  This program will dump the display buffers from the display of the
656   !  HP 3563A using the DUMP VECTOR BINARY command.  The program decode the
657   !  HP 1345 B2D2 display commands and translates them to HP-GL, then
658   !  plots it on the controller's CRT display.
659   !
660   !  SUB PROGRAMS USED:
661   !
662   !    Ggplot
663   !
664   !    Read binary
665   !
666   !  DATA DICTIONARY:
667   !
668   !  Ibuf(*)             Integer buffer for storing the dumped
669   !                      data prior to plotting.
670   !
671   !  Ix,Iy,Idx           Integer in range 0 to 2048 corresponding
672   !                      to the current value of the increment for
673   !                      use with the graph command.
674   !
675   !  Chr_buf             Common arrays containing the definition of
676   !                      the character set not standard to HP-GL.
677   !
678   !  First_dspbuf        First display buffer
679   !
680   !  Num_dspbuf          Number of display buffers used
681   !
682   !  Data_len            Length of buffer in bytes
683   !
684   !  @Io                 HP-IB code assignment of the 3563
685   !
686   !/////////////////////////////////////////////////////////////////////////
687     OPTION BASE 1
688     COM /Char_buf/ Triangle(6,3),Tri_2(5,3),Sqr_rt(5,3)
689     COM /Char_buf/ Rgt_arrow(6,3),Hp_log(20,3),Micro(8,3)
690     DIM A$[1]
691     INTEGER Ibuf(4096),Ix,Iy,Idx,Data_len,I,J
```

```
692   !
693   ! The following DATA statements describe non HP-GL standard characters
694   !   which are used in the HP1345 display.
695   !
696 Triangle:DATA 0,0,6, 0,0,-2, 9,4,-1
697     DATA 0,8,-1, 0,0,-1, 0,0,7
698 Tri_2:DATA 1,0,-2, 8,0,-1, 5,8,-1
699     DATA 1,0,-1, 0,0,-2
700 Sqr_rt:DATA 0,3,-2, 1,4,-1, 3,0,-1
701     DATA 6,8,-1, 0,0,-2
702 Rgt_arrow:DATA 0,4,-2, 8,4,-1, 6,6,-1
703     DATA 6,2,-2, 8,4,-1, 0,0,-2
704 Hp_log:DATA 6,4,-2, 6,12,-1, 6,8,-2
705     DATA 9,8,-1, 9,4,-1, 11,0,-2
706     DATA 11,8,-1, 14,8,-1, 14,4,-1
707     DATA 11,4,-1
708     DATA 0,2,-2, 0,10,-1, 2,12,-1
709     DATA 18,12,-1,  20,10,-1, 20,2,-1
710     DATA 18,0,-1, 2,0,-1, 0,2,-1
711     DATA 20,0,-2
712 Micro:DATA 0,0,-2, 2,8,-1, 1,4,-2
713     DATA 5,4,-1, 6,8,-1, 5,4,-2, 6,4,-1
714     DATA 0,0,-2
715   !
716   ! Load the non-standard characters into the following arrays.
717   !
718     READ Triangle(*),Tri_2(*),Sqr_rt(*),Rgt_arrow(*),Hp_log(*),Micro(*)
719   !
720 Begin:   !
721     ASSIGN @Io TO 720                 ! Set up I/O path to 3563A
722     GINIT                             ! Initialize graphics raster
723     GCLEAR                            ! Clear graphics raster
724     CLEAR SCREEN                      ! Clear alpha raster
725     KEY LABELS OFF                    ! Turn off softkeys.
726     PLOTTER IS CRT,"INTERNAL"         ! Designate crt as plot device
727     LORG 1                            ! Specify label origin
728     GRAPHICS ON                       ! Turn graphics raster on
729     CSIZE 3.4,.67                     ! Set character size
730     VIEWPORT 0,RATIO*100,20,100       ! Specify hard-clip area
731     WINDOW 0,2100,0,2100              ! Set window in HP1345 units
732     DEG                               ! Default to degrees
733     OUTPUT @Io;"COMD"                 ! Disable command echo
734     First_dspbuf=4                    ! Start with buffer 4 (ie. ignore
735                                       !    menu and command fields)
736     FOR J=First_dspbuf TO 17          ! 17 is last buffer
737       CALL Read_binary(@Io,"DVBN",J,Data_len,Ibuf(*))
738       IF Data_len<>0 THEN             ! Ignore if buffer empty
739         FOR I=1 TO Data_len DIV 2     ! Translate the commands to HP-GL
740           CALL Ggplot(Ibuf(I),Ix,Iy,Idx)! and plot to crt display
741         NEXT I
742       END IF
743     NEXT J
744     DISP CHR$(129)&" Program finished. "&CHR$(128)
745     END
```

```
746   !//////////////////////////////////////////////////////////////////////////
747   SUB Ggplot(INTEGER Coordt,Xcoord,Ycoord,Delta_x)
748   !
749   !  This subprogram translate the HP1745 commands to HP-GL commands.
750   !
751     OPTION BASE 1
752     COM /Char_buf/ Triangle(6,3),Tri_2(5,3),Sqr_rt(5,3)
753     COM /Char_buf/ Rgt_arrow(6,3),Hp_log(20,3),Micro(8,3)
754     INTEGER Op_code
755     DIM A$[1]
756 Decode_instr:   !
757     Op_code=(BINAND(Coordt,24576))                  ! Mask out opcode
758     SELECT Op_code
759     CASE =0
760       GOSUB Plot                                    ! Plot vector
761     CASE =8192
762       GOSUB Graph                                   ! Graph vector
763     CASE =16384
764       GOSUB Text                                    ! Write text
765     CASE =24576
766       GOSUB Set_cond                                ! Set condition
767     END SELECT
768     SUBEXIT
769     !
770 Set_cond:       ! Sets Linetype
771     IF BIT(Coordt,8)=1 THEN LINE TYPE 4
772     IF BIT(Coordt,8)=0 THEN LINE TYPE 1
773     RETURN
774     !
775 Plot:           ! Plots/move pen
776     IF BIT(Coordt,12)=0 THEN
777       Xcoord=BINAND(Coordt,2047)
778     ELSE
779       Ycoord=BINAND(Coordt,2047)
780       IF BIT(Coordt,11)=0 THEN
781         Pflag=-2
782       ELSE
783         Pflag=-1
784       END IF
785       PLOT Xcoord,Ycoord,Pflag
786     END IF
787     RETURN
788     !
789 Graph:          ! Graph data
790     IF BIT(Coordt,12)=0 THEN
791       Delta_x=BINAND(Coordt,2047)
792     ELSE
793       Ycoord=BINAND(Coordt,2047)
794       Xcoord=Xcoord+Delta_x
795       IF BIT(Coordt,11)=0 THEN
796         Pflag=-2
797       ELSE
798         Pflag=-1
799       END IF
```

