Programmer's Guide

HP 8753D Network Analyzer

HEWLETT PACKARD

HP Part No. 08753-90256
Printed in USA June 1994

Notice.

The information contained in this document is subject to change without notice.

Hewlett-Packard makes no warranty of any kind with regard to this material, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

1400 Fountaingrove Parkway, Santa Rosa CA, 95403-1799, USA

Caution	Always use the three-prong ac power cord supplied with this instrument. Failure to ensure adequate earth grounding by not using this cord may cause instrument damage.

Caution	This instrument has autoranging line voltage input; be sure the supply voltage is within the specified range.

Caution	**Ventilation Requirements:** When installing the instrument in a cabinet, the convection into and out of the instrument must not be restricted. The ambient temperature (outside the cabinet) must be less than the maximum operating temperature of the instrument by 4 °C for every 100 watts dissipated in the cabinet. If the total power dissipated in the cabinet is greater than 800 watts, then forced convection must be used.

 The instruction documentation symbol. The product is marked with this symbol when it is necessary for the user to refer to the instructions in the documentation.

"CE" The CE mark is a registered trademark of the European Community. (If accompanied by a year, it is when the design was proven.)

"ISM1-A" This is a symbol of an Industrial Scientific and Medical Group 1 Class A product.

"CSA" The CSA mark is a registered trademark of the Canadian Standards Association.

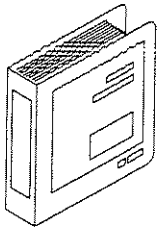
How to Use This Guide

This guide uses the following conventions:

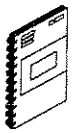
Front-Panel Key	This represents a key physically located on the instrument.
`Softkey`	This represents a "softkey," a key whose label is determined by the instrument's firmware.
`Screen Text`	This represents text displayed on the instrument's screen.

HP 8753D Network Analyzer Documentation Map

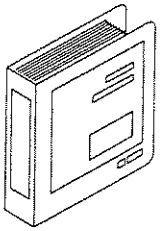
The **Installation and Quick Start Guide** familiarizes you with the HP 8753D network analyzer's front and rear panels, electrical and environmental operating requirements, as well as procedures for installing, configuring, and verifying the operation of the analyzer.



The **User's Guide** shows how to make measurements, explains commonly-used features, and tells you how to get the most performance from your analyzer.



The **Quick Reference Guide** provides a summary of all available user features.



The **Programmer's Guide** provides programming information including: an HP-IB command reference, an HP-IB programming reference, as well as programming examples.

The **System Verification and Test Guide** provides the system verification and performance tests and the Performance Test Record for your HP 8753D network analyzer.

Contents

Figures

Tables

Contents

Figures

Tables

1

Command Reference

This chapter is a reference for operation of the HP 8753D network analyzer under HP-IB control. You should already be familiar with making measurements with the analyzer. Information about the HP-IB commands is organized as follows:

- overview of HP-IB, including:
  - HP-IB capabilities of the analyzer
  - held commands
  - operation complete (OPC) commands
  - interrogate commands
  - syntax
- key select codes arranged by front-panel hardkey
- HP-IB only commands
- measurement calibration sequence
- display graphics

For information about manual operation of the analyzer, refer to the *HP 8753D Network Analyzer User's Guide.*

HP-IB Capabilities

As defined by the IEEE 488.1 standard, the analyzer has the following capabilities:

SH1	Full source handshake capability.
AH1	Full acceptor handshake capability.
T6	Can be a basic talker, answers serial poll, unaddresses if MLA is issued.
TE0	No extended talker capabilities.
L4	Acts as a basic listener and unaddresses if MTA is issued.
LE0	No extended listener capabilities.
SR1	Can issue service requests.
RL1	Will do remote, local, and local lockout.
PP0	Does not respond to parallel poll.
DC1	Device clear capability.
DT1	Will respond to device trigger in hold mode.
C1, C2, C3, C10	No controller capabilities in talker/listener mode. System controller mode can be selected under the LOCAL menu. Pass control capability in pass control mode.
E2	Tri-state drivers.

These codes are completely explained in the IEEE Std 488-1978 document, published by the Institute of Electrical and Electronic Engineers, Inc., 345 East 47th Street, New York, New York 11017.

Held Commands

The analyzer cannot process HP-IB commands while executing certain key commands, called held commands. Once a held command is received, the analyzer will read new commands into the input buffer, but it will not begin the execution of any commands until the completion of the held command. When the 15-character input buffer is full, the analyzer will hold off the bus until it is able to process the commands in the buffer.

Note Commands that call a calibration class are held if there is just one standard in the class, since such commands trigger a measurement.

Operation Complete

You can synchronize programs with the execution of certain held commands with the operation complete command. This function is enabled by issuing `OPC;` or `OPC?;` prior to an OPC-compatible command. The operation complete bit will then be set at the completion of the OPC-compatible command's execution. For example, issuing `OPC;SING;` causes the OPC bit to be set when the single sweep is finished. Issuing `OPC?;` causes the analyzer to output a one when the command execution is complete. Addressing the analyzer to talk after issuing `OPC?;` will not cause an "addressed to talk without selecting output" error, but the analyzer will halt the computer by not transmitting the one until the command has completed. For example, issuing `OPC?;PRES;`, and then immediately interrogating the analyzer causes the bus to halt until the instrument preset is complete and the analyzer outputs a one.

Table 1-1. OPC-compatible Commands

CHAN1	FWDM[1]	RECAREG<01 to 32>
CHAN2	FWDT[1]	REFD
CLASS11A[1]	HARMOFF	RESPDONE
CLASS11B[1]	HARMSEC	REVI[1]
CLASS11C[1]	HARMTHIR	REVM[1]
CLASS22A[1]	INSMEXSA	REVT[1]
CLASS22B[1]	INSMEXSM	RST
CLASS22C[1]	INSMNETA	SAV1
CLEARALL	INSMTUNR	SAV2
DATI	ISOD	SAVC
DONE	MANTRIG	SAVE<1 to 5>
EDITDONE	NOOP	SAVEREG<01 to 32>
EXTTOFF	NUMG	SING
EXTTON	OMII[1]	STAN<A to G>
EXTTPOIN	PRES	TRAD
FREQOFFS<ON|OFF>	RAID	WAIT
FWDI[1]	RECA<1 to 5>	

1 The class commands are OPC-compatible if there is only one standard in the class.

Command Interrogate

To interrogate one of the front-panel-equivalent commands listed in Table 1-2, append a question mark to the command root. This causes the analyzer to output the state of that function as a single number in ASCII format, form 4. If the function is a settable function, such as power or sweep time, the analyzer will output the current value of that function. If the function is either ON/OFF or one selection of several, the analyzer outputs a one for ON or selected, and a zero for OFF. If a command that does not have a defined response is interrogated, the analyzer outputs a zero. Interrogating a function does not put it in the active entry area.

Identification

The analyzer's response to `IDN?;` is "HEWLETT PACKARD,8753D,0,X.XX" where X.XX is the firmware revision of the instrument.

Output Queue

Whenever a command to output data is received, the analyzer puts the data into the output queue to be copied out by the next read operation. The queue, however, is only one event long: the next command to output data will overwrite the data already in the queue. Therefore, it is important to read the output queue immediately after every interrogation or request to output data.

Units

The analyzer outputs data in basic units such as Hz, dB, seconds, ohms, etc.

Input data is assumed to be in basic units unless one of the following units expressions qualifies the data input (upper and lower case are equivalent):

S	Seconds	HZ	Hertz
MS	Milliseconds	KHZ	Kilohertz
US	Microseconds	MHZ	Megahertz
NS	Nanoseconds	GHZ	Gigahertz
PS	Picoseconds	DB	dB or dBm
FS	Femtoseconds	V	Volts

Input Data

Input Syntax

The HP-IB commands accepted by the analyzer can be grouped into four input syntax types. The analyzer does not distinguish between upper and lower case letters.

General Structure

[code][appendage] [data] [unit] [terminator]

[code] The root mnemonic (these codes are described in the "Alphabetical Code Listing" later in this chapter.)

[appendage] A qualifier attached to the root mnemonic. Possible appendages are ON or OFF, which toggle a function ON or OFF, or integers, which specify one option out of several. There can be no spaces or symbols between the code and the appendage.

[data] A single operand used by the root mnemonic, usually to set the value of a function. The data can be a number or a character string. Numbers are accepted as integers or decimals, with power of ten specified by E, as in STAR 0.2E+10, which sets the start frequency to 2 GHz. Character strings must be enclosed by double quotation marks.

[unit] The units of the operand, if applicable. If no units are specified, the analyzer assumes the basic units as described under "Units" earlier in this chapter. The data is entered into the function when either units or a terminator is received.

[terminator] Indicates the end of the command, enters the data, and turns OFF the active entry area. The terminator should be a semicolon. Terminators are not necessary for the analyzer to interpret commands correctly, but in the case of a syntax error, the analyzer will attempt to recover at the next terminator. The analyzer also interprets line feeds and HP-IB END OR IDENTIFY (EOI) messages as terminators.

The specific syntaxes are as follows:

SYNTAX TYPE 1: [code] [terminator]

These are simple action commands that require no complementary information, such as `AUTO;` (auto scales the active channel.)

SYNTAX TYPE 2: [code][appendage] [terminator]

These are simple action commands requiring limited customization, such as `CORRON;`, `CORROFF;` (turn error correction ON or OFF,) or`RECA1;`, `RECA2;`, `RECA3;` ... (recall register 1, 2, 3 ...) There can be no characters or symbols between the code and the appendage. In the following cases: CLEAREG[D], RECAREG[D], SAVEREG[D], and TITREG[D], [D] must be 2 characters. For example, `CLEAREG01;` is correct; `CLEAREG1;` is not.

SYNTAX TYPE 3: [code] [data] [unit] [terminator]

These are data input commands such as `STAR 1.0 GHZ;` (set the start frequency to 1 GHz).

SYNTAX TYPE 4: [code][appendage] [data] [terminator]

These are titling and marker commands that have an appendage, such as TITL "FILTER" (enter FILTER as the CRT title), TITR1 "STATE1", TITR2 "EMPTY" (title register 1 STATE1, title register 2 EMPTY.)

INTERROGATE SYNTAX: [code][?]

To interrogate a front-panel-equivalent function, append a question mark to the root mnemonic. (For example POWE?, AVERO?, or REAL?) To interrogate commands with integer appendages, place the question mark after the appendage.

Valid Characters

The analyzer will accept letters, changing lower case to upper case, numbers, decimal points, +, semicolons, carriage returns and line feeds. Leading zeros, spaces, carriage returns, and unnecessary terminators are ignored, except when inserted into or between a mnemonic and/or an appendage. If the analyzer does not recognize a character as appropriate, it generates a syntax error message and recovers at the next terminator.

Programming Data

The front-panel-equivalent command mnemonics are listed in Table 1-2. These commands perform the same function as a front-panel key, and are arranged functionally by front-panel key. Table 1-3 lists functions that have no logical equivalent in manual operation. They concern data transmission, status reporting, and special HP-IB functions. Following these tables is a list of all these mnemonics with their descriptions.

In most cases, the commands were named following these rules:

1. Simple commands are the first four letters of the function they control, as in POWE. If the function label is two words, the first three mnemonic letters are the first three letters of the first word, and the fourth mnemonic letter is the first letter of the second word. For example, ELED is derived from electrical delay.
2. If there are many commands grouped together in a class, as in markers or plotting pen numbers, the command is increased to eight letters. The first four letters are the class label derived using rule one. The last four letters are the function specifier, again derived using rule one. An example of this is the class pen numbers PENN, which is used with several functions such as PENNDATA, PENNMEMO.

These rules were not always followed in order to maintain compatibility with other products, to make commands more meaningful and easier to remember, and when technical considerations prevented their use.

Key Select Codes for the HP 8753D Network Analyzer

The HP-IB mnemonics in this table are functionally arranged by their front-panel key equivalent.

Column headings:

Function	The front-panel function affected by the mnemonic.
Action	The effects of the mnemonic on that function.
Mnemonic	The HP-IB mnemonic.
S	Syntax type. See "Input Syntax."
?	Interrogate response. If a response is defined, it is listed.
O	OPC-compatible command.
Range	The range of acceptable inputs and corresponding units.

Symbol conventions:

[ ]	An optional operand.
D	A numerical operand.
I	An integer appendage that is part of the command. For example, CLEA<I>, where I=1 to 5, indicates that the actual commands are CLEA1, CLEA2, CLEA3, CLEA4, and CLEA5.
$	A character string operand which must be enclosed by quotes.
< >	A necessary appendage.
|	An either/or choice in appendages.

Table 1-2. Key Select Codes

Function	Action	Mnemonic	S	?	O	Range
AVG						
Averaging	Restart	AVERREST	1			
	Factor	AVERFACT[D]	3	D		0 to 999
	On/off	AVERO<ON|OFF>	2	1,0		
Smoothing	Set aperture	SMOOAPER[D]	3	D		0.05 to 20%
	On/off	SMOOO<ON|OFF>	2	1,0		
IF bandwidth	Set bandwidth	IFBW[D]	3	D		10, 30, 100, 300, 1000, 3000 Hz
CAL-error correction, calibration						
Correction	On/off	CORR<ON|OFF>	2	1,0		
Interpolative correction	On/off	CORI<ON|OFF>	2	1,0		
Cal sequence	Resume	RESC	1			
Port extensions	Port 1	PORT1[D]	3	D		±10 s
	Port 2	PORT2[D]	3	D		±10 s
	Input A	PORTA[D]	3	D		±10 s
	Input B	PORTB[D]	3	D		±10 s
	Off	PORE<ON|OFF>	2	1,0		
Velocity factor	Set value	VELOFACT[D]	3	D		0 to 10
ZO	Set Value	SETZ[D]	3	D		0.1 to 500Ω
Test set switching	Continuous/full 2-port cal	CSWION	2	1,0		
	Hold/quasi 2-port cal	CSWIOFF	2	1,0		
Begin cal sequence	Response	CALIRESP	1	0,1		
	Response and Isol	CALIRAI	1	0,1		
	S11 1-port	CALIS111	1	0,1		
	S22 1-port	CALIS221	1	0,1		
	Full 2-port	CALIFUL2	1	0,1		
	One path 2-port	CALIONE2	1	0,1		
	TRL*/LRM* 2-port	CALITRL2	1	0,1		
	None	CALN	1	0,1		

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
Intermediate cal steps, 1 path/2-port	Isolation	ISOOP	1			
	Reflection	REFOP	1			
	Transmission	TRAOP	1			
Intermediate cal steps, 2-port cal	Transmission	TRAN	1			
	Reflection (incl. Line in TRL)	REFL	1			
	(incl. Line in TRL)					
	Isolation	ISOL	1			
Select response & isol. class	Response	RAIRESP	1			
	Isolation	RAIISOL	1			
Select reflection class	S11A (open)	CLASS11A	1		OPC††	
	S11B (short)	CLASS11B	1		OPC††	
	S11C (load)	CLASS11C	1		OPC††	
	S22A (open)	CLASS22A	1		OPC††	
	S22B (short)	CLASS22B	1		OPC††	
	S22C (load)	CLASS22C	1		OPC††	
Select transmission class	Fwd transmission	FWDT	1		OPC††	
	Rev transmission	REVT	1		OPC††	
	Fwd match	FWDM	1		OPC††	
	Rev match	REVM	1		OPC††	
Select isolation class	Forward isolation	FWDI	1		OPC††	
	Reverse isolation	REVI	1		OPC††	
	Omit isolation	OMII	1		OPC††	
Select standard in class	Standard A	STANA	1		OPC	
	Standard B	STANB	1		OPC	
	Standard C	STANC	1		OPC	
	Standard D	STAND	1		OPC	
	Standard E	STANE	1		OPC	
	Standard F	STANF	1		OPC	
	Standard G	STANG	1		OPC	
Sliding load	Set	SLIS	1			
	Done	SLID	1			
Done with:	Class	DONE	1		OPC	
	Isolation	ISOD	1		OPC	
	Reflection	REFD	1		OPC	
	Transmission	TRAD	1		OPC	

†† The class commands are OPC-compatible if there is only one standard in the class. If there is just one standard, that standard is measured automatically. If there is more than one standard in the class, the class command only calls another menu. Also, the reflection classes "B" and "C" (e.g., S11B, S11C) are actually transmission measurements of the "LINE" when the TRL calibration type is selected. The transmission classes are used for the "THRU" when TRL calibration is selected.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
Save cal	Response	RESPDONE	1		OPC	
	Resp and isol	RAID	1		OPC	
	1-port cal	SAV1	1		OPC	
	2-port cal	SAV2	1		OPC	
CAL-calibration kits						
Select default kits	7 mm	CALK7MM	1	1,0		
	3.5 mmC	CALK35MM	1	1,0		
	3.5 mmD	CALK35MD	1	1,0		
	Type N, 50 ohm	CALKN50	1	1,0		
	Type N, 75 ohm	CALKN75	1	1,0		
	User-defined	CALKUSED	1	1,0		
Modify kit	Modify current	MODI1	1			
Define std. number (begin std. definition)		DEFS[D]	3			
Define std. type	Open	STDTOPEN	1	1,0		
	Short	STDTSHOR	1	1,0		
	Load	STDTLOAD	1	1,0		
	Delay/thru	STDTDELA	1	1,0		
	Arbitrary imped.	STDTARBI	1	1,0		
Define std. parameters	Open cap. C0	C0[D]	3			±10k (10^{-15} F)
	Open cap. C1	C1[D]	3			±10k (10^{-27} F/Hz)
	Open cap. C2	C2[D]	3			±10k (10^{-36} F/Hz^2)
	Open cap. C3	C3[D]	3			±10k (10^{-45} F/Hz^3)
	Fixed load	FIXE	1			
	Sliding load	SLIL	1			
	Terminal imped.	TERI[D]	3			0 to 1 kohm;
Define std. offsets	Delay	OFSD[D]	3			±1 s
	Loss	OFSL[D]	3			0 to 1000 TΩ/s
	Z0	OFSZ[D]	3			0.1 to 500Ω
	Min. frequency	MINF[D]	3			0 to 1000 GHz
	Max. frequency	MAXF[D]	3			0 to 1000 GHz
	Coaxial	COAX	1	0,1		
	Waveguide	WAVE	1	0,1		
Std. done	Standard defined	STDD	1			
Label std		LABS[$]	3			10 char.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
Specify class	Response	SPECRESP[I,I..]	3			Std numbers
	Resp & Isol	SPECRESI[I,I..]	3			Std numbers
	SllA (open)	SPECS11A[I,I..]	3			Std numbers
	SllB (short)	SPECS11B[I,I..]	3			Std numbers
	SllC (load)	SPECS11C[I,I..]	3			Std numbers
	S22A (open)	SPECS22A[I,I..]	3			Std numbers
	S22B (short)	SPECS22B[I,I..]	3			Std numbers
	S22C (load)	SPECS22C[I,I..]	3			Std numbers
	Forward Trans	SPECFWDT[I,I..]	3			Std numbers
	Forward Match	SPECFWDM[I,I..]	3			Std numbers
	Reverse Trans	SPECREVT[I,I..]	3			Std numbers
	Reverse Match	SPECREVM[I,I..]	3			Std numbers
	TRL,Reflect,Forward,Match	SPECTRFM[I,I..]	3			Std numbers
	TRL,Reflect,Reverse,Match	SPECTRRM[I,I..]	3			Std numbers
	TRL,Line ,Forward,Match	SPECTLFM[I,I..]	3			Std numbers
	TRL,Line ,Forward,Trans	SPECTLFT[I,I..]	3			Std numbers
	TRL,Line ,Reverse,Match	SPECTLRM[I,I..]	3			Std numbers
	TRL,Line ,Reverse,Trans	SPECTLRT[I,I..]	3			Std numbers
	TRL,Thru ,Forward,Match	SPECTTFM[I,I..]	3			Std numbers
	TRL,Thru ,Forward,Trans	SPECTTFT[I,I..]	3			Std numbers
	TRL,Thru ,Reverse,Match	SPECTTRM[I,I..]	3			Std numbers
	TRL,Thru ,Reverse,Trans	SPECTTRT[I,I..]	3			Std numbers
Class done		CLAD	1			
Label class	Response	LABERESP[$]	3			10 char.
	Resp. & isolation	LABERESI[$]	3			10 char.
	SllA	LABES11A[$]	3			10 char.
	SllB	LABES11B[$]	3			10 char.
	SllC	LABES11C[$]	3			10 char.
	S22A	LABES22A[$]	3			10 char.
	S22B	LABES22B[$]	3			10 char.
	S22C	LABES22C[$]	3			10 char.
	Forward Trans	LABEFWDT[$]	3			10 char.
	Forward Match	LABEFWDM[$]	3			10 char.
	Reverse Trans	LABEREVT[$]	3			10 char.
	Reverse Match	LABEREVM[$]	3			10 char.
	TRL, Reflect, Forward, Match	LABETRFM[$]	3			10 char.
	TRL, Reflect, Reverse, Match	LABETRRM[$]	3			10 char.
	TRL, Line, Forward, Match	LABETLFM[$]	3			10 char.
	TRL, Line, Forward, Trans	LABETLFT[$]	3			10 char.
	TRL, Line, Reverse, Match	LABETLRM[$]	3			10 char.
	TRL, Line, Reverse, Trans	LABETLRT[$]	3			10 char.
	TRL, Thru, Forward, Match	LABETTFM[$]	3			10 char.
	TRL, Thru, Forward, Trans	LABETTFT[$]	3			10 char.
	TRL, Thru, Reverse, Match	LABETTRM[$]	3			10 char.
	TRL, Thru, Reverse, Trans	LABETTRT[$]	3			10 char.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
Label kit		LABK[$]	3			10 char.
Kit done		KITD	1			
Save kit	Into user kit	SAVEUSEK	1			
CAL-power meter calibration						
Power meter cal	Off	PWMCOFF[D]	3	D		Cal power: −100 to 100 dB
	Each sweep	PWMCEACS[D]	3	D		Cal power: −100 to 100 dB
	One sweep	PWMCONES[D]	3	D		Cal power: −100 to 100 dB
	Take cal sweep§	TAKCS	1			
	Number of readings	NUMR[D]	3	D		1 to 100
	Set port cal pwr	PWRMCAL	1			
Edit power loss table	On/off	PWRLOSS<ON|OFF>	2	1,0		
	Edit list	POWLLIST	1			
	Use sensor A or B	USES<ENSA|ENSB>	2			Sensor B - HP 438A only
	Add segment	SADD	1			
	Edit segment N	SEDI[D]	3	D		1 to 12
	Done with segment	SDON	1			
	Delete segment	SDEL	1			
	Done	EDITDONE	1		OPC	
	Clear list	CLEL	1			
Edit power loss segment	Frequency	POWLFREQ[D]	3	D		Stimulus range†
	Value	POWLLOSS[D]	3	D		−9900 to 9900 dB
Edit cal sensor table	Edit sensor menu A	CALFSENA	1			
	Edit sensor menu B	CALFSENB	1			HP 438A only
	Add segment	SADD	1			
	Edit segment N	SEDI[D]	3	D		1 to 12
	Done with segment	SDON	1			
	Delete segment	SDEL	1			
	Done	EDITDONE	1		OPC	
	Clear list	CLEL	1			
Edit cal sensor segment	Frequency	CALFFREQ[D]	3	D		Stimulus range†
	Cal factor	CALFCALF[D]	3	D		0 to 200%

† For frequency sweeps: 30 kHz to 3 GHz. (To 6 GHz for option 006). For power sweeps: −15 to 20 dBm in range 0, 25 dB maximum in other ranges (−100 to +100 with power meter cal on). For CW time: 0 to 24 hours. For frequency sweep, transform on: ± 1/frequency step. For CW time sweep, transform on: ±1/time step.

§ Requires pass control mode when using the HP-IB port.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
CHANNEL						
Channel	CH 1 active	CHAN1	1		OPC	
	CH 2 active	CHAN2	1		OPC	
COPY						
Copy display	To printer§	PRINALL	1			
	To plotter§	PLOT	1			
Printer	Auto feed	PRNTRAUTF<ON|OFF>	2	1,0		
Printer	Form feed	PRNTRFORF	1			
Printer setup	Default	DEFLPRINT	1			
Plotter	Auto feed	PLTTRAUTF<ON|OFF>	2	1,0		
Plotter	Form feed	PLTTRFORF	1			
Plotter setup	Default	DFLT	1			
List values		LISV	1			
Operating parameters		OPEP	1			
Print list values or operating and marker parameters	Raster display dump to HP-IB§	PRINTALL	1			
Next page		NEXP	1			
Restore display		RESD	1			
Select print color	Monochrome	PRIS	1			
	Color	PRIC	1			

§ Requires pass control mode when using the HP-IB port.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
Print feature color	Data channel 1	PCOLDATA1<color>	2			Colors‡
	Data channel 2	PCOLDATA2<color>	2			Colors‡
	Memory channel 1	PCOLMEMO1<color>	2			Colors‡
	Memory channel 2	PCOLMEMO2<color>	2			Colors‡
	Graticule	PCOLGRAT<color>	2			Colors‡
	Text	PCOLTEXT<color>	2			Colors‡
	Warning	PCOLWARN<color>	2			Colors‡
Quadrant	Left lower	LEFL	1	0,1		
	Left upper	LEFU	1	0,1		
	Right lower	RIGL	1	0,1		
	Right upper	RIGU	1	0,1		
	Full page	FULP	1	0,1		
Features to be plotted	Data	PDATA<ON|OFF>	2	1,0		
	Memory	PMEM<ON|OFF>	2	1,0		
	Graticule	PGRAT<ON|OFF>	2	1,0		
	Text	PTEXT<ON|OFF>	2	1,0		
	Marker	PMKR<ON|OFF>	2	1,0		
Pen number	Data	PENNDATA[D]	3			0,1,2 . . . 10
	Memory	PENNMEMO[D]	3			0,1,2 . . . 10
	Graticule	PENNGRAT[D]	3			0,1,2 . . . 10
	Text	PENNTEXT[D]	3			0,1,2 . . . 10
	Marker	PENNMARK[D]	3			0,1,2 . . . 10
Line type	Data	LINTDATA[D]	3			0,1,2 . . . 10
	Memory	LINTMEMO[D]	3			0,1,2 . . . 10
Plot scale	Full page	SCAPFULL	1			
	Graticule to p1,p2	SCAPGRAT	1			
Plot speed	Slow	PLOSSLOW	1			
	Fast	PLOSFAST	1			

‡ Colors = white|cyan|magenta|blue|yellow|green|red|black

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
		DISPLAY				
Channels	Dual on/off	DUAC<ON|OFF>	2	1,0		
	Split on/off	SPLD<ON|OFF>	2	1,0		
	D2/D1 to D2	D1DIVD2<ON|OFF>	2	1,0		
Display	Data	DISPDATA	1	0,1		
	Memory only	DISPMEMO	1	0,1		
	Data and mem	DISPDATM	1	0,1		
	Data/mem	DISPDDM	1	0,1		
	Data — mem	DISPDMM	1	0,1		
	Data to mem	DATI	1	0,1	OPC	
Beeper	On done	BEEPDONE<ON|OFF>	2	1,0		
	On warning message	BEEPWARN<ON|OFF>	2	1,0		
CRT	Intensity	INTE[D]	3	D		0 to 100
	Focus	FOCU[D]	3	D		0 to 100
	Title	TITL[$]	4	$		48 char.
Frequency notation	Blank	FREO	1			
Adjust display	Background intensity	BACI[D]	3	D		0 to 100
	Save colors	SVCO	1			
	Recall colors	RECO	1			
	Default colors	DEFC	1			
Modify colors	Ch 1 data/lim ln	COLOCH1D[D]	3	D		0 to 100
	Ch 1 memory	COLOCH1M[D]	3	D		0 to 100
	Ch 2 data/lim ln	COLOCH2D[D]	3	D		0 to 100
	Ch 2 memory	COLOCH2M[D]	3	D		0 to 100
	Graticule	COLOGRAT[D]	3	D		0 to 100
	Text	COLOTEXT[D]	3	D		0 to 100
	Warning	COLOWARN[D]	3	D		0 to 100
Adjust color	Brightness	CBRI[D]	3	D		0 to 100
	Color	COLOR[D]	3	D		0 to 100
	Tint	TINT[D]	3	D		0 to 100
	Reset	RSCO	1			

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
ENTRY						
Step keys	Up	UP	1			
	Down	DOWN	1			
Entry off		ENTO	1			
FORMAT						
Format	Log mag	LOGM	1	0,1		
	Phase	PHAS	1	0,1		
	Delay	DELA	1	0,1		
	Smith chart	SMIC	1	0,1		
	Polar	POLA	1	0,1		
	Lin mag	LINM	1	0,1		
	Real	REAL	1	0,1		
	Imaginary	IMAG	1	0,1		
	SWR	SWR	1	0,1		
LOCAL						
HP-IB modes	Talker/listener	TALKLIST	1	0,1		
	Use pass control	USEPASC	1	0,1		
Debug	Display commands	DEBU<ON|OFF>	2	1,0		
Disk drive	Unit	DISCUNIT[D]	3	D		0 to 30
	Volume	DISCVOLU[D]	3	D		0 to 30
HP-IB addresses	Plotter	ADDRPLOT[D]	3	D		0 to 30
	Printer	ADDRPRIN[D]	3	D		0 to 30
	Disk drive	ADDRDISC[D]	3	D		0 to 30
	Controller	ADDRCONT[D]	3	D		0 to 30
Power meter	Address	ADDRPOWM[D]	3	D		0 to 30
	Type	POWM<ON|OFF>	2	0,1		On = 436A, Off = 438A/437B
Plotter type	Plotter	PLTTYPPLTR	1			
	HPGL printer	PLTTYPHPGL	1			
Printer type	ThinkJet	PRNTYPTJ	1			
	DeskJet	PRNTYPDJ	1			
	LaserJet	PRNTYPLJ	1			
	PaintJet	PRNTYPPJ	1			
	Epson-P2	PRNTYPEP	1			

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
Printer port	HP-IB	PRNPRTHPIB	1			
	Parallel	PRNPRTPARA	1			
	Serial	PRNPRTSERI	1			
Plotter port	HP-IB	PLTPRTHPIB	1			
	Parallel	PLTPRTPARA	1			
	Serial	PLTPRTSERI	1			
Printer serial port	Baud rate	PRNTRBAUD[D]	3	D		1200, 2400, 4800, 9600, 19200
Printer serial port	Handshake	PRNHNDSHK<XON|DTR>	2	1,0		
Plotter serial port	Baud rate	PLTTRBAUD[D]	3	D		1200, 2400, 4800, 9600, 19200
Plotter serial port	Handshake	PLTHNDSHK<XON|DTR>	2	1,0		
Parallel port	Configure	PARAL<GPIO|CPY>	2	0,1		GPIO=Gen.Purpose I/O, CPY=COPY use
MEAS						
Input ports	A/R	AR	1	0,1		
	B/R	BR	1	0,1		
	A/B	AB	1	0,1		
	A	MEASA	1	0,1		
	B	MEASB	1	0,1		
	R	MEASR	1	0,1		
	Selects testport 1 or 2	TSTP<P1|P2>	2			
S-parameters	S11	S11	1	0,1		
	S12	S12	1	0,1		
	S21	S21	1	0,1		
	S22	S22	1	0,1		
Conversion to alternate parameters	Off	CONVOFF	1	0,1		
	Z:reflection	CONVZREF	1	0,1		
	Z:transmission	CONVZTRA	1	0,1		
	Y:reflection	CONVYREF	1	0,1		
	Y:transmission	CONVYTRA	1	0,1		
	1/S	CONV1DS	1	0,1		
Analog input		ANAI[D]	1*	0,1		

* Syntax type 1 when ANABOFF. Syntax type 3, and range = 1 to 31, when ANABON.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
MENU (stimulus)						
Power	Level	POWE[D]	3	D		−85 to +20 dBm
	Trip	POWT<ON|OFF>	2	1,0		
	Always couple power	COUP<ON|OFF>	2	1,0		
	Port power coupling	PORTP<CPLD|UNCPLD>	2			
	Range 0	PRAN0	2			
	Range 1	PRAN1	2			
	Range 2	PRAN2	2			
	Range 3	PRAN3	2			
	Range 4	PRAN4	2			
	Range 5	PRAN5	2			
	Range 6	PRAN6	2			
	Range 7	PRAN7	2			
	Power range auto/manual	PWRR<PAUTO/PMAN>	2			
	Source power on/off	SOUP<ON|OFF>	2			
Time	Specify	SWET[D]	3	D		0.01 to 86,400 s
Measurement	Restart	REST	1			
Trigger	Hold	HOLD	1	0,1		
	Single	SING	1		OPC	
	Number of groups	NUMG[D]	3	D	OPC	1 to 999
	Continuous	CONT	1	0,1		
	External trigger off	EXTTOFF	2	0,1	OPC	
	External trigger on	EXTTON	2	0,1	OPC	
	External trigger on point	EXTTPOIN	1	0,1	OPC	
	Manual trigger on point	MANTRIG	1	0,1	OPC	
Points	Specify	POIN[D]	3	D		3, 11, 26, 51, 101 201, 401, 801, 1601
Coupled channels	On/off	COUC<ON|OFF>	2	1,0		
CW freq	Set value	CWFREQ[D]	3	D		Stimulus range†
Power slope	Value	SLOPE[D]	3	D		0 to 2 dB/GHz
	On/off	SLOPO<ON|OFF>	2	1,0		

† For frequency sweeps: 30 kHz to 3 GHz. (To 6 GHz for option 006). For power sweeps: −15 to 20 dBm in range 0, 25 dB maximum in other ranges (−100 to +100 with power meter cal on). For CW time: 0 to 24 hours. For frequency sweep, transform on: ± 1/frequency step. For CW time sweep, transform on: ±1/time step.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
MENU (stimulus) (continued)						
Sweep type	Linear	LINFREQ	1	0,1		
	Log	LOGFREQ	1	0,1		
	List	LISFREQ	1	0,1		
	Select a segment	SSEG[D]	3	0,1		1 to 30
	Select all segments	ASEG	1	0,1		
	Power	POWS	1	0,1		
	CW time	CWTIME	1	0,1		
Edit list	Begin	EDITLIST	1			
	Add segment	SADD	1			
	Edit segment N	SEDI[D]	3	D		1 to 30
	Done with segment	SDON	1			
	Delete segment	SDEL	1			
	Done	EDITDONE	1		OPC	
	Clear list	CLEL	1			
Edit segment	Start	STAR[D]	3	D		Stimulus range†
	Stop	STOP[D]	3	D		Stimulus range†
	Center	CENT[D]	3	D		Stimulus range†
	Span	SPAN[D]	3	D		Stimulus range†
	Points	POIN[D]	3	D		1 to 1632
	Stepsize	STPSIZE[D]	3	D		Stimulus range†
	CW	CWFREQ[D]	3	D		Stimulus range†
MARKER						
Select active	1 to 4	MARK<I>[D]	3	D		Stimulus range†
	All off	MARKOFF	1	0,1		
Marker zero	Zero offsets	MARKZERO	1			
Delta reference	1 to 4	DELR<I>	2	0,1		1 to 4
	Fixed marker	DELRFIXM	1	0,1		
	Mode off	DELO	1	0,1		
Fixed mkr position	Stimulus	MARKFSTI[D]	3	D		Stimulus range†
	Value	MARKFVAL[D]	3	D		Amplitude range#
	Aux value	MARKFAUV[D]	3	D		Amplitude range#

† For frequency sweeps: 30 kHz to 3 GHz. (To 6 GHz for option 006). For power sweeps: −15 to 20 dBm in range 0, 25 dB maximum in other ranges (−100 to +100 with power meter cal on). For CW time: 0 to 24 hours. For frequency sweep, transform on: ± 1/frequency step. For CW time sweep, transform on: ±1/time step.