```
800       PLOT Xcoord,Ycoord,Pflag
801     END IF
802     RETURN
803     !
804 Text:          ! Text control and output
805     LINE TYPE 1
806     A$=CHR$(BINAND(Coordt,255))
807     IF BIT(Coordt,8)0 THEN
808       IF BIT(Coordt,10)=0 AND BIT(Coordt,9)=0 THEN LDIR 0
809       IF BIT(Coordt,10)=0 AND BIT(Coordt,9)=1 THEN LDIR 90
810       IF BIT(Coordt,10)=1 AND BIT(Coordt,9)=0 THEN LDIR 180
811       IF BIT(Coordt,10)=1 AND BIT(Coordt,9)=1 THEN LDIR 270
812       IF BIT(Coordt,12)=0 AND BIT(Coordt,11)=0 THEN CSIZE 3.4,.67
813       IF BIT(Coordt,12)=0 AND BIT(Coordt,11)=1 THEN CSIZE 5.1,.67
814       IF BIT(Coordt,12)=1 AND BIT(Coordt,11)=0 THEN CSIZE 6.8,.67
815       IF BIT(Coordt,12)=1 AND BIT(Coordt,11)=1 THEN CSIZE 8.5,.67
816     END IF
817 Type:          ! Special non-standard characters.
818     Char_no=NUM(A$)
819     SELECT Char_no
820     CASE =1
821       SYMBOL Hp_log(*)
822     CASE =9
823       IMOVE 0,-12
824     CASE =12
825       IMOVE 0,18
826     CASE =17
827       IMOVE -13,-28
828       LABEL USING "#,A";CHR$(111)
829     CASE =21
830       SYMBOL Rgt_arrow(*)
831       LABEL USING "#,A";CHR$(32)
832     CASE =22
833       SYMBOL Sqr_rt(*)
834       LABEL USING "#,A";CHR$(32)
835     CASE =24
836       SYMBOL Tri_2(*)
837       LABEL USING "#,A";CHR$(32)
838     CASE =25
839       SYMBOL Micro(*)
840       LABEL USING "#,A";CHR$(32)
841     CASE =95
842       LABEL USING "#,A";CHR$(8)
843       IMOVE 0,-20
844       LABEL USING "#,A";A$
845       IMOVE 0,20
846     CASE =127
847       SYMBOL Triangle(*)
848       LABEL USING "#,A";CHR$(32)
849     CASE ELSE
850       IMOVE 0,-20
851       LABEL USING "#,A";A$
852       IMOVE 0,20
853     END SELECT
```

```
854     RETURN
855   SUBEND
856   !//////////////////////////////////////////////////////////////////////////
857   SUB Read_binary(@Io,Com$,INTEGER Block_no,Data_len,INTEGER Buf(*))
858   !
859   !  This routine performs a "generic" read binary from the HP 3563A.
860   !
861     OPTION BASE 1
862     ASSIGN @Io;FORMAT ON                        ! Turn ascii formatter on.
863     OUTPUT @Io;"VBLK ";Block_no                 ! Output requested buffer.
864     OUTPUT @Io;Com$                             ! Output command.
865     ENTER @Io USING "%,2A,W";A$,Data_len        ! Enter format specifier.
866     IF A$<>"#A" THEN                            ! Check for correct response.
867       DISP "NOT CORRECT BLOCK MODE"
868       CLEAR @Io
869     ELSE
870       IF Data_len=0 THEN SUBEXIT
871       REDIM Buf(Data_len DIV 2)                 ! Set buffer to proper length.
872       ASSIGN @Io;FORMAT OFF                     ! Turn ascii formatter off.
873       ENTER @Io;Buf(*)                          ! Read data into buffer.
874       ASSIGN @Io;FORMAT ON                      ! Turn ascii formatter on.
875       DISP CHR$(129)&"Read Binary Data-block ";Block_no;" finished,
          length     = ";Data_len;CHR$(128)
876     END IF
877     LOCAL @Io                                   ! Put HP 3563A in local mode.
878   SUBEND
```

APPENDIX B - EXAMPLE PROGRAM 6

```
879   !//////////////////////////////////////////////////////////////////////////
880   !
881   !//////////////////////////////////////////////////////////////////////////
882   !
883   !              COMPOSITE DEMO PROGRAM
884   !
885   !      (c)  COPYRIGHT 1985, Hewlett-Packard Co.
886   !                last update 8-14-89
887   !                    BASIC 5.0
888   !
889   !  PURPOSE:
890   !
891   !  This program demostrates many of the features of the HP 3563A.  It
892   !  is a softkey driven program that allows the user to 1) send hpib
893   !  commands to the instrument, 2) perform all types of data transfers
894   !  to and from the instrument in any of the three formats discussed
895   !  in the programming manual, and 3) view several different short
896   !  demonstration routines that use the HP 3563A's display screen.  The
897   !  program contains many generic subprograms that perform specific
898   !  functions that could be used in many other programs with little
899   !  or no modification.
900   !
901   !  DATA DICTIONARY:
902   !
903   !    Buf(*)              Real buffer used for ascii and ansi data transfers
904   !
905   !    Ibuf(*)             Integer buffer used for binary data transfers
906   !
907   !    Data_len            Active data buffer length (bytes)
908   !
909   !    Hp3563a             Contains address of the HP 3563A
910   !
911   !    Disk_drive          Contains address of a disk drive on the bus
912   !
913   !    Plottr              Contains address of a plotter on the bus
914   !
915   !    Demo_flg            Flag denotes demo in progress when true
916   !
917   !    Err_flag            Flag denotes HP 3563A has an error when true
918   !
919   !    Ignore_intr_flg     Flag disables HP-IB interrupts when true
920   !
921   !    Take_ctrl_flag      Flag enables HP 3563A to become active controller
922   !                        when true
923   !
924   !    Disk_flg            Flag denotes end-of-disk-action when true
925   !
926   !    End_plot            Flag denotes end-of-plot-action when true
927   !
928   !    Entry_changed       Flag denotes entry knob value changed when true
```

```
929   !
930   !    Rep_flg            Flag denotes disk report enabled when true
931   !
932   !//////////////////////////////////////////////////////////////////////
933     OPTION BASE 1
934     COM /Addresses/ Hp3563a,Disk_drive,Plottr
935     COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
936     COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
937     COM /Data/ Buf(2200),INTEGER Ibuf(4400),Data_len
938     INTEGER True,False,End_flg
939     CLEAR SCREEN
940     OUTPUT KBD;"*PSCRATCH C";"*E";"RUN Start";"*E";
941 Start:   !
942   !
943   ! Initialize variables
944   !
945     True=1
946     False=0
947   !
948     Disk_drive=1
949     Plottr=5
950     Hp3563a=20
951   !
952     Rep_flg=True
953     End_flg=False
954     Ignore_intr_flg=False
955     Take_ctrl_flag=False
956   !
957   ! Set up main softkeys.
958   !
959     ON KEY 0 LABEL "",1 CALL Do_nothing
960     ON KEY 1 LABEL "Command",1 CALL Send_command
961     ON KEY 2 LABEL " Address",1 CALL Change_address
962     ON KEY 3 LABEL " Serial  Poll",1 CALL Serial_poll
963     ON KEY 4 LABEL "Transfer",1 CALL Xfer_mainsub
964     ON KEY 5 LABEL " Clear  3563A",1 CALL Device_clear
965     ON KEY 6 LABEL " Demos",1 CALL Demo_mainsub
966     ON KEY 7 LABEL "",1 CALL Do_nothing
967     ON KEY 8 LABEL "  Exit",1 GOSUB Set_exit_flg
968     ON KEY 9 LABEL "",1 CALL Do_nothing
969     USER 1 KEYS                              ! User level one softkeys.
970     KEY LABELS ON                            ! Turn on softkeys.
971     REMOTE 7                                 ! Address bus to listen.
972     ON INTR 7,15 CALL Hpib_interrupt         ! Set up interrupt branch.
973     ENABLE INTR 7;2                          ! Enable interrupts.
974   !
975   ! Wait for a softkey or interrupt.
976   !
977     REPEAT
978     UNTIL End_flg
979   !
980     WAIT .5                                  ! Allow bus to settle.
981     LOCAL 7                                  ! Return local control.
982     CLEAR SCREEN
```