For log mag: ± 500 dB. For phase: ± 500 degrees. For Smith chart and Polar: ± 500 units. For linear magnitude: ± 500 units. For SWR: ± 500 units. The scale is always positive, and has minimum values of .001 dB, 10e-12 degrees, 10e-15 seconds, and 10 picounits.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
Marker placement	Continuous	MARKCONT	1	0,1		
	Discrete	MARKDISC	1	0,1		
Coupled	Couple channels	MARKCOUP	1	0,1		
	Uncouple	MARKUNCO	1	0,1		
Displayed	On/off	DISM<ON|OFF>	2	1,0		
Polar markers	Log	POLMLOG	1	0,1		
	Linear	POLMLIN	1	0,1		
	Re/Im	POLMRI	1	0,1		
Smith markers	Linear	SMIMLIN	1	0,1		
	Log	SMIMLOG	1	0,1		
	Re/Im	SMIMRI	1	0,1		
	R+jX	SMIMRX	1	0,1		
	G+jB	SMIMGB	1	0,1		
MARKER FCTN						
Set function to marker value	Start	MARKSTAR	1			
	Stop	MARKSTOP	1			
	Center	MARKCENT	1			
	Span	MARKSPAN	1			
	Reference	MARKREF	1			
	Delay	MARKDELA	1			
Search	Off	SEAOFF	1	0,1		
	Maximum	SEAMAX	1	0,1		
	Minimum	SEAMIN	1	0,1		
	Target	SEATARG[D]	3	D		Amplitude range#
	Search left	SEAL	1			
	Search right	SEAR	1			
Width	Value	WIDV[D]	3	D		Amplitude range#
Search	Search on/off	WIDT<ON|OFF>	2	1,0		
Tracking search	On/off	TRACK<ON|OFF>	2	1,0		
Statistics	On/off	MEASTAT<ON|OFF>	2	1,0		

For log mag: ± 500 dB. For phase: ± 500 degrees. For Smith chart and Polar: ± 500 units. For linear magnitude: ± 500 units. For SWR: ± 500 units. The scale is always positive, and has minimum values of .001 dB, 10e-12 degrees, 10e-15 seconds, and 10 picounits.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
SAVE/RECALL-internal registers						
Save	Selected reg	SAVE<I>	2		OPC	1 to 5
	Selected reg	SAVEREG<I>	2		OPC	01 TO 32
Clear	Selected reg	CLEA<I>	2			1 to 5
	Selected reg	CLEAREG<I>	2			01 to 32
	All regs	CLEARALL	1		OPC	
Recall	Selected reg	RECA<I>	2		OPC	1 to 5
	Selected reg	RECAREG<I>	2		OPC	01 to 32
Title	Internal reg	TITR<I>[$]	4			1 to 5, 10 char.
	Internal reg	TITREG<I>[$]	4			01 to 32, 10 char.
SAVE/RECALL-disk files						
Purge	Selected file§	PURG<I>	2			1 to 5
Store	To disk§	STOR<I>	2			1 to 5
Title	Disk file	TITF<I>[$]	4			1 to 5, 10 char.
Include with disk files	Data	EXTMDATA<ON|OFF>	2	1,0		
	Raw data	EXTMRAW<ON|OFF>	2	1,0		
	Formatted data	EXTMFORM<ON|OFF>	2	1,0		
	User graphics	EXTMGRAP<ON|OFF>	2	1,0		
	Data only	EXTMDATO<ON|OFF>	2	1,0		
Save format	ASCII/citifile	SAVUASCI	1			
	Binary	SAVUBINA	1			
Load	From disk§	LOAD<I>	2			1 to 5
	File titles§	REFT	1			
Initialize	Internal disk	INID	1			
	External disk§	INIE	1			
	LIF Directory size	DIRS[D]	3	D		8 to 8192
Select storage	Internal disk	INTD	1			
	Internal memory	INTM	1			
	External disk	EXTD	1			
Disk format	DOS	FORMATDOS	1			
	LIF	FORMATLIF	1			

§ Requires pass control mode when using the HP-IB port.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
SCALE REF						
Scale	Auto	AUTO	1			
	Value	SCAL[D]	3	D		Amplitude range#
Reference	Position	REFP[D]	3	D		0<D<10
	Value	REFV[D]	3	D		Amplitude range#
	Set to mkr	MARKREF	1			
Delay	Set delay	ELED[D]	3	D		+1.0 s
	Set to mkr	MARKDELA	1			
Phase	Offset	PHAO[D]	3	D		+360 deg
SEQ-sequencing						
Sequencing menu	Continue sequence	CONS	1			
	Do sequence	DOSEQ<I>	2			1 to 6
	Gosub sequence	GOSUB	1			
	New/modify sequence	NEWSEQ<I>	2			1 to 6
	Pause to select seq.	PTOS	1			
	Done modify	DONM	1			
	Select sequence	SEQ<I>	2	I		1 to 6
TTL I/O	TTL out high continuously	TTLOH	1			
	TTL out low continuously	TTLOL	1			
	TTL low - end sweep high	TTLHPULS	1			
	TTL high - end sweep low	TTLLPULS	1			
	Programs all GPIO output bits	PARAOUT[D]	3			0 to 255
	Set specified bit on GPIO	SETBIT[D]	2			0 to 7
	Clear specified bit on GPIO	CLEABIT[D]	2			0 to 7
	Specify input GPIO bit for IFBI	PARAIN[D]	2			0 to 4
	Input GPIO bit high - do SEQ<I>	IFBIHIGH	1			
	Input GPIO bit low - do SEQ<I>	IFBILOW	1			

For log mag: ± 500 dB. For phase: ± 500 degrees. For Smith chart and Polar: ± 500 units. For linear magnitude: ± 500 units. For SWR: ± 500 units. The scale is always positive, and has minimum values of .001 dB, 10e-12 degrees, 10e-15 seconds, and 10 picounits.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
Save/recall sequences	Store to disk§	STORSEQ<I>	2			1 to 6
	Recall from disk§	LOADSEQ<I>	2			1 to 6
Special functions	Peripheral address	ADDRPERI	1			
	Title to peripheral meter/HP-IB	TITTPERI	1			
	Wait D seconds	SEQWAIT[D]	3	D		0.1 to 3000 s
	Pause	PAUS	1			
	Marker to CW freq.	MARKCW	1			
	Emit beep	EMIB	1			
	Title to printer	TITTPRIN	1			
	Show menus	SHOM	1			
	Assert seq. status bit	ASSS	1			
	Read pwr mtr/HP-IB into title string	PMTRTTIT	1			
	Send number into trace memory	TITTMEM	1			
	Duplicate seq. X to seq. Y	DUPLSEQ<X>SEQ<Y>	2			X, Y = 1 to 6
	Print sequence I	PRINSEQ<I>	2			1 to 6
	Title sequence I	TITSEQ<I>	2			1 to 6
	Clear sequence I	CLEASEQ<I>	2			1 to 6
Decision making	If limit test pass then do sequence	IFLTPASSSEQ<I>	2			1 to 6
	If limit test fail then do sequence	IFLTFAILSEQ<I>	2			1 to 6
Loop counter	Set value	LOOC[D]	3			0 to 32,760
	Increment by 1	INCRLOOC				
	Decrement by 1	DECRLOOC				
	If counter equals 0 then do sequence	IFLCEQZESEQ<I>	2			1 to 6
	If counter not eq. 0 then do sequence	IFLCNEZESEQ<I>	2			1 to 6
STIMULUS						
Stimulus	Center	CENT[D]	3	D		Stimulus range†
	Span	SPAN[D]	3	D		Stimulus range†
	Start	STAR[D]	3	D		Stimulus range†
	Stop	STOP[D]	3	D		Stimulus range†

† For frequency sweeps: 30 kHz to 3 GHz. (To 6 GHz for option 006). For power sweeps: −15 to 20 dBm in range 0, 25 dB maximum in other ranges (−100 to +100 with power meter cal on). For CW time: 0 to 24 hours. For frequency sweep, transform on: ± 1/frequency step. For CW time sweep, transform on: ±1/time step.

§ Requires pass control when using the HP-IB port.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
		SYSTEM				
Set clock	Time stamp	TIMESTAM<ON|OFF>	2	1,0		
	Set date	SETDATE[$]	3			DD MMM YYY
	Set time	SETTIME[$]	3			HH:MM:SS
Harmonic mode	Off	HARMOFF	1	0,1	OPC	
	Second	HARMSEC	1	0,1	OPC	
	Third	HARMTHIR	1	0,1	OPC	
Instrument mode	Network analyzer	INSMNETA	1	0,1	OPC	
	Ext. source auto	INSMEXSA	1	0,1	OPC	
	Ext. source manual	INSMEXSM	1	0,1	OPC	
	Tuned receiver	INSMTUNR	1	0,1	OPC	
Service	Analog bus	ANAB<ON|OFF>	2			
Frequency offset	On/off	FREQOFFS<ON|OFF>	2	1,0	OPC	
	Value	VOFF[D]	3			0 to 3 GHz (0 to 6 GH option 006)
	Set RF > LO	RFGTLO	1			
	Set RF < LO	RFLTLO	1			
	Select up converter	UCONV	1			
	Select down converter	DCONV	1			
	View measurement/mixer setup	VIEM<ON|OFF>	2	1,0		

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
SYSTEM-limit testing						
Limit line	On/off	LIMILINE<ON|OFF> |2	1,0			
Limit test	On/off	LIMITEST<ON|OFF>	2	1,0		
	Beeper	BEEPFAIL<ON|OFF>	2	1,0		
Limit offset	Stimulus	LIMISTIO[D]	3	D		Stimulus range†
	Amplitude	LIMIAMPO[D]	3	D		Amplitude range#
	Marker to offset	LIMIMAOF	1			
Edit table	Begin edit	EDITLIML	1			
	Add segment	SADD	1			
	Edit segment D	SEDI[D]	3	D		1 to 18
	Segment done	SDON	1			
	Delete segment	SDEL	1			
	Done with edit	EDITDONE	1			
	Clear list	CLEAL	1			
Edit segment	Stimulus value	LIMS[D]	3	D		Stimulus range†
	Marker to stimulus	MARKSTIM	1			
	Upper limit	LIMU[D]	3	D		Amplitude range#
	Lower limit	LIML[D]	3	D		Amplitude range#
	Delta limits	LIMD[D]	3	D		Amplitude range#
	Middle value	LIMM[D]	3	D		Amplitude range#
	Marker to middle	MARKMIDD	1			
	Flat line type	LIMTFL	1	0,1		
	Sloping line type	LIMTSL	1	0,1		
	Single point type	LIMTSP	1	0,1		

† For frequency sweeps: 30 kHz to 3 GHz. (To 6 GHz for option 006). For power sweeps: −15 to 20 dBm in range 0, 25 dB maximum in other ranges (−100 to +100 with power meter cal on). For CW time: 0 to 24 hours. For frequency sweep, transform on: ± 1/frequency step. For CW time sweep, transform on: ±1/time step.

For log mag: ± 500 dB. For phase: ± 500 degrees. For Smith chart and Polar: ± 500 units. For linear magnitude: ± 500 units. For SWR: ± 500 units. The scale is always positive, and has minimum values of .001 dB, 10e-12 degrees, 10e-15 seconds, and 10 picounits.

Table 1-2. Key Select Codes (continued)

Function	Action	Mnemonic	S	?	O	Range
SYSTEM-transform						
Transform	On/off	TIMDTRAN<ON|OFF>	2			
Set freq	Low pass	SETF	1			
Mode	Low pass impulse	LOWPIMPU	1	0,1		
	Low pass step	LOWPSTEP	1	0,1		
	Bandpass	BANDPASS	1	0,1		
	Specify gate menu	SPEG	1			
Window	Maximum	WINDMAXI	1			
	Normal	WINDNORM	1			
	Minimum	WINDMINI	1			
	Any value	WINDOW[D]	3	D	State dependent	
Window shape	Use trace memory	WINDUSEM<ON|OFF>	2	1,0		
Demodulation	Off	DEMOOFF	1	0,1		
	Amplitude	DEMOAMPL	1	0,1		
	Phase	DEMOPHAS	1	0,1		
Gate	On/off	GATEO<ON|OFF>	2	1,0		
	Start	GATESTAR[D]	3	D		Stimulus range†
	Stop	GATESTOP[D]	3	D		Stimulus range†
	Center	GATECENT[D]	3	D		Stimulus range†
	Span	GATESPAN[D]	3	D		Stimulus range†
Gate shape	Maximum	GATSMAXI	1	0,1		
	Wide	GATSWIDE	1	0,1		
	Normal	GATSNORM	1	0,1		
	Minimum	GATSMINI	1	0,1		

† For frequency sweeps: 30 kHz to 3 GHz. (To 6 GHz for option 006). For power sweeps: −15 to 20 dBm in range 0, 25 dB maximum in other ranges (−100 to +100 with power meter cal on). For CW time: 0 to 24 hours. For frequency sweep, transform on: ± 1/frequency step. For CW time sweep, transform on: ±1/time step.

HP-IB Only Commands

Table 1-3. HP-IB Only Commands

Action	Mnemonic	Syntax	Description
MISCELLANEOUS			
Identity	IDN?	1	Outputs the identification string: "HEWLETT PACKARD, 8753D,0,X.XX", where X.XX is the firmware revision of the instrument.
Key	KEY[D]	1	Imitates pressing a key. The data transmitted is the key code, as defined in Figure 1-2. Range for D = 1 to 8.
Key code	KOR?	1	Outputs last key code or knob count. If the reply is positive, it is a key code. If it is negative, then set bit 15 equal to bit 14, and the resulting two byte integer is the RPG knob count. It can be either positive or negative. There are about 120 counts per turn.
Move marker	MARKBUCK[D]	2	Moves the marker to the selected point on the trace. On a 201 point sweep, D can range from 0 to 200.
On completion	OPC	1	Reports completion of the last OPC-compatible command received since `OPC;` or `OPC?;` was received.
Plot/print softkeys	PSOFT<ON|OFF>	2	Includes the menu keys when printing or plotting the screen.
Copy default	DEFLCPIO	1	Sets up a default state for copy.
Revision	SOFR	1	Displays the software revision on the analyzer.
Learn string	SELL[D]	2	Selects the learn string revision to input to and output from the analyzer. The valid parameters are: 0: Defaults to current revision. 201: Revision 8753B 2.01 300: Revision 8753B 3.00 401: Revision 8753C 4.01 402: Revision 8753C 4.02 412: Revision 8753C 4.12 413: Revision 8753C 4.13
Sampler	SAMC<ON|OFF>	2	Turns sampler correction off. To be used only when data is being taken to create custom calibration coefficients.
Test set	TESS?	1	Returns a one on the standard analyzer. (The standard analyzer does not recognize external test sets.) For option 011, returns a one if an HP 85046A/B S-parameter test set is present. Returns a two if an HP 85047A S-parameter test set is present.

Table 1-3. HP-IB Only Commands (continued)

Action	Mnemonic	Syntax	Description
MISCELLANEOUS (continued)			
External trigger	EXTTHIGH	1	Sets the trigger polarity high.
	EXTTLOW	1	Sets the trigger polarity low.
INPUT			
Data	INPUDATA[D]	3	Accepts error-corrected data.
Formatted	INPUFORM[D]	3	Accepts formatted data.
Uncorrected	INPURAW1[D]	3	Accepts raw data.
	INPURAW2[D]	3	
	INPURAW3[D]	3	
	INPURAW4[D]	3	
Error coefficient	INPUCALC<01, 02, ... 12>	2	Accepts the individual error coefficient arrays. Issue the command that begins the calibration the coefficients are from (e.g. `CALIS11;`), then input the data. Finally, issue `SAVC;` and trigger a sweep.
Power meter cal.	INPUPMCAL<1,2>	3	Accepts power meter cal array. Values should be entered as 100 times the desired source power in dB.
Cal kit	INPUCALK[D]	3	Accepts a cal kit.
Learn string	INPULEAS[D]	3	Accepts the learn string. Preceded by SELL if learn string is not current revision.
MENUS			
Averaging	MENUAVG	1	
Calibration	MENUCAL	1	
Copy	MENUCOPY	1	
Display	MENUDISP	1	
Format	MENUFORM	1	
Marker	MENUMARK	1	
Meas	MENUMEAS	1	
Marker function	MENUMRKF	1	
Off	MENUO<ON|OFF>	2	
Recall	MENURECA	1	
Scale	MENUSCAL	1	
Stimulus	MENUSTIM	1	
System	MENUSYST	1	

Table 1-3. HP-IB Only Commands (continued)

Action	Mnemonic	Syntax	Description
OUTPUT			
NOTE: Except as noted, these commands output data according to the current output format. The data is transmitted in pairs of numbers, the number of pairs being the same as the number of points in the sweep.			
Active function	OUTPACTI	1	Outputs value of function in active entry area in ASCII format.
Error coefficient	OUTPCALC<01,02 . . . 12>	2	Outputs the selected error coefficient array from the active channel. Each array is the same as a data array. See Table 1-5, for the contents of the arrays.
Interp. cal.	OUTPICAL<01,02 . . . 12>	2	Outputs the selected interpolated cal coefficient array.
Cal kit	OUTPCALK	1	Outputs the active cal kit, a less than 1000 byte string in form 1.
Data	OUTPDATA	1	Outputs the error corrected data from the active channel in real/imaginary pairs. See Figure 1-1.
Error	OUTPERRO	1	Outputs the oldest error in the error queue. The error number is transmitted, then the error message, in ASCII format.
Formatted	OUTPFORM	1	Outputs the formatted trace data from the active channel in current display units. See Table 1-6 for data transmitted.
Power meter cal.	OUTPIPMCAL<1,2>	2	Outputs the interpolated power meter cal array for channel 1 or channel 2.
Power meter cal.	OUTPPMCAL<1,2>	2	Outputs power meter cal array for channel 1 or channel 2. Values are sent as 100 times the source power in dB.
Identity	OUTPIDEN	1	Outputs identification string, same as IDN?.
Key code	OUTPKEY	1	Outputs the code of the last key pressed, in ASCII format. See Figure 1-2 for key codes. A −1 is transmitted for a knob turn.
Learn string	OUTPLEAS	1	Outputs the learn string, a less than 3,000 byte string in form 1.
External source	OUTPRFFR	1	Outputs external source RF frequency when in external source instrument mode.

Table 1-3. HP-IB Only Commands (continued)

Action	Mnemonic	Syntax	Description
OUTPUT (Continued)			
Sequencing	OUTPSEQ<I>	2	Outputs sequence I (I = 1 to 6) listing over HP-IB.
Limit failures	OUTPLIMF	1	Outputs the limit results as described under OUTPLIML for only those stimulus points that failed.
Limit list	OUTPLIML	1	Outputs the limit test results for each stimulus point. The results consist of four numbers. The first is the stimulus value tested, the second is the test result:—1 for no test, 0 for fail, 1 for pass. The third number is the upper limit value, and the fourth is the lower limit value. This is a form 4 transfer.
Limit marker	OUTPLIMM	1	Outputs the limit test results as described for OUTPLIML at the marker.
Marker	OUTPMARK	1	Outputs the active marker values in 3 numbers. The first two numbers are the marker values, and the last is the stimulus value. See Table 1-6 for the marker values.
Memory	OUTPMEMO	1	Outputs the memory trace from the active channel. It is error corrected data in real/imaginary pairs, and can be treated the same as data from OUTPDATA.
Marker statistics	OUTPMSTA	1	Outputs marker statistics: mean, standard deviation, and peak to peak deviation. ASCII format.
Bandwidth	OUTPMWID	1	Outputs results of bandwidth search: bandwidth, center, and Q. ASCII format.
Clock	READDATE	1	Outputs the date of the clock in the following format: DD MMM YYYY
Clock	READTIME	1	Outputs the time of the clock in the following format: HH:MM:SS
Plot	OUTPPLOT	1	Outputs the plot string in ASCII format to the HP-IB port. Can be directed to an HP-GL plotter.

Table 1-3. HP-IB Only Commands (continued)

Action	Mnemonic	Syntax	Description
OUTPUT (Continued)			
Print	OUTPPRNALL	1	Prints all list values or operating and marker parameters in text mode to HP-IB. Requires pass control mode.
Raw data	OUTPRAW1	1	Outputs uncorrected data arrays for the active channel. Raw 1 holds the data unless a 2-port calibration is on, in which case the arrays hold S11, S21, S12, and S22, respectively. The data is in real/imaginary pairs.
	OUTPRAW2		
	OUTPRAW3		
	OUTPRAW4		
Status byte	OUTPSTAT	1	Outputs the status byte. ASCII format.
Display title	OUTPTITL	1	Outputs the display title. ASCII format.
OUTPUT FORMATS			
	FORM1	1	HP 8753 internal format, with header.
	FORM2	1	32 bit floating point, with header.
	FORM3	1	64 bit floating point, with header.
	FORM4	1	ASCII format. No header.
	FORM5	1	32 bit PC format (bytes reversed).
SOFTKEYS			
Press	SOFT[I]	2	Activates softkey I, I = 1 to 8.
Label	WRSK<1 TO 8>[$]	4	Writes label (10 char) to indicated softkey.
STATUS REPORTING			
Clear	CLES	1	Clears the status byte.
Interrogate	ESB?	1	Returns event-status register B.
	ESR?	1	Returns the event-status register.
	OUTPSTAT	1	Returns the status byte.
Enable	ESE[D]	1	Enables event-status register. (0<D<255)
	ESNB[D]	1	Enables event-status register B. (0<D<255)
	SRE[D]	1	Enables SRQ. (0<D<255)

Calibration

Measurement calibration over HP-IB follows the same command sequence as a calibration from the front-panel:

1. Start by selecting a calibration kit, such as 50 ohm type N (`CALKN50;` over HP-IB.)
2. Select a calibration type, such as S11 1-port (`CALIS111;` over HP-IB.)
3. Call each class used by the calibration type, such as `FORWARD: OPEN` (`CLASS11A;` over HP-IB.) During a 2-port calibration, the reflection, transmission, and isolation subsequences must be opened before the classes in the subsequence are called, and then closed at the end of each subsequence.
4. If a class has more than one standard in it, select a standard from the menu presented (STANA to STANG over HP-IB.)
5. If, during a calibration, two standards are measured to satisfy one class, the class must be closed with `DONE;`.
6. Declare the calibration done, such as with `DONE 1-PORT CAL` (`SAV1;` over HP-IB.)

The STANA to STANG commands are all held commands because they trigger a sweep. If a class has only one standard in it, which means that it will trigger a sweep when called, the class command will be held also.

Note Since different cal kits can have a different number of standards in a given class, any automated calibration sequence is valid only for a specific cal kit.

Table 1-4. Relationship between Calibrations and Classes

Class	Response	Response and Isolation	S11 1-port	S22 1-port	One path 2-port	Full 2-port	TRL*/LRM*
Reflection:[1]					•	•	•
S11A, RE FW MTCH			•		•	•	•
S11B, LN FW MTCH			•		•	•	•
S11C, LN FW TRAN			•		•	•	•
S22A, LN RV MTCH				•		•	•
S22B, LN RV TRAN				•		•	•
S22C, LN RV TRAN				•		•	•
Transmission:[1]					•	•	•
Forward match					•	•	•
Forward trans					•	•	•
Reverse match						•	•
Reverse trans						•	•
Isolation:[1]					•	•	•
Forward					•	•	•
Reverse						•	•
Response	•						
Response and isolation:							
Response		•					
Isolation		•					

1 These subheadings must be called when doing 2-port calibrations.

Table 1-5. Calibration Arrays

Array	Response	Response and Isolation	1-port	2-port[1]	TRL*/LRM*
1	ER or ET	E_X (E_D)[2]	E_D	E_{DF}	E_{DF}
2		E_T (E_R)	E_S	E_{SF}	E_{SF}
3			E_R	E_{RF}	E_{RF}
4				E_{XF}	E_{XF}
5				E_{LF}	E_{LF}
6				E_{TF}	E_{TF}
7				E_{DR}	E_{DR}
8				E_{SR}	E_{SR}
9				E_{RR}	E_{RR}
10				E_{XR}	E_{XR}
11				E_{LR}	E_{LR}
12				E_{TR}	E_{TR}

1 One path, 2-port cal duplicates arrays 1 to 6 in arrays 7 to 12.

2 Response and isolation corrects for crosstalk and transmission tracking in transmission measurements, and for directivity and reflection tracking in reflection measurements.

Meaning of first subscript:

D = directivity
S = source match
R = reflection tracking
X = crosstalk
L = load match
T = transmission tracking

Meaning of second subscript:

F = forward
R = reverse

Display Graphics

User Graphics Units

Size of Graticule only

- length = 350 to 4915
- height = 150 to 3950

Size of Complete Display (graticule plus annotation and softkey labels)

- length = 0 to 5850
- height = 0 to 4095

HP-GL subset:

Command	Description
AF;	Erases the user graphics display.
CS;	Turns off the measurement display.
DF;	Sets the default values.
LB[text][etx];	Labels the display, placing the symbols starting at the current pen position. All incoming characters are printed until the etx symbol is received. The default etx symbol is the ASCII value 3 (not the character 3).
LTa;	Specifies line type:

a	line
0	solid
1	solid
2	short dashes
3	long dashes

OP;	Outputs P1 and P2, the scaling limits: 0,0,5850,4095.
PAx,y;	Draws from the current pen position to x,y. There can be many pairs of x,y coordinates within one command. They are separated by commas, and the entire sequence is terminated with a semicolon.
PD;	Pen down. A line is drawn only if the pen is down.

PG; Erases the user graphics display.

PRx,y; Plot relative: draws a line from the current pen position to a position y up and x over.

PU; Pen up. Stops anything from being drawn.

RS; Turns ON the measurement display.

SIh,w; Sets the character size, for height h and width w in centimeters:

h	w	size
0.16	0.20	smallest
0.25	0.30	
0.33	0.39	
0.41	0.49	largest

SPn; Selects pen n:

n	brightness
0	blank
1	brightest
2	
3	dimmest

Accepted but ignored HP-GL commands:

IM Input service request mask
IP Input P1,P2 scaling points
IW Input window
OC Output current pen position
OE Output error
OI Output identity
OS Output status
SL Character slant
SR Relative character size

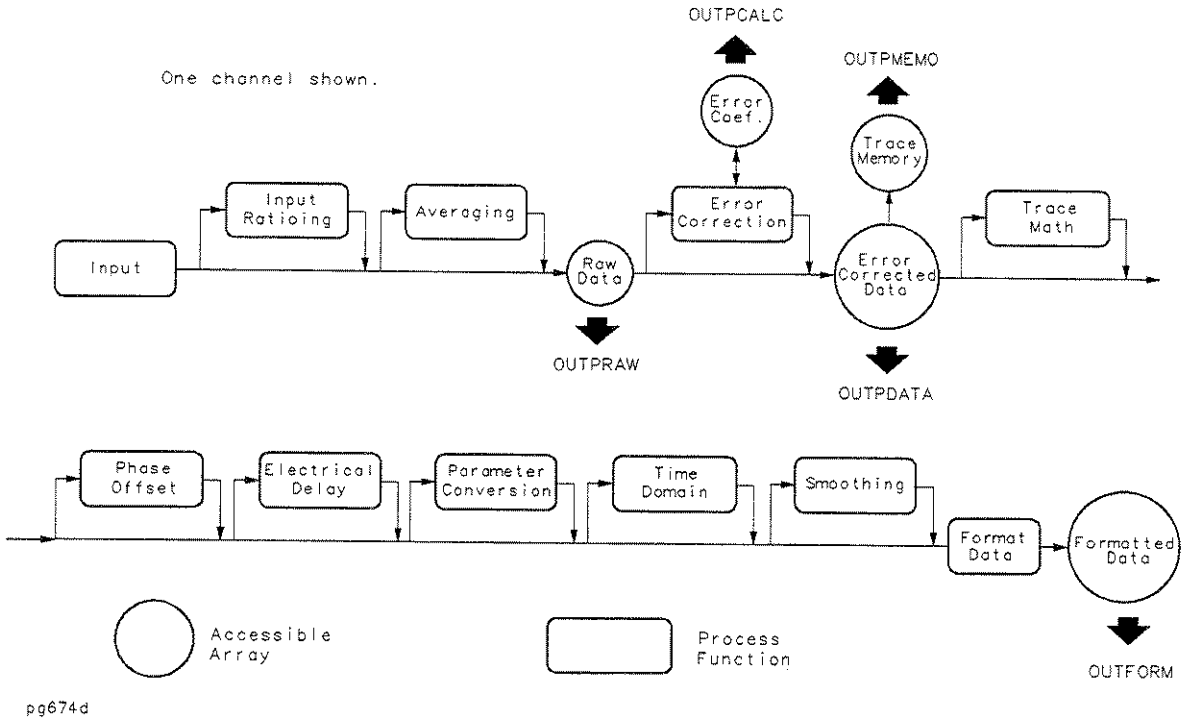


Figure 1-1. Data Processing Chain

Table 1-6.
Marker and Data Array Units as a Function of Display Format

Display Format	Marker Mode	OUTPMARK value 1, value 2	OUTFORM value 1, value 2	Marker Readout value, aux value*
LOG MAG		dB	dB†	dB†
PHASE		degrees†	degrees†	degrees†
DELAY		seconds†	seconds†	seconds†
SMITH CHART	LIN MKR	lin mag, degrees	real,imag	lin mag, degrees
	LOG MKR	dB, degrees	real, imag	dB, degrees
	Re/Im	real, imag	real, imag	real, imag
	R + jX	real, imag ohms	real, imag	real, imag ohms
	G + jB	real, imag Siemans	real, imag	real, imag Siemans
POLAR	LIN MKR	lin mag, degrees	real, imag	lin mag, degrees
	LOG MKR	dB, degrees	real, imag	dB, degrees
	Re/Im	real, imag	real, imag	real, imag
LIN MAG		lin mag†	lin mag†	lin mag†
REAL		real†	real†	real†
SWR		SWR†	SWR†	SWR†

* The marker readout values are the marker values displayed in the upper left hand corner of the display. They also correspond to the value and aux value associated with the fixed marker.