```
983     DISP " Program finished."
984     KEY LABELS OFF
985     STOP
986   !
987 Set_exit_flg:  !
988     End_flg=True
989     RETURN
990     END
991   !//////////////////////////////////////////////////////////////////////////
992 Read_ascii:SUB Read_ascii(Addr,Com$,INTEGER Data_len,REAL Buf(*))
993   !
994   ! Reads data from the HP 3563A in ASCII format.
995   !
996     OPTION BASE 1
997     DIM A$[2]
998     REAL Float_len
999     INTEGER I
1000    ASSIGN @Io TO Addr
1001    ASSIGN @Io;FORMAT ON
1002    DISP "Start Read Ascii"
1003    OUTPUT @Io;Com$
1004    ENTER @Io USING "2A,K";A$,Float_len
1005    Data_len=INT(Float_len+.5)
1006    REDIM Buf(Data_len)
1007    IF A$<>"#I" THEN
1008      DISP "NOT CORRECT BLOCK MODE"
1009      CLEAR @Io
1010    ELSE
1011      FOR I=1 TO Data_len
1012        ENTER @Io;Buf(I)
1013      NEXT I
1014      DISP "End Read, Data_length =  ";Data_len
1015    END IF
1016    LOCAL @Io
1017  SUBEND
1018  !//////////////////////////////////////////////////////////////////////////
1019 Write_ascii:SUB Write_ascii(Addr,Com$,INTEGER Data_len,REAL Buf(*))
1020  !
1021  ! Writes data to the HP 3563A in ASCII format.
1022  !
1023    OPTION BASE 1
1024    INTEGER I
1025    ASSIGN @Io TO Addr
1026    ASSIGN @Io;FORMAT ON
1027    DISP "Start Write Ascii, Data_len = ";Data_len
1028    OUTPUT @Io;Com$
1029    OUTPUT @Io;"#I";Data_len
1030    FOR I=1 TO Data_len-1
1031      OUTPUT @Io;Buf(I)
1032    NEXT I
1033    OUTPUT @Io;Buf(I),END
1034    DISP "End Write"
1035    LOCAL @Io
1036  SUBEND
```

```
1037  !/////////////////////////////////////////////////////////////////////////
1038 Read_binary:SUB Read_binary(Addr,Com$,INTEGER Data_len,INTEGER Buf(*))
1039  !
1040  ! Reads data from the HP 3563A in internal binary format.
1041  !
1042    OPTION BASE 1
1043    DIM A$[2]
1044    ASSIGN @Io TO Addr
1045    ASSIGN @Io;FORMAT ON
1046    DISP "Read Binary"
1047    OUTPUT @Io;Com$
1048    ENTER @Io USING "%,2A,W";A$,Data_len
1049    IF A$<>"#A" THEN
1050      DISP "NOT CORRECT BLOCK MODE"
1051      CLEAR @Io
1052    ELSE
1053      REDIM Buf(Data_len DIV 2)
1054      ASSIGN @Io;FORMAT OFF
1055      ENTER @Io;Buf(*)
1056      ASSIGN @Io;FORMAT ON
1057      DISP "End read, DATA_LEN =  ";Data_len
1058    END IF
1059    LOCAL @Io
1060  SUBEND
1061  !/////////////////////////////////////////////////////////////////////////
1062 Write_binary:SUB Write_binary(Addr,Com$,INTEGER Data_len,INTEGER Buf(*))
1063  !
1064  ! Writes data to the HP 3563A in internal binary format.
1065  !
1066    OPTION BASE 1
1067    ASSIGN @Io TO Addr
1068    ASSIGN @Io;FORMAT ON
1069    DISP "Start Write Binary, DATA_LEN = ";Data_len
1070    OUTPUT @Io;Com$
1071    REDIM Buf(Data_len DIV 2)
1072    OUTPUT @Io USING "#,2A,W";"#A";Data_len
1073    ASSIGN @Io;FORMAT OFF
1074    OUTPUT @Io;Buf(*)
1075    ASSIGN @Io;FORMAT ON
1076    DISP "End Write"
1077    LOCAL @Io
1078  SUBEND
1079  !/////////////////////////////////////////////////////////////////////////
1080 Read_ansi:SUB Read_ansi(Addr,Com$,INTEGER Data_len,REAL Buf(*))
1081  !
1082  ! Reads data from the HP 3563A in ANSI binary format.
1083  !
1084    OPTION BASE 1
1085    DIM A$[2]
1086    ASSIGN @Io TO Addr
1087    ASSIGN @Io;FORMAT ON
1088    DISP "Start Read float"
1089    OUTPUT @Io;Com$
1090    ENTER @Io USING "%,2A,W";A$,Data_len
```

```
1091     IF (A$<>"#A") OR (Data_len MOD 8<>0) THEN
1092       DISP "NOT CORRECT BLOCK MODE"
1093       CLEAR @Io
1094     ELSE
1095       ASSIGN @Io;FORMAT OFF
1096       REDIM Buf(Data_len DIV 8)
1097       ENTER @Io;Buf(*)
1098       DISP "End Read, DATA_LEN = ";Data_len
1099       ASSIGN @Io;FORMAT ON
1100     END IF
1101     LOCAL @Io
1102   SUBEND
1103   !//////////////////////////////////////////////////////////////////////////
1104 Write_ansi:SUB Write_ansi(Addr,Com$,INTEGER Data_len,REAL Buf(*))
1105   !
1106   ! Writes data to the HP 3563A in ANSI binary format.
1107   !
1108     OPTION BASE 1
1109     ASSIGN @Io TO Addr
1110     ASSIGN @Io;FORMAT ON
1111     DISP "Start Write Float, DATA_LEN = ",Data_len
1112     OUTPUT @Io;Com$
1113     OUTPUT @Io USING "#,2A,W";"#A",Data_len
1114     ASSIGN @Io;FORMAT OFF
1115     REDIM Buf(Data_len DIV 8)
1116     OUTPUT @Io;Buf(*)
1117     ASSIGN @Io;FORMAT ON
1118     DISP "End Write."
1119     LOCAL @Io
1120   SUBEND
1121   !//////////////////////////////////////////////////////////////////////////
1122 Do_nothing:SUB Do_nothing
1123   !
1124   ! Subprogram to trap unwanted keystrokes.
1125   !
1126     ALLOCATE Key$[256]
1127     Key$=KBD$
1128   SUBEND
1129   !//////////////////////////////////////////////////////////////////////////
1130 Send_command:SUB Send_command
1131   !
1132   ! Subprogram to output HP-IB command mnemonics to the HP 3563A.
1133   !
1134     COM /Addresses/ Hp3563a,Disk_drive,Plottr
1135     DIM Command$[80]
1136     LINPUT " Enter command mnemonic to send to the HP3563A.",Command$
1137     IF Command$"" THEN
1138       OUTPUT 700+Hp3563a;Command$
1139       LOCAL 700+Hp3563a
1140     END IF
1141   SUBEND
1142   !//////////////////////////////////////////////////////////////////////////
1143 Change_address:SUB Change_address
1144   !
```