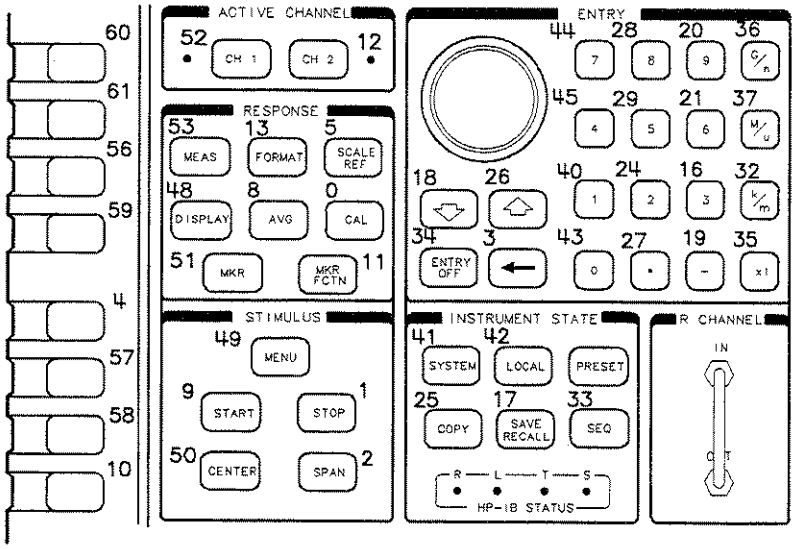
† Value not significant in this format, but is included in data transfers.

Disk File Names

Disk files created by the analyzer consist of a state name of up to eight characters, such as FILTER, appended with up to two characters. In LIF format, the file name is FILTERXX. In DOS format, the filename is FILTER.XX. The first appended character is the file type, telling the kind of information in the file. The second appended character is a data index, used to distinguish files of the same type. Data and calibration files are form 3 data (without a header) which can be read off the disk. The other files are not meant to be decoded. Table 1-7 lists the appended characters and their meanings.

Table 1-7. Disk File Names

Char 1	Meaning	Char 2	Meaning
I	Instrument state		
G	Graphics	1	Display graphics
		0	Graphics index
D	Error corrected data	1	Channel 1
		2	Channel 2
R	Raw data	1 to 4	Channel 1, raw arrays 1 to 4
		5 to 8	Channel 2, raw arrays 1 to 4
F	Formatted data	1	Channel 1
		2	Channel 2
M	Memory trace	1	Channel 1
		2	Channel 2
P	Instrument state appendix		
C	Cal kit	K	
1	Cal data, channel 1		
		O	Stimulus state
		1 to 9	Coefficients 1 to 9
		A	Coefficient 10
		B	Coefficient 11
		C	Coefficient 12
2	Cal data, channel 2	0 to C	Same as channel 1



cg61d

Figure 1-2. Key Codes

Note 1: Key code 63 is invalid key.

Note 2: `OUTPKEY;` reports a knob turn as a −1.

Note 3: If the two byte integer sent back from KOR? is negative, it is a knob count. If the knob count was negative, no modification is needed. If the knob count was positive, however, bit 14 will not be set. In this case, the number must be decoded by clearing the most significant byte, as by AND'ing the integer with 255.

Alphabetical Code Listing

Mnemonic	Description
AB	Measure and display A/B on the active channel.
ADDRCONT[D]	Controller HP-IB address: the address where control is returned after a pass control.
ADDRDISC[D]	Disk HP-IB address.
ADDRPERI[D]	Peripheral HP-IB address.
ADDRPLOT[D]	Plotter HP-IB address.
ADDRPOWM[D]	Power meter HP-IB address.
ADDRPRIN[D]	Printer HP-IB address.
ALTAB	Places the analyzer in the alternate inputs measurement mode, where inputs A and B are measured on alternate sweeps. As opposed to `CHOPAB;`.
ANAB<ON|OFF>	Enables the analog bus for service use.
ANAI	Measure and display the data at the auxiliary input (ANALOG IN).
AR	Measure and display A/R on the active channel.
ASEG	Use all segments for list frequency sweep.
ASSS	Assert the sequence status bit.
AUTO	Auto scale the active channel.
AVERFACT[D]	Set the averaging factor on the active channel.
AVERO<ON|OFF>	Turn averaging ON and off on the active channel.
AVERREST	Restart the averaging on the active channel.
BACI[D]	Sets the background intensity of the display.
BANDPASS	Select the time domain bandpass mode.

These commands control the warning beeper, causing it to sound if the indicated condition occurs:

BEEPDONE<ON|OFF>	The completion of functions such as save, done with calibration standard, and data trace saved.
BEEPFAIL<ON|OFF>	A limit test failure.
BEEPWARN<ON|OFF>	The generation of a warning message.
BR	Measure and display B/R on the active channel.

These commands set the open capacitance values of an open circuit while it is being defined as a calibration standard.

C0[D]

C1[D]

C2[D]

C3[D]

CAL1 Accepted for compatibility with the HP 8510A, where its function is to begin a calibration sequence.

These commands set the power meter calibration factor corrections for the particular sensor used. Sensor B is only valid for the HP 438A which has two input channels:

CALFCALF[D] Set the calibration factor.

CALFFREQ[D] Select the frequency for the calibration factor correction.

CALFSENA Edit the sensor A calibration factor table.

CALFSENB Edit the sensor B calibration factor table.

These commands begin a calibration sequence:

CALIFUL2 Short, load, open, thru (SLOT) 2-port.

CALIONE2 One-path 2-port.

CALIRAI Response and isolation.

CALIRESP Response.

CALIS111 S11 1-port.

CALIS221 S22 1-port.

CALITRL2 Thru, reflect, line or Line, reflect, match (TRL*/LRM*) 2-port.

These commands select a default calibration kit:

CALK35MD 3.5 mm.

CALK35MM 3.5 mm.

CALK7MM 7 mm.

CALKN50 Type-N 50 ohm.

CALKN75 Type-N 75 ohm.

CALKUSED User-defined calibration kit.

CALN Calibration: none. Turns calibration type to off.

CBRI[D] Adjusts the color brightness of the selected display feature.

CENT[D] Sets the center stimulus value. If a list frequency segment is being edited, sets the center of the list segment.

CHAN1 Make channel 1 the active channel. OPC-compatible.

CHAN2 Make channel 2 the active channel. OPC-compatible.

CHOPAB Places the analyzer in the chop measurement mode. As opposed to ALTAB;.

CLAD Class done, modify cal kit, specify class.

These commands call reflection standard classes during a calibration sequence. If only one standard is in the class, it is measured. If there is more than one, the standard being used must be selected with STAN<A|B|C|D|E|F|G>. If there is only one standard in the class, these commands are OPC-compatible.

CLASS11A S11A: S11 1-port, opens.

CLASS11B S11B: S11 1-port, shorts.

CLASS11C S11C: S11 1-port, loads.

CLASS22A S22A: S22 1-port, opens.

CLASS22B S22B: S22 1-port, shorts.

CLASS22C S22C: S22 1-port, loads.

These commands clear the indicated save/recall registers:

CLEA1 Clears save/recall register 1.

CLEA2 Clears save/recall register 2.

CLEA3 Clears save/recall register 3.

CLEA4 Clears save/recall register 4.

CLEA5 Clears save/recall register 5.

CLEARALL Clears all the save/recall registers. OPC-compatible.

CLEAREG[I] Clears save/recall registers 01 through 32. CLEAREG01 through CLEAREG05 are the same as CLEA1 through CLEA5.

CLEABIT Clears the specified bit on the GPIO.

CLEL Clear the desired list. This could be a frequency list, power loss list, or limit test list.

These commands clear the sequence from the internal registers:

CLEASEQ1 Sequence 1.

CLEASEQ2 Sequence 2.

CLEASEQ3 Sequence 3.

CLEASEQ4 Sequence 4.

CLEASEQ5 Sequence 5.

CLEASEQ6 Sequence 6.

CLES Clears the status register, the event-status registers, and the enable registers.

CLS Same as CLES.

COAX Selects coaxial offsets instead of waveguide while defining a standard during a cal kit modification.

These commands select the indicated display feature for color modification:

COLOCH1D	Channel 1 data and limit lines.
COLOCH1M	Channel 1 memory.
COLOCH2D	Channel 2 data and limit lines.
COLOCH2M	Channel 2 memory.
COLOGRAT	Graticule.
COLOTEXT	Text.
COLOWARN	Warning.

COLOR[D]	Adjusts the color saturation for the selected display feature.
CONS	Continues the paused sequence.
CONT	Continuous sweep trigger mode.

These commands convert the S-parameter data to:

CONV1DS	Inverted S-parameters.
CONVOFF	Conversion OFF.
CONVYREF	Y:reflection.
CONVYTRA	Y:transmission.
CONVZREF	Z:reflection.
CONVZTRA	Z:transmission.

COPYFRFT	Copies the file titles into the register titles.
COPYFRRT	Copy save/recall register titles to the disk register titles.
CORI<ON|OFF>	Turns interpolative error correction ON and OFF.
CORR<ON|OFF>	Turns error correction ON and OFF.
COUC<ON|OFF>	Couples and uncouples the stimulus between the channels.
COUP<ON|OFF>	Couple the power when coupled channels is turned OFF, COUCOFF.
CSWI<ON|OFF>	Selects test set continuous switching (ON) or test set hold (OFF) when there is a 2-port calibration active. Continuous switching is allowed only when the power ranges on both attenuator ports are set the same. When continuous switching is ON, the analyzer measures all four S-parameters each time before displaying the data for a full 2-port cal measurement. In test set hold mode, the analyzer measures all four S-parameters once and then measures the desired parameter continuously. This is known as a quasi 2-port cal measurement and it is less accurate than a full 2-port calibrated measurement.
CWFREQ[D]	Sets the CW frequency for power sweep and CW frequency modes. While the list frequency table segment is being edited, it sets the center frequency of the current segment.
CWTIME	Selects the CW time sweep type.

D1DIVID2<0N|OFF> This command divides the data in channel 1 by the data in channel 2 and displays the result on channel 2.

DATI Stores trace in channel memory. OPC-compatible.

DCONV Selects down converter for mixer measurements.

DEBU<ON|OFF> Turns the HP-IB debug mode ON and OFF. When ON, the HP 8753D scrolls incoming HP-IB commands across the display.

DECRLOOC Decrements the sequencing loop counter by 1. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.

DEFC Sets the default colors for all display features.

DEFLCPIO Sets up the following default state for copy. HP-IB only command. There is no equivalent front-panel key.

Plotter Type:	PLOTTER	Printer Type:	DESKJET
Plotter Port:	SERIAL	Printer Port:	PARALLEL
Baud Rate:	9600	Baud Rate:	19200
Handshake:	Xon-Xoff	Handshake:	Xon-Xoff
HP-IB Address:	5	HP-IB Address:	1
Parallel Port:	COPY		

DEFLPRINT Sets the printer to the following default setup conditions:

Print	Monochrome
Auto-feed	On
Print Colors:	
Ch 1 Data	Magenta
Ch 1 Memory	Green
Ch 2 Data	Blue
Ch 2 Memory	Red
Graticule	Cyan
Warning	Black
Text	Black

DEFS[D] Begins standard definition during cal kit modification. D is the standard number.

DELA Displays the data formatted as group delay.

DELO Turns the delta marker mode OFF.

These commands make the indicated marker the delta reference:

DELR1 Marker 1.

DELR2 Marker 2.

DELR3 Marker 3.

DELR4 Marker 4.

DELRFIXM Fixed marker.

DEMOAMPL Sets the transform demodulation to amplitude demodulation. Only has an effect with a CW time transform.

DEMOOFF Turns the transform demodulation function OFF.

DEMOPHAS Sets the transform demodulation to phase demodulation.

DFLT Sets the plotter to the following default setup conditions.

Plot Data	On	Pen Number	
Plot Mem	On	Data	2
Plot Grat	On	Memory	5
Plot Text	On	Graticule	1
Plot Mkr	On	Text	7
Auto-feed	On	Marker	7
Scale Plot	Full	Line Type	
Plot Speed	Fast	Data	7
		Memory	7

DIRS[D] Sets the number of files in the directory at disk initialization.

DISCUNIT[D] Specifies which disk in a multiple-disk disk drive is to be used for disk registers.

DISCVOLU[D] Specifies which volume of a multiple-volume disk drive (e.g. a Winchester) is to be used for disk registers.

DISM<ON|OFF> Displays the response and stimulus values for all markers that are turned on.

These commands display the indicated combinations of data and trace memory on the active channel:

DISPDATA Data only.

DISPDATM Data and memory.

DISPDDM Data divided by memory (linear division, log subtraction).

DISPDMM Data minus memory (linear subtraction).

DISPMEMO Memory only.

DIVI Same as DISPDDM.

DONE	Done with a class of standards, during a calibration. Only needed when multiple standards are measured to complete the class. OPC-compatible.
DONM	Done modifying a test sequence.
DOSEQ<I>	Begin execution of the selected sequence.
DOWN	Decrements the value in the active entry area (down key).
DUAC<ON|OFF>	Dual channel display ON or OFF.
DUPLSEQ[X]SEQ[Y]	Duplicates sequence X to sequence Y.
EDITDONE	Done editing list frequency or limit table. OPC-compatible.
EDITLIML	Begin editing limit table.
EDITLIST	Begin editing list frequency table.
ELED[D]	Sets the electrical delay offset.
EMIB	Send out a beep during a sequence. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.
ENTO	Turns the active entry area OFF.
ESB?	Outputs event-status register B.
ESE[D]	Enables the selected event-status register bits to be summarized by bit 5 in the status byte. An event-status register bit is enabled when the corresponding bit in the operand D is set.
ESNB[D]	Enables the selected event-status register B bits to be summarized by bit 2 of the status byte. Much like ESE;.
ESR?	Outputs the value of the event-status register.
EXTD	Selects the external disk as the active storage device. Used with STORSEQ<I>, TITF<I>, and PURG<I>.

These commands include the indicated information when a register is stored on disk. See Figure 1-1 for data types:

EXTMDATA<ON|OFF>	Error corrected data.
EXTMDATO<ON|OFF>	Data array only.
EXTMFORM<ON|OFF>	Formatted trace data.
EXTMGRAP<ON|OFF>	User graphics.
EXTMRAW<ON|OFF>	Raw data arrays.
EXTTHIGH	Sets the external trigger line high.
EXTTLOW	Sets the external trigger line low.
EXTTOFF	Deactivates the external trigger mode. OPC-compatible.
EXTTON	Activates the external trigger mode. OPC-compatible.
EXTTPOIN	Sets the external trigger to auto trigger on point. OPC-compatible.

FIXE Specifies a fixed load, as opposed to a sliding load, when defining a standard during a cal kit modification.

FOCU[D] Adjusts display focus, 0 to 100 percent.

These commands set the data format for array transfers in and out of the instrument:

FORM1 HP 8753D internal format. Preceded by 4 byte header.

FORM2 32 bit floating point format. Preceded by 4 byte header.

FORM3 64 bit floating point format. Preceded by 4 byte header.

FORM4 ASCII format. No header.

FORM5 32 bit floating point PC format. Bytes reversed.

These commands define the format to use on disk initializations:

FORMATDOS Selects DOS as the disk format.

FORMATLIF Selects LIF as the disk format.

FREQOFFS<ON|OFF> Activates the frequency offset instrument mode. OPC-compatible.

FREO Frequency blank. Turns OFF frequency notation.

FRER HP-IB free run. Acts the same as `CONT;` .

FULP Selects full page plotting, as opposed to plotting in one of the four quadrants.

These commands select a forward calibration class, during a 2-port calibration sequence. They are OPC-compatible if there is only one standard in the class:

FWDI Isolation.

FWDM Match.

FWDT Transmission.

These commands control the time domain gate (available only with option 010, time domain):

GATECENT[D] Center time.

GATEO<ON|OFF> Gate ON/OFF.

GATESPAN[D] Span time.

GATESTAR[D] Start time.

GATESTOP[D] Stop time.

These commands set the gate shape:

GATSMAXI Maximum.

GATSMINI Minimum.

GATSNORM Normal.

GATSWIDE Wide.

GOSUB Invokes a sequence as a subroutine. Used with SEQ<I>.

These commands activate the harmonic measurement mode, option 002. They are all OPC-compatible:

HARMOFF Turns OFF harmonic mode.

HARMSEC Measures the second harmonic.

HARMTHIR Measures the third harmonic.

HOLD Puts the sweep trigger into hold.

IDN? Outputs the identification string: "HEWLETT PACKARD,8753D,0,X.XX", where X.XX is the firmware revision of the instrument.

IFBIHIGH Tests the specified input GPIO bit (PARAIN). If high, invokes the sequence specified by SEQ<I>.

IFBILOW Tests the specified input GPIO bit (PARAIN). If low, invokes the sequence specified by SEQ<I>.

IFBW[D] Sets the IF bandwidth.

These commands branch an executing sequence to a new sequence if the following condition is satisfied. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified:

IFLCEQZESEQ<I> If loop counter equals zero, then do sequence <I>.

IFLCNEZESEQ<I> If loop counter does not equal zero, then do sequence <I>.

IFLTFAILSEQ<I> Limit test fails.

IFLTPASSSEQ<I> Limit test passes.

IMAG Selects the imaginary display format.

INCRLOOC Increments the sequencing loop counter by 1. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.

INID Initializes the internal disk. All information on the disk will be destroyed.

INIE Initializes the external disk. All information on the disk will be destroyed. Requires pass control when using the HP-IB port.

These commands input individual calibration coefficient arrays. Before sending the array, issue a `CALIXXXX;` command, where XXX specifies the calibration type of the data. Then input the cal arrays. Lastly store the data with `SAVC;`. The instrument goes into hold, displaying uncorrected data: `SING;` completes the process by displaying error corrected data. See Table 1-5, for the contents of the different arrays.

INPUCALC01[D] Array 1.

INPUCALC02[D] Array 2.

INPUCALC03[D] Array 3.

INPUCALC04[D] Array 4.

INPUCALC05[D] Array 5.

INPUCALC06[D] Array 6.

INPUCALC07[D] Array 7.

INPUCALC08[D] Array 8.

INPUCALC09[D] Array 9.

INPUCALC10[D] Array 10.

INPUCALC11[D] Array 11.

INPUCALC12[D] Array 12.

INPUCALK[D] Inputs a cal kit read out with `OUTCALK;` . After the transfer, the data should be saved into the user cal kit area with `SAVEUSEK;`.

INPUDATA[D] Inputs an error corrected data array, using current format. The instrument stops sweeping, and then formats and displays the data.

INPUFORM[D] Inputs a formatted data array, using current format. The instrument stops sweeping and displays the data.

INPULEAS[D] Inputs a learn string read out by `OUTPLEAS;`.

These commands input power meter calibration arrays into the instrument. Values should be entered as 100 × desired source power in dB.

INPUPMCAL1 Channel 1.

INPUPMCAL2 Channel 2.

These commands input a raw data array using the current format. See OUTPRAW for the meaning of the arrays. The instrument stops sweeping, error corrects the data, then formats and displays the data.

INPURAW1[D] Array 1.

INPURAW2[D] Array 2.

INPURAW3[D] Array 3.

INPURAW4[D] Array 4.

These commands select the instrument mode. They are all OPC-compatible.:

INSMEXSA External source, auto.

INSMEXSM External source, manual.

INSMNETA Standard network analyzer.

INSMTUNR Tuned receiver.

INTD Selects the internal disk as the active storage device. Used with STORSEQ<I>, TITF<I>, PURG<I>, STOR<I>, LOAD<I>, and LOADSEQ<I>.

INTE[D] Sets the display intensity, 0 to 100 percent.

INTM Selects the internal memory as the active storage device.

ISOD Done with isolation subsequence in a 2-port calibration. OPC-compatible.

ISOL Begins the isolation subsequence step in a 2-port calibration.

ISOOP Selects isolation for one path, two port calibration.

KEY[D] Sends a keycode, equivalent to actually pressing the key. It does not matter if the front-panel is in remote mode. See Figure 1-2 for the key codes.

KITD Calibration kit done: the last step in modifying a cal kit.

KOR? Outputs a two byte key code/knob count. If the number is positive, it is a key code. Otherwise, it has to be converted to a knob count by clearing the upper 8 bits if bit 14 is not set. The resulting integer is the knob count, either positive or negative, depending on the direction of turn. There are approximately 120 counts per knob turn .

These commands enter labels for the standard classes during a cal kit modification:

LABEFWDM[$] Forward match.

LABEFWDT[$] Forward transmission.

LABERESI[$] Response, response and isolation.

LABERESP[$] Response.

LABEREVM[$] Reverse match.

LABEREVT[$] Reverse transmission.

LABES11A[$] S11A (opens).

LABES11B[$] S11B (shorts).

LABES11C[$] S11C (loads).

LABES22A[$] S22A (opens).

LABES22B[$] S22B (shorts).

LABES22C[$] S22C (loads).

LABETLFM[$] TRL, Line, Forward, Match

LABETLFT[$] TRL, Line, Forward, Trans

LABETLRM[$] TRL, Line, Reverse, Match

LABETLRT[$] TRL, Line, Reverse, Trans

LABETRFM[$] TRL, Reflect, Forward, Match

LABETRRM[$] TRL, Reflect, Reverse, Match

LABETTFM[$] TRL, Thru, Forward, Match

LABETTFT[$] TRL, Thru, Forward, Trans

LABETTRM[$] TRL, Thru, Reverse, Match

LABETTRT[$] TRL, Thru, Reverse, Trans

LABK[$] Enters a cal kit label during a cal kit modification.

LABS[$] Enters a standard's label during standard definition.

LEFL Selects a plot in the left lower quadrant.

LEFU Selects a plot in the left upper quadrant.

LIMIAMPO[D] Enters the limit line amplitude offset.

LIMILINE<ON|OFF> Turns the display of the limit lines ON and OFF.

LIMIMAOF[D] Marker to limit offset. Centers the limit lines about the current marker position using the limit amplitude offset function.

LIMISTIO[D] Enters the stimulus offset of the limit lines.

LIMITEST<ON|OFF> Turns limit testing ON and OFF.

These commands edit a limit test segment. The limit table editing is begun with `EDITLIML;`, and a segment is brought up for editing with either `SADD;` or `SEDI N;`. The segment is closed with `SDON;`, the table is closed with `EDITDONE;`

LIMD[D] Sets the limit delta value while editing a limit line segment.

LIML[D] Sets the lower limit value.

LIMM[D] Sets the middle limit value.

LIMS[D] Sets the limit stimulus break point.

LIMTFL Make the segment a flat line.

LIMTSL Make the segment a sloping line.

LIMTSP Make the segment a single point.

LIMU[D] Set the upper limit value.

LINFREQ Selects a linear frequency sweep.

LINM Selects the linear magnitude display format.

LINTDATA[D] Enters the line type for plotting data.

LINTMEMO[D] Enters the line type for plotting memory.

LISFREQ Selects the list frequency sweep mode.

LISV Activates the list values function. The next page of values can be called with `NEXP;`. The current page can be plotted or printed, in raster graphics mode, with `PLOT;`, or `PRINALL;`. The entire list can be printed, in ASCII text mode, with `PRINTALL;`.

These commands load the file from disk with the name indicated by the previous TITFn command. The actual file recalled depends on the file title in the file position specified. Requires pass control mode when using the HP-IB port.

LOAD1 Loads the file from disk using the file name provided by the preceding `TITF1;` command.

LOAD2 Loads the file from disk using the file name provided by the preceding `TITF2;` command.

LOAD3 Loads the file from disk using the file name provided by the preceding `TITF3;` command.

LOAD4 Loads the file from disk using the file name provided by the preceding `TITF4;` command.

LOAD5 Loads the file from disk using the file name provided by the preceding `TITF5;` command.

These commands load the indicated sequence from disk. Requires pass control mode when using the HP-IB port.

LOADSEQ1	Loads sequence 1 from disk.
LOADSEQ2	Loads sequence 2 from disk.
LOADSEQ3	Loads sequence 3 from disk.
LOADSEQ4	Loads sequence 4 from disk.
LOADSEQ5	Loads sequence 5 from disk.
LOADSEQ6	Loads sequence 6 from disk.
LOGFREQ	Selects a log frequency sweep.
LOGM	Selects the log magnitude display format.
LOOC[D]	Sets the value of the sequencing loop counter. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.
LOWPIMPU	Turns ON the low pass impulse transform (option 010).
LOWPSTEP	Turns ON the low pass step transform (option 010).
LRN?	Same as OUTPLEAS.
LRN[D]	Same as INPULEAS .
MANTRIG	Sets the external trigger to manual trigger on point. OPC-compatible.

These commands make the indicated marker active and set its stimulus:

MARK1[D]	Marker 1.
MARK2[D]	Marker 2.
MARK3[D]	Marker 3.
MARK4[D]	Marker 4.
MARKBUCK[D]	Places the marker on a specific sweep point (bucket). D is the bucket number, ranging from 0 to number of points less 1.
MARKCENT	Enters the marker stimulus as the center stimulus.
MARKCONT	Places the markers continuously on the trace, not on discrete sample points.
MARKCOUP	Couples the markers between the channels, as opposed to MARKUNCO.
MARKCW	Sets the CW frequency to the marker frequency.
MARKDELA	Sets electrical length so group delay is zero at the marker stimulus.
MARKDISC	Places the markers in discrete placement mode.
MARKFAUV[D]	Sets the auxiliary value of the fixed marker position. Works in coordination with MARKFVAL and MARKFSTI.

MARKFSTI[D]	Sets the stimulus position of the fixed marker.
MARKFVAL[D]	Sets the value of the fixed marker position. See Table 1-6 for the meaning of value and auxiliary value as a function of display format.
MARKMAXI	Same as SEAMAX.
MARKMIDD	During a limit segment edit, makes the marker amplitude the limit segment middle value.
MARKMINI	Same as SEAMIN.
MARKOFF	Turns all markers and marker functions OFF.
MARKREF	Enters the marker amplitude as the reference value.
MARKSPAN	Enters the span between the active marker and the delta reference as the sweep span.
MARKSTAR	Enters the marker stimulus as the start stimulus.
MARKSTIM	During a limit segment edit, enters the marker stimulus as the limit stimulus break point .
MARKSTOP	Enters the marker stimulus as the stop stimulus.
MARKUNCO	Uncouples the markers between channels, as opposed to MARKCOUP.
MARKZERO	Places the fixed marker at the active marker position and makes it the delta reference.
MAXF[D]	Sets the maximum valid frequency of a standard being defined during a cal kit modification.
MEASA	Measures and displays input A on the active channel.
MEASB	Measures and displays input B on the active channel.
MEASR	Measures and displays input R on the active channel.
MEASTAT<ON|OFF>	Turns trace statistics ON and OFF.
MENU<ON|OFF>	Blanks the softkey menu.

These commands bring up the menu associated with the indicated front-panel key:

MENUAVG	AVG
MENUCAL	CAL
MENUCOPY	COPY
MENUDISP	DISPLAY
MENUFORM	FORMAT
MENUMARK	MARKER
MENUMEAS	MEAS
MENUMRKF	MARKER FCTN
MENURECA	RECALL
MENUSCAL	SCALE
MENUSTIM	STIMULUS MENU

MENUSYST SYSTEM

MINF[D] Sets the minimum valid frequency of a standard being defined during a cal kit modification.

MINU Displays data minus memory, the same as DISPDMM .

MODI1 Begins the modify cal kit sequence.

NEWSEQ<I> Begin modifying a sequence.

NEXP Displays the next page of the operating parameters list.

NOOP No operation. OPC-compatible.

NUMG[D] Activates D number of groups of sweeps. A group is whatever is needed to update the current parameter once. This function restarts averaging if ON. OPC-compatible.

NUMR[D] Sets the number of power meter readings per point used during a power meter calibration.

These commands specify the offset value for the indicated parameter for a standard being defined during a cal kit modification:

OFSD[D] Delay offset.

OFSL[D] Loss offset.

OFSZ[D] Impedance offset.

OMII Omits the isolation step of a calibration sequence.

OPC Operation complete. Reports the completion of the next command received by setting bit 0 in the event-status register, or by replying to an interrogation if `OPC?;` is issued. See "Command Interrogate" earlier in this chapter.

OPEP Presents a list of key operating parameters. `NEXP;` scrolls to the next page of parameters. Requesting a plot or print copies the current page. The current page can be plotted or printed, in raster graphics mode, with `PLOT;`, or `PRINALL;`. The entire list can be printed, in ASCII text mode, with `PRINTALL;`.

OUTPACTI Outputs the value of the active function, or the last active function if the active entry area is OFF.

OUTPAPER Outputs the smoothing aperture in stimulus units, rather than as a percentage.

These commands output the error correction arrays for the active calibration on the active channel. See Table 1-5, for the contents of the arrays. Each array comes out in the current output format. They contain real/imaginary pairs, the same number of pairs as points in the sweep.

OUTPCALC01 Array 1.

OUTPCALC02 Array 2.

OUTPCALC03 Array 3.

OUTPCALC04 Array 4.

OUTPCALC05 Array 5.

OUTPCALC06 Array 6.

OUTPCALC07 Array 7.

OUTPCALC08 Array 8.

OUTPCALC09 Array 9.

OUTPCALC10 Array 10.

OUTPCALC11 Array 11.

OUTPCALC12 Array 12.

These commands output over HP-IB the interpolated calibration coefficient arrays for the active calibration on the active channel.

OUTPICALC01 Array 1.

OUTPICALC02 Array 2.

OUTPICALC03 Array 3.

OUTPICALC04 Array 4.

OUTPICALC05 Array 5.

OUTPICALC06 Array 6.

OUTPICALC07 Array 7.

OUTPICALC08 Array 8.

OUTPICALC09 Array 9.

OUTPICALC10 Array 10.

OUTPICALC11 Array 11.

OUTPICALC12 Array 12.

These commands output over HP-IB the interpolated power meter calibration arrays for channels 1 and 2.

OUTPIPMCAL1 Channel 1.

OUTPIPMCAL2 Channel 2.

OUTPCALK Outputs the currently active calibration kit, as a less than 1000 byte string. The data is in form 1.

OUTPDATA Outputs the error corrected data from the active channel in the current format. See Figure 1-1.

OUTPERRO Outputs the oldest error message in the error queue. Sends first the error number, and then the error message itself as a string no longer than 50 characters.

OUTPFORM Outputs the formatted display data array from the active channel in the current format. See Table 1-6 for the contents of the array positions as a function of display format .

OUTPIDEN Outputs the identification string for the analyzer: "HEWLETT PACKARD,8753D,0,X.XX" where X.XX is the firmware revision.

OUTPKEY Outputs the key code of the last key pressed. An invalid key is reported with a 63, a knob turn with a −1. See Figure 1-2 for the front-panel key codes.

OUTPLEAS Outputs the learn string, which contains the entire front panel state, the limit table, and the list frequency table. It is always in form 1.

These commands output the limit test results. The results consist of four fields. First is the stimulus value for the point. Second is an integer indicating test status. Third is the upper limit at that point. Fourth is the lower limit at that point. If there are no limits at that point, the third and fourth fields are zero. The test status is—1 for no test, 0 for fail, and 1 for pass.

OUTPLIMF Outputs the limit test results for each failed point.

OUTPLIML Outputs the limit test results for each point in the sweep. This is a form 4 transfer.

OUTPLIMM Outputs the limit test results at the marker.

OUTPMARK Outputs the marker values. The first two numbers are the marker response values, and the last is the stimulus value. See Table 1-6 for the meaning of the response values as a function of display format.

OUTPMEMO Outputs the memory trace from the active channel. The data is in real/imaginary pairs, and can be treated the same as data read with the OUTPDATA command.

OUTPMSTA Outputs the marker statistics: mean, standard deviation, and peak-to-peak variation in that order. If statistics is not ON, it is turned ON to generate current values and turned OFF again.

OUTPMWID Outputs the marker bandwidths search results: bandwidth, center, and Q in that order. If widths is not ON, it is turned ON to generate current values and turned OFF again .

OUTPPLOT Outputs the plot string to the HP-IB ports. Can be directed to a plotter, or read into the computer. PSOFT<ON|OFF> controls whether the softkeys are included in the plot.

These commands output the power meter calibration array. Values should be entered as 100 times the source power in dB. A default array is used if a power meter calibration sweep, TAKCS, has not been taken:

OUTPPMCAL1 Channel 1.

OUTPPMCAL2 Channel 2.