```
1145  ! Subprogram to alter the HP-IB address of devices in the system or toggle
1146  !   the flag that allows the program to pass control to the HP 3563A.
1147  !
1148    COM /Addresses/ Hp3563a,Disk_drive,Plottr
1149    COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
1150    COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
1151    INTEGER Addr,Priority,Key_pressed,I,Exit_flag
1152    Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1153    FOR I=0 TO 9
1154      ON KEY I LABEL "",Priority CALL Do_nothing
1155    NEXT I
1156    Exit_flag=0
1157    REPEAT
1158      ON KEY 1 LABEL "HP 3563A address",Priority GOSUB Hpib_addr
1159      ON KEY 2 LABEL "Disk drv address",Priority GOSUB Disk_addr
1160      ON KEY 3 LABEL "TakeCtrlflag",Priority GOSUB Set_tct
1161      ON KEY 8 LABEL "  Main    Menu",Priority GOSUB Set_exit_flag
1162      Key_pressed=0
1163      REPEAT
1164      UNTIL Key_pressed
1165    UNTIL Exit_flag
1166    CLEAR SCREEN
1167    SUBEXIT
1168    !
1169 Hpib_addr:   !
1170    Key_pressed=1
1171    REPEAT
1172      OUTPUT KBD;Hp3563a;
1173      INPUT " Enter HP-IB address of the HP 3563A.",Addr
1174    UNTIL Addr>=0 AND Addr<=31
1175    Hp3563a=Addr
1176    RETURN
1177    !
1178 Disk_addr:  !
1179    Key_pressed=1
1180    REPEAT
1181      OUTPUT KBD;Disk_drive;
1182      INPUT " Enter HP-IB address of the disk drive.",Addr
1183    UNTIL Addr>=0 AND Addr<=31
1184    Disk_drive=Addr
1185    RETURN
1186    !
1187 Set_tct:  !
1188    Take_ctrl_flag=NOT Take_ctrl_flag
1189    IF Take_ctrl_flag THEN
1190      DISP "Take Control flag now enabled."
1191    ELS"
1192      DISP "Take Control flag now disabled."
1193    END IF
1194    RETURN
1195    !
1196 Set_exit_flag:     !
1197    Key_pressed=1
1198    Exit_flag=1
```

```
1199    RETURN
1200    !
1201  SUBEND
1202  !///////////////////////////////////////////////////////////////////////
1203 Serial_poll:SUB Serial_poll
1204  !
1205  !  Subprogram to do a serial poll of the HP 3563A's status byte.
1206  !
1207    COM /Addresses/ Hp3563a,Disk_drive,Plottr
1208    INTEGER Poll_byte
1209    Poll_byte=SPOLL(700+Hp3563a)
1210    DISP "Serial Poll = ";Poll_byte
1211  SUBEND
1212  !///////////////////////////////////////////////////////////////////////
1213 Device_clear:SUB Device_clear
1214  !
1215  !  Sends the HP-IB device clear command to the HP 3563A.
1216  !
1217    COM /Addresses/ Hp3563a,Disk_drive,Plottr
1218    CLEAR 700+Hp3563a
1219    DISP " Device clear sent to the HP 3563A."
1220    ENABLE INTR 7
1221  SUBEND
1222  !///////////////////////////////////////////////////////////////////////
1223 Demo_mainsub:SUB Demo_mainsub
1224  !
1225  ! Main control subprogram for accessing demo routines.
1226  !
1227    COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
1228    COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
1229    INTEGER Priority,Exit_flag
1230    Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1231    FOR I=0 TO 9
1232      ON KEY I LABEL "",Priority CALL Do_nothing
1233    NEXT I
1234    Exit_flag=0
1235    ON KEY 1 LABEL "Demo 1",Priority CALL Demo_1
1236    ON KEY 2 LABEL "Demo 2",Priority CALL Demo_2
1237    ON KEY 3 LABEL "Rpg Demo",Priority CALL Rpg_demo
1238    ON KEY 4 LABEL "Control via hpib",Priority CALL Canned_demo
1239    ON KEY 5 LABEL "Intrupt flag",Priority CALL Toggle_intr_flg
1240    ON KEY 8 LABEL "  Main    Menu",Priority GOSUB Set_exit_flag
1241    REPEAT
1242    UNTIL Exit_flag
1243    SUBEXIT
1244 Set_exit_flag:    !
1245    Exit_flag=1
1246    RETURN
1247  SUBEND
1248  !///////////////////////////////////////////////////////////////////////
1249 Toggle_intr_flg:SUB Toggle_intr_flg
1250  !
1251  ! Subprogram to toggle the interrupt enable flag.
1252  !
```

```
1253     COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
1254     COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
1255     Ignore_intr_flg=NOT Ignore_intr_flg
1256     IF Ignore_intr_flg THEN
1257       DISP " Interrupts disabled."
1258     ELSE
1259       DISP " Interrupts enabled."
1260     END IF
1261     ENABLE INTR 7
1262   SUBEND
1263   !//////////////////////////////////////////////////////////////////////
1264 Rpg_demo:SUB Rpg_demo
1265   !
1266   ! Subprogram to run a demo of the RPG entry knob functionality.
1267   !
1268     COM /Addresses/ Hp3563a,Disk_drive,Plottr
1269     COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
1270     COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
1271     INTEGER I,Priority,Entry_value
1272     Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1273     FOR I=0 TO 9
1274       ON KEY I LABEL "",Priority CALL Do_nothing
1275     NEXT I
1276     ON KEY 8 LABEL "End demo",Priority CALL Reset_demo_flag
1277     IF Ignore_intr_flg THEN
1278       DISP " Interrupts must be enabled to run RPG demo."
1279       WAIT 3
1280     ELSE
1281       OUTPUT 700+Hp3563a;"COMD;ISM 4096;RENE"
1282       Demo_flg=1
1283       Entry_changed=1
1284       WHILE Demo_flg
1285         WHILE Entry_changed
1286           Entry_changed=0
1287           OUTPUT 700+Hp3563a;"RENV?"
1288           ENTER 700+Hp3563a;Entry_value
1289           DISP "Rotate RPG knob to generate SRQ; Entry = ";Entry_value
1290         END WHILE
1291       END WHILE
1292       OUTPUT 700+Hp3563a;"REND;ISM 0;COME"
1293     END IF
1294     CLEAR SCREEN
1295   SUBEND
1296   !//////////////////////////////////////////////////////////////////////
1297 Reset_demo_flag:SUB Reset_demo_flag
1298   !
1299   ! This subprogram resets the demo flag to false.
1300   !
1301     COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
1302     COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
1303     Demo_flg=0
1304   SUBEND
1305   !//////////////////////////////////////////////////////////////////////
1306 Demo_1:SUB Demo_1
```

```
1307   !
1308   !  Runs the first program demostration routine.
1309   !
1310     COM /Addresses/ Hp3563a,Disk_drive,Plottr
1311     COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
1312     COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
1313     REAL P1x,P1y,P1dir,P2x,P2y,P2dir
1314     INTEGER I,Priority
1315     Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1316     FOR I=0 TO 9
1317       ON KEY I LABEL "",Priority CALL Do_nothing
1318     NEXT I
1319     ON KEY 8 LABEL "End demo",Priority CALL Reset_demo_flag
1320     I=0
1321     Demo_flg=1
1322     P1x=0
1323     P1y=250
1324     P1dir=1
1325     P2x=1750
1326     P2y=2000
1327     P2dir=3
1328     OUTPUT 700+Hp3563a;"PAUS"
1329     WAIT 1
1331     OUTPUT 700+Hp3563a;"COMD;DBSZ 200,0,2"
1332     WHILE Demo_flg
1333       OUTPUT 700+Hp3563a;"DBAC ";I
1334       OUTPUT 700+Hp3563a;"PU;PA ";P1x,P1y
1335       OUTPUT 700+Hp3563a;"PD;PA ";P2x,P2y
1336       OUTPUT 700+Hp3563a;"LT 0"
1337       OUTPUT 700+Hp3563a;"DBSW ";I,1-I
1338       I=1-I
1339       CALL Nextpoint(P1x,P1y,P1dir)
1340       CALL Nextpoint(P2x,P2y,P2dir)
1341     END WHILE
1342   SUBEND
1343   !//////////////////////////////////////////////////////////////////////////
1344 Demo_2:SUB Demo_2
1345   !
1346   !  Runs the second program demostration routine.
1347   !
1348     COM /Addresses/ Hp3563a,Disk_drive,Plottr
1349     COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
1350     COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
1351     REAL P1x,P1y,P2x,P2y,Incr
1352     INTEGER I,Priority
1353     Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1354     FOR I=0 TO 9
1355       ON KEY I LABEL "",Priority CALL Do_nothing
1356     NEXT I
1357     ON KEY 8 LABEL "End demo",Priority CALL Reset_demo_flag
1358     Demo_flg=1
1359     WHILE Demo_flg
1360       P1x=0
1361       P1y=250
```

```
1362      P2x=1750
1363      P2y=2000
1364      Incr=50
1365      OUTPUT 700+Hp3563a;"PAUS"
1366      WAIT 1
1368      OUTPUT 700+Hp3563a;"COMD;DBSZ 300,0;DBAC 0;PU"
1369      OUTPUT 700+Hp3563a;"PA ";P1x,P1y
1370      OUTPUT 700+Hp3563a;"PD;DBUP 0"
1371      LOOP
1372      EXIT IF NOT Demo_flg
1373        OUTPUT 700+Hp3563a;"DBAA 0;PA ";P2x,P1y
1374        OUTPUT 700+Hp3563a;"DBUP 0"
1375        P2y=P2y-Incr
1376      EXIT IF P2yy
1377        OUTPUT 700+Hp3563a;"DBAA 0;PA ";P2x,P2y
1378        OUTPUT 700+Hp3563a;"DBUP 0"
1379        P1x=P1x+Incr
1380      EXIT IF P1xP2x
1381        OUTPUT 700+Hp3563a;"DBAA 0;PA ";P1x,P2y
1382        OUTPUT 700+Hp3563a;"DBUP 0"
1383        P1y=P1y+Incr
1384      EXIT IF P1yP2y
1385        OUTPUT 700+Hp3563a;"DBAA 0;PA ";P1x,P1y
1386        OUTPUT 700+Hp3563a;"DBUP 0"
1387        P2x=P2x-Incr
1388      EXIT IF P2x
1389      END LOOP
1390    END WHILE
1391  SUBEND
1392  !///////////////////////////////////////////////////////////////////////
1393 Canned_demo:SUB Canned_demo
1394  !
1395  ! This subprogram runs a demo program using the HP 3563A's display screen
1396  !   to display the text of each step of the demo.
1397  !
1398    COM /Addresses/ Hp3563a,Disk_drive,Plottr
1399    DIM Message$[80]
1400    CLEAR SCREEN
1401    INTEGER I,Priority
1402    Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1403    FOR I=0 TO 9
1404      ON KEY I LABEL "",Priority CALL Do_nothing
1405    NEXT I
1406    ON KEY 8 LABEL "End demo",Priority RECOVER Abort_demo
1407    DISP " Watch 3563A display screen..."
1408    Message$="HP-IB control program"
1409    GOSUB Out_hpib
1410    Message$="Change Setup State"
1411    GOSUB Out_hpib
1412    OUTPUT 700+Hp3563a;"SRQE;STAT"
1413    WAIT 2
1414    OUTPUT 700+Hp3563a;"SMES;TRGD;FREQ"
1415    OUTPUT 700+Hp3563a;"RNG"
1416    WAIT 2
```

```
1417     Message$="Synthesize Data"
1418     GOSUB Out_hpib
1419     OUTPUT 700+Hp3563a;"CTRC"
1420     WAIT 5
1421     Message$="Display LOG MAG data"
1422     GOSUB Out_hpib
1423     OUTPUT 700+Hp3563a;"CORD;MGLG"
1424     WAIT 5
1425     Message$="Display REAL data"
1426     GOSUB Out_hpib
1427     OUTPUT 700+Hp3563a;"CORD;REAL"
1428     WAIT 5
1429     Message$="Display IMAGINARY data"
1430     GOSUB Out_hpib
1431     OUTPUT 700+Hp3563a;"CORD;IMAG"
1432     WAIT 5
1433     Message$="End of HP-IB test"
1434     GOSUB Out_hpib
1435     OUTPUT 700+Hp3563a;"STAT;COME"
1436     WAIT 1
1437 Abort_demo:    !
1438     LOCAL 700+Hp3563a
1439     DISP
1440     SUBEXIT
1441 Out_hpib:OUTPUT 700+Hp3563a;"COMD;DBSZ 100,0"
1442     OUTPUT 700+Hp3563a;"DBAC 0;PU;PA 20,1000"
1443     OUTPUT 700+Hp3563a;"CHSZ 2"
1444     OUTPUT 700+Hp3563a;"WRIT '";Message$;"'"
1445     OUTPUT 700+Hp3563a;"CHSZ 0;DBUP 0"
1446     WAIT 3
1447     RETURN
1448  SUBEND
1449  !/////////////////////////////////////////////////////////////////////////
1450 Hpib_interrupt:SUB Hpib_interrupt
1451  !
1452  !  Hpib interrupt service routine.
1453  !
1454     COM /Addresses/ Hp3563a,Disk_drive,Plottr
1455     COM /Flags/ Demo_flg,Err_flag,Ignore_intr_flg,Take_ctrl_flag
1456     COM /Flags/ Disk_flg,End_plot,Entry_changed,Rep_flg
1457     INTEGER A_commnd,A_data,A_dsj,A_format,A_read,A_write,As_word,Commnd
1458     INTEGER Idisk,Ilisten,Iplot,Isdc,Iskey,Italk,Keycode,Poll_byte
1459     INTEGER Ppoll_byte,Report,Stat_word,Xecute
1460     DIM Keystr$[18]
1461     !
1462     ! Declare constants
1463     !
1464     Report=16
1465     Commnd=5
1466     Xecute=14
1467     A_commnd=8
1468     A_data=0
1469     A_dsj=16
1470     A_format=12
```

```
1471    A_read=10
1472    A_write=9
1473    Italk=128
1474    Ilisten=136
1475    Isdc=14
1476    Iskey=13
1477    Iplot=0
1478    Idisk=1
1479    !
1480    Poll_byte=SPOLL(700+Hp3563a)
1481    PRINT " Status register =",Poll_byte
1482    IF BIT(Poll_byte,5) THEN Err_flag=1
1483    Poll_byte=BINAND(Poll_byte,143)          ! Mask out bits 4,5,6.
1484    IF Ignore_intr_flg THEN End_intr
1485    SELECT Poll_byte
1486    CASE 14 TO 143
1487      IF Take_ctrl_flag THEN
1488        DISP "PASS CONTROL"                  ! Pass control to 3563A.
1489        PASS CONTROL Hp3563a
1490      END IF
1491    CASE =9
1492      Disk_flg=1                             ! End of disk action.
1493    CASE =10
1494      End_plot=1                             ! End of plot action.
1495    CASE =Italk+Iplot                        ! Talk plotter.
1496      SEND 7;UNL UNT LISTEN Hp3563a CMD 1 TALK Plottr DATA
1497    CASE =Ilisten+Iplot                      ! Listen plotter.
1498      SEND 7;UNL UNT LISTEN Plottr TALK Hp3563a DATA
1499    CASE =Ilisten+Idisk                      ! Listen disk command.
1500      SEND 7;UNL UNT LISTEN Disk_drive SEC Commnd TALK Hp3563a DATA
1501      Rep_flg=0
1502    CASE =Ilisten+Idisk+1                    ! Listen disk execution.
1503      GOSUB Parallel_poll
1504      SEND 7;UNL UNT LISTEN Disk_drive SEC Xecute TALK Hp3563a DATA
1505      Rep_flg=0
1506    CASE =Ilisten+Idisk+2                    ! Listen Amigo command.
1507      SEND 7;UNT UNL TALK Hp3563a LISTEN Disk_drive
1508      WAIT .001
1509      SEND 7;SEC A_commnd
1510      WAIT .001
1511      SEND 7;DATA
1512    CASE =Ilisten+Idisk+3                    ! Listen Amigo data.
1513      SEND 7;UNT UNL TALK Hp3563a LISTEN Disk_drive
1514      WAIT .001
1515      SEND 7;SEC A_data
1516      WAIT .001
1517      SEND 7;DATA
1518    CASE =Ilisten+Idisk+4                    ! Listen Amigo read command.
1519      SEND 7;UNT UNL TALK Hp3563a LISTEN Disk_drive
1520      WAIT .001
1521      SEND 7;SEC A_read
1522      WAIT .001
1523      SEND 7;DATA
1524    CASE =Ilisten+Idisk+5                    ! Listen Amigo write command.
```