OUTPPRIN Outputs to the HP-IB port a raster dump of the display, intended for a graphics printer. PSOFT<ON|OFF> controls whether the soft keys are included in the printout.

OUTPPRNALL Prints all list values or operating and marker parameters in text mode to HP-IB.

These commands output the raw measurement data. See Figure 1-1 for the meaning of the data. Normally, array 1 holds the current parameter. If a 2-port calibration is active, the arrays hold S11, S21, S12, and S22, respectively:

OUTPRAW1 Array 1.

OUTPRAW2 Array 2.

OUTPRAW3 Array 3.

OUTPRAW4 Array 4.

OUTPRFFR Outputs the external source RF frequency. The instrument must be in external source mode, either INSMECSA or INSEXSM.

OUTPSEQ<I> Outputs a sequence listing over HP-IB.

OUTPSTAT Outputs the status byte.

OUTPTITL Outputs the display title.

PARAIN Specified the input GPIO bit to be used by IFBIHIGH and IFBILOW tests.

PARAL<GPIO|CPY> Selects use of the parallel port: for general purpose I/O or for the copy function.

PARAOUT[D] Programs all GPIO output bits (0 to 255) at once.

PAUS Inserts a pause into a sequence. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.

PCB[D] Same as ADDRCONT. Indicates where control will be passed in pass control mode.

These commands select the color for the indicated display feature where <COLOR> equals one of the following colors: white, cyan, magenta, blue, yellow, green, red, or black.

PCOLDATA1<COLOR> Channel 1 data.

PCOLDATA2<COLOR> Channel 2 data.

PCOLMEMO1<COLOR> Channel 1 memory.

PCOLMEMO2<COLOR> Channel 2 memory.

PCOLGRAT<COLOR> Graticule.

PCOLTEXT<COLOR> Display text.

PCOLWARN<COLOR> Warning text.

PDATA<ON|OFF> Selects whether trace data is plotted.

These commands select the pen for plotting the indicated display feature for the active channel:

PENNDATA[D] Data trace.

PENNGRAT[D] Graticule.

PENNMARK[D] Markers and marker text.

PENNMEMO[D] Memory trace.

PENNTEXT[D] Text and user graphics.

PGRAT<ON|OFF> Selects whether the graticule is plotted.

PHAO[D] Sets the phase offset.

PHAS Selects the phase display format.

PLOS<SLOW|FAST> Selects the pen speed for plotting.

PLOT	Initiates a plot. Requires pass control mode when using the HP-IB port.
PLTHNDSHK<XON|DTR>	Selects the plotter handshake mode as either Xon-Xoff or DTR-DSR.
PLTPRTHPIB	Sets the plotter port to HP-IB.
PLTPRTPARA	Sets the plotter port to parallel.
PLTPRTSERI	Sets the plotter port to serial.
PLTTRAUTF<ON|OFF>	Turns ON and OFF the plotter auto feed.
PLTTRBAUD[D]	Sets the plotter baud rate.
PLTTRFORF	Sends a form feed to the plotter.
PLTTYPHPGL	Selects HPGL-compatible printer as the plotter type.
PLTTYPPLTR	Selects plotter as the plotter type.
PMEM<ON|OFF>	Selects whether memory is plotted.
PMKR<ON|OFF>	Selects whether markers are plotted.
PMTRTTIT	Reads power meter/HP-IB value into title string. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.
POIN[D]	Sets the number of points in the sweep.
POLA	Selects the polar display format.

These commands select the marker readout format for polar display:

POLMLIN	Linear markers.
POLMLOG	Log markers.
POLMRI	Real/imaginary markers.

PORE<ON|OFF>	Turn port extensions ON and OFF.

These commands set the port extension length for the indicated port or input. Ports 1 and 2 refer to the test set ports:

PORT1[D]	Port 1.
PORT2[D]	Port 2.
PORTA[D]	Input A.
PORTB[D]	Input B.

PORTP<CPLD|UNCPLD>	Selects either coupled or uncoupled for the port powers for a given channel.
POWE[D]	Sets the output power level.
POWLFREQ[D]	Selects the frequency for which a power loss correction is entered. This must be followed by a POWLLOSS[D], which sets the value.
POWLLIST	Begins editing a power loss list for a power meter calibration.
POWLLOSS[D]	Sets the loss value for a particular frequency, POWLFREQ[D], in the power loss list.

POWM<ON|OFF>	Selects whether the HP 436A (ON) or the HP 437B/438A (OFF) is to be used as the power meter in service procedures.
POWS	Selects power sweep, from the sweep type menu.
POWT<ON|OFF>	Turning power trip OFF clears a power trip after an overload condition is detected at one of the input ports.
PRES	Presets the instrument. OPC-compatible.
PRAN0	Selects power range 0 when in manual power range. Used with PWRR and PMAN.
PRAN1	Selects power range 1 when in manual power range. Used with PWRR and PMAN.
PRAN2	Selects power range 2 when in manual power range. Used with PWRR and PMAN.
PRAN3	Selects power range 3 when in manual power range. Used with PWRR and PMAN.
PRAN4	Selects power range 4 when in manual power range. Used with PWRR and PMAN.
PRAN5	Selects power range 5 when in manual power range. Used with PWRR and PMAN.
PRAN6	Selects power range 6 when in manual power range. Used with PWRR and PMAN.
PRAN7	Selects power range 7 when in manual power range. Used with PWRR and PMAN.
PRES	Presets the analyzer to the factory preset state.
PRIC	Selects color print.
PRINALL	Copies the display, in raster graphics mode, to a printer. Requires pass control mode when using the HP-IB port.
PRINSEQ<I>	Begins printing the sequence selected.
PRINTALL	Prints all list values or operating and marker parameters in ASCII text mode. Requires pass control mode when using the HP-IB port.
PRIS	Selects standard (monochrome) print.
PRNHNDSHK<XON|DTR>	Selects the printer handshake mode as either Xon-Xoff or DTR-DSR.
PRNPRTHPIB	Sets the printer port to HP-IB.
PRNPRTPARA	Sets the printer port to parallel.
PRNPRTSERI	Sets the printer port to serial.
PRNTRAUTF<ON|OFF>	Turns ON and OFF the printer auto feed.
PRNTRBAUD[D]	Sets the printer baud rate.
PRNTRFORF	Sends a form feed to the printer.
PRNTYPDJ	Selects the DeskJet printer as the printer type.
PRNTYPEP	Selects the Epson ESC/P2 printer control language-compatible printer as the printer type.

PRNTYPLJ	Selects the LaserJet printer as the printer type.
PRNTYPPJ	Selects the PaintJet printer as the printer type.
PRNTYPTJ	Selects the ThinkJet printer as the printer type.
PSOFT<ON|OFF>	Controls whether softkeys are included in the hardcopy print or plot when using one of the following commands: `PLOT;`, `PRINALL;`, `OUTPPLOT;` or `OUTPPRIN;`.
PTEXT<ON|OFF>	Selects whether text is plotted.
PTOS	Pauses the sequence to be followed by selection one of the 6 sequences (SEQ<I>).

These commands purge the indicated file from disk. Requires pass control mode when using the HP-IB port.

PURG1	File 1.
PURG2	File 2.
PURG3	File 3.
PURG4	File 4.
PURG5	File 5.

These commands select the type of power meter calibration desired. A calibration sweep should be taken, TAKCS, after selecting a "one sweep" Power meter calibration, to ensure a valid calibration. No calibration sweep is needed for "each sweep" power meter calibrations. They are all OPC-compatible:

PWMCEACS[D]	Each sweep.
PWMCOFF[D]	Off.
PWMCONES[D]	One sweep.
PWRLOSS<ON|OFF>	Selects whether or not to use the power loss table for a power meter calibration.
PWRMCAL	Displays the power meter cal menu and sets the drive port cal power.
PWRR<AUTO|PMAN>	Select the source power range auto or manual mode.
RAID	Completes the response and isolation cal sequence. OPC-compatible.
RAIISOL	Calls the isolation class for the response and isolation calibration. OPC-compatible if only one standard in class.
RAIRESP	Calls the response class for the response and isolation calibration. OPC-compatible if only one standard in class.
READDATE	Outputs the date in the following string format: DD MMM YYYY. HP-IB only command. There is no front-panel equivalent.
READTIME	Outputs the time in the following string format: HH:MM:SS. HP-IB only command. There is no front-panel equivalent.
REAL	Selects the real display format.
RECO	Recalls previously saved display colors.

These commands are used in mixer measurements.

RFGTLO	Sets RF greater than LO.
RFLTLO	Sets RF less than LO.

These commands recall the indicated internal register. They are all OPC-compatible:

RECA1	Register 1.
RECA2	Register 2.
RECA3	Register 3.
RECA4	Register 4.
RECA5	Register 5.
RECAREG<I>	Recalls save/recall registers 01 through 32. RECAREG01 through RECAREG05 are the same as RECA1 through RECA5.
REFD	Completes the reflection calibration subsequence of a 2-port calibration. OPC-compatible.
REFL	Begins the reflection calibration subsequence of a 2-port calibration.
REFOP	Selects reflection for one path, two port calibration.
REFP[D]	Enters the reference position. 0 is the bottom, 10 is the top of the graticule.
REFT	Recall file titles from disk. Requires pass control mode when using the HP-IB port.
REFV[D]	Enters the reference line value.
RESC	Resume cal sequence.
RESD	Restores the measurement display after viewing the operating parameters or list values.
RESPDONE	Completes the response calibration sequence. OPC-compatible.
REST	Measurement restart.

These commands call the reverse calibration classes, during a full 2-port calibration. They are OPC-compatible if there is only one standard in the class:

REVI	Isolation.
REVM	Match.
REVT	Transmission.
RIGL	Selects a plot in the lower right quadrant.
RIGU	Selects a plot in the upper right quadrant.
RSCO	Resets colors for the selected group.
RST	Presets the instrument. OPC-compatible.

These commands select the parameter displayed on the active channel:

S11

S12

S21

S22

SADD	During either a list frequency or limit table edit, adds a new segment to the table.
SAMC<ON|OFF>	Turns sampler correction ON and OFF. Sampler correction is only turned off to take data for custom calibration coefficients.
SAV1	Completes the 1-port calibration sequence. OPC-compatible.
SAV2	Completes the 2-port calibration sequence. OPC-compatible.
SAVC	Completes the transfer of error correction coefficients back into the instrument. OPC-compatible.

These commands store the current instrument state in the indicated internal register. These commands are all OPC-compatible.

SAVE1	Register 1.
SAVE2	Register 2.
SAVE3	Register 3.
SAVE4	Register 4.
SAVE5	Register 5.
SAVEREG<I>	Saves to save/recall registers 01 through 32. SAVEREG01 through SAVEREG05 are the same as SAVE1 through SAVE5.

The following commands define the format for saving files to disk.

SAVUASCI	Selects ASCII format for saving to disk. Also known as citifile format.
SAVUBINA	Selects binary format for saving to disk.
SAVEUSEK	Stores the active calibration kit as the user kit.
SCAL[D]	Sets the trace scale factor.
SCAP<FULL|GRAT>	Selects a full plot, or a plot where the graticule is expanded to P1 and P2.
SDEL	During either a list frequency or a limit table edit, deletes the current segment.
SDON	During either a list frequency or a limit table edit, closes a segment after editing.

These commands control the marker searches. The marker searches place the active marker according to the indicated search criteria. The search is continuously updated if tracking is ON:

SEAL	Search left for next occurrence of the target value.
SEAMAX	Trace maximum.

SEAMIN Trace minimum.

SEAOFF Turns the marker search OFF.

SEAR Search right for next occurrence of the target value.

SEATARG[D] Arbitrary target amplitude.

SEDI[D] During either a frequency or a limit table edit, selects segment D for editing.

SELL[D] Selects the learnstring revision (LRN) to input to or output from the analyzer. The valid parameters are:

0: Defaults to current revision.

201: Revision 8753B 2.01

300: Revision 8753B 3.00

401: Revision 8753C 4.01

402: Revision 8753C 4.02

412: Revision 8753C 4.12

413: Revision 8753C 4.13

This is an HP-IB only command. There is no front-panel equivalent.

SEQ<I> Selects sequence 1 through 6.

SEQWAIT[D] Tells the instrument to wait D seconds during a sequence. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.

SETBIT[D] Sets the specified bit (0 to 7) on the GPIO.

SETDATE[$] Sets the date in the following format: DD MMM YYYY, where DD is the day and must be 2 digits, MMM is the month and must be three alpha characters (JAN, FEB, MAR, APR, MAY, JUN, FUL, AUG, SEP, OCT, NOV, DEC), and YYYY is the year and must be 2 digits.

SETF Set frequency for low pass transform, option 010.

SETTIME[$] Sets the time in the following format: HH:MM:SS, where HH is the hour, MM is minutes, SS is seconds, and each must be 2 digits.

SETZ Set the characteristic impedance of the measurement system.

SHOM Displays the desired softkey menu during a sequence. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.

SING Single sweep. OPC-compatible.

SLID Sliding load done.

SLIL Specifies the standard as a sliding load during a standard definition as part of a cal kit modification.

SLIS Sliding load set.

SLOPE[D] Enters the power slope value.

SLOPO<ON|OFF> Turns the power slope ON and OFF.

SMIC Select Smith chart display format.

The following commands select the marker readout format on a Smith chart:

SMIMGB G+jB.

SMIMLIN Linear.

SMIMLOG Log.

SMIMRI Real/imaginary pairs.

SMIMRX R+jX.

SMOOAPER[D] Sets the smoothing aperture as a percent of the trace.

SMOOO<ON|OFF> Turns smoothing ON and OFF.

The following commands press the indicated soft key:

SOFT1 Softkey 1.

SOFT2 Softkey 2.

SOFT3 Softkey 3.

SOFT4 Softkey 4.

SOFT5 Softkey 5.

SOFT6 Softkey 6.

SOFT7 Softkey 7.

SOFT8 Softkey 8.

SOFR Displays the firmware revision on the screen.

SOUP<ON|OFF> Turns the source power ON and OFF.

SPAN[D] Sets the stimulus span. If a list frequency segment is being edited, sets the span of the segment.

SPEG Displays the specify gate menu.

The following commands initiate the `SPECIFY CLASS` part of modifying a cal kit. After issuing each command, send the analyzer a series of standard numbers to be included in the class. When the class is full, send `CLAD;` to terminate the sequence.

SPECFWDM Forward match.

SPECFWDT Forward transmission.

SPECRESP Response.

SPECRESI Resp & Isol, response.

SPECREVM Reverse match.

SPECREVT Reverse transmission.

SPECS11A S11A (opens).

SPECS11B S11B (shorts).

SPECS11C S11C (loads) .

SPECS22A	S22A (opens).
SPECS22B	S22B (shorts).
SPECS22C	S22C (loads).
SPECTRFM	TRL, Reflect, Forward, Match
SPECTRRM	TRL, Reflect, Reverse, Match
SPECTLFM	TRL, Line, Forward, Match
SPECTLFT	TRL, Line, Forward, Trans
SPECTLRM	TRL, Line, Reverse, Match
SPECTLRT	TRL, Line, Reverse, Trans
SPECTTFM	TRL, Thru, Forward, Match
SPECTTFT	TRL, Thru, Forward, Trans
SPECTTRM	TRL, Thru, Reverse, Match
SPECTTRT	TRL, Thru, Reverse, Trans

SPLD<ON|OFF>	Turns the split display mode ON and OFF.
SRE[D]	Service request enable. A bit set in D enables the corresponding bit in the status byte to generate an SRQ.
SSEG[D]	Selects the desired segment of the frequency list for a list frequency sweep.
STB?	Outputs the status byte.

The following commands select a standard from a class during a calibration sequence. If a class is requested, as in CLASS11A (open, S11 1-port cal,) the analyzer will do one of two things. If there is only one standard in the class, it will measure that standard automatically. If there are several standards in the class, then one of the following commands must be used to select one, causing it to be measured. All of these commands are OPC-compatible:

STANA	Standard listed under softkey 1.
STANB	Standard listed under softkey 2.
STANC	Standard listed under softkey 3.
STAND	Standard listed under softkey 4.
STANE	Standard listed under softkey 5.
STANF	Standard listed under softkey 6.
STANG	Standard listed under softkey 7.

STAR[D]	Enters the start stimulus value. If a list frequency segment is being edited, sets the start of the segment.
STDD	Standard done, define standard sequence, while modifying a cal kit.

The following commands select the standard type after the standard number has been entered during a modify cal kit sequence:

STDTARBI	Arbitrary impedance.

STDTDELA Delay/thru.

STDTLOAD Load.

STDTOPEN Open.

STDTSHOR Short.

STOP[D] Sets the stimulus stop value. If a list frequency segment is being edited, sets the stop of the segment.

These commands store the indicated file on disk. Used with the INTD and EXTD commands to designate the internal or external disk. Requires pass control mode when using the HP-IB port.

STOR1 Stores the current instrument state to disk using the file name provided by the preceding `TITF1;` command.

STOR2 Stores the current instrument state to disk using the file name provided by the preceding `TITF2;` command.

STOR3 Stores the current instrument state to disk using the file name provided by the preceding `TITF3;` command.

STOR4 Stores the current instrument state to disk using the file name provided by the preceding `TITF4;` command.

STOR5 Stores the current instrument state to disk using the file name provided by the preceding `TITF5;` command.

These commands store the instrument state to the indicated sequence to disk. Used with the INTD and EXTD commands to designate the internal or external disk. Requires pass control mode when using the HP-IB port.

STORSEQ1 Sequence 1.

STORSEQ2 Sequence 2.

STORSEQ3 Sequence 3.

STORSEQ4 Sequence 4.

STORSEQ5 Sequence 5.

STORSEQ6 Sequence 6.

STPSIZE While editing a list frequency segment, sets step size.

SVCO Saves display colors.

SWET[D] Sets the sweep time.

SWR Selects the SWR display format.

TAKCS Begins a power meter calibration sweep. Requires pass control. OPC-compatible.

TALKLIST Puts the analyzer in talker listener mode.

TERI[D] Specifies the terminal impedance of an arbitrary impedance standard during a cal kit modification.

TESS? Returns a one on the standard analyzer. (The standard analyzer does not recognize external test sets.) For option 011, returns a one if an

	HP 85046A/B S-parameter test set is present. Returns a two if an HP 85047A S-parameter test set is present.
TIMDTRAN<ON|OFF>	Turns the option 010 (time domain) transform ON and OFF.
TIMESTAM<ON|OFF>	Turns on the clock time for prints and plots.
TINT[D]	Adjusts the tint for the selected display feature.

These commands title the indicated file numbers:

TITF1[$]	File 1.
TITF2[$]	File 2.
TITF3[$]	File 3.
TITF4[$]	File 4.
TITF5[$1	File 5.
TITL[$]	Enters a new CRT title. A maximum of 50 characters (alphanumeric and mathematical symbols) are allowed.

These commands title the indicated internal register:

TITR1[$]	Register 1.
TITR2[$]	Register 2.
TITR3[$]	Register 3.
TITR4[$]	Register 4.
TITR5[$]	Register 5.
TITREG<I>[$]	Titles save/recall registers 01 through 32. TITREG01 through TITREG05 are the same as TITR1 through TITR5.
TITSEQ<I>	Selects the sequence to be titled.
TITTMEM	Sends the title string to trace memory. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.
TITTPMTR	Sends the title string to the power meter address. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.
TITTPERI	Sends the title string to the peripheral address. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.
TITTPRIN	Sends the title string to the printer address. NEWSEQ<I> must precede to ensure that a sequence is currently being created or modified.
TRACK<ON|OFF>	Turns marker search tracking ON and OFF.
TRAD	Completes the transmission calibration subsequence of a 2-port calibration. OPC-compatible.
TRAN	Begins the transmission calibration subsequence of a 2-port calibration.
TRAOP	Selects reflection for one path, two port calibration.
TRIG	HP-IB trigger. Puts instrument into hold mode.

TST? Causes a self test and returns a zero if the test is passed.

TSTP<P1|P2> Selects test port 1 or 2 for non-S-parameter measurements.

These commands set the TTL output and end of sweep pulse:

TTLHPULS TTL normally low, high pulse at end of sweep.

TTLLPULS TTL normally high, low pulse at end of sweep.

TTLOH Sets TTL continuously high.

TTLOL Sets TTL continuously low.

UCONV Selects up converter for mixer measurements.

UP Increments the value in the active entry area (up key).

USEPASC Puts the analyzer in pass control mode.

These commands select the sensor input being used with the HP 438A power meter. For the HP 436A or 437B, the A sensor is always used:

USESENSA Sensor A.

USESENSB Sensor B.

VELOFACT[D] Enters the velocity factor of the transmission medium.

VIEM<ON|OFF> Displays the measurement trace (ON) or the mixer setup (OFF).

VOFF[D] Sets the frequency offset value.

WAIT Waits for a clean sweep. OPC-compatible.

WAVE Specifies a waveguide standard while defining a standard as part of a cal kit modification.

WIDT<ON|OFF> Turns the bandwidth search ON and OFF.

WIDV[D] Enters the widths search parameter.

These commands set the window for the transform (option 010, time domain):

WINDMAXI Maximum.

WINDMINI Minimum.

WINDNORM Normal.

WINDOW[D] Enters arbitrary window.

WINDUSEM<ON|OFF> Turns the trace memory ON as the window shape.

These commands enter new softkey labels into the indicated softkey positions.

WRSK1[$]	Softkey 1.
WRSK2[$]	Softkey 2.
WRSK3[$]	Softkey 3.
WRSK4[$]	Softkey 4.
WRSK5[$]	Softkey 5.
WRSK6[$]	Softkey 6.
WRSK7[$]	Softkey 7.
WRSK8[$]	Softkey 8.

Index

M

T

Contents

Figures

Tables

2

HP-IB Programming Reference

This chapter includes information on the following topics:

- HP-IB operation
- HP-IB requirements
- analyzer command syntax
- analyzer operation
- reading analyzer data
- data-processing chain
- error reporting
- measurement process
- programming examples
  - measurement setup
    - setting parameters
    - verifying parameters
  - measurement calibration
    - S_{11} 1-port measurement calibration
    - full 2-port measurement calibration
  - data transfer
    - data transfer using markers
    - ASCII data transfer using FORM 4
    - data transfer using floating-point numbers
    - data transfer using frequency array information
    - internal-binary data transfer using FORM 1
  - measurement-process synchronization
    - using the error queue
    - generating interrupts
    - power meter calibration
  - network analyzer system setups
    - using the learn string
    - reading calibration data
    - saving and restoring the instrument state
  - limit-line testing
    - setting up a list-frequency sweep
    - selecting a single segment from a table of segments
    - setting up limit lines
    - performing PASS/FAIL tests while tuning
  - report generation
    - operation using the talker/listener mode
    - controlling peripherals using pass-control mode
    - printing with the serial port
    - plotting to a file and transfering file data to a plotter
    - utilizing PC-graphics applications using the plot file
    - reading ASCII disk files to the instrument controller disk file

Where to Look for More Information

Additional information covering many of the topics discussed in this chapter is located in the following:

- Chapter 1, "HP-IB Command Reference," includes both functional and alphabetical lists of all analyzer HP-IB commands.
- *Tutorial Description of the Hewlett-Packard Interface Bus*, presents a description and discussion of all aspects of the HP-IB. A thorough overview of all technical details as a broad tutorial. HP publication, HP part number 5021-1927.
- *IEEE Standard Digital Interface for Programmable Instrumentation ANSI/IEEE std 488.1-1987* contains detailed information on IEEE-488 operation. Published by the Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, New York 10017.
- *IEEE Standard Codes, Formats, Protocols and Common Commands ANSI/IEEE std 488.2-1987* contains detailed information on IEEE-488 operation. Published by the Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, New York 10017.
- Chapter 3, "HP BASIC Programming Reference," includes programming examples in HP BASIC.

HP-IB Operation

The Hewlett-Packard Interface Bus (HP-IB) is Hewlett-Packard's hardware, software, documentation, and support for IEEE 488.1 and IEC-625 worldwide standards for interfacing instruments. This interface allows you to operate the analyzer and peripherals in two methods:

- by an external system controller
- by the network analyzer in system-controller mode

Device Types

The HP-IB employs a party-line bus structure in which up to 15 devices can be connected on one contiguous bus. The interface consists of 16 signal lines and 8 ground lines within a shielded cable. With this cabling system, many different types of devices including instruments, computers, power meters, plotters, printers, and disk drives can be connected in parallel.

Every HP-IB device must be capable of performing one or more of the following interface functions:

Talker

A talker is a device capable of transmitting device-dependent data when addressed to talk. There can be only one active talker at any given time. Examples of this type of device include:

- power meters
- disk drives
- voltmeters
- counters
- tape readers

The network analyzer is a talker when it sends trace data or marker information over the bus.

Listener

A listener is a device capable of receiving device-dependent data over the interface when addressed to listen. There can be as many as 14 listeners connected to the interface at any given time. Examples of this type of device include:

- printers
- power supplies
- signal generators

The network analyzer is a listener when it is controlled over the bus by a system controller.

Controller

A controller is defined as a device capable of:

1. managing the operation of the bus
2. addressing talkers and listeners

There can be only one active controller on the interface at any time. Examples of controllers include desktop computers, minicomputers, workstations, and the network analyzer. In a multiple-controller system, active control can be passed between controllers, but there can only be one *system* controller connected to the interface. The system controller acts as the master and can regain active control at any time. The analyzer is an active controller when it plots, prints, or stores to an external disk drive in the pass-control mode. The analyzer is also a system controller when it is operating in the system-controller mode.

HP-IB Bus Structure

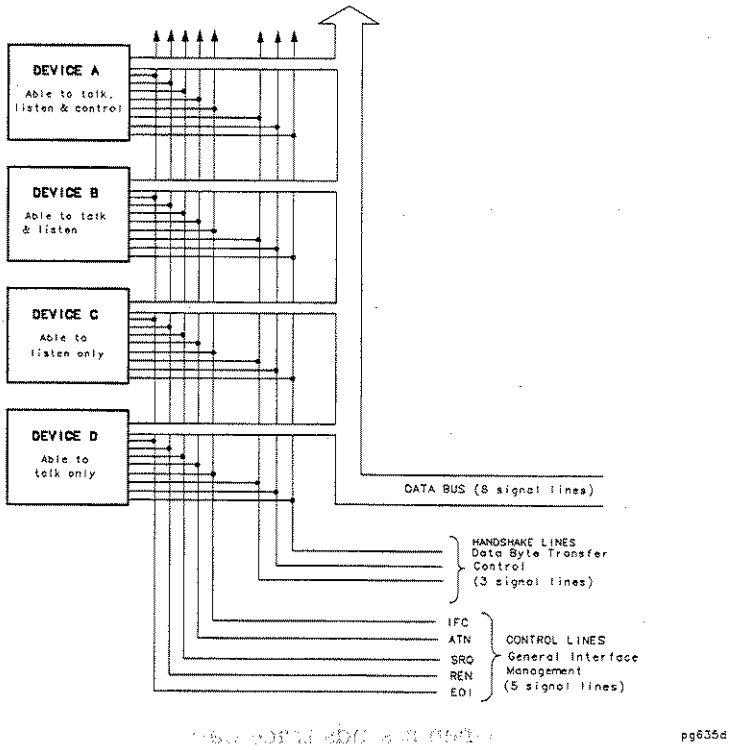


Figure 2-1. HP-IB Bus Structure

Data Bus

The data bus consists of 8 bi-directional lines that are used to transfer data from one device to another. Programming commands and data transmitted on these lines are typically encoded in ASCII, although binary encoding is often used to speed up the transfer of large arrays. Both ASCII- and binary-data formats are available to the analyzer. In addition, every byte transferred over HP-IB undergoes a handshake to insure valid data.

Handshake Lines

A three-line handshake scheme coordinates the transfer of data between talkers and listeners. To insure data integrity in multiple-listener transfers, this technique forces data transfers to occur at the transfer rate of the slowest device connected to the interface. With most computing controllers and instruments, the handshake is performed automatically, making it transparent to the programmer.

Control Lines

The data bus also has five control lines. The controller uses these lines to address devices and to send bus commands.

IFC (Interface Clear)	This line is used exclusively by the system controller. When this line is true (low), all devices (whether addressed or not) unaddress and revert to an idle state.
ATN (Attention)	The active controller uses this line to define whether the information on the data bus is command-oriented or data-oriented. When this line is true (low), the bus is in the command mode, and the data lines carry bus commands. When this line is false (high), the bus is in the data mode, and the data lines carry device-dependent instructions or data.
SRQ (Service Request)	This line is set true (low) when a device requests service and the active controller services the requesting device. The network analyzer can be enabled to pull the SRQ line for a variety of reasons such as requesting control of the interface, for the purposes of printing, plotting, or accessing a disk.
REN (Remote Enable)	This line is used exclusively by the system controller. When this line is set true (low), the bus is in the remote mode, and devices are addressed by the controller to either listen or talk. When the bus is in remote and a device is addressed, it receives instructions from the system controller via HP-IB rather than from its front panel (pressing LOCAL returns the device to front-panel operation). When this line is set false (high), the bus and all of the connected devices return to local operation.
EOI (End or Identify)	This line is used by a talker to indicate the last data byte in a multiple-byte transmission, or by an active controller to initiate a parallel-poll sequence. The analyzer recognizes the EOI line as a terminator, and it pulls the EOI line with the last byte of a message output (data, markers, plots, prints, error messages). The analyzer does not respond to parallel poll.

HP-IB Requirements

Number of Interconnected Devices:	15 maximum.
Interconnection Path Maximum Cable Length:	20 meters maximum or 2 meters-per-device, whichever is less.
Message Transfer Scheme:	Byte serial/bit parallel asynchronous data transfer using a 3-line handshake system.
Data Rate:	Maximum of 1 megabyte-per-second over the specified distances with tri-state drivers. Actual data rate depends on the transfer rate of the slowest device connected to the bus.
Address Capability:	Primary addresses: 31 talk, 31 listen. A maximum of 1 talker and 14 listeners can be connected to the interface at given time.
Multiple-Controller Capability:	In systems with more than one controller (like the analyzer system), only one controller can be active at any given time. The active controller can pass control to another controller, but only the system controller can assume unconditional control. Only one *system* controller is allowed. The system controller is hard-wired to assume bus control after a power failure.

HP-IB Operational Capabilities

On the network analyzer's rear panel, next to the HP-IB connector, there is a list of HP-IB device subsets as defined by the IEEE 488.2 standard. The analyzer has the following capabilities:

SH1	Full-source handshake.
AH1	Full-acceptor handshake.
T6	Basic talker, answers serial poll, unaddresses if MLA is issued. No talk-only mode.
L4	Basic listener, unaddresses if MTA is issued. No listen-only mode.
SR1	Complete service request (SRQ) capabilities.
RL1	Complete remote/local capability including local lockout.
PP0	Does not respond to parallel poll.
DC1	Complete device clear.
DT1	Responds to a Group Execute Trigger (GET) in the hold-trigger mode.
C1,C2,C3	System controller capabilities in system-controller mode.
C10	Pass control capabilities in pass-control mode.
E2	Tri-state drivers.
LE0	No extended listener capabilities.
TE0	No extended talker capabilities.

HP-IB Status Indicators

When the analyzer is connected to other instruments over the HP-IB, the HP-IB status indicators illuminate to display the current status of the analyzer. The HP-IB status indicators are located in the instrument-state function block on the front panel of the network analyzer.

R = Remote Operation

L = Listen mode

T = Talk mode

S = Service request (SRQ) asserted by the analyzer

Bus Device Modes

The analyzer uses a single-bus architecture. The single bus allows both the analyzer and the host controller to have complete access to the peripherals in the system.

Three different controller modes are possible in and HP-IB system:

- system-controller mode
- talker/listener mode
- pass-control mode

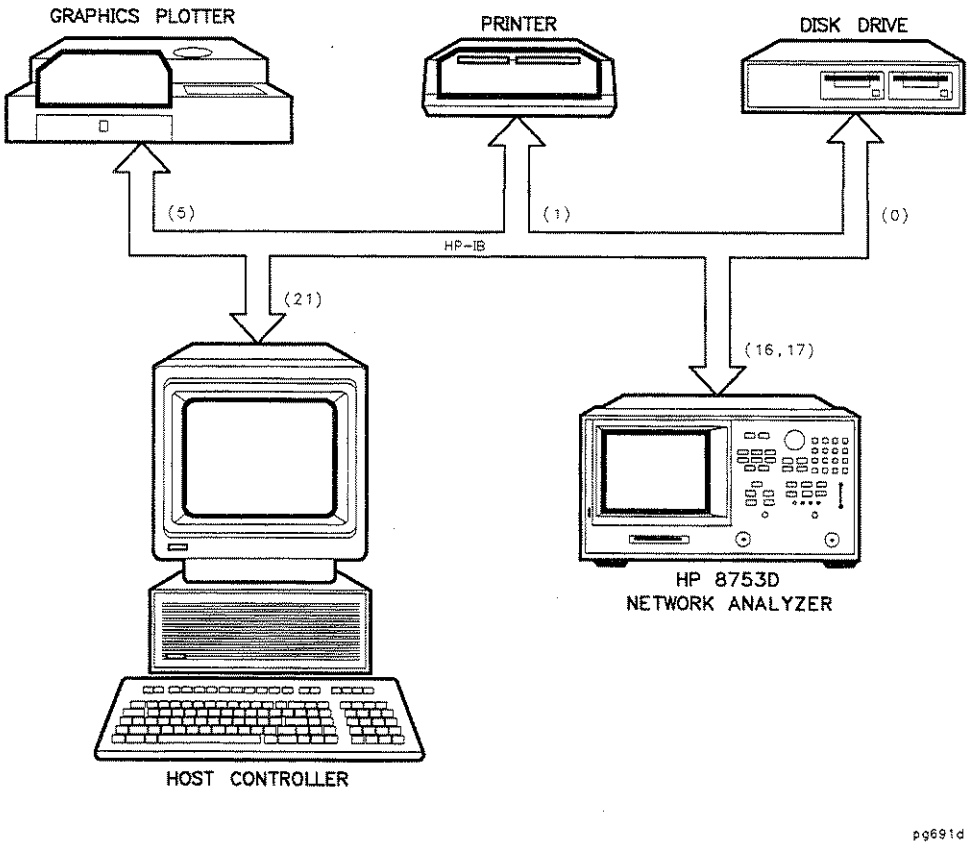


Figure 2-2. Analyzer Single Bus Concept

System-Controller Mode

This mode allows the analyzer to control peripherals directly in a stand-alone environment (without an external controller). This mode can only be selected manually from the analyzer's front panel. It can only be used if no active computer or instrument controller is connected to the system via HP-IB. If an attempt is made to set the network analyzer to the system-controller mode when another controller is connected to the interface, the following message is displayed on the analyzer's display screen:

```
"CAUTION: CAN'T CHANGE - ANOTHER CONTROLLER ON BUS"
```

The analyzer must be set to the system-controller mode in order to access peripherals from the front panel. In this mode, the analyzer can directly control peripherals (plotters, printers, disk drives, power meters, etc.) and the analyzer may plot, print, store on disk or perform power meter functions.

Note Do not attempt to use this mode for programming. HP recommends using an external instrument controller when programming. See the following section, "Talker/Listener Mode."

Talker/Listener Mode

This is the mode that is normally used for remote programming of the analyzer. In talker/listener mode, the analyzer and all peripheral devices are controlled from an external instrument controller. The controller can command the analyzer to talk and other devices to listen. The analyzer and peripheral devices cannot talk directly to each other unless the computer sets up a data path between them. This mode allows the analyzer to act as either a talker or a listener, as required by the controlling computer for the particular operation in progress.

Pass-Control Mode

This mode allows the computer to control the analyzer via HP-IB (as with the talker/listener mode), but also allows the analyzer to take control of the interface in order to plot, print, or access a disk. During an analyzer-controlled peripheral operation, the host computer is free to perform other internal tasks (i.e. data or display manipulation) while the analyzer is controlling the bus. After the analyzer-controlled task is completed, the analyzer returns control to the system controller.

Note Performing an instrument preset does not affect the selected bus mode, although the bus mode will return to talker/listener mode if the line power is cycled.

Note "Specifications and Measurement Uncertainties" in the *HP 8753D Network Analyzer User's Guide* provides information on setting the correct bus mode from the front-panel menu.

Analyzer Bus Modes

As discussed earlier, under HP-IB control, the analyzer can operate in one of three modes: talker/listener, pass-control, or system-controller mode.

In talker/listener mode, the analyzer behaves as a simple device on the bus. While in this mode, the analyzer can make a plot or print using the `OUTPPLOT;` or `OUTPPRIN;` commands. The analyzer will wait until it is addresses to talk by the system controller and then dump the display to a plotter/printer that the system controller has addressed to listen. Use of the commands `PLOT;` and `PRINALL;` require control to be passed to another controller.

In pass-control mode, the analyzer can request control from the system controller and take control of the bus if the controller addresses it to take control. This allows the analyzer to take control of printers, plotters, and disk drives on an as-needed basis. The analyzer sets event-status-register bit 1 when it needs control of the interface, and the analyzer will transfer control back to the system controller at the completion of the operation. It will pass control back to its controller address, specified by ADDRCONT.

The analyzer can also operate in the system-controller mode. This mode is only used when there is no remote controller on the bus. In this mode, the analyzer takes control of the bus, and uses it whenever it needs to access a peripheral. While the analyzer is in this mode, no other devices on the bus can attempt to take control. Specifically, the REN, ATN, and IFC lines must remain unasserted, and the data lines must be freed by all but the addressed talker.

Setting HP-IB Addresses

In systems interfaced using HP-IB, each instrument on the bus is identified by an HP-IB address. This address code must be different for each instrument on the bus. These addresses are stored in short-term, non-volatile memory and are not affected when you press (PRESET) or cycle the power. The analyzer occupies two HP-IB addresses: the instrument itself and the display. The display address is derived from the instrument address by complementing the instrument's least-significant bit. Hence, if the instrument is at an even address, the display occupies the next higher address. If the instrument is at an odd address, the display occupies the next lower address. A change in the next instrument preset following the address change.

The analyzer addresses are set by pressing (LOCAL) SET ADDRESSES. In system-controller mode, the addresses must be set for the plotter, printer, disk drive, and power meter.

The default address for the analyzer is device 16, and the display address is device 17.

Note There is also an address for the system controller. This address refers to the controller when the network analyzer is being used in pass-control mode. This is the address that control is passed back to when the analyzer-controlled operation is complete.

Response to HP-IB Meta-Messages (IEEE-488 Universal Commands)

Abort

The analyzer responds to the abort message (IFC) by halting all listener, talker, and controller functions.

Device Clear

The analyzer responds to the device clear commands (DCL, SDC) by clearing the input and output queues, and clearing any HP-IB errors. The status registers and the error queue are unaffected.

Local

The analyzer will go into local mode if: the local command (GTL) is received, the remote line is unasserted, or the front-panel local key is pressed. Changing the analyzer's HP-IB status from remote to local does not affect any of the front-panel functions or values.

Local Lockout

If the analyzer receives the local-lockout command (LLO) while it is in remote mode, it will disable the entire front panel except for the line power switch. A local-lockout condition can only be cleared by releasing the remote line, although the local command (GTL) will place the instrument temporarily in local mode.

Parallel Poll

The analyzer does not respond to parallel-poll configure (PPC) or parallel-poll unconfigure (PPU) messages.

Pass Control

If the analyzer is in pass-control mode, is addressed to talk, and receives the take-control command (TCT), from the system control it will take active control of the bus. If the analyzer is not requesting control, it will immediately pass control to the system controller's address.

Otherwise, the analyzer will execute the function for which it sought control of the bus and then pass control back to the system controller.

Remote

The analyzer will go into remote mode when the remote line is asserted and the analyzer is addressed to listen. While the analyzer is held in remote mode, all front-panel keys (with the exception of (LOCAL)) are disabled. Changing the analyzer's HP-IB status from remote to local does not affect any front-panel settings or values.

Serial Poll

The analyzer will respond to a serial poll with its status byte, as defined in the "Status Reporting" section of this chapter. To initiate the serial-poll sequence, address the analyzer to talk and issue a serial-poll enable command (SPE). Upon receiving this command, the analyzer will return its status byte. End the sequence by issuing a serial-poll disable command (SPD). A serial poll does not affect the value of the status byte, and it does not set the instrument to remote mode.

Trigger

In hold mode, the analyzer responds to device trigger by taking a single sweep. If a one-path, 2-port measurement calibration is active, the analyzer will set the waiting-for-Group-Execute-Trigger bits in the status byte. If waiting-for-forward-GET is set, the analyzer will assume the device is connected for forward measurement and take a sweep when GET is received. Similarly, if the waiting-for-reverse-GET bit is set, the analyzer will assume the device is connected for reverse measurement. The analyzer responds only to selected-device trigger (SDT). This means that it will not respond to group execute-trigger (GET) unless it is addressed to listen. The analyzer will not respond to GET if it is not in hold mode.

Analyzer Command Syntax

Code Naming Convention

The analyzer HP-IB commands are derived from their front-panel key titles (where possible), according to the naming convention described in this section.

Simple commands are the first four letters of the function they control, as in POWE, the command name for power. If the function label contains two words, the first three mnemonic letters are the first three letters of the first word, and the fourth mnemonic letter is the first letter of the second word. For example, ELED is derived from electrical delay.

If there are many commands grouped together in a class, as in markers or plotting pen numbers, the command is increased to 8 letters. The first 4 letters are the class label derived using rule 1. The last 4 letters are the function specifier, again derived using rule 1. As an example, class pen numbers are represented by the command PENN, which is used in combination with several functions such as PENNDATA, PENNMEMO.

The code naming guidelines, listed in Table 2-1, are used in order to:

- make commands more meaningful and easier to remember
- maintain compatibility with other products

Note There are times when these guidelines are not followed due to technical considerations.

Table 2-1. Code Naming Convention

Convention	Key Title	For HP-IB Code Use	Example
One Word	Power	First Four Letters	POWE
	Start		STAR
Two Words	Electrical Delay	First Three Letters of First Word, First Letter of Second Word	ELED
	Search Right		SEAR
Two Words in a Group	Marker →Center	Four Letters of Both	MARKCENT
	Gate →Span		GATESPAN
Three Words	Cal Kit N 50Ω	First Three Letters of First Word, First Letter of Second Word, First Four Letters of Third Word	CALKN50
	Pen Num Data		PENNDATA

Some codes require appendages (ON, OFF, 1, 2, etc.). Codes that do not have a front-panel equivalent are HP-IB only commands. They use a similar convention based on the common name of the function. Where possible, analyzer codes are compatible with HP 8510A/B codes.

Note Front-panel-equivalent codes and HP-IB-only codes are summarized in "Command Reference" in this guide.

Valid Characters

The analyzer accepts ASCII letters, numbers, decimal points, +/-, semicolons (;), quotation marks ("), carriage returns (CR), and linefeeds (LF). Both upper- and lower-case letters are acceptable. Carriage returns, leading zeros, spaces, and unnecessary terminators are ignored, except for those within a command or appendage. An invalid character causes a syntax error. Syntax errors are described in more detail in "Error Messages" of the *HP 8753D Network Analyzer User's Guide.*

Units and Terminators

The analyzer outputs data in basic units and assumes these basic units when it receives an input (unless the input is otherwise qualified). The basic units and allowable expressions can be found below. Both upper- and lower-case letters are acceptable.

Terminator codes

S	Seconds	HZ	Hertz
MS	Milliseconds	KHZ	Kilohertz
US	Microseconds	MHZ	Megahertz
NS	Nanoseconds	GHZ	Gigahertz
PS	Picoseconds	DB	dB or dBm
FS	Femtoseconds	V	Volts

Terminators are used to indicate the end of a command. This allows the analyzer to recover to the next command in the event of a syntax error. The semicolon (;) is the recommended command terminator. The line-feed character (LF) and the HP-IB EOI line can also be used as terminators. Again, the analyzer will ignore the carriage-return character (CR).

Command Formats

The HP-IB commands accepted by the analyzer can be grouped into five input-syntax types. The analyzer does not distinguish between upper- and lower-case letters.

General Structure:

The general syntax structure is: [code][appendage][data][unit][terminator]

The individual sections of the syntax code are explained below.

[code] The root mnemonic, as found in "Command Reference."

[appendage] A qualifier attached to the root mnemonic. Possible appendages are ON or OFF (toggle a function ON or OFF), or integers, which specify one option out of several. There can be no spaces or symbols between the code and the appendage.

[data] A single operand used by the root mnemonic, usually to set the value of a function. The data can be a number or a character string. Numbers are accepted as integers or decimals, with power of ten specified by E (for example, `STAR 0.2E+10;` sets the start frequency to 2 GHz). Character strings must be preceded and followed by double quotation marks (for example `"STAR 0.2E+10"` must be sent to the analyzer, not STAR 0.2E+10.)

[unit] The units of the operand, if applicable. If no units are specified, the analyzer assumes the basic units as described above. The data is entered into the function when either units or a terminator are received.

[terminator] Indicates the end of the command, enters the data, and switches the active-entry area OFF. A semicolon (;) is the recommended terminator.

The analyzer also interprets line feed as a terminator. It is a advantageous to carefully and methodically use semicolons as terminators. This practice insures correct command interpretation and operation.

Syntax Types

The specific syntax types are:

SYNTAX TYPE 1: [code] [terminator]

These are simple action commands that require no complementary information, such as `AUTO;` (autoscales the active channel).

SYNTAX TYPE 2: [code][appendage][terminator]

These are simple action commands requiring limited customization, such as `CORRON;` and `CORROFF;` (Switch error correction ON or OFF) or `RECA1;`, `RECA2;`, `RECA3;` (recall register 1, 2, 3). There can be no characters or symbols between the code and the appendage. In the following cases: CLEAREG[D], RECAREG[D], SAVEREG[D], and TITREG[D], [D] must be 2 characters. For example, `CLEAREG01;` will execute; `CLEAREG1;` will generate a syntax error.

SYNTAX TYPE 3: [code] [data] [unit][terminator]

These are data-input commands such as `STAR 1.0 GHZ;` (set the start frequency to 1 GHz).

SYNTAX TYPE 4: [code] [appendage] [data] [terminator]

These are titling and marker commands that have an appendage, such as TITL "FILTER" (enter FILTER as the CRT title), TITR1 "STATE1", TITR2 "EMPTY" (title register 1 STATE1, title register 2 EMPTY.)

INTERROGATE SYNTAX: [code][?]

To interrogate a front-panel-equivalent function, append a question mark (?) to the root mnemonic, for example POWE?, AVERO?, or REAL?. To interrogate commands with integer appendages, place the question mark after the appendage.

Analyzer Operation

Held Commands

The analyzer cannot process HP-IB commands while executing certain key commands known as "held" commands. Once a held command is received, the analyzer will read new commands into the input buffer, but it will not begin the execution of any commands until the completion of the held command. When the 15-character input buffer is full, the analyzer will put hold on the bus until it is able to process the commands in the buffer.

Operation Complete

Note For a complete list of OPC-compatible commands, see the section titled "HP-IB Capabilities" at the beginning of Chapter 1.

Occasionally, there is a need to know when certain analyzer operations have been completed. There is an operation-complete function (OPC) that allows a synchronization of programs with the execution of certain key commands. This mechanism is activated by issuing `OPC;` or `OPC?;` prior to an OPC-compatible command. The status byte or ESR operation-complete bit will then be set after the execution of the OPC-compatible command. For example, issuing `OPC;SING;` causes the OPC bit to be set when the single sweep is finished. Issuing `OPC?;` in place of the `OPC;` causes the analyzer to output a one (1) when the command execution is complete. Addressing the analyzer to talk after issuing `OPC?;` will not cause an `"addressed to talk without selecting output"` error, but the analyzer will halt the computer by not transmitting the one (1) until the command has completed. For example, executing `OPC?;PRES;`, and then immediately interrogating the analyzer causes the bus to halt until the instrument preset is complete and the analyzer outputs a one (1).

Note Commands that call a calibration class are held if there is only one standard in the class, since these commands trigger a measurement.

As another example, consider the timing of sweep completion. Send the command string `"SWET 3 S;OPC?;SING;"` to the analyzer. This string sets the analyzer sweep time to 3 seconds, and then waits for completion of a single sweep to respond with a one (1). The computer is programmed to read the one (1) response from the analyzer as the completion of the single sweep. The program then waits until the sweep is completed before continuing operation. At this point a valid trace exists and the trace data can be read into the computer.

Reading Analyzer Data

Output Queue

Whenever a output-data command is received, the analyzer puts the data into the output queue (or buffer) where it is held until the system controller outputs the next read command. The queue, however, is only one event long: the next output-data command will overwrite the data already in the queue. Therefore, it is important to read the output queue immediately after every interrogation or data request from the analyzer.

Command Interrogate

All instrument functions can be interrogated to find the current ON/OFF state or value. For instrument state commands, append the question mark character (?) to the command to interrogate the state of the functions. Suppose the operator has changed the power level from the analyzer's front panel. The computer can ascertain the new power level using the analyzer's command-interrogate function. If a question mark is appended to the root of a command, the analyzer will output the value of that function. For instance, `POWE 7 DB;` sets the source power to 7 dB, and `POWE?;` outputs the current RF source power at the test port. When the analyzer receives `POWE?;`, it prepares to transmit the current RF source power level. This condition illuminates the analyzer front-panel talk light (T). In this case, the analyzer transmits the output power to the controller.

ON/OFF commands can be also be interrogated. The reply is a one (1) if the function is ON or a zero (0) if it is OFF. For example, if a command controls an active function that is underlined on the analyzer display, interrogating that command yields a one (1) if the command is underlined or a zero (0) if it is not. As another example, there are nine options on the format menu and only one option is underlined at a time. Only the underlined option will return a one when interrogated.

For instance, send the command string `DUAC?;` to the analyzer. If dual-channel display is switched ON, the analyzer will return a one (1) to the instrument controller.

Similarly, to determine if phase is being measured and displayed, send the command string `PHAS?;` to the analyzer. In this case, the analyzer will return a one (1) if phase is currently being displayed. Since the command only applies to the active channel, the response to the `PHAS?;` query depends on which channel is active.

Output Syntax

The following three types of data are transmitted by the analyzer in ASCII format:

- response to interrogation
- certain output commands
- ASCII floating-point (FORM 4) array transfers

Marker-output commands and interrogated commands are output in ASCII format only, meaning that each character and each digit is transmitted as a separate byte, leaving the receiving computer to reconstruct the numbers and strings. Numbers are transmitted as 24-character strings, consisting of:

-DDD.DDDDDDDDDDDDDDE-DD

When multiple numbers are sent, the numbers are separated by commas. See Table 2-2 for character definitions.

Table 2-2. FORM 4 (ASCII) Data-Transfer Character Definitions

Character(s)	Definition
Sign	'-' for negative, blank for positive.
3 digits	Digits to the left of the decimal point.
Decimal point	
15 digits	Digits to the right of the decimal point.
E	Exponent notation.
Sign	'-' for negative, '+' for positive.
Exponent	Two digits for the exponent.

Marker data

The network analyzer offers several options for outputting trace-related data. Trace information can be read out of the analyzer in several methods. Data can be selectively read from the trace using the markers, or the entire trace can be read by the controller. If only specific information is required (such as a single point on the trace or the result of a marker search), the marker output command can be used to read the information.

To read the trace data using the marker, the marker must first be assigned to the desired frequency. This is accomplished using the marker commands. The controller sends a marker command followed by a frequency within the trace-data range. If the actual desired frequency was not sampled, the markers can be set to continuous mode and the desired marker value will be linearly interpolated from the two nearest points. This interpolation can be prevented by putting the markers into discrete mode. Discrete mode allows the marker to only be positioned on a measured trace-data point.

As an alternative, the analyzer can be programmed to choose the stimulus value by using the MARKER SEARCH function. Maximum, minimum, target value, or bandwidths search can be automatically determined with MARKER SEARCH. To continually update the search, switch the marker tracking ON. The trace-maximum search will remain activated until:

- The search is switched OFF.
- The tracking is switched OFF.
- All markers are switched OFF.

Marker data can be output to a controller with a command to the analyzer. This set of commands causes the analyzer to transmit three numbers: marker value 1, marker value 2, and marker stimulus value. In log-magnitude display mode we get the log magnitude at marker 1, zero, and the marker frequency. See Table 2-3 for a complete listing of all the possibilities for values 1 and 2. The three possibilities for the third parameter are:

- frequency
- time (as in time domain, option 010 only)
- CW time

Table 2-3. Units as a Function of Display Format

Display Format	Marker Mode	OUTPMARK		OUTPFORM		MARKER READOUT*	
		value 1	value 2	value 1	value 2	value	aux value
LOG MAG		dB	†	dB	†	dB	†
PHASE		degrees	†	degrees	†	degrees	†
DELAY		seconds	†	seconds	†	seconds	†
SMITH CHART	LIN MKR	lin mag	degrees	real	imag	lin mag	degrees
	LOG MKR	dB	degrees	real	imag	dB	degrees
	Re/Im	real	imag	real	imag	real	imag
	R + jX	real	imag ohms	real	imag	real	imag ohms
	G + jB	real	imag Siemens	real	imag	real	imag Siemens
POLAR	LIN MKR	lin mag	degrees	real	imag	lin mag	degrees
	LOG MKR	dB	degrees	real	imag	dB	degrees
	Re/Im	real	imag	real	imag	real	imag
LIN MAG		lin mag	†	lin mag	†	lin mag	†
REAL		real	†	real	†	real	†
SWR		SWR	†	SWR	†	SWR	†

*The marker readout values are the marker values displayed in the upper right-hand corner of the display. They also correspond to the value and auxiliary value associated with the fixed marker.

†Value 2 is not significant in this format, though it is included in data transfers.

Array-Data Formats

The analyzer can transmit and receive arrays in the analyzer's internal binary format as well as four different numeric formats. The current format is set with the FORM1, FORM2, FORM3, FORM4, and FORM5 commands. These commands do not affect learn-string transfers, calibration-kit string transfers, or non-array transfers, such as command interrogate, or output marker values. A transmitted array will be output out in the current format, and the analyzer will attempt to read incoming arrays according to the current format. Each data point in an array is a pair of numbers, usually a real/imaginary pair. The number of data points in each array is the same as the number of points in the current sweep.

The five formats are described below:

FORM 1 — The analyzer's internal binary format, 6 bytes-per-data point. The array is preceded by a four-byte header. The first two bytes represent the string "#A", the standard block header. The second two bytes are an integer representing the number of bytes in the block to follow. FORM 1 is best applied when rapid data transfers, not to be modified by the computer nor interpreted by the user, are required.

FORM 2 — IEEE 32-bit floating-point format, 8 bytes-per-data point. The data is preceded by the same header as in FORM1. Each number consists of a 1-bit sign, an 8-bit biased exponent, and a 23-bit mantissa. FORM 2 is the format of choice if your computer supports single-precision floating-point numbers.

FORM 3 — IEEE 64-bit floating-point format, 16 bytes-per-data point. The data is preceded by the same header as in FORM 1. Each number consists of a 1-bit sign, an 11-bit biased exponent, and a 52-bit mantissa. This format may be used with double-precision floating-point numbers. No additional precision is available in the analyzer data, but FORM 3 may be a convenient form for transferring data to your computer.

FORM 4 — ASCII floating-point format. The data is transmitted as ASCII numbers, as described in "Output Syntax".

There is no header. The analyzer uses FORM 4 to transfer data that is not related to array transfers (i.e. marker responses and instrument settings).

FORM 5 — PC-DOS 32-bit floating-point format with 4 bytes-per-number, 8 bytes-per-data point. The data is preceded by the same header as in FORM 1. The byte order is reversed to comply with PC-DOS formats. If you are using a PC-based controller, FORM 5 is the most effective format to use.

The analyzer terminates each transmission by asserting the EOI interface line with the last byte transmitted. Table 2-4 offers a comparative overview of the five array-data formats.

Table 2-4. HP 8753D Network Analyzer Array-Data Formats

Format type	Type of Data	Bytes per Data Value	Bytes-per-point 2 data values	201 Bytes-per-trace	Total Bytes add header
FORM 1	Internal Binary	3	6	1206	1210
FORM 2	IEEE 32-bit Floating-Point	4	8	1608	1612
FORM 3	IEEE 64-bit Floating-Point	8	16	3216	3220
FORM 4	ASCII Numbers	24	50	10,050	10,050*
FORM 5	PC-DOS 32-bit Floating-Point	4	8	1608	1612

*No header is used in FORM 4.

Trace-Data Transfers

Transfering trace data from the analyzer using an instrument controller can be divided into three steps:

1. allocating an array to receive and store the data
2. commanding the analyzer to transmit the data
3. accepting the transferred data

Data residing in the analyzer is always stored in pairs for each data point (to accommodate real/imaginary pairs). Hence, the receiving array has to be two elements wide, and as deep as the number of points in the array being transferred. Memory space for the array must be declared before any data can be transferred from the analyzer to the computer.

As mentioned earlier, the analyzer can transmit data over HP-IB in five different formats. The type of format affects what kind of data array is declared (real or integer), because the format determines what type of data is transferred. Examples of data transfers using different formats are discussed below. The first, Example 3A, illustrates an ASCII transfer using FORM 4. For more information on the various data formats, turn to "Array Data Formats" located earlier

in this chapter. For information on the various types of data that can be obtained (raw data, error-corrected data, etc.), see "Data Levels," located later in this chapter.

Note "Example 7C: Reading ASCII Disk Files to the Instrument Controller's Disk File," located later in this chapter, explains how to access disk files from a computer.

Frequency-Related Arrays

Frequency-related values are calculated for the analyzer displays. The only data available to the programmer are the start and stop frequencies or center and span frequencies of the selected frequency range.

In a linear frequency range, the frequency values can be easily calculated because the trace data points are equally spaced across the trace. Relating the data from a linear frequency sweep to frequency can be done by interrogating the start frequency, the frequency span, and the number of points in the trace.

Given that information, the frequency of point n in a linear-frequency sweep is represented by the equation:

$$F = \text{Start frequency} + (n-1) \times \text{Span}/(\text{Points}-1)$$

In most cases, this is an easy solution for determining the related frequency value that corresponds with a data point. This technique is illustrated in "Example 3B: Data Transfer Using FORM 4 (ASCII Format)."

When using log sweep or a list-frequency sweep, the points are not evenly spaced over the frequency range of the sweep. In these cases, the frequencies can be read directly out of the instrument with the OUTPLIML command.

Executing `OUTPLIML;` reports the limit-test results by transmitting:

- the stimulus point tested
- a number indicating the limit-test results
- the upper test limit at the stimulus point (if available)
- the lower test limit at the stimulus point (if available)

The numbers used to indicate the limit-test results are:

- a negative one (−1) for no test
- a zero (0) for fail
- a positive one (1) for pass

If there are no limits available, the analyzer transmits zeros.

This data is very useful when testing with limit lines. It provides a method of obtaining accurate frequency-stimulus values to correspond with the trace data. The other limit-related values may be discarded and the stimulus values used with the trace-data points.

Data-Processing Chain

This section describes the manner in which the analyzer processes measurement data. It includes information on data arrays, common output commands, data levels, the learn string, and the calibration kit string.

Data Arrays

Figure 2-3 shows the different kinds of data available within the instrument:

- raw measured data
- error-corrected data
- formatted data
- trace memory
- calibration coefficients

Trace memory can be directly output to a controller with `OUTPMEMO;`, but it cannot be directly transmitted back. If an analyzer with time domain (option 10) contains 1601 points, the formatted data array will have only 401 points.

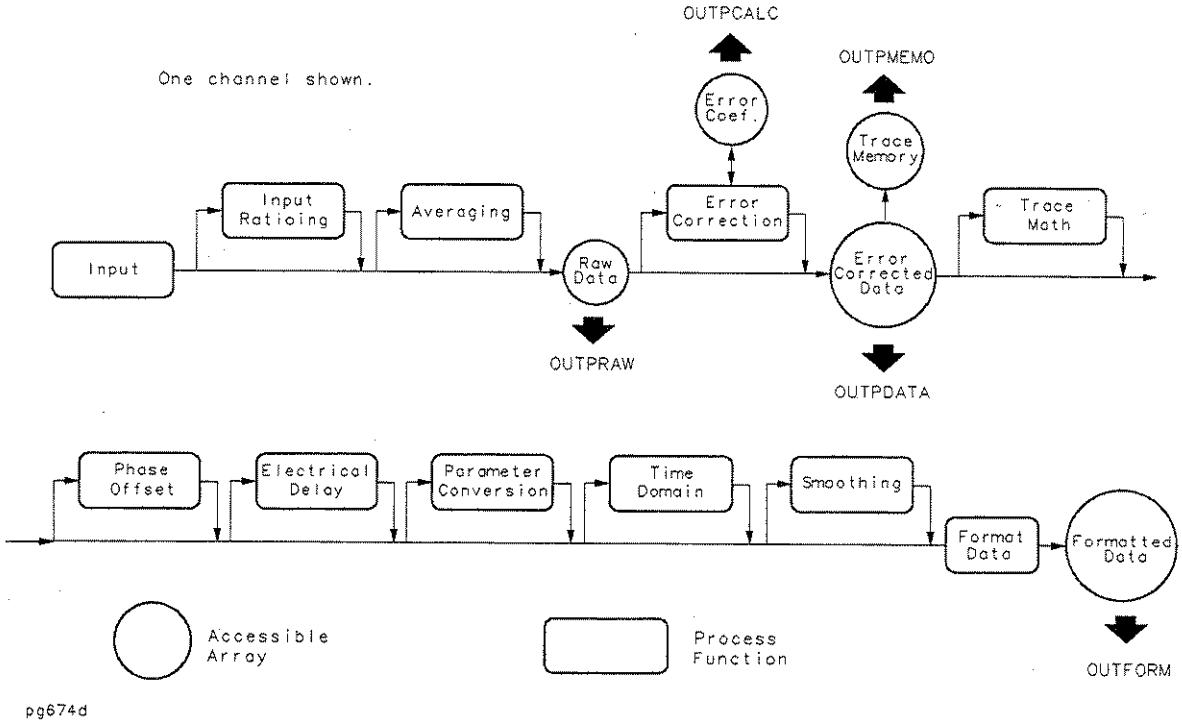


Figure 2-3. The Data-Processing Chain

All the data-output commands are designed to insure that the data transmitted reflects the current state of the instrument:

- OUTPDATA, OUTPRAW, and OUTPFORM will not transmit data until all formatting functions have completed.
- OUTPLIML, OUTPLIMM, and OUTPLIMF will not transmit data until the limit test has occurred (if activated).
- OUTPMARK will activate a marker if a marker is not already selected. It will also insure that any current marker searches have been completed before transmitting data.

- OUTPMSTA insures that the statistics have been calculated for the current trace before transmitting data. If the statistics are not activated, it will activate the statistics long enough to update the current values before deactivating the statistics.
- OUTPMWID insures that a bandwidth search has been executed for the current trace before transmitting data. If the bandwidth-search function is not activated, it will activate the bandwidth-search function long enough to update the current values before switching OFF the bandwidth-search functions.

Data Levels

Different levels of data can be read out of the instrument. Refer to the data-processing chain in Figure 2-3. The following list describes the different types of data that are available from the network analyzer.

Raw data	The basic measurement data, reflecting the stimulus parameters, IF averaging, and IF bandwidth. If a full 2-port measurement calibration is activated, there are actually four raw arrays kept: one for each raw S-parameter. The data can be output to a controller with the commands OUTPRAW1, OUTPRAW2, OUTPRAW3, OUTPRAW4. Normally, only raw 1 is available, and it holds the current parameter. If a 2-port measurement calibration is active, the four arrays refer to S_{11}, S_{21}, S_{12}, and S_{22} respectively. This data is represented in real/imaginary pairs.
Error-corrected data	This is the raw data with error-correction applied. The array represents the currently measured parameter, and is stored in real/imaginary pairs. The error-corrected data can be output to a controller with the `OUTPDATA;` command. The `OUTPMEMO;` command reads the trace memory, if available. The trace memory also contains error-corrected data. Note that neither raw nor error-corrected data reflect such post-processing functions as electrical-delay offset, trace math, or time-domain gating.
Formatted data	This is the array of data actually being displayed. It reflects all post-processing functions such as electrical delay and time domain. The units of the array output depend on the current display format. See Table 2-3 for the various units defined as a function of display format.
Calibration coefficients	The results of a measurement calibration are arrays containing calibration coefficients. These calibration coefficients are then used in the error-correction routines. Each array corresponds to a specific error term in the error model. The *HP 8753D Network Analyzer User's* details which error coefficients are used for specific calibration types, as well as the arrays those coefficients can be found in. Not all calibration types use all 12 arrays. The data is stored as real/imaginary pairs.

Generally, formatted data is the most useful of the four data levels, because it is the same information the operator sees on the display. However if post-processing is unnecessary (e.g. possibly in cases involving smoothing), error-corrected data may be more desirable. Error-corrected data also affords the user the opportunity to input the data to the network analyzer and apply post-processing at another time.

Learn String and Calibration-Kit String

The learn string is summary of the instrument state. It includes all the front-panel settings, the limit-test tables, and the list-frequency table for the current instrument state. It does not include calibration data or the information stored in the save/recall registers.

The learn string can be output to a controller with the `OUTPLEAS;` executable, which commands the analyzer to start transmitting the binary string. The string has a fixed length for a given firmware revision. It can not be more than 3000 bytes in length. The array has the same header as in FORM 1.