```
1525      SEND 7;UNT UNL TALK Hp3563a LISTEN Disk_drive
1526      WAIT .001
1527      SEND 7;SEC A_write
1528      WAIT .001
1529      SEND 7;DATA
1530    CASE =Ilisten+Idisk+6                    ! Listen Amigo format.
1531      SEND 7;UNT UNL TALK Hp3563a LISTEN Disk_drive
1532      WAIT .001
1533      SEND 7;SEC A_format
1534      WAIT .001
1535      SEND 7;DATA
1536    CASE =Italk+Idisk                        ! Talk disk execution.
1537      GOSUB Parallel_poll
1538      SEND 7;UNL UNT LISTEN Hp3563a CMD 1 TALK Disk_drive SEC Xecute
1539      Rep_flg=0
1540    CASE =Italk+Idisk+1                      ! Talk disk report.
1541      IF NOT Rep_flg THEN GOSUB Parallel_poll
1542      SEND 7;UNL UNT LISTEN Hp3563a CMD 1 TALK Disk_drive
1543      WAIT .001
1544      SEND 7;SEC Report DATA
1545      Rep_flg=1
1546    CASE =Italk+Idisk+2                      ! Talk Amigo Status / Command.
1547      SEND 7;UNT UNL LISTEN Hp3563a CMD 1 TALK Disk_drive
1548      WAIT .001
1549      SEND 7;SEC A_commnd DATA
1550    CASE =Italk+Idisk+3                      ! Talk Amigo data.
1551      SEND 7;UNT UNL LISTEN Hp3563a CMD 1 TALK Disk_drive
1552      WAIT .001
1553      SEND 7;SEC A_data DATA
1554    CASE =Italk+Idisk+4                      ! Talk Amigo DSJ.
1555      SEND 7;UNT UNL LISTEN Hp3563a CMD 1 TALK Disk_drive
1556      WAIT .001
1557      SEND 7;SEC A_dsj DATA
1558    CASE =Italk+Idisk+5                      ! Talk Disk Ident.
1559      SEND 7;UNT UNL LISTEN Hp3563a CMD 1 UNT
1560      WAIT .001
1561      SEND 7;SEC Disk_drive DATA
1562    CASE =Italk+Idisk+6                      ! Talk Amigo Parallel Poll.
1563      SEND 7;UNT UNL DATA
1564      GOSUB Parallel_poll
1565    CASE =Isdc                               ! Clear plotter.
1566      CLEAR 700+Plottr
1567      SEND 7;UNT UNL LISTEN Hp3563a CMD 1 DATA
1568    CASE =Isdc+1                             ! Unaddress bus.
1569      SEND 7;UNT UNL LISTEN Hp3563a CMD 1 UNL DATA
1570    CASE =Iskey                              ! Redirected key hit.
1571      OUTPUT 700+Hp3563a;"COM?"
1572      ENTER 700+Hp3563a;Keycode,Keystr$
1573      DISP "KEY CODE = ",Keycode,"STR-";Keystr$;""
1574      IF Keycode>0 THEN OUTPUT 700+Hp3563a;"KEY ";Keycode
1575    CASE =11                                 ! Instrument Status change.
1576      OUTPUT 700+Hp3563a;"IS?"
1577      ENTER 700+Hp3563a;Stat_word
1578      PRINT " Instrument Status register = ",Stat_word
```

```
1579      IF BINAND(Stat_word,4096) THEN Entry_changed=1
1580      IF BINAND(Stat_word,8192) THEN
1581        OUTPUT 700+Hp3563a;"AS?"
1582        ENTER 700+Hp3563a;As_word
1583        PRINT " Activity Status register = ",As_word
1584      END IF
1585      LOCAL 700+Hp3563a
1586    END SELECT
1587 End_intr:ENABLE INTR 7
1588    SUBEXIT
1589  !
1590 Parallel_poll:Ppoll_byte=PPOLL(7)
1591    IF BIT(Ppoll_byte,7-Disk_drive)=0 THEN Parallel_poll
1592    RETURN
1593  SUBEND
1594  !//////////////////////////////////////////////////////////////////////
1595 Xfer_mainsub:SUB Xfer_mainsub
1596  !
1597  ! Main control subprogram for all data transfer demonstrations.
1598  !
1599    CLEAR SCREEN
1600    INTEGER Exit_flag,I,Priority
1601    Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1602    FOR I=0 TO 9
1603      ON KEY I LABEL "",Priority CALL Do_nothing
1604    NEXT I
1605    Exit_flag=0
1606    ON KEY 1 LABEL " Print   Buffer",Priority CALL Print_buffer
1607    ON KEY 2 LABEL "Transfer Synth",Priority CALL Xfer_synth_tbl
1608    ON KEY 3 LABEL "Transfer State",Priority CALL Xfer_state
1609    ON KEY 4 LABEL "Transfer Trace",Priority CALL Xfer_data_trace
1610    ON KEY 5 LABEL "Transfer Block",Priority CALL Xfer_xfrm_block
1611    ON KEY 6 LABEL "Transfer Display",Priority CALL Xfer_disp_block
1612    ON KEY 8 LABEL "  Main    Menu",Priority GOSUB Set_exit_flag
1613    REPEAT
1614    UNTIL Exit_flag
1615    SUBEXIT
1616    !
1617 Set_exit_flag:     !
1618    Exit_flag=1
1619    RETURN
1620  SUBEND
1621  !//////////////////////////////////////////////////////////////////////
1622 Print_buffer:SUB Print_buffer
1623  !
1624  ! Prints all or part of the most recently loaded data buffer.
1625  !
1626    COM /Data/ Buf(*),INTEGER Ibuf(*),Data_len
1627    INTEGER Exit_flag,I,Priority,Divisor,Icount,Istart
1628    CLEAR SCREEN
1629    Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1630    FOR I=0 TO 9
1631      ON KEY I LABEL "",Priority CALL Do_nothing
1632    NEXT I
```