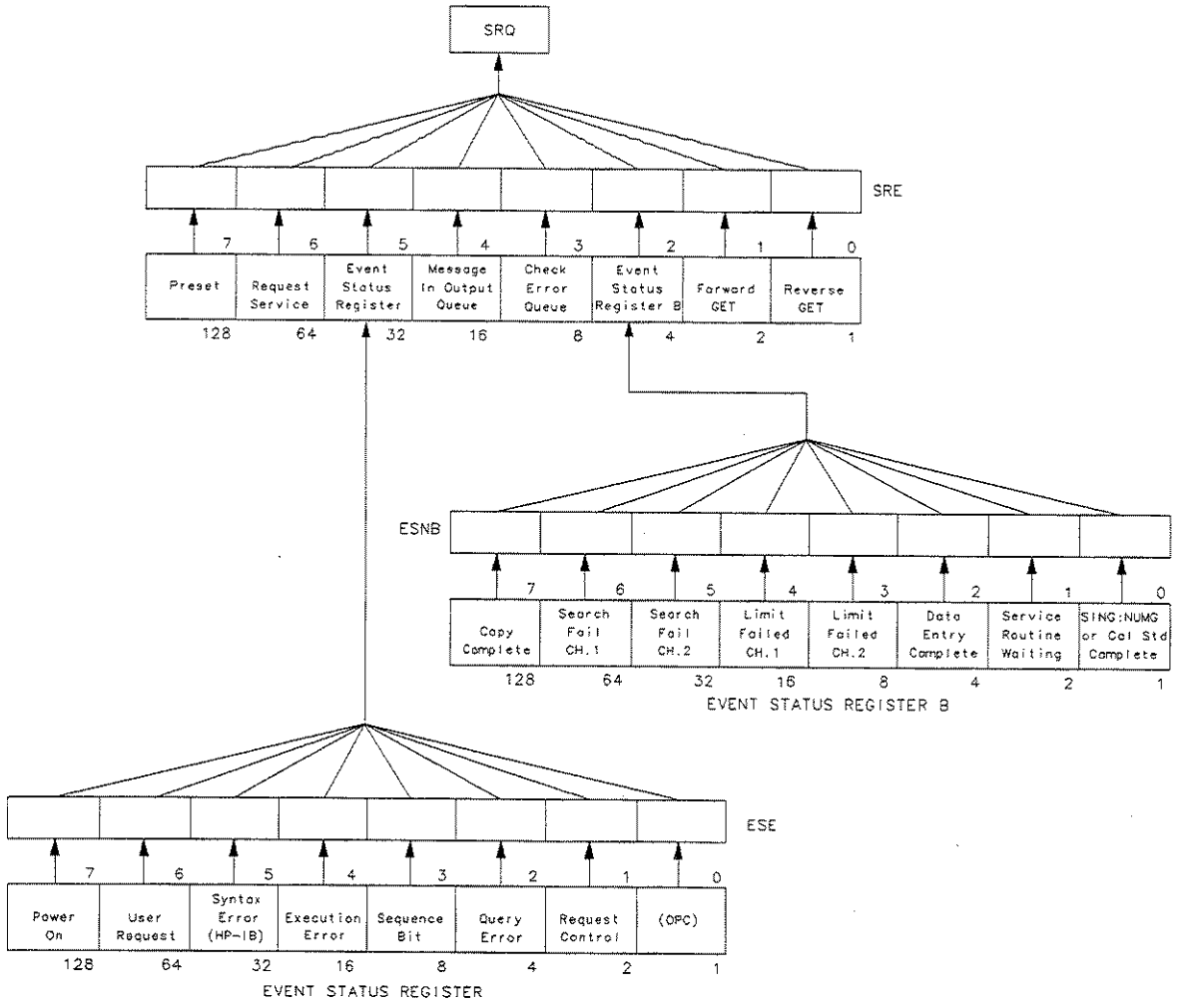
The calibration kit is a set of key characteristics of the calibration standards used to increase the calibration accuracy. There are default kits for several different connector types. There is also space for a user-defined calibration kit. The command OUTPCALK outputs the currently active calibration kit as a binary string in FORM 1. As with the learn string, the calibration-kit string has a fixed length for a given firmware revision. It can not be longer than 1000 bytes.

Error Reporting

This section describes the analyzer's error-reporting process. It includes information on status reporting, the status byte, the event-status registers, and the error output.

Status Reporting

The analyzer status reporting structure is depicted in Figure 2-4.



pg6151d

Figure 2-4. Status Reporting Structure

Table 2-5. Status Bit Definitions

Status Byte		
Bit	**Name**	**Definition**
0	Waiting for reverse GET	A one path, 2-port measurement calibration is active, and the instrument has stopped, waiting for the operator to connect the device for reverse measurement.
1	Waiting for forward GET	A one path, 2-port measurement calibration is active, and the instrument has stopped, waiting for the operator to connect the device for forward measurement.
2	Check event-status-register B	One of the enabled bits in event status register B has been set.
3	Check error queue	An error has occurred and the message has been placed in the error queue, but has not been read yet.
4	Message in output queue	A command has prepared information to be output, but it has not been read yet.
5	Check event-status register	One of the enabled bits in the event-status register has been set.
6	Request service	One of the enabled status-byte its is causing an SRQ.
7	Preset	An instrument preset has been executed.
Event-Status Register		
Bit	**Name**	**Definition**
0	Operation complete	A command for which OPC has been enabled has completed operation.
1	Request control	The analyzer has been commanded to perform an operation that requires control of a peripheral, and needs control of HP-IB. Requires pass-control mode.
2	Query error	The analyzer has been addressed to talk but there is nothing in the output queue to transmit.
3	Sequence Bit	A sequence has executed the assert SRQ command.
4	Execution error	A command was received that could not be executed.
5	Syntax error	The incoming HP-IB commands contained a syntax error. The syntax error is can only be cleared by a device clear or an instrument preset.
6	User request	The operator has pressed a front-panel key or turned the RPG.
7	Power on	A power-on sequence has occurred since the last read of the register.
Event-Status-Register B		
Bit	**Name**	**Definition**
0	Single sweep, number of groups, or calibration step complete	A single sweep, group, or calibration step has been completed since the last read of the register.
1	Service routine waiting or done	An internal service routine has completed operation, or is waiting for an operator response.
2	Data entry complete	A terminator key has been pressed or a value entered over HP-IB since the last read of the register.
3	Limit failed, Channel 2	Limit test failed on Channel 2.
4	Limit failed, Channel 1	Limit test failed on Channel 1.
5	Search failed, Channel 2	A marker search was executed on Channel 2, but the target value was not found.
6	Search failed, Channel 1	A marker search was executed on Channel 1, but the target value was not found.
7	Copy Complete	A copy has been completed since the last read of the register.

The Status Byte

The analyzer has a status-reporting mechanism that reports information about specific analyzer functions and events. The status byte (consisting of summary bits) is the top-level register. Each bit reflects the condition of another register or queue. If a summary bit is set (equals 1), the corresponding register or queue should be read to obtain the status information and clear the condition. Reading the status byte does not affect the state of the summary bits. The summary bits always reflect the condition of the summarized queue or register. The status byte can be read by a serial poll or by using the command OUTPSTAT. When using this command, the sequencing bit can be set by the operator during the execution of a test sequence. OUTPSTAT does not automatically put the instrument in remote mode, thus giving the operator access to the analyzer front-panel functions.

The status byte:

- summarizes the error queue
- summarizes two event-status registers that monitor specific conditions inside the instrument
- contains a bit that is set when the instrument is issuing a service request (SRQ) over HP-IB
- contains a bit that is set when the analyzer has data to transmit over HP-IB

Any bit in the status byte can be selectively enabled to generate a service request (SRQ) when set. Setting a bit in the service-request-enable register with the `SREnn;` executable enables the corresponding bit in the status byte. The units variable *nn* represents the binary equivalent of the bit in the status byte. For example, `SRE24;` enables status-byte bits 3 and 4 (since $2^3 + 2^4 = 24$) and disables all the other bits. SRE will not affect the state of the status-register bits.

The status byte also summarizes two queues: the output queue and the error queue. (The error queue is described in the next section.) When the analyzer outputs information, it puts the information in the output queue where it resides until the controller reads it. The output queue is only one event long. Therefore, the next output request will clear the current data. The summary bit is set whenever there is data in the output queue.

The Event-Status Register and Event-Status-Register B

The event-status register and event-status-register B are the other two registers in the status-reporting structure. They are selectively summarized by bits in the status byte via enable registers. The event-status registers consist of latched bits. A latched bit is set at the beginning of a specific trigger condition in the instrument. It can only be cleared by reading the register. The bit will not be reactivated until the condition occurs again. If a bit in one of these two registers is enabled, it is summarized by the summary bit in the status byte. The registers are enabled using the commands `ESEnn;` and `ESNBnn;`, both of which work in the same manner as `SREnn`. The units variable *nn* represents the binary equivalent of the bit in the status byte.

If a bit in one of the event-status registers is enabled, and therefore, summary bit in the status byte is enabled, an SRQ will be generated. The SRQ will not be cleared until one of the five following conditions transpire:

1. The event-status register is read, clearing the latched bit.
2. The summary bit in the status byte is disabled.
3. The event-status-register bit is disabled.
4. The status registers are cleared with the `CLES;` command.
5. An instrument preset is performed.

Service requests generated when there are error messages or when the instrument is waiting for the Group Execute Trigger (GET) command are cleared by:

- reading the errors
- issuing GET (disabling the bits)
- clearing the status registers

Error Output

When an error condition is detected in the analyzer, an message is generated, displayed on the analyzer's display screen, and placed in the error queue. Error messages consist of an error number followed by an ASCII string no more than 50-characters long. The string contains the same message that appears on the analyzer's display. The error queue holds up to 20 error messages in the order in which they occur. The error messages remain in the error queue until the errors are read by the system controller using the command OUTPERRO. The OUTPERRO command outputs one error message.

Note The error queue can only be cleared by performing an instrument preset or by cycling the line power. In order to keep the queue up-to-date, it is important to read all of the messages out of the queue each time errors are detected.

Measurement Process

This section explains how to organize instrument commands into a measurement sequence. A typical measurement sequence consists of the following steps:

1. setting up the instrument
2. calibrating the test setup
3. connecting the device under test
4. taking the measurement data
5. post-processing the measurement data
6. transfering the measurement data

Step 1. Setting Up the Instrument

Define the measurement by setting all of the basic measurement parameters. These include:

- the sweep type
- the frequency span
- the sweep time
- the number of points (in the data trace)
- the RF power level
- the type of measurement
- the IF averaging
- the IF bandwidth

You can quickly set up an entire instrument state, using the save/recall registers and the learn string. The learn string is a summary of the instrument state compacted into a string that the computer reads and retransmits to the analyzer. See "Example 5A: Using the Learn String."

Step 2. Calibrating the Test Setup

After you have defined an instrument state, you should perform a measurement calibration. Although it is not required, a measurement calibration improves the accuracy of your measurement data.

The following list describes several methods to calibrate the analyzer:

- Stop the program and perform a calibration from the analyzer's front panel.
- Use the computer to guide you through the calibration, as discussed in "Example 2A: S_{11} 1-Port Measurement Calibration" and "Example 2B: Full 2-Port Measurement Calibration."
- Transfer the calibration data from a previous calibration back into the analyzer, as discussed in "Example 5C: Saving and Restoring the Analyzer Instrument State."

Step 3. Connecting the Device under Test

After you connect your test device, you can use the computer to speed up any necessary device adjustments such as limit testing, bandwidth searches, and trace statistics.

Step 4. Taking the Measurement Data

Measure the device response and set the analyzer to hold the data. This captures the data on the analyzer display.

By using the single-sweep command (SING), you can insure a valid sweep. When you use this command, the analyzer completes all stimulus changes before starting the sweep, and does not release the HP-IB hold state until it has displayed the formatted trace. Then when the analyzer completes the sweep, the instrument is put into hold mode, freezing the data. Because single sweep is OPC-compatible, it is easy to determine when the sweep has been completed.

The number-of-groups command (NUMGn) triggers multiple sweeps. It is designed to work the same as single-sweep command. NUMGn is useful for making a measurement with an averaging factor n (n can be 1 to 999). Both the single-sweep and number-of-groups commands restart averaging.

Step 5. Post-Processing the Measurement Data

Figure 2-3 shows the process functions used to affect the data after you have made an error-corrected measurement. These process functions have parameters that can be adjusted to manipulate the error-corrected data prior to formatting. They do not affect the analyzer's data gathering. The most useful functions are trace statistics, marker searches, electrical-delay offset, time domain, and gating.

After Performing and activating a full 2-port measurement calibration, any of the four S-parameters may be viewed without taking a new sweep.

Step 6. Transferring the Measurement Data

Read your measurement results. All the data-output commands are designed to insure that the data transmitted reflects the current state of the instrument.

Data transfer is also discussed in Example 3.

Programming Examples

This section contains a description of the operations and the required commands for the following programming examples used to make device measurements with the network analyzer:

- Example 1: Measurement Setup
  - Example 1A: Setting Parameters
  - Example 1B: Verifying Parameters
- Example 2: Measurement Calibration
  - Example 2A: S_{11} 1-Port Measurement Calibration
  - Example 2B: Full 2-Port Measurement Calibration
- Example 3: Measurement Data Transfer
  - Example 3A: Data Transfer Using Markers
  - Example 3B: Data Transfer Using FORM 4 (ASCII Format)
  - Example 3C: Data Transfer Using Floating-Point Numbers
  - Example 3D: Data Transfer Using Frequency-Array Information
  - Example 3E: Data Transfer Using FORM 1 (Internal Binary Format)
- Example 4: Measurement Process Synchronization
  - Example 4A: Using the Error Queue
  - Example 4B: Generating Interrupts
  - Example 4C: Power Meter Calibration
- Example 5: Network Analyzer System Setups
  - Example 5A: Using the Learn String
  - Example 5B: Reading Calibration Data
  - Example 5C: Saving and Restoring the Analyzer Instrument State
- Example 6: Limit-Line Testing
  - Example 6A: Setting up a List-Frequency Sweep
  - Example 6B: Selecting a Single Segment from a Table of Segments
  - Example 6C: Setting Up Limit Lines
  - Example 6D: Performing PASS/FAIL Tests While Tuning
- Example 7: Report Generation
  - Example 7A1: Analyzer Operation Using the Talker/Listener Mode
  - Example 7A2: Controlling Peripherals with the Pass-Control Mode
  - Example 7A3: Printing with the Serial Port
  - Example 7B: Plotting to a File and Transferring the File Data to a Plotter
    - Utilizing PC-Graphics Applications Using the Plot File
  - Example 7C: Reading ASCII Disk Files to the Instrument Controller's Disk File

Program Information

The following information is provided for every example program included on the "HP 8753D Programming Examples" disk:

- A program description
- An outline of the program's processing sequence
- A step-by-step instrument-command-level tutorial explanation of the program including:
  - The command mnemonic and command name for the HP-IB instrument command used in the program.
  - An explanation of the operations and affects of the HP-IB instrument commands used in the program.

Note The HP BASIC programming code for each of these examples is contained in "HP BASIC Programming Examples."

Analyzer Features Helpful in Developing Programming Routines

Analyzer-Debug Mode

The analyzer-debug mode aids you in developing programming routines. The analyzer displays the commands being received. If a syntax error occurs, the analyzer displays the last buffer and points to the first character in the command line that it could not understand.

You can enable this mode from the front panel by pressing (LOCAL) HP-IB DIAG ON. The debug mode remains activated until you preset the analyzer or deactivate the mode. You can also enable this mode over the HP-IB using the `DEBUON;` executable and disable the debug mode using the `DEBUOFF;` executable.

User-Controllable Sweep

There are two important advantages to using the single-sweep mode:

1. The user can initiate the sweep.
2. The user can determine when the sweep has completed.
3. The user can be confident that the trace data has be derived from a valid sweep.

Execute the command string `OPC?;SING;` to place the analyzer in single-sweep mode and trigger a sweep. Once the sweep is complete, the analyzer returns an ASCII character one (1) to indicate the completion of the sweep.

Note The measurement cycle and the data acquisition cycle must always be synchronized. The analyzer must complete a measurement sweep for the data to be valid.

Example 1: Measurement Setup

Example 1A: Setting Parameters

Note This program is stored as EXAMP1A on the "HP 8753D Programming Examples" disk received with the network analyzer.

You can use the same command sequence for manually setting up measurements or remotely setting up measurements via HP-IB. There is no required order, as long as you set the desired frequency range, number of points, and power level prior to performing a measurement calibration.

This example sets the following parameters:

- reflection log magnitude on channel 1
- reflection phase on channel 2
- dual channel display mode
- frequency range from 100 MHz to 500 MHz

The following is an outline of the program's processing sequence:

- The system is initialized.
- The analyzer is adjusted to measure return loss on channel 1 and display it in log magnitude.
- The analyzer is adjusted to measure return loss on channel 2 and display the phase.
- The dual-channel display mode is activated.
- The system operator is prompted to enter the frequency range of the measurement.
- The displays are autoscaled.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 1A: Setting Parameters" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference" in this guide.

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing (PRESET) on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

`"CHAN1;"` **(Select Channel 1)**

Selects channel 1 as the active channel.

`"S11;"` **(Select Reflection)**

Selects a reflection-measurement mode.

`"LOGM;"` **(Select Log-Magnitude Display)**

Sets the data display to a log-magnitude format.

This command could be joined together with other commands in a single string of characters and sent to the analyzer in one operation as follows:

```
"CHAN1;S11;LOGM;"
```

Notice the use of semicolons as the individual command deliminators and terminators.

`"CHAN2;"` **(Select Active Channel 2)**

Selects channel 2 as the active channel.

`"S11;"` **(Select Reflection)**

Selects a reflection-measurement mode.

`"PHAS;"` **(Select Phase Display)**

Sets the analyzer to display phase data.

`"DUACON;"` **(Activate Dual-Channel Display)**

Displays both channels simultaneously in a dual-display configuration.

`ENTER START FREQUENCY (MHz):,F_start` **(Operator Prompt)**

Prompts the operator to enter the start frequency to define the beginning of the network analyzer's frequency span. `F_start` is the program's input variable for start frequency, input by the operator (in MHz).

`ENTER STOP FREQUENCY (MHz):,F_stop` **(Operator Prompt)**

Prompts the operator to enter the stop frequency to define the end of the network analyzer's frequency span. `F_stop` is the program's input variable for the stop frequency data, input by the operator (in MHz).

`"STAR";F_start;"MHZ"` **(Select Start Frequency in MHz)**

Inputs the operator-defined start frequency, using MHz as the basic units, to the network analyzer.

`"STOP";F_stop;"MHZ"` **(Select Stop Frequency in MHz)**

Inputs the operator-defined stop frequency, using MHz as the basic units, to the network analyzer. The selection of the analyzer's frequency span is now complete.

`"CHAN1;AUTO;"` **(Select Channel 1 and Autoscale)**

Selects channel 1 and autoscales the display to provide the optimum view of the measurement data.

`"CHAN2;AUTO;"` **(Select Channel 2 and Autoscale)**

Selects channel 2 and autoscales the display to provide the optimum view of the measurement data.

`"OPC?;WAIT;"` **(Operation Complete Query and Wait for Reply)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one, the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

The analyzer is initialized and the operator is queried for the measurement's start and stop frequencies. The analyzer is set up to display the S_{11} reflection measurement as a function of log magnitude and phase over the selected frequency range. The displays are autoscaled and the program ends.

Example 1B: Verifying Parameters

Note This program is stored as EXAMP1B on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example shows how to read analyzer settings into your program. The "Reading Analyzer Data" section, located earlier, contains information on the command formats and operations. Appending a question mark (?) to a command that sets an analyzer parameter will return the value of that setting from the analyzer to the controller. Parameters that are set as ON or OFF when queried will return a one (1) if active or a zero (0) if OFF . Parameters are returned in ASCII format, FORM 4. This format varies in length from 1 to 24 characters-per-value. In cases of marker or other multiple responses, these values are separated by commas.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The number of points in the trace is queried and printed on the controller display.
- The start frequency is queried and printed on the controller display.
- The averaging is queried and printed on the controller display.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 1B: Verifying Parameters" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing [PRESET] on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

`"POIN?;"` **(Number of Points Query)**

Queries the analyzer for the number of points in the measurement trace. The returned value is displayed on both the analyzer's and the controller's CRT.

`"STAR?;"` **(Start Frequency Query)**

Queries the start frequency. The returned value is displayed on both the analyzer's and the controller's CRT.

`"AVERO?;"` **(Averaging Status Query)**

Queries the averaging function to determine whether averaging is currently active. Notice that this is an ON or OFF command. The ON or OFF is replaced by a "?" to return the analyzer's averaging state to the controller.

The returned value is tested as a flag, and `"Averaging ON"` or `"Averaging OFF"` is displayed on the controller as a result of that test.

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one, the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

The analyzer is preset. The preset values are returned for: the number of points, the start frequency, and the state of the averaging function. The analyzer is released from remote control and the program ends.

Example 2: Measurement Calibration

This section shows you how to coordinate a measurement calibration over HP-IB. You can use the following sequence for performing either a manual measurement calibration, or a remote measurement calibration via HP-IB:

1. Select the calibration type.
2. Measure the calibration standards.
3. Declare the calibration done.

The actual sequence depends on the calibration kit and changes slightly for full 2-port measurement calibrations, which are divided into three calibration sub-sequences.

Calibration kits

The calibration kit tells the analyzer what standards to expect at each step of the calibration. The set of standards associated with a given calibration is termed a "class." For example, measuring the short during an S_{11} 1-port measurement calibration is one calibration step. All of the shorts that can be used for this calibration step make up the class, which is called class S11B. For the 7-mm and the 3.5-mm cal kits, class S11B uses only one standard. For type-N cal kits, class S11B contains two standards: male and female shorts.

When doing an S_{11} 1-port measurement calibration use a 7- or 3.5-mm calibration kit. Selecting SHORT automatically measures the short because the class contains only one standard. When doing the same calibration in type-N, selecting SHORT brings up a second menu, allowing the operator to select which standard in the class is to be measured. The sex listed refers to the test port: if the test port is female, then the operator selects the female short option.

Doing an S_{11} 1-port measurement calibration over HP-IB is very similar. When using a 7- or 3.5-mm calibration kit, sending CLASS11B will automatically measure the short. In type-N, sending CLASS11B brings up the menu with the male and female short options. To select a standard, use STANA or STANB. The STAN command is appended with the letters A through G, corresponding to the standards listed under softkeys 1 through 7, softkey 1 being the topmost softkey.

The STAN command is OPC-compatible. A command that calls a class is only OPC-compatible if that class has only one standard in it. If there is more than one standard in a class, the command that calls the class brings up another menu, and there is no need to query it.

Example 2A: S_{11} 1-Port Measurement Calibration

Note This program is stored as EXAMP2A on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example shows you how to coordinate an S_{11} 1-port measurement calibration over HP-IB, using the HP 85032B 50Ω type-N calibration kit.

1. Set up the desired instrument state.
2. Run the program.
3. Connect the standards as prompted.
4. Press ENTER on the controller keyboard to measure the standard.

Note Some computers may have a (RETURN) hardkey as the activating switch for carriage return/line feed. Throughout this chapter, the carriage-return/line-feed hardkey is represented by (ENTER).

Information on selecting calibration standards can be found earlier in the "Measurement Process" section.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The appropriate calibration kit is selected.
- The softkey menu is deactivated.
- The S_{11} 1-port measurement calibration sequence is run.
- The S_{11} 1-port measurement calibration data is saved.
- The softkey menu is activated.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 2A: S_{11} 1-Port Measurement Calibration" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

`"CALKN50;"` **(Select 50Ω Type-N Calibration Kit)**

Selects the 50Ω type-N calibration kit.

`"MENUOFF;"` **(Deactivate Softkey Menu)**

Deactivates the softkey menu.

`"CALIS111;"` **(Initiate S_{11} 1-Port Measurement Calibration)**

Begins the calibration cycle by selecting the S_{11} 1-port measurement calibration.

"CONNECT OPEN AT PORT 1" **(Operator Prompt)**

Prompts the operator to connect the first calibration standard, the open, at PORT 1.

CALL Waitforkey

Subroutine: Waitforkey:

Note In order to present a streamlined explanation of the example programs, subroutines and their associated commands are only discussed at the first point they are accessed in the program.

"PG;PU;PA390,3700;PD;LB;Lab$;", PRESS ENTER WHEN READYA;" **(Position and Display Text Commands and Operator Prompt)**

A subroutine is accessed to position and display the text prompt on the controller's display console. It is accessed every time the operator is required to press (ENTER). The variable A; is read and discarded. To continue the program (ENTER) is pressed.

" Press ENTER when ready" **(Operator Prompt)**

Prompts the operator to press (ENTER) on the controller keyboard after connecting the open at PORT 1. The controller reads the (ENTER) key-press and the calibration sequence continues.

"PG;" **(Page)**

Removes the last prompt message ("Press ENTER when ready") from the controller's display.

This is the last subroutine command and the subroutine ends.

Main Program Resumes

"CLASS11A;" **(Select Open Reflection Class)**

Measures the open. There is more than one standard in this loads class, so you must identify the specific standard within that class. The female open is the second softkey selection from the top in the menu.

"OPC?;STANB;" **(Select Standard B with Operation Complete Query)**

The command STANB selects a lowband load to be used as the standard. The analyzer returns an ASCII one (1) when the measurement is complete.

"CONNECT SHORT AT PORT 1" **(Operator Prompt)**

Prompts the operator to connect the next calibration standard, the short, at PORT 1.

CALL Waitforkey

"CLASS11B;" **(Select Short Reflection Class)**

Measures the short. There is more than one standard in this loads class, so you must identify the specific standard within that class. The female short is the second softkey selection from the top in the menu (choice "B").

`"OPC?;STANB;"` **(Select Standard B with Operation Complete Query)**

The command STANB selects a lowband load to be used as the standard. The analyzer returns an ASCII one (1) when the measurement is complete.

`"CONNECT LOAD AT PORT 1"` **(Operator Prompt)**

Prompts the operator to connect the 50Ω load to PORT 1. The program operation continues when you connect the load and press ENTER on the controller keyboard.

CALL `Waitforkey`

`"OPC?;CLASS11C;"` **(Select Load Reflection with Operation Complete Query)**

Measures the load. There is only one standard choice in this class, so the CLASS command is OPC-compatible. Using the OPC? command causes the program to wait until the standard is measured before continuing. This is very important, because the prompt to connect the next standard must appear after the first standard is measured. An ASCII one (1) is returned once the analyzer has measured the load standard.

`"PG;"` **(Page)**

Removes the last prompt message (`"CONNECT LOAD AT PORT 1"`) from the controller's display.

`"COMPUTING CALIBRATION COEFFICIENTS"` **(Program Message)**

Indicates that the calibration constants are being calculated by the analyzer and put into the calibration arrays for use with the measurement.

`"DONE;"` **(Calibration Complete)**

Completes the calibration process.

`"OPC?;SAV1;"` **(Save 1-Port Measurement Calibration with Operation Complete Query)**

Saves the instrument state and resulting calibration arrays as a 1-port measurement calibration. The OPC? command creates a wait-for-completion condition until the analyzer has finished calculating and storing the calibration coefficients.

`"S11 1-Port measurement cal COMPLETED. CONNECT TEST DEVICE."` **(Prompt)**

Indicates that the calibration is complete.

`"MENUON;"` **(Activate Softkey Menu)**

Reactivates the analyzer's softkey menu.

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

This program assumes the following test port characteristics:

- 50 ohm
- type-N connectors
- female

The prompts appear just above the message line on the analyzer's and the controller's display. Pressing (ENTER) on the controller continues the program and measures the standard. The program displays a message when the measurement calibration is complete.

Example 2B: Full 2-Port Measurement Calibration

Note This program is stored as EXAMP2B on the "HP 8753D Programming Examples" disk received with the network analyzer.

The following example shows how to perform a full 2-port measurement calibration using the HP 85032B calibration kit. The main difference between this example and Example 2A is that the full 2-port measurement calibration process allows removal of both the forward- and reverse-error terms, so that all four S-parameters of the device under test can be measured. PORT 1 is a female test port and PORT 2 is a male test port.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The appropriate calibration kit is selected.
- The softkey menu is deactivated.
- The 2-port calibration sequence is run.
- The operator is prompted to choose or skip the isolation calibration.
- The softkey menu is activated.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 2B: Full 2-Port Measurement Calibration" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "HP-IB Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Reflection Portion of the Full 2-Port Measurement Calibration Program

`"CALKN50;"` **(Select 50Ω Type-N Calibration Kit)**

Selects the 50Ω type-N calibration kit.

`"MENUOFF;"` **(Deactivate Softkey Menu)**

Deactivates the softkey menu.

`"CALIFUL2;"` **(Initiate Full 2-Port Measurement Calibration)**

Begins the full 2-port measurement calibration cycle.

`"REFL;"` **(Select Reflection Calibration)**

Selects the reflection calibration subsequence first.

`"CONNECT OPEN AT PORT 1"` **(Operator Prompt)**

Prompts the operator to connect the first calibration standard, the open, at PORT 1.

CALL `Waitforkey`

Subroutine: `Waitforkey:`

Note In order to present a streamlined explanation of the example programs, subroutines and their associated commands are only discussed at the first point they are accessed in the program.

`"PG;PU;PA390,3700;PD;LB;Lab$;", PRESS ENTER WHEN READYA;"` **(Position and Display Text Commands and Operator Prompt)**

A subroutine is accessed to position and display the text prompt on the controller's display console. It is accessed every time the operator is required to press ENTER. The variable `A;` is read and discarded. To continue the program ENTER is pressed.

`" Press ENTER when ready"` **(Operator Prompt)**

Prompts the operator to press ENTER on the controller keyboard after connecting the open at PORT 1. The controller reads the ENTER key-press and the calibration sequence continues.

`"PG;"` **(Page)**

Removes the last prompt message (`"Press ENTER when ready"`) from the controller's display.

This is the last subroutine command and the **subroutine ends**.

Reflection Portion of the Full 2-Port Measurement Calibration Program Resumes

`"CLASS11A;"` **(Select Open Reflection Class)**

Measures the open. There is more than one standard in this loads class, so you must identify the specific standard within that class. The female open is the second softkey selection from the top in the menu.

`"OPC?;STANB;"` **(Select Standard B with Operation Complete Query)**

The command STANB selects a lowband load to be used as the standard. The analyzer returns an ASCII one (1) when the measurement is complete.

`"CONNECT SHORT AT PORT 1"` **(Operator Prompt)**

Prompts the operator to connect the next calibration standard, the short, at PORT 1.

CALL `Waitforkey`

`"CLASS11B;"` **(Select Short Reflection Class)**

Measures the short. There is more than one standard in this loads class, so you must identify the specific standard within that class. The female short is the second softkey selection from the top in the menu (choice "B").

`"OPC?;STANB;"` **(Select Standard B with Operation Complete Query)**

The command STANB selects a lowband load to be used as the standard. The analyzer returns an ASCII one (1) when the measurement is complete.

`"CONNECT LOAD AT PORT 1"` **(Operator Prompt)**

Prompts the operator to connect the 50Ω load to PORT 1. The program operation continues when you connect the load and press ENTER on the controller keyboard.

`"OPC?;CLASS11C;"` **(Select Load Reflection with Operation Complete Query)**

Measures the load. There is only one standard choice in this class, so the CLASS command is OPC-compatible. Using the OPC? command causes the program to wait until the standard is measured before continuing. This is very important, because the prompt to connect the next standard must appear after the first standard is measured. An ASCII one (1) is returned once the analyzer has measured the load standard.

`"CONNECT OPEN AT PORT 2"` **(Operator Prompt)**

Prompts the operator to connect the next calibration standard, the open, at PORT 2.

`"CLASS22A;"` **(Select Open Reflection Class)**

Measures the open. There is more than one standard in this loads class, so you must identify the specific standard within that class. The male open is the first softkey selection from the top in the menu.

"OPC?;STANA;" **(Select Standard A with Operation Complete Query)**

The command STANA selects a lowband load to be used as the standard. The analyzer returns an ASCII one (1) when the measurement is complete.

"CONNECT SHORT AT PORT 2" **(Operator Prompt)**

Prompts the operator to connect the next calibration standard, the short, at PORT 2.

"CLASS22B;" **(Select Short Reflection Class)**

Measures the short. There is more than one standard in this loads class, so you must identify the specific standard within that class. The male short is the first softkey selection from the top in the menu (choice "A").

"OPC?;STANA" **(Select Standard A with Operation Complete Query)**

The command STANA selects a lowband load to be used as the standard. The analyzer returns an ASCII one (1) when the measurement is complete.

"CONNECT LOAD AT PORT 2" **(Operator Prompt)**

Prompts the operator to connect the 50Ω load to PORT 2.

"OPC?;CLASS11C;" **(Select Load Reflection with Operation Complete Query)**

Measures the load. There is only one standard choice in this class, so the CLASS command is OPC-compatible. Using the OPC? command causes the program to wait until the standard is measured before continuing. This is very important, because the prompt to connect the next standard must appear after the first standard is measured. An ASCII one (1) is returned once the analyzer has measured the load standard.

"COMPUTING REFLECTION CALIBRATION COEFFICIENTS" **(Program Comment)**

Indicates that the reflection calibration coefficients are being calculated by the analyzer.

"REFD;" **(Reflection Calibration Done)**

Completes the reflection cycle and the ends the reflection calibration routine.

The analyzer now computes the reflection calibration coefficients for the calibration arrays.

Transmission Portion of the Full 2-Port Measurement Calibration Program Resumes

"TRAN;" **(Select Transmission Calibration)**

Begins the transmission calibration cycle.

"CONNECT THRU [PORT 1 TO PORT 2]" **(Operator Prompt)**

Prompts the operator to make a thru connection between PORT 1 and PORT 2.

"MEASURING FORWARD TRANSMISSION" **(Program Comment)**

Displays a message on the controller console showing that the forward transmission is being measured.

`"OPC?;FWDT;"` **(Select FWD Transmission with Operation Complete Query)**

Measures the forward transmission. A one (1) is received from the analyzer after the measurement is complete.

`"OPC?;FWDM;"` **(Select FWD Trans Load Match with Operation Complete Query)**

Measures the forward load match. Again, a one (1) is received when the analyzer completes the measurement.

`"MEASURING REVERSE TRANSMISSION"` **(Program Comment)**

The prompt is displayed on the controller console informing the operator that the reverse transmission measurements are taking place.

`"OPC?;REVT;"` **(Select Reverse Transmission with Operation Complete Query)**

Measures the reverse transmission. A one (1) is received from the analyzer after the measurement is complete.

`"OPC?;REVM;"` **(Select REV Trans Load Match with Operation Complete Query)**

Measures the reverse load match. Again, a one (1) is received when the analyzer completes the measurement.

`"TRAD;"` **(Transmission Calibration Done)**

Completes the transmission cycle and the ends the transmission-calibration routine.

Isolation Portion of the Full 2-Port Measurement Calibration Program Resumes

`"SKIP ISOLATION CAL? Y OR N."` **(Operator Prompt)**

Prompts the operator to choose whether to perform the isolation-calibration step of the full 2-port measurement calibration cycle. The response is read into a string and tested.

- If the operator responds by entering `"Y"`, the isolation cycle is skipped with the omit-isolation command executable `"OMII;"`. The program then branches to the isolation-done command at the end of the isolation tests.
- If the operator responds with any other character to the prompt, the isolation tests are performed.

`"ISOLATE TEST PORTS"` **(Operator Prompt)**

Prompts the operator to connect loads to the test ports in preparation for the isolation measurement.

`"ISOL;"` **(Select Isolation Calibration)**

Selects the reverse-isolation calibration.

`"AVERFACT10;"` **(Select Averaging Factor {10})**

Selects averaging for ten sweeps.

"AVEROON;" **(Activate Averaging)**

Activates the averaging function.

"MEASURING REVERSE ISOLATION" **(Program Comment)**

Displays "MEASURING REVERSE ISOLATION" on the controller's console.

"OPC?;REVI;" **(Select Reverse Isolation with Operation Complete Query)**

Performs a reverse-isolation calibration. Ten sweeps are taken and measured. The noise level is averaged to obtain the best test results. A one (1) is returned from the analyzer to indicate the completion of the operation.

"MEASURING FORWARD ISOLATION" **(Program Comment)**

Displays "MEASURING FORWARD ISOLATION" on the controller's console.

"OPC?;FWDI;" **(Select Forward Isolation with Operation Complete Query)**

Measures the forward isolation. A one (1) is returned from the analyzer to indicate the completion of the operation.

"ISOD;AVEROOFF;" **(Isolation Calibration Done and Deactivate Averaging)**

This completes the isolation measurements and the isolation calibration cycle ends.

"PG;" **(Page)**

Removes the last prompt message ("MEASURING FORWARD ISOLATION") from the controller's display.

"COMPUTING CALIBRATION COEFFICIENTS" **(Program Comment)**

Displays "COMPUTING CALIBRATION COEFFICIENTS" on the controller's console.

"DONE;" **(Completed)**

Completes the calibration process.

"OPC?;SAV2;" **(Save Full 2-Port Measurement Calibration with Operation Complete Query)**

Saves the instrument state and resulting calibration arrays as a 2-port measurement calibration. The OPC? command creates a wait-for-completion condition until the analyzer responds that it has finished calculating and storing the calibration coefficients in its internal registers.

"DONE WITH FULL 2-PORT CAL. CONNECT TEST DEVICE." **(Operator Prompt)**

Displays the final message on the controller console to alert the operator that the full 2-port measurement calibration process is complete.

"MENUON;" **(Activate Softkey Menu)**

Reactivates the analyzer's softkey menu.

`"OPC?;WAIT;"` (Wait for Reply with Operation Complete Query)

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

The program assumes the following situation:

- test ports are type-N
- PORT 1 is a female test port
- PORT 2 is a male test port
- HP 85032B 50Ω type-N calibration kit is used

The prompts appear just above the message line on the analyzer's display as well as the computer's console. Pressing (ENTER) on the computer keyboard continues the program and measures the standard. You have the option of omitting the isolation portion of the measurement calibration. If you perform the isolation portion of the calibration, averaging is automatically employed to insure a valid calibration. The program will display a message when the measurement calibration is complete.

Example 3: Measurement Data Transfer

There are two methods that can be used to read trace information from the analyzer:

- selectively, using the trace markers
- completely, using the trace-data array

If only specific information (such as a single point on the trace or the result of a marker search) is required, the marker output command can be used to read the information. If all of the trace data is required, see Examples 3B through 3E for examples of the various formats available.

Trace-Data Formats and Transfers

Refer to Table 2-4. This table shows the number of bytes required to transfer a 201-point trace in the different formats. As you will see in the first example (FORM 4), ASCII data is the easiest to transfer, but the most time consuming due to the number of bytes in the trace. If you are using a PC-based controller, a more suitable format would be FORM 5. To use any trace data format other than FORM 4 (ASCII data) requires some care in transferring the data to the computer. Data types must be matched to read the bytes from the analyzer directly in to the variable array. The computer must be told to stop formatting the incoming data and treat it as a binary-data transfer. All of the other data formats also have a four byte header to deal with. The first two bytes are the ASCII characters "#A" that indicate that a fixed length block transfer follows, and the next two bytes form an integer containing the number of bytes in the block to follow. The header must be read in to separate the header from the rest of the block data to be mapped into an array. "Array-Data Formats," located earlier in this chapter, discusses the different types of formats and their compositions.

Data may also be transferred from several different locations in the trace-processing chain. These examples will illustrate formatted-data transfers, but other locations in the trace-data processing chains may be accessed. See Figure 2-3.

In this section, an example of each of the data formats will be shown for comparison. A general rule of thumb is to use FORM 1 (internal binary format) for traces that are not being utilized for data content. Learn strings, state transfers, and calibration data that are being transferred to a file and back are good examples. See Example 3D.

Arrays which will be interpreted or processed within your program should be in FORM 2, 3 or 5, whichever is appropriate for your computer. Example 3C shows how to transfer a trace in these formats.

In Examples 3B and 3C, the frequency counterpart of each data point in the array is also determined. Many applications generate a frequency and magnitude, or a phase array for the test results. Such data may be required for other data processing applications (such as comparing data from other measurements).

In Example 3B, the frequency data is constructed from the frequency span information. Alternatively, it is possible to read the frequencies directly out of the instrument with the OUTPLIML command. OUTPLIML reports the limit-test results by transmitting the stimulus point tested, a number indicating the limit-test results, and then the upper and lower limits at that stimulus point (if available). The number indicating the limit results is a −1 for no test, 0 for fail, and 1 for pass. If there are no limits available, the analyzer transmits zeros. For this example, we delete the limit test information and keep the stimulus information.

In Example 3C, the limit-test array is read into the controller and used to provide the values directly from the analyzer memory. Reading from the limit-test array is convenient, although it outputs the results in ASCII format (form 4), which may be slow. If there is no other way to obtain the frequency data, this transfer time may be acceptable. Frequency information becomes more difficult to determine when not using the linear sweep mode. Log-frequency

sweeps and list-frequency sweeps have quite different values for each data point. For these special cases, the additional time spent reading out the limit test results is an acceptable solution for obtaining the valid frequency information for each data point in the trace.

Example 3A: Data Transfer Using Markers

Note This program is stored as EXAMP3A on the "HP 8753D Programming Examples" disk received with the network analyzer.

Markers are the simplest form of trace-data transfer. You can position a marker in the following locations on the trace:

- a frequency location
- an actual data point location
- a trace data value

The marker data is always returned in FORM 3, ASCII format. Each number is sent as a 24-character string; each character being a digit, sign, or decimal point. In the case of markers, three numbers are sent. The display format determines the values of the marker responses. See Table 2-4 for additional information.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The selected frequency span is swept once.
- The marker is activated and placed on the maximum trace value.
- The three marker values (value 1, value 2, and stimulus) are output to the controller and displayed.
- The instrument is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 3A: Data Transfer Using Markers" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing (PRESET) on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Places the analyzer in single-sweep mode and activates a sweep. The analyzer returns a one (1) at the completion of the sweep.

`"MARK1;"` **(Select Active Marker {1})**

Marker 1 is selected and activated.

`"SEAMAX;"` **(Search Maximum)**

The marker is placed at the maximum data value on the trace.

`"OUTPMARK;"` **(Output Marker)**

Outputs the active marker values in 3 numbers. The first two numbers are the marker values, and the last is the stimulus value. The data values are read from the marker and output to the controller.

The analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Execute the program. After performing an instrument preset, the analyzer switches into the log-magnitude mode and measures the S_{11} reflection values of the device under test.

The three values returned to the controller are:

1. reflection, in dB
2. a non-significant value
3. the stimulus frequency at the maximum point

A non-significant value means that the analyzer returned a value that is meaningless in this data format.

Table 2-3, located earlier in this chapter, provides an easy reference for the types of data returned with the various data-format operational modes.

Example 3B: Data Transfer Using FORM 4 (ASCII Transfer)

Note This program is stored as EXAMP3B on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example shows you how to transfer a trace array from the analyzer using FORM 4, an ASCII data transfer.

Table 2-3 shows the relationship of the two values-per-point that are transferred to the analyzer. When FORM 4 is used, each number is sent as a 24-character string (each character represented by a digit, sign, or decimal point). Since there are two numbers-per-point, plus a comma and line-feed, a 201-point transfer in FORM 4 takes 10,050 bytes. This form is useful only when input-data formatting is difficult with the instrument controller. Refer to Table 2-4 for a comparison with the other formats.

Another example is included with the ASCII data transfer. A fairly common requirement is to create frequency-amplitude data pairs from the trace data. No frequency information is included with the trace-data transfer. Relating the data from a linear frequency sweep to frequency can be done by interrogating the analyzer start frequency, the frequency span, and the number of points. Given that information, the frequency of point n in a linear frequency sweep is defined by the equation:

$$F = \text{Start frequency} + (n-1) \times \text{Span}/(\text{Points}-1)$$

This example illustrates the technique of generating corresponding frequency data. This is a straight-forward solution for linear uniform sweeps. For other sweep types, frequency data is more difficult to construct and may be best read from the analyzer directly from the limit-test array. See "Example 3D: Data Transfer Using Frequency Array Information" for an example of this technique.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The trace-data array is allocated.
- The trace length is set to 11.
- The selected frequency span is swept once.
- The FORM 4, ASCII format is set.
- The formatted trace is read from the analyzer.
- The frequency increments between the points are calculated.
- The marker is activated and placed at 30 kHz.
- The instrument is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 3B: Data Transfer Using FORM 4 (ASCII Transfer)" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing (PRESET) on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

`"POIN 11;"` **(Edit Segment Points {11})**

Sets the number of points in the trace to 11.

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Places the analyzer in single-sweep mode and activates a sweep. The analyzer returns a one (1) at the completion of the sweep.

`"FORM4;"` **(Select Array-Data Format 4)**

Sets array-data format 4, ASCII floating-point format.

`"OUTPFORM;"` **(Output Formatted Trace-Data Array from Active Channel)**

Outputs the formatted trace data from the active channel in current display units. The analyzer's trace data is output as ordered pairs separated by commas and terminated with a line-feed character for each trace-data point. The trace-data array is then read by the controller.

`"POIN?;"` **(Number of Points Query)**

Queries the analyzer for the number of points in the measurement trace. The returned value is displayed on both the analyzer's and the controller's CRT.

`"STAR?;"` **(Start Frequency Query)**

Queries the start frequency. The returned value is displayed on both the analyzer's and the controller's CRT.

`"SPAN?;"` **(Frequency Span Query)**

Queries the frequency span. The returned value is displayed on both the analyzer's and the controller's CRT.

`F_inc=Span/(Num_points-1)` **(Fixed-Frequency Increment Equation)**

The program calculates the frequency increments between data points. The values are then displayed on the controller's display console. The data is then formatted for display on the controller's console. The program loops through the data points and calculates the frequency values for each point on the 11-point trace, as defined by `"POIN 11;"`. The display-formatted frequency-value data is then shown on the controller's display console. The program then exits the loop and continues to the next command line.

Note In this format, the second value in the array is not significant. See Table 2-4.

`"MARKDISC;"` **(Select Discrete-Marker Mode)**

Sets the analyzer to display discrete-data points with the marker. In discrete-marker mode, the trace data is not interpolated between points. This allows the operator to easily identify and view the datapoints in the trace.

`"MARK1 .3E+6;"` **(Select Active Marker 1 and position at 30 kHz)**

Positions marker 1 at 30 KHz (the left-hand edge of the display).

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB.

`"Position the marker with the knob and compare values"` **(Operator Prompt)**

The operator is prompted to position the marker with the RPG knob along the points in the trace and the program ends.

Running the Program

Run the program and watch the controller console. The analyzer will perform an instrument preset. The program will then print out the data values received from the analyzer. The marker is activated and placed at the left-hand edge of the analyzer display. Position the marker with the knob and compare the values read with the active marker with the results printed on the controller console. The data points should agree exactly. Keep in mind that no matter how many digits are displayed, the analyzer is specified to measure:

- magnitude to a resolution of 0.001 dB
- phase to a resolution of 0.01 degrees
- group delay to a resolution of 0.01 ps

Changing the display format will change the data sent with the OUTPFORM transfer. See Table 2-4 for a list of the specific data that is provided with each format. The data from OUTPFORM reflects all the post processing such as:

- time domain
- gating
- electrical delay
- trace math
- smoothing

Note Note that if time domain (option 010 only) is activated, operation is limited to 201 points in the lowpass mode.

Example 3C: Data Transfer Using Floating-Point Numbers

Note This program is stored as EXAMP3C on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example program illustrates data transfer using FORM 3 in which data is transmitted in the floating-point formats. FORM 2 is nearly identical except for the IEEE 32-bit format of 4 bytes-per-value. FORM 5 reverses the order of the bytes to conform with the PC conventions for defining a real number.

The block-data formats have a four-byte header. The first two bytes are the ASCII characters `#A` that indicate that a fixed-length block transfer follows, and the next two bytes form an integer containing the number of bytes in the block to follow. The header must be read in so that data order is maintained.

This transfer is more than twice as fast than a FORM 4 transfer. With the FORM 4 transfer, 10,050 bytes are sent (201 points × 2 values-per-point × 24 bytes-per-value). Using FORM 2 to transfer the data, only 1612 bytes are sent (201 points × 2 values-per-point × 4 bytes-per-value). See "Array-Data Formats."

The following is an outline of the program's processing sequence:

- The system is initialized.
- The integer variables are defined to contain the header information.
- The number of points in the trace is set to 11.
- The selected frequency span is swept once.
- Data-transfer format 3 is set.
- The headers are read from the trace.
- The array size is calculated and allocated.
- The trace data is read in and printed on the controller display.
- The marker is activated and placed at 30 kHz.
- The instrument is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 3C: Data Transfer Using Floating-Point Numbers" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB

instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing [PRESET] on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

First, two integers are defined to hold the header information. Since an integer takes two bytes, the header and length variables will form the entire four-byte header.

The program variable `Numpoints` is defined as 11. The value "11" will be used in the next command line to set the number of points in the trace.

The program command line `"POIN";Numpoints;";"` edits the number of points in the trace using the value (11) for the variable `Numpoints`. This operation is virtually identical to executing the `"POIN 11;"` command used in Example 3B. Both operations set the number of points in the trace to 11.

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Places the analyzer in single-sweep mode and activates a sweep. The analyzer returns a one (1) at the completion of the sweep.

`"FORM3;"` **(Select Array-Data Format 3)**

Selects array-data format 3. Data will now be output to the controller, formatted as a 64-bit floating point, with header.

`"OUTPFORM;"` **(Output Formatted Data Array from Active Channel)**

Outputs the formatted trace data from the active channel in current display units.

The header for the trace is read into the header- and length-integer variables. A data array is allocated to the size of the length statement. Using the length variable allows dynamic trace-array allocation. This can be very useful when using varying-length traces in your application.

The trace data is then input to the controller. The array size as well as the number of bytes contained in the array are displayed on the controller's display console.

The data points are then looped and displayed on the controller's display.

`"MARKDISC;"` **(Select Discrete-Marker Mode)**

Sets the analyzer to display discrete-data points with the marker. In discrete-marker mode, the trace data is not interpolated between points. This allows the operator to easily identify and view the datapoints in the trace.

`"MARK1 .3E+6;"` **(Select Active Marker 1 and position at 30 kHz)**

Activates marker 1 and positions it at 30 KHz (the left-hand edge of the display).

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB.

`"Position the marker with the knob and compare values"` **(Operator Prompt)**

The operator is prompted to position the marker with the RPG knob along the points in the trace and the program ends.

Running the Program

Run the program. The computer displays the number of elements and bytes associated with the transfer of the trace, as well as the first 10 data points. Position the marker and examine the data values. Compare the received values with the analyzer's marker values.

Example 3D: Data Transfer Using Frequency Array Information

Note This program is stored as EXAMP3D on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example explains how to use the limit-test array to read the corresponding frequency values for the completed trace array into the controller. The analyzer is set to sweep from 10 MHz to 200 MHz in log-frequency mode with the number of points in the trace set to 11. This makes it very difficult to compute the frequency-point spacing in the trace. The points are equally spaced across the trace, but not equally spaced in relation to frequency (because the frequency span is displayed in a logarithmic scale, as opposed to a linear scale). The limit-test data array may be read from the analyzer to provide the frequency values for each data point. Four values are read for each data point on the analyzer. The test results and limit values are not used in this example. Only the frequency values are used. This technique is the only method of obtaining the non-linear frequency data from the analyzer display. The test data and frequencies are printed on the controller display and the marker is enabled to allow the operator to examine the actual locations on the analyzer display.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The integer variables for the header information are defined.
- The number of points in the trace is set to 11.
- The frequency span (10 MHz to 200 MHz) is selected.
- The log-frequency sweep is selected.
- The data-transfer format 3 is set.
- The headers are read from the trace.
- The array size is calculated and allocated.
- The trace data is read in.
- The limit-test array is calculated and allocated.
- The limit-line test array is read in.
- The table header is printed.
- The program cycles through the trace values.
- The trace data and frequency are printed.
- The discrete-marker mode is activated.
- The marker is activated and placed at 10 MHz.
- The instrument is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 3D: Data Transfer Using Frequency Array Information" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing [PRESET] on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

First, two integers are defined to hold the header information. Since an integer takes two bytes, the header and length variables will form the entire four-byte header.

The program variable `Numpoints` is defined as 11. The value "11" will be used in the next command line to set the number of points in the trace.

The program command line `"POIN";Numpoints;";"` edits the number of points in the trace using the value (11) for the variable `Numpoints`. This operation is virtually identical the executing a `"POIN 11;"` command used in Example 3B. Both operations set the number of points in the trace to 11.

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Places the analyzer in single-sweep mode and activates a sweep. The analyzer returns a one (1) at the completion of the sweep.

`"FORM3;"` **(Select Array-Data Format 3)**

Selects array-data format 3. Data will now be output to the controller formatted as a 64-bit floating point, with header.

`"POIN 11;"` **(Edit Segment Points {11})**

Sets the number of points in the trace to 11.

`"STAR 10.E+6;"` **(Select Start Frequency as 10 MHz)**

A start frequency of 10 MHz is selected.

`"STOP 200.E+6;"` **(Select Stop Frequency as 200 MHz)**

A stop frequency of 200 MHz is selected. This frequency range allows 2 cycles of log frequency scaling on the display.

`"LOGFREQ;"` **(Select Log Frequency Sweep)**

Selects a log frequency sweep.

`"FORM3;"` **(Select Array-Data Format 3)**

Selects array-data format 3. Data will now be output to the controller formatted as a 64-bit floating point, with header.

`"OUTPFORM;"` **(Output Formatted Data Array from Active Channel)**

Outputs the formatted trace data from the active channel in current display units.

The header for the trace is read into the header- and length-integer variables. A data array is allocated to the size of the length statement. Using the length variable allows dynamic trace array allocation, which is very useful when using varying-length traces in your application.

The trace data is then input to the controller.

Next, a limit-line-results array is allocated in the controller.

`"OUTPLIML;"` **(Output Limit-Line Test Results)**

Outputs the limit test results for each stimulus point. The results consist of four numbers. The first number represents the stimulus value tested. The second represents the test result: −1 for no test, 0 for fail, 1 for pass. The third number is the upper limit value, and the fourth is the lower limit value.

One shortcoming of this technique is that the array data is transferred in FORM 4, ASCII format. Although all the values are transferred, this example is only concerned with the frequency values.

As the program continues, a table heading with columns titled "Freq (MHz)" and "Mag (dB)" is printed on the controller's display. The frequency and data-point values are cycled through and the resulting values are printed beneath their respective column headings on the controller display.

`"MARKDISC;"` **(Select Discrete-Marker Mode)**

Sets the analyzer to display discrete-data points with the marker. In discrete-marker mode, the trace data is not interpolated between points. This allows the operator to easily identify and view the datapoints in the trace.

`"MARK1 10.E+6;"` **(Select Active Marker 1 and position at 10 MHz)**

Activates marker 1 and positions it at 10 MHz (at the left side of the analyzer display).

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB.

`"Position the marker with the knob and compare values"` **(Operator Prompt)**

The operator is prompted to position the marker with the RPG knob along the points in the trace and the program ends.

Running the Program

Run the program. Observe the controller display. The corresponding frequency values are shown with the trace-data values. Position the marker and observe the relationship between the frequency values and the point spacing on the trace. Compare the trace-data values on the analyzer with those received.

Example 3E: Data Transfer Using FORM 1 (Internal Binary Format)

Note This program is stored as EXAMP3E on the "HP 8753D Programming Examples" disk received with the network analyzer.

FORM 1 is used for rapid I/O transfer of analyzer data. It contains the least number of bytes-per-trace and does not require re-formatting in the analyzer. This format is more difficult to convert into a numeric array in the controller. Analyzer-state information, such as learn strings and calibration arrays, may be easily transferred in this format because data conversion is not required. Recalling an instrument state that has been stored in a file and transferring instrument-state information to the analyzer are excellent applications of a FORM 1 data transfer.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The integer variables for the header information are defined.
- The string variable for the header is defined.
- The selected frequency span is swept once.
- The internal-binary format is selected.
- The error-corrected data is output from the analyzer.
- The two data-header characters and the two length bytes are read in.
- The string buffer is allocated for data.
- The trace data is read into the string buffer.
- The analyzer is restored to continuous-sweep mode and queried for command completion.
- The instrument is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 3E: Data Transfer Using FORM 1 (Internal Binary Format)" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing [PRESET] on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

First, two integers are defined to hold the header information. Since an integer takes two bytes, the header and length variables will form the entire four-byte header.

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Places the analyzer in single-sweep mode and activates a sweep. The analyzer returns a one (1) at the completion of the sweep.

`"FORM1;"` **(Select Array-Data Format 1)**

Selects array-data format 1. Data will now be output to the controller formatted in the analyzer internal format, with header.

`"OUTPDATA;"` **(Output Error-Corrected Data)**

Outputs the error-corrected data from the active channel in real/imaginary pairs.

In this case, the data is not converted to trace offsets and conversions. The trace data is read as error-corrected only. See Figure 2-3. The first two bytes of the header are read as a string and placed in the variable header. The next two bytes are read into an integer variable which is the length of the data block to follow. The header value and the number of bytes are displayed on the controller console. The block is allocated and read in as a long string. Once the transfer is complete, the length of the data string received is compared to the header value.

`"CONT;"` **(Select Continuous-Sweep Mode)**

The analyzer is restored to continuous-sweep mode.

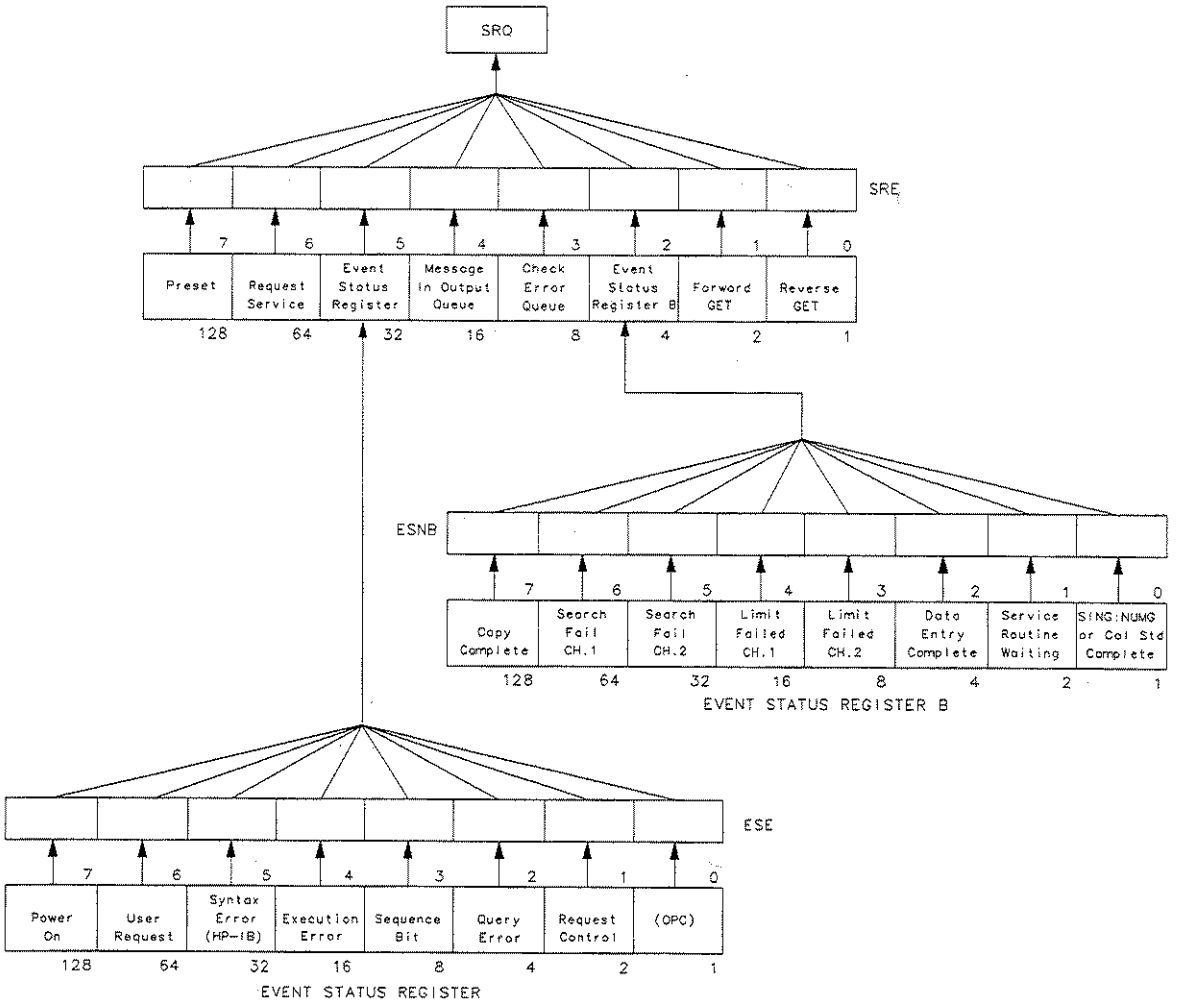
`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Run the program. The analyzer is initialized. The header and the number of bytes in the block transfer are printed on the controller display. Once the transfer is complete, the number of bytes in the data string is printed. Compare the two numbers to be sure that the transfer was completed.

Example 4: Measurement Process Synchronization



pg6151d

Figure 2-5. Status Reporting Structure

Status Reporting

The analyzer has a status reporting mechanism, illustrated in Figure 2-5, that provides information about specific analyzer functions and events. The status byte is an 8-bit register with each bit summarizing the state of one aspect of the instrument. For example, the error queue summary bit will always be set if there are any errors in the queue. The value of the status byte can be read with the HP-IB serial poll operation. This command does not automatically put the instrument in remote mode, which gives you access to the analyzer front-panel functions. The status byte can also be read by sending the command OUTPSTAT. Reading the status byte does not affect its value.

The status byte summarizes the error queue, as mentioned before. It also summarizes two event-status registers that monitor specific conditions inside the instrument. The status byte also has a bit (6) that is set when the instrument is issuing a service request over HP-IB, and

a bit (0) that is set in the event-status register when the analyzer has data to send out over HP-IB. See "The Event-Status Registers," located earlier in this chapter, for a discussion of the event-status registers.

Example 4A: Using the Error Queue

Note This program is stored as EXAMP4A on the "HP 8753D Programming Examples" disk received with the network analyzer.

The error queue holds up to 20 instrument errors and warnings in the order that they occurred. Each time the analyzer detects an error condition, the analyzer displays a message on the CRT, and puts the error in the error queue. If there are any errors in the queue, bit 3 of the status byte will be set. The errors can be read from the queue with the OUTPERRO command. OUTPERRO causes the analyzer to transmit the error number and message of the oldest error in the queue.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The error-message string is allocated.
- The analyzer is released from remote control.
- The program begins an endless loop to read the error queue.
- The status byte is read with a serial poll.
- The program tests to see if an error is present in the queue.
- The error-queue bit is set.
- The program requests the content of the error queue.
- The error number and string are read.
- The error messages are printed until there are no more errors in the queue.
- The instrument is returned to local control.
- The controller emits a beep to attach the attention of the operator and resumes searching for errors.

Tutorial Program Explanation

The following section explains "Example 4A: Using the Error Queue" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB

instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing (PRESET) on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

An error-message string variable is defined to contain the error messages from the analyzer.

Next, the analyzer is returned to local control. The program then reads the error queue in an endless loop waiting to detect an error message.

The status byte for the analyzer is read by the serial-poll command until it finds that the error-queue summary bit is set. Serial poll is an HP-IB function dedicated specifically to reading the status byte of an instrument quickly, and does not cause the analyzer to go into remote operation. If the bit is not set, the program will loop until the error bit gets set.

Once the bit is set (the error queue contains an error), the analyzer is instructed to output the error number and the error message.

`"OUTPERRO;"` **(Output Error Queue Contents)**

Outputs the oldest error in the error queue. First, the error number is transmitted, then the error message, both in ASCII format. This data string returns the analyzer to remote mode and the controller then reads the error message(s) from the analyzer. The analyzer returns the error number (2) and the error message(s) that appear(s) on the analyzer display. The error queue is read until the error number is zero, which states that there are no more errors in the error queue.

These values are read into the controller and displayed on the controller console.

The analyzer is addressed and returned to local control with the HP-IB Meta- Message: Go To Local (GTL) on the HP-IB. The front-panel functions are now accessible and the controller gives an audible signal that there is a problem.

Afterwards, the program returns to looping and reading the status byte to look for another error message.

Running the Program

Run the program. The analyzer goes through the preset cycle. Nothing will happen at first. The program is waiting for an error condition to activate the error queue. To cause an error, press a blank softkey. The message `CAUTION: INVALID KEY` will appear on the analyzer's display. The computer will beep and print two error messages. The first line will be the invalid key error message, and the second line will be the `NO ERRORS` message. To clear the error queue, you can either loop until the `NO ERRORS` message is received, or until the bit in the status register is cleared. In this case, we wait until the status bit in the status register is clear. Note that while the program is running, the analyzer remains in the local mode and the front-panel keys may be accessed.