```
1633     Exit_flag=0
1634     REPEAT
1635       ON KEY 5 LABEL "Prt buff Ascii",Priority GOSUB Disp_asc
1636       ON KEY 6 LABEL "Prt buff Binary",Priority GOSUB Disp_bin
1637       ON KEY 7 LABEL "Prt buff Ansi",Priority GOSUB Disp_ans
1638       ON KEY 8 LABEL " Return ",Priority GOSUB Set_exit_flag
1639     UNTIL Exit_flag
1640     SUBEXIT
1641 Disp_asc:      !
1642     Divisor=1
1643     GOSUB Print_it
1644     RETURN
1645 Disp_ans:      !
1646     Divisor=8
1647     GOSUB Print_it
1648     RETURN
1649 Print_it:      !
1650     OFF KEY
1651     OUTPUT KBD;" 1,";Data_len/Divisor;
1652     INPUT "Enter starting element and # of elements to print.",Istart,Icount
1653     ON KEY 1 LABEL " Pause   print",Priority+1 GOSUB Pause_print
1654     ON KEY 3 LABEL " Abort   print",Priority+1 GOSUB Abort_print
1655     FOR I=Istart TO Istart+Icount-1
1656       PRINT I;" = ";Buf(I)
1657     NEXT I
1658     OFF KEY 1
1659     OFF KEY 2
1660     OFF KEY 3
1661     RETURN
1662 Disp_bin:      !
1663     OFF KEY
1664     OUTPUT KBD;" 1,";Data_len/2;
1665     INPUT "Enter starting element and # of elements to print.",Istart,Icount
1666     ON KEY 1 LABEL " Pause   print",Priority+1 GOSUB Pause_print
1667     ON KEY 3 LABEL " Abort   print",Priority+2 GOSUB Abort_print
1668     FOR I=Istart TO Istart+Icount-1
1669       PRINT I;" = ";Ibuf(I)
1670     NEXT I
1671     OFF KEY 1
1672     OFF KEY 2
1673     OFF KEY 3
1674     RETURN
1675 Pause_print:    !
1676     OFF KEY 1
1677     ON KEY 2 LABEL "  Cont   print",Priority+2 GOSUB Cont_print
1678     Print_paused=1
1679     REPEAT
1680     UNTIL NOT Print_paused
1681     RETURN
1682 Cont_print:    !
1683     OFF KEY 2
1684     ON KEY 1 LABEL " Pause   print",Priority+1 GOSUB Pause_print
1685     Print_paused=0
1686     RETURN
```

```
1687 Abort_print:    !
1688    I=Istart+Icount-1
1689    Print_paused=0
1690    CLEAR SCREEN
1691    RETURN
1692 Set_exit_flag:    !
1693    Exit_flag=1
1694    RETURN
1695  SUBEND
1696  !///////////////////////////////////////////////////////////////////////
1697 Xfer_synth_tbl:SUB Xfer_synth_tbl
1698  !
1699  !  Allows loading and dumping of the Synthesis table.
1700  !
1701    COM /Data/ Buf(*),INTEGER Ibuf(*),Data_len
1702    COM /Addresses/ Hp3563a,Disk_drive,Plottr
1703    INTEGER Exit_flag,I,Priority
1704    CLEAR SCREEN
1705    Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1706    FOR I=0 TO 9
1707      ON KEY I LABEL "",Priority CALL Do_nothing
1708    NEXT I
1709    Exit_flag=0
1710    REPEAT
1711      IF Data_len0 THEN
1712        ON KEY 1 LABEL "Load Tbl Ascii",Priority GOSUB Ltas
1713        ON KEY 2 LABEL "Load Tbl Binary",Priority GOSUB Ltbn
1714        ON KEY 3 LABEL "Load Tbl Ansi",Priority GOSUB Ltan
1715      END IF
1716      ON KEY 5 LABEL "Dump Tbl Ascii",Priority GOSUB Dtas
1717      ON KEY 6 LABEL "Dump Tbl Binary",Priority GOSUB Dtbn
1718      ON KEY 7 LABEL "Dump Tbl Ansi",Priority GOSUB Dtan
1719      ON KEY 8 LABEL " Return ",Priority GOSUB Set_exit_flag
1720    UNTIL Exit_flag
1721    SUBEXIT
1722 Set_exit_flag:    !
1723    Exit_flag=1
1724    RETURN
1725 Ltas:      !
1726    CALL Write_ascii(700+Hp3563a,"LTAS",Data_len,Buf(*))
1727    RETURN
1728 Dtas:      !
1729    CALL Read_ascii(700+Hp3563a,"DTAS",Data_len,Buf(*))
1730    RETURN
1731 Ltbn:      !
1732    CALL Write_binary(700+Hp3563a,"LTBN",Data_len,Ibuf(*))
1733    RETURN
1734 Dtbn:      !
1735    CALL Read_binary(700+Hp3563a,"DTBN",Data_len,Ibuf(*))
1736    RETURN
1737 Ltan:      !
1738    CALL Write_ansi(700+Hp3563a,"LTAN",Data_len,Buf(*))
1739    RETURN
1740 Dtan:      !
```

```
1741    CALL Read_ansi(700+Hp3563a,"DTAN",Data_len,Buf(*))
1742    RETURN
1743  SUBEND
1744  !/////////////////////////////////////////////////////////////////////////
1745 Xfer_state:SUB Xfer_state
1746  !
1747  ! Allows loading and dumping of the instrument setup state.
1748  !
1749    COM /Data/ Buf(*),INTEGER Ibuf(*),Data_len
1750    COM /Addresses/ Hp3563a,Disk_drive,Plottr
1751    INTEGER Exit_flag,I,Priority
1752    CLEAR SCREEN
1753    Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1754    FOR I=0 TO 9
1755      ON KEY I LABEL "",Priority CALL Do_nothing
1756    NEXT I
1757    Exit_flag=0
1758    REPEAT
1759      IF Data_len0 THEN
1760        ON KEY 1 LABEL "Ld State Ascii",Priority GOSUB Lsas
1761        ON KEY 2 LABEL "Ld State Binary",Priority GOSUB Lsbn
1762        ON KEY 3 LABEL "Ld State Ansi",Priority GOSUB Lsan
1763      END IF
1764      ON KEY 5 LABEL "Dmp Stat Ascii",Priority GOSUB Dsas
1765      ON KEY 6 LABEL "Dmp Stat Binary",Priority GOSUB Dsbn
1766      ON KEY 7 LABEL "Dmp Stat Ansi",Priority GOSUB Dsan
1767      ON KEY 8 LABEL " Return ",Priority GOSUB Set_exit_flag
1768    UNTIL Exit_flag
1769    SUBEXIT
1770 Set_exit_flag:    !
1771    Exit_flag=1
1772    RETURN
1773 Lsas:             !
1774    CALL Write_ascii(700+Hp3563a,"LSAS",Data_len,Buf(*))
1775    RETURN
1776 Dsas:             !
1777    CALL Read_ascii(700+Hp3563a,"DSAS",Data_len,Buf(*))
1778    RETURN
1779 Lsbn:             !
1780    CALL Write_binary(700+Hp3563a,"LSBN",Data_len,Ibuf(*))
1781    RETURN
1782 Dsbn:             !
1783    CALL Read_binary(700+Hp3563a,"DSBN",Data_len,Ibuf(*))
1784    RETURN
1785 Lsan:             !
1786    CALL Write_ansi(700+Hp3563a,"LSAN",Data_len,Buf(*))
1787    RETURN
1788 Dsan:             !
1789    CALL Read_ansi(700+Hp3563a,"DSAN",Data_len,Buf(*))
1790    RETURN
1791  SUBEND
1792  !/////////////////////////////////////////////////////////////////////////
1793 Xfer_data_trace:SUB Xfer_data_trace
1794  !
```