The error queue will hold up to 20 errors until all the errors are read out or the instrument is preset. It is important to clear the error queue whenever errors are detected. Otherwise, old errors may be mistakenly associated with the current instrument state.

Press [SYSTEM] and then the unlabeled key several times quickly and watch the display. The number of errors observed should correspond to the number of times you pressed the key.

As another example, press [CAL]. Then the CALIBRATE MENU key. Select the RESPONSE calibration. Press DONE RESPONSE without performing any calibrations. Note the error message on the analyzer and on the controller display. Push the THROUGH key and then DONE RESPONSE. We are not concerned with the validity of the calibration, just setting a simple calibration on the analyzer. Note that COR is displayed in the upper left-hand section of the graticule. Now, press START and [↑]. This will generate an error because the start frequency has been changed, invalidating the calibration. This error is reported on the controller display as well. A complete list of error messages and their descriptions can be found in "Error Messages" of the *HP 8753D Network Analyzer User's Guide.*

The program is in an infinite loop waiting for errors to occur. End the program by pressing [RESET] or [BREAK] on the controller keyboard.

Note Not all messages displayed by the analyzer are put in the error queue: operator prompts and cautions are not included.

Example 4B: Generating Interrupts

Note This program is stored as EXAMP4B on the "HP 8753D Programming Examples" disk received with the network analyzer.

Interrupts can be generated using the status-reporting mechanism. The status-byte bits can be enabled to generate a service request (SRQ) when set. In turn, the instrument controller can be set up to generate an interrupt on the SRQ and respond to the condition which caused the SRQ.

To generate an SRQ, a bit in the status byte is enabled using the SREn; command. A one (1) in a bit position enables that bit in the status byte. Hence, executing SRE 8; enables an SRQ on bit 3, the check-error queue, since the decimal value 8 equals 00001000 in binary representation. Whenever an error is put into the error queue and bit 3 is set, the SRQ line is asserted, illuminating the (S) indicator in the HP-IB status block on the front panel of the analyzer. The only way to clear the SRQ is to disable bit 3, re-enable bit 3, or read out all the errors from the queue.

A bit in the event-status register can be enabled so that it is summarized by bit 5 of the status byte. If any enabled bit in the event-status register is set, bit 5 of the status byte will also be set. For example, executing ESE 66; enables bits 1 and 6 of the event-status register, since in binary, the decimal number 66 equals 01000010. Hence, whenever active control is requested or a front-panel key is pressed, bit 5 of the status byte will be set. Similarly, executing ESNBn; enables bits in event-status-register B so that they will be summarized by bit 2 in the status byte.

To generate an SRQ from an event-status register, enable the desired event-status-register bit. Then enable the status byte to generate an SRQ. For instance, executing ESE 32;SRE 32; enables the syntax-error bit. When the syntax-error bit is set, the summary bit in the status byte will be set. This will, in turn, enable an SRQ on bit 5 of the status byte, the summary bit for the event-status register.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The status registers are cleared.

- The event-status register bit 5 is enabled.
- The status-register bit 5 is enabled.
- The interrupt pointer is enabled and points to a subroutine.
- Two bad commands are set to the analyzer to generate errors.
- The controller reads a serial-poll byte from HP-IB in the event of an interrupt.
- The program tests for an SRQ.
- If the SRQ is not generated by the analyzer, the subroutine stops and displays `SRQ FROM OTHER DEVICE`.
- If the SRQ was generated by the analyzer, the program reads the status byte and event-status register.
- If bit 5 in the event-status register is set, program prints: `SYNTAX ERROR FROM ANALYZER`.
- If bit 5 in the event-status register is NOT set, program prints: `SYNTAX ERROR BIT NOT SET`.
- The SRQ interrupt is re-enabled on the bus.
- At the finish, the interrupt is deactivated.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 4B: Generating Interrupts" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing PRESET on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

`"CLES;"` **(Clear Status)**

Clears the status register, the event-status registers, and the enable registers.

"ESE 32;" **(Event Status Enable {Bit 5 of the Event-Status Register})**

Enables bit 5 of the event-status register.

"SRE 32;" **(Service Request Enable {Bit 5 of the Status Byte})**

Service request enable. Enables bit 5 of the status byte so that an SRQ is generated when a syntax error is detected.

Subroutine: `Srq_det:`

The program sets up a subroutine to execute when it detects the interrupt.

If there is more than one instrument on the bus capable of generating an SRQ, it is necessary to use serial poll to determine which device has issued the SRQ. Each serial poll returns a status byte from the instrument. Bit 6 is set in the status byte of the instrument that caused the request. A branch to the SRQ-detection routine disables the interrupt assignment. Therefore, the return from Srq_det must re-enable the controller interrupt so it can detect another SRQ interrupt.

Upon the detection of SRQ, the subroutine is executed. The subroutine issues a serial poll of the analyzer's status byte to test for a service request (enabled bit 6).

If the analyzer's status-byte bit 6 is set (the status-byte-SRQ bit enabled) the following comment is displayed on the controller's console:

"SRQ received from analyzer" **(Program Comment)**

Acknowledges on the controller display console that an SRQ was received from the analyzer if bit 6 is set in the status byte.

"SRQ from other device" **(Program Comment)**

Acknowledges on the controller display console that an SRQ was not generated by the analyzer if bit 6 is not set in the status byte. If this is the case, the program halts the reading of the analyzer's status byte and continues looking for the SRQ on the bus.

"ESR?;" **(Event-Status Register Query)**

The controller uses this command to query the event-status register.

The controller then reads the event-status register with a serial poll.

"Event Status Register caused SRQ" **(Program Comment)**

If bit 5 of the status byte is set, this message is displayed on the controller's CRT.

"Some other bit caused SRQ" **(Program Comment)**

If bit 5 of the status byte is not set, this message is displayed on the controller's CRT.

"ESE?;" **(Event-status register Query)**

Queries the event-status register. The controller then reads the status-register byte with another serial poll.

`"Syntax error from analyzer"` **(Program Comment)**

If bit 5 of the event-status register byte is set, this message is displayed on the controller's CRT.

`"Syntax error bit not set"` **(Program Comment)**

If bit 5 of the event-status register byte is not set, this message is displayed on the controller's CRT and the **subroutine ends**.

Main Program Resumes

To continue operation with SRQ detection enabled, the operator must re-enable the interrupt and then return to normal programming operation.

The degree of testing depends upon the number of operations enabled to generate an SRQ. The check-error queue is always set as a consequence of the error occurring. On a complex system with other devices and multiple bits enabled, the SRQ-handler routine can become quite complex in order to test all possible conditions.

The SRQ can be cleared by reading the event-status register and clearing the latched bit, or by clearing the enable registers with the `CLES;` command. The syntax-error message on the analyzer display can only be cleared by the HP-IB Device Clear (DCL) message or Selected Device Clear (SDC) message. Device Clear is not commonly used because it clears every device on the bus. Selected Device Clear can be used to reset the input and output queue and the registers, but will also clear all the interrupt definitions.

The controller issues a serial poll to read and clear any pending service requests. The program then sets an interrupt on the HP-IB bit 2 (SRQ).

`"Waiting for bad syntax"` **(Program Comment)**

Displays `"Waiting for bad syntax"` on the controller's CRT while the program pauses for 2 seconds.

`"STIP 2GHZ;"` **(Incorrect Analyzer Stop Frequency Command)**

This incorrect command generates a syntax error. Indicates the computer is waiting for an interrupt and pauses for two seconds. The interrupt causes the program to branch to the interrupt routine `Srq_det`.

The analyzer will display `CAUTION: SYNTAX ERROR` and the incorrect command, pointing to the first character it did not understand.

`"STAR 10 HZ;"` **(Incorrect Analyzer Start Frequency Command)**

Generates a syntax error because the out-of-range command cannot be interpreted by the input parser of the analyzer.

The program displays `FINISHED`, switches OFF the interrupt, addresses the analyzer and returns it to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Run the program. The computer will preset the analyzer, then pause for a second or two. After pausing, the program sends an invalid command string `"STIP 2GHZ;"` to cause a syntax error. This command is intended to be `"STOP 2 GHZ;"`. The computer will display a series of messages

from the SRQ-handler routine. The analyzer will display `CAUTION: SYNTAX ERROR` and the incorrect command, pointing to the first character it did not understand.

The SRQ can be cleared by reading the event-status register and clearing the latched bit, or by clearing the enable registers with the `CLES;` command. The syntax-error message on the analyzer display can only be cleared by the HP-IB Device Clear (DCL) message or Selected Device Clear (SDC) message. Device Clear is not commonly used because it clears every device on the bus. Selected Device Clear can be used to reset the input and output queue and the registers of a specific instrument on the bus. This will also clear all the interrupt definitions.

After a 4 second pause, the out-of-range command `"STAR 10 HZ;"` is sent to the analyzer.

Note An impossible data condition does not generate a syntax error.

The analyzer simply sets the start frequency to 30 kHz, the minimum start frequency of the analyzer. Syntax errors are created when a command cannot be interpreted by the input parser of the analyzer. The controller releases the analyzer from remote control and the program ends.

Example 4C: Power Meter Calibration

Note This program is stored as EXAMP4C on the "HP 8753D Programming Examples" disk received with the network analyzer.

For increased accuracy of the analyzer's PORT 1-output power, a power meter calibration is available. This measurement-accuracy enhancement technique is described in "Error Messages" of the *HP 8753D Network Analyzer User's Guide*. The example described will perform the sample and sweep calibration under HP-IB remote control.

The power meter is usually connected to PORT 1 for the forward measurements. Its address must be set correctly and it must be connected to the HP-IB. The power meter address can be set by pressing: (LOCAL) `SET ADDRESSES` `ADDRESS P MTR/HPIB` and using the ⇧ and ⇩ keys or the numeric key pad to complete the process. The appropriate command must be selected for the model number of power meter being used. Press `POWER MTR: [ ]` until the model being used is displayed between the brackets. The correction factors for the power sensor are entered into the analyzer. All of these steps are explained in "Error Messages" of the *HP 8753D Network Analyzer User's Guide*.

The number of readings-per-point must also be selected before starting. The number of points directly affects the measurement time of the calibration sequence. The power meter must interact with the analyzer for each of the selected points and read the number of values specified for each trace point. Typically, two readings-per-point is considered appropriate. More than two readings-per-point could lead to unacceptable processing time.

To control a power meter calibration via HP-IB, the analyzer must be set to pass-control mode. The analyzer must position the local oscillator to a point in the sweep and read the power present at the power meter sensor. For this operation to take place, the system controller must set up the measurement and then pass control to the analyzer to read each data point in the sweep. After reading the data point from the power meter, the analyzer passes control back to the system controller. The analyzer then sets up to measure the next point and again requests control from the system controller. This process continues until the analyzer signals that the entire sweep has been measured point-by-point.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The number of points in the trace is set.
- The number of readings-per-point is set.
- The frequency span is set.
- The reference channel is measured.
- The power meter-calibration array is allocated.
- The power meter model is chosen.
- The status registers are cleared.
- The request-control summary bit is enabled.
- The pass-control mode is enabled.
- A calibration sweep is taken to begin the sequence.
- The status byte is read until control is requested.
- The computer passes control to the analyzer.
- The display is cleared and the analyzer is set to talker/listener mode.
- The HP-IB interface status is read until control is returned.
- The program loops until all the points have been measured.
- The power meter calibration is enabled.
- The calibration data is output to the controller in FORM 4, ASCII format.
- The power meter-calibration factors are read into the controller.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 4C: Power Meter Calibration" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing PRESET on the front panel, though it does not initiate an analyzer self-test.

An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

The controller must set up the measurement parameters and then pass control to the analyzer so the analyzer can read the power at each measurement point.

For this operation to take place, the controller must set up the measurement and then pass control to the analyzer to read in each point in the sweep. After the analyzer reads the power, the analyzer passes control back to the controller. The analyzer sets up the next measurement parameters and requests HP-IB control from the controller. This process continues until the analyzer signals that all the data points in the entire sweep have been measured.

If control is passed before requested, the analyzer will just ignore it and return control to the controller. If control is passed while the analyzer is in control, the pass is ignored. The program accomplishes this pass control synchronization by using the event-status registers to signal the state of the analyzer. The program monitors the analyzer's status byte by performing a serial poll and reading the status byte.

The program sets the value of the variables by defining:

- the serial poll as an integer
- the number of points in the trace as 11
- the number of reading-per-point as 2
- the start frequency as 10 MHz
- the stop frequency as 50 MHz

The controller then uses these variables to set the analyzer's instrument state.

`"MEASR;"` (Select and Measure Input R)

Once the instrument state has been set, the program selects and measures Input R.

At this point in the program, the power meter calibration array is created.

`"DATI;"` (Store Data in Channel Memory)

Stores the trace data in the active-channel memory array.

`"DISPDATM;"` (Display Data and Memory)

Displays the combinations of active data and stored trace memory on the active channel.

`"AUTO;"` (Autoscale Selected Channel)

Autoscales the active channel.

`"POWMOFF;DEBUON;"` (Deactivate Power Meter Menu/Activate Debug Mode)

Selects the HP 437B/438A power meter and activates the debug-mode display. This debug display shows the control transfers.

`"CLES;"` (Clear Status)

Clears the status register, the event-status registers, and the enable registers.

`"ESE2;"` (Event-Status Enable {Control-Summary Bit 2})

Enables the request-control bit when the analyzer is ready to perform a power meter measurement over the HP-IB. This is part of the pass-control operation synchronization. This bit is enabled in the event-status register as bit 1, decimal 2.

When this bit is set, the event-status-register bit is set in the analyzer-status register as bit 5, decimal 32.

Bit 5, decimal 32 will indicate that a request for pass control is active.

`"SRE 32;"` **(Service Request Enable {Bit 5 of the Status Byte})**

Service request enable. Enables bit 5 of the status byte so that an SRQ is generated when the pass-control bit is detected.

`"Beginning Power Meter Cal"` **(Program Comment)**

Displays the message `"Beginning Power Meter Cal"` on the controller's Display console.

`"USEPASC;"` **(Enable Pass-Control Mode)**

Enables pass control mode. The HP-IB control must be in pass-control mode for the analyzer to control the power meter calibration.

`"TAKCS;"` **(Take Calibration Sweep)**

Initiates a power meter calibration sweep at the number of points for the number of readings plus one, as defined at the beginning of the program. Repeat when the controller detects a service request by issuing a serial poll.

`"Stat Waiting for request"` **(Program Message)**

The message `"Stat Waiting for request"` is displayed on the controller's CRT. The controller continues to read the status byte with a serial poll until bit 6 is enabled on the status byte.

An interrupt is then set on the HP-IB status byte bit (SRQ).

`"ESR?;"` **(Event-Status-Register Query)**

Queries the analyzer's event-status register to clear the serial poll. The value returned is read by the controller and discarded.

`"Passing Control"` **(Program Message)**

This message is displayed on the controller's CRT as control is passed to the analyzer. The controller's HP-IB interface register is read and tested until control is returned from the analyzer.

`"Waiting for request"` **(Program Message)**

This message is displayed as the system control continues to read the event-status register while waiting to take control back from the analyzer. Reading the event-status register does not interact with the analyzer's operation. When bit 6 in the event-status-register byte is set, control is returned to the system controller.

`"Finished with Power meter Cal"` **(Program Message)**

This message is displayed on the controller's console at the end of the power meter calibration process. The display is then cleared.

`"TALKLIST;"` **(Select Talker/Listener Mode)**

Enables the talker/listener mode in the analyzer.

`"CLES;"` **(Clear Status)**

Clears the status register, the event-status registers, and the enable registers.

`"PWMCONES;"` **(Activate Power Meter Calibration Sweep for One Sweep)**

Activates a power meter calibration sweep after taking one calibration sweep.

`"FORM4;"` **(Select Array-Data Format 4)**

Sets format 4, ASCII floating-point format to be used in reading the power meter calibration correction factors.

`"OUTPPMCAL1;"` **(Output Channel 1 Power Meter Calibration Array)**

Outputs the channel 1 power meter calibration array. The power meter calibration factors are then read by the controller. The power meter calibration factors are cycled through and printed on the controller's display.

The analyzer is addressed and returned to local control with the HP-IB Meta- Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

The analyzer is preset and the power meter-calibration routine begins. The analyzer displays the message `"WAITING FOR HP-IB CONTROL"'` when it is requesting control. The system controller display prints `"Passing Control"` when control is passed to the analyzer. The controller displays `"Waiting for request"` while the analyzer has control and is reading the power meter.

The interaction of the messages and the movement of the cursor allow observation of the calibration process. Once the calibration is complete, the analyzer displays `"POWER METER CAL IS COMPLETE"` and the system controller displays `"Finished with Power meter Cal"`.

The power meter-calibration mode (with one sweep of correction data) is enabled and the calibration is switched ON. At the completion of the program, talker/listener mode is restored, the event-status registers are cleared (to halt the status-byte interaction), the power meter correction factors are displayed, the sweep is placed in continuous-sweep mode, the analyzer is released from HP-IB control, and the program ends.

Example 5: Network Analyzer System Setups

This section provides several different examples of performing analyzer system setups.

Storing and Recalling Instrument States

Note The most efficient option for storing and recalling analyzer states is using the analyzer's internal registers to save the CAL data. Recalling these registers is the fastest solution to restoring analyzer setups. See "Printing, Plotting, or Saving Measurement Results" in the *HP 8753D Network Analyzer User's Guide* for detailed information on the analyzer's internal storage registers.

In the event that all the registers have been used, the internal disk drive is not used, or if internal memory limitations exist, then these external solutions become viable.

The purpose of this example is to demonstrate several programming options for storing and recalling entire instrument states over HP-IB. The examples describe two different processes for storing and recalling instrument states. The first example accomplishes the task using the learn string. The second example involves reading both the learn string and the calibration arrays out of the analyzer and storing them to disk or storing them in the system controller itself.

Using the learn string is a very rapid way of saving the instrument state, but using direct disk access has the advantage of automatically storing calibrations, cal kits, and data along with the instrument state.

A complete analyzer setup requires sending the learn string and a calibration array to set the analyzer parameters. The CAL array may also be placed in the analyzer, just as if a calibration was performed. By sending both sets of data, the analyzer may be quickly setup for a measurement.

Several different measurements may be required in the course of testing a device. An efficient way of performing multiple measurements is to send both the calibration array and the learn string, and then perform the measurements.

Example 5A: Using the Learn String

Note This program is stored as EXAMP5A on the "HP 8753D Programming Examples" disk received with the network analyzer.

The learn string is a very fast and easy way to read an instrument state. The learn string includes all front-panel settings, the limit table for each channel, and the list-frequency table. It can be read out of the analyzer with the command OUTPLEAS, and input to the analyzer with the command INPULEAS. This array is always transmitted in FORM 1, the internal format for the analyzer. It cannot be longer than 3000 bytes. The example for a FORM 1 transfer could also have been used. However, Example 5A is the simplest solution for reading the learn string from the analyzer.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The string storage is allocated.
- The learn string is requested.
- The string is read without any processing.

- The analyzer is released from remote control.
- The instrument state is changed by the operator.
- The learn string is sent back to the analyzer.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 5A: Using the Learn String" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

First, a string is defined for contents to receive the instrument state learn-string data.

`"OUTPLEAS;"` **(Output Learn String)**

Outputs the learn string. The learn string is in binary data and must not be interpreted by the input process. The EOI terminator is sent with the last byte in the string and is used to terminate the entry. The conventional terminators (carriage return or line feed) may appear in the string and must be treated as data.

The controller releases the analyzer from HP-IB control.

`"Change state and press ENTER"` **(Operator Prompt)**

Prompts the operator to change the analyzer settings after the analyzer has received the learn string. The analyzer is placed in local mode to allow you to change the analyzer settings. If you press [PRESET], the analyzer restores the settings to the default state.

Pressing [ENTER] on the controller keyboard continues the program.

`"INPULEAS;"` **(Input Learn String to Analyzer)**

Sends the analyzer input learn string. Notice that no semicolon is used as a terminator. The command is terminated by the end of the learn string. The previous state of the analyzer is restored with the learn string.

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1),

The analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Run the program. When the program stops, change the instrument state and press (ENTER) on the controller. The analyzer will be returned to its original state by using the learn string.

Example 5B: Reading Calibration Data

Note This program is stored as EXAMP5B on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example demonstrates:

- how to read measurement calibration data out of the network analyzer
- how to read it back into the analyzer
- how to determine which calibration is active

The data used to perform measurement-error correction is stored inside the analyzer in one (or more) of twelve calibration-coefficient arrays. Each array is a specific error coefficient, and is stored and transmitted as an error-corrected data array. Each point is a real/imaginary pair, and the number of points in the array is the same as the number of points in the sweep. The five data formats also apply to the transfer of calibration-coefficient arrays. "Printing, Plotting, or Saving Measurement Results" in the *HP 8753D Network Analyzer User's Guide* contains information on the storage locations for calibration coefficients and different calibration types.

A computer can read out the error coefficients using the commands OUTPCALC01, OUTPCALC02, ... through OUTPCALC12. Each calibration type uses only as many arrays as required, beginning with array 1. Hence, it is necessary to know the type of calibration about to be read out: attempting to read an array not being used in the current calibration causes the `"REQUESTED DATA NOT CURRENTLY AVAILABLE"` warning.

A computer can also store calibration coefficients in the analyzer. To do this, declare the type of calibration data about to be stored in the analyzer just as if you were about to perform that calibration. Then, instead of calling up different classes, transfer the calibration coefficients using the `INPUCALCnn;` commands. The variables `nn` are a data pair appended to the command representing a calibration number from `01` through `12`. When all the coefficients are stored in the analyzer, activate the calibration by issuing the mnemonic `SAVC;`, and trigger a sweep on the analyzer.

This example reads the calibration coefficients into a very large array, from which they can be examined, modified, stored, or put back into the instrument. If the data is to be directly stored on to disk, it is usually more efficient to use FORM 1 (analyzer's internal-binary format), and to store each coefficient array as it is read in.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The calibration types and number of arrays are defined.
- The integer variables for reading the headers are defined.
- The calibration type and number of arrays are read by the controller.
- The output is formatted in FORM 3.
- The number of points in the trace is read.

- The memory is allocated for the calibration arrays.
- Each calibration array is requested from the analyzer.
- The elements from each calibration array are read in.
- The next calibration array is requested until all the arrays have been read.
- The calibration type is sent to the analyzer.
- Each calibration array is sent.
- The calibration is activated.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 5B: Reading Calibration Data" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

Data defined for determining calibration types and numbers of arrays.

The analyzer responds with a one (1) if one of the following calibrations are active:

- `"CALIRESP"`, 1 which is a response calibration with one array
- `"CALIRAI"`, 2 which is a response and isolation calibration with 2 arrays
- `"CALIS111"`, 3 which is an input one-port measurement calibration with 3 arrays
- `"CALIS221"`, 3 which is an output one-port measurement calibration with 3 arrays
- `"CALIFUL2"`, 12 which is a full 2-port measurement calibration with 12 arrays

Once the calibration type is determined, the type and number of arrays is shown on the console display.

If no calibration type is present the program ends.

`Caltype"?;"` **(Calibration Type Query)** Each calibration type followed by a "?" is sent to the analyzer as a query. a one (1) is returned for the active calibration type.

If there is an active calibration, the program uses the number of arrays associated with the calibrations to query the active calibration present in the analyzer. If no calibration is found, the program ends.

`"FORM3;"` **(Select Array-Data Format 3)**

Selects array-data format 3. Data will now be output to the controller formatted as a 64-bit floating point, with header.

`"POIN?;"` **(Number of Points Query)**

Queries the analyzer for the number of points in the measurement trace.

An array to store the calibration arrays is created.

Each of the calibration arrays are read from the controller with the command `"OUTPCALCnn;"`, where *nn* is the calibration number 01 through 12. A leading zero (0) is required to generate a two-digit command.

The selected error coefficient array is output from the active channel formatted to add a zero (0) into the command. Each array is the same as a data array for use by the program to read in the values. The "I" variable must be in a fixed, two-digit form with a leading zero (if needed).

The required number of arrays for the active calibration are read from the analyzer and stored into the calibration array.

A 4-byte header, containing `"#A"` and two bytes which identify the length of the array is read by the program before putting the calibration-array data into the multi-dimensioned array. Each pair is read from the analyzer and the array is indexed for the next element in the array. After all the calibration arrays have been read from the analyzer, the operator is prompted:

`"PRESS RETURN TO RE-TRANSMIT CALIBRATION"` **(Operator Prompt)**

Note If your system controller has an [ENTER] rather than a [RETURN], this function should be performed by pressing [ENTER].

Prompts the operator to send the calibration arrays of coefficients back to the analyzer. When [ENTER] is pressed the calibration coefficients are sent back to the analyzer.

`"FORM3;"` **(Select Array-Data Format 3)**

Selects array-data format 3. Data will now be output to the controller formatted as a 64-bit floating point, with header. This is the same format with which the data originally read.

Caltype;";"'' **(Output Calibration Type)**

Output the calibration type to be determined at the beginning of the program to the analyzer. Each calibration array is separately transmitted to the analyzer and `"TRANSMITTING ARRAY :nn"` (where *nn* represents the array number) is displayed on the controller's screen.

The header and array length are sent to the analyzer along with each array element and the real/imaginary pairs.

The program then moves to the next calibration array and performs the transmission functions again until all of the calibration arrays have been transmitted back to the analyzer.

`"SAVC;"` **(Calibration-Array Transmission Complete)**

Completes the transfer of error correction coefficients back into the instrument.

Activates the calibration after all the arrays have been indexed and sent to the analyzer.

`"CONT;"` **(Select Continuous-Sweep Mode)**

Sets the sweep mode to continuous.

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Before executing the program, perform a calibration.

The program is able to detect which type of calibration is active. With that information, it predicts how many arrays to read out. When all the arrays have been sent to the computer, the program prompts the operator. The operator then switches OFF the calibration or performs a completely different calibration on the analyzer and continues the program. The computer reloads the old calibration. The operator should not preset the analyzer because the instrument settings must be the same as those that were present when the calibration was taken.

Note The re-transmitted calibration is associated with the current instrument state: the instrument has no way of knowing the original state associated with the calibration data. For this reason, it is recommended that the learn string be used to store the instrument state whenever calibration data is stored. The next example demonstrates how to reload the analyzer state with both the learn string and the calibration arrays.

Example 5C: Saving and Restoring the Analyzer Instrument State

Note This program is stored as EXAMP5C on the "HP 8753D Programming Examples" disk received with the network analyzer.

Note The instrument state may also be stored in the analyzer's internal registers. This is the fastest and most efficient method for toggling between instrument states. This example is for use when the analyzer's internal memory is full, or when the are other internal-memory limitations.

This example demonstrates using both the learn string and the calibration arrays to completely re-program the analyzer state. If you were performing two entirely different measurements on a device and wanted to quickly change between instrument states and perform the measurements, this example program is a potential solution.

The example will request the learn string and a calibration array from the analyzer and store them in a disk file on the system controller. Once the storage is complete, the operator will be prompted to change the state of the analyzer and then re-load the state that was previously stored in the disk file. Once the file is created on the disk, the state information can be retrieved from the controller and restored on the analyzer.

Note The disk file can only be created once. Errors will occur if the operator repeatedly tries to re-create the file.

For this example, only a thru response calibration will be performed and transferred. This means only one calibration array will be read from the analyzer and written to the disk file with the instrument state. To work with more elaborate calibrations, additional arrays will need to be defined and transferred to the disk file. This is not difficult, but requires some additional programming steps which were omitted in the interest of presenting a simple example.

The following is an outline of the program's processing sequence:

- The integers for reading the headers are defined.
- The system is initialized.
- An array is created to hold the learn string.
- The learn string is requested by the controller.
- The number of points in the trace is read.
- The controller allocates an array for the calibration data.
- The calibration data is read into the controller.
- The controller creates and assigns a data file for the calibration array and the learn string.
- The learn string and calibration array are stored in the disk file.
- The operator presses ENTER on the controller to read the calibration data back into the analyzer.
- The learn string is read from the disk file and output to the analyzer.
- The calibration array is read in from the disk file and stored in the analyzer.
- The analyzer is returned to continuous-sweep mode.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 5C: Saving and Restoring the Analyzer Instrument State" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

The instrument state is going to be transferred in the form of a learn string and a calibration array. In the first section of the program, a string array is dimensioned and the learn string is read from the analyzer.

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB

instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Places the analyzer in single-sweep mode and activates a sweep. The analyzer returns a one (1) at the completion of the sweep.

`"OUTPLEAS;"` **(Output Learn String)**

Outputs the learn string. A 3000-byte string is defined to receive the learn-string data. The learn string is in binary data and must not be interpreted by the input process. The EOI terminator is sent with the last byte in the string. EOI is used to terminate the entry. The conventional terminators (carriage return or line feed) may appear in the string and must be treated as data.

`"POIN?;"` **(Number of Points Query)**

Queries the analyzer for the number of points in the measurement trace.

The analyzer outputs the trace length to determine the size of the calibration array. An array was dimensioned to this length with two entries-per-point (the real and imaginary calibration values).

`"FORM3;"` **(Select Array-Data Format 3)**

Selects array-data format 3. Data will now be output to the controller formatted in IEEE 64-bit floating point format, with header.

`"OUTPCALCO1;"` **(Output Selected Calibration Array)**

Outputs the selected error coefficient array from the active channel. Each array is the same as a data array.

Used here, this command queries the calibration array to read the first and only array. The array header and length bytes are read in as integers and the array read as floating-point numbers. Once the array and the learn string are stored in arrays, the next step is to transfer them to a disk file.

A file is created large enough to hold 3000 bytes of learn string and the number of points times 16 bytes-per-point. In this case, [3000 + (201 × 16)] or 6,216 bytes. The learn string, header and length integers and the calibration array are written to the file and the file closed.

`"Cal data received. Press RETURN to send it back."` **(Operator Prompt)**

Prompts the operator to press ENTER on the controller keyboard to reload the calibration arrays back into the analyzer.

A new learn-string variable is dimensioned to 3000 bytes. The data file is opened and the learn string is read along with the header and length integers. The length variable is necessary to dimension the array for the calibration array. The length integer divided by 16 (two 8-byte values-per-data pair) dimensions the array. The calibration data can now be read from the file into the array.

Pressing ENTER on the controller keyboard continues the program.

"INPULEAS" **(Input Learn String)**

Sends the learn string array.

"INPCALCO1;" **(Input Selected-Calibration Array)**

Sends the calibration data to define the location for the calibration array. The header and length integers are sent first, followed by the calibration array.

"OPC?;SAVC;" **(Cal-Array Transmission Done with Operation Complete Query)**

Completes the transfer of error-correction coefficients back into the instrument.

Activates the calibration after all the arrays have been indexed and sent to the analyzer. The analyzer returns a one (1) at the completion of the array transmission process.

"CONT;" **(Select Continuous-Sweep Mode)**

Sets the sweep mode to continuous.

"OPC?;WAIT;" **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Setup the analyzer and perform a through calibration.

Run the program. The program prompts the operator to change the state of the analyzer and then press (ENTER) to continue. At this point, the analyzer state is stored on the disk file in the controller. Pressing (ENTER) will begin the transfer from the disk file to internal arrays within the controller and then on to the analyzer. Once completed: the original state will be restored, the analyzer will be sweeping, the analyzer will be calibrated, and COR will be displayed on the analyzer's CRT.

Example 6: Limit-Line Testing

Using List-Frequency Mode

The analyzer normally takes data points spaced at regular intervals across the overall frequency range of the measurement. For example, for a 2 GHz frequency span using 201 points, data will be taken at intervals of 10 MHz. The list-frequency mode allows the operator to select the specific points, or frequency spacing between points, at which measurements are to be made. This mode of operation allows flexibility in setting up tests that insure device performance in an efficient manner. By only sampling specific points, measurement time is reduced. List-frequency sweeps are also discussed in "Application and Operation Concepts" of the *HP 8753D Network Analyzer User's Guide*. These programs emulate operation from the analyzer's front panel when using list sweeps.

The following two examples illustrate the use of the analyzer's list-frequency mode to perform arbitrary frequency testing. Example 6A allows the operator to construct a table of list-frequency segments which is then loaded into the analyzer's list-frequency table. There are a maximum of 30 segments available. Each segment stipulates a start and stop frequency, and the number of data points to be taken over that frequency range. Example 6B lets the operator select a specific segment to "zoom-in." A single instrument can be programmed to measure several different devices, each with its own frequency range, using a single calibration. When a specific device is connected, the operator selects the appropriate segment for that device. Note that list-frequency segments can be overlapped, but the total number of points in all the segments must not exceed 1632.

Example 6A: Setting Up a List-Frequency Sweep

Note This program is stored as EXAMP6A on the "HP 8753D Programming Examples