```
1795  ! Allows the loading and dumping of the current data trace.
1796  !
1797    COM /Data/ Buf(*),INTEGER Ibuf(*),Data_len
1798    COM /Addresses/ Hp3563a,Disk_drive,Plottr
1799    INTEGER Exit_flag,I,Priority
1800    CLEAR SCREEN
1801    Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1802    FOR I=0 TO 9
1803      ON KEY I LABEL "",Priority CALL Do_nothing
1804    NEXT I
1805    Exit_flag=0
1806    REPEAT
1807      IF Data_len0 THEN
1808        ON KEY 1 LABEL "Ld Trace Ascii",Priority GOSUB Ldas
1809        ON KEY 2 LABEL "Ld Trace Binary",Priority GOSUB Ldbn
1810        ON KEY 3 LABEL "Ld Trace Ansi",Priority GOSUB Ldan
1811      END IF
1812      ON KEY 5 LABEL "DmpTrace Ascii",Priority GOSUB Ddas
1813      ON KEY 6 LABEL "DmpTrace Binary",Priority GOSUB Ddbn
1814      ON KEY 7 LABEL "DmpTrace Ansi",Priority GOSUB Ddan
1815      ON KEY 8 LABEL " Return ",Priority GOSUB Set_exit_flag
1816    UNTIL Exit_flag
1817    SUBEXIT
1818 Set_exit_flag:    !
1819    Exit_flag=1
1820    RETURN
1821 Ldas:             !
1822    CALL Write_ascii(700+Hp3563a,"LDAS",Data_len,Buf(*))
1823    RETURN
1824 Ddas:             !
1825    CALL Read_ascii(700+Hp3563a,"DDAS",Data_len,Buf(*))
1826    RETURN
1827 Ldbn:             !
1828    CALL Write_binary(700+Hp3563a,"LDBN",Data_len,Ibuf(*))
1829    RETURN
1830 Ddbn:             !
1831    CALL Read_binary(700+Hp3563a,"DDBN",Data_len,Ibuf(*))
1832    RETURN
1833 Ldan:             !
1834    CALL Write_ansi(700+Hp3563a,"LDAN",Data_len,Buf(*))
1835    RETURN
1836 Ddan:             !
1837    CALL Read_ansi(700+Hp3563a,"DDAN",Data_len,Buf(*))
1838    RETURN
1839  SUBEND
1840  !///////////////////////////////////////////////////////////////////////
1841 Xfer_xfrm_block:SUB Xfer_xfrm_block
1842  !
1843  ! Allows dumping of the coordinate transform block.
1844  !
1845    COM /Data/ Buf(*),INTEGER Ibuf(*),Data_len
1846    COM /Addresses/ Hp3563a,Disk_drive,Plottr
1847    INTEGER Exit_flag,I,Priority
1848    CLEAR SCREEN
```

```
1849     Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1850     FOR I=0 TO 9
1851       ON KEY I LABEL "",Priority CALL Do_nothing
1852     NEXT I
1853     Exit_flag=0
1854     ON KEY 5 LABEL "DmpBlock Ascii",Priority GOSUB Dcas
1855     ON KEY 6 LABEL "DmpBlock Binary",Priority GOSUB Dcbn
1856     ON KEY 7 LABEL "DmpBlock Ansi",Priority GOSUB Dcan
1857     ON KEY 8 LABEL " Return ",Priority GOSUB Set_exit_flag
1858     REPEAT
1859     UNTIL Exit_flag
1860     SUBEXIT
1861 Set_exit_flag:     !
1862     Exit_flag=1
1863     RETURN
1864 Dcas:              !
1865     CALL Read_ascii(700+Hp3563a,"DCAS",Data_len,Buf(*))
1866     RETURN
1867 Dcbn:              !
1868     CALL Read_binary(700+Hp3563a,"DCBN",Data_len,Ibuf(*))
1869     RETURN
1870 Dcan:              !
1871     CALL Read_ansi(700+Hp3563a,"DCAN",Data_len,Buf(*))
1872     RETURN
1873  SUBEND
1874  !////////////////////////////////////////////////////////////////////////////
1875 Xfer_disp_block:SUB Xfer_disp_block
1876  !
1877  ! Dumps the vector display buffer that contains the softkey labels.
1878  !
1879     COM /Data/ Buf(*),INTEGER Ibuf(*),Data_len
1880     COM /Addresses/ Hp3563a,Disk_drive,Plottr
1881     INTEGER Exit_flag,I,Priority
1882     CLEAR SCREEN
1883     Priority=VAL(SYSTEM$("SYSTEM PRIORITY"))+1
1884     FOR I=0 TO 9
1885       ON KEY I LABEL "",Priority CALL Do_nothing
1886     NEXT I
1887     Exit_flag=0
1888     ON KEY 5 LABEL "DmpBlock Ascii",Priority GOSUB Dvas
1889     ON KEY 6 LABEL "DmpBlock Binary",Priority GOSUB Dvbn
1890     ON KEY 7 LABEL "DmpBlock Ansi",Priority GOSUB Dvan
1891     ON KEY 8 LABEL " Return ",Priority GOSUB Set_exit_flag
1892     REPEAT
1893     UNTIL Exit_flag
1894     SUBEXIT
1895 Set_exit_flag:     !
1896     Exit_flag=1
1897     RETURN
1898 Dvas:              !
1899     CALL Read_ascii(700+Hp3563a,"DVAS",Data_len,Buf(*))
1900     RETURN
1901 Dvbn:              !
1902     CALL Read_binary(700+Hp3563a,"DVBN",Data_len,Ibuf(*))
```

```
1903    RETURN
1904 Dvan:                !
1905    CALL Read_ansi(700+Hp3563a,"DVAN",Data_len,Buf(*))
1906    RETURN
1907  SUBEND
1908  !////////////////////////////////////////////////////////////////////////////
1909 Nextpoint:SUB Nextpoint(X,Y,Dir)
1910  !
1911  ! Used by 'Demo_1' to increment x and y locations for graphics demo.
1912  !
1913    INTEGER Incr
1914    Incr=100
1915    ON Dir GOSUB Dright,Dup,Dleft,Ddown
1916    SUBEXIT
1917    !
1918 Dright:X=X+Incr
1919    Y=250
1920    IF X+Incr>1750 THEN Dir=2
1921    RETURN
1922 Dup:Y=Y+Incr
1923    X=1750
1924    IF Y+Incr>2000 THEN Dir=3
1925    RETURN
1926 Dleft:X=X-Incr
1927    Y=2000
1928    IF X-Incr<0 THEN Dir=4
1929    RETURN
1930 Ddown:Y=Y-Incr
1931    X=0
1932    IF Y-Incr<250 THEN Dir=1
1933    RETURN
1934  SUBEND
1935  !////////////////////////////////////////////////////////////////////////////
```

Index

E

S

T

U

V

W

X

Hewlett-Packard Sales and Service Offices

To obtain Servicing information or to order replacement parts, contact the nearest Hewlett-Packard Sales and Service Office listed in HP Catalog, or contact the nearest regional office listed below:

In the United States

California
P.O. Box 4230
1421 South Manhattan Avenue
Fullerton 92631

Georgia
P.O. Box 105005
2000 South Park Place
Atlanta 30339

Illinois
5201 Tollview Drive
Rolling Meadows 60008

New Jersey
W. 120 Century Road
Paramus 07652

In Canada
Hewlett-Packard (Canada) Ltd.
17500 South Service Road
Trans-Canada Highway
Kirkland, Quebec H9J 2M5

In France
Hewlett-Packard France
F-91947 Les Ulis Cedex
Orsay

In German Federal Republic
Hewlett-Packard GmbH
Vertriebszentrale Frankfurt
Berner Strasse 117
Postfach 560 140
D-6000 Frankfurt 56

In Great Britain
Hewlett-Packard Ltd.
King Street Lane
Winnersh, Wokingham
Berkshire RG11 5AR

In Other European Countries
Switzerland
Hewlett-Packard (Schweiz) AG
7, rue du Bois-du-Lan
Case Postale 365
CH-1217 Meyrin

In All Other Locations
Hewlett-Packard Inter-Americas
3155 Porter Drive
Palo Alto, California 94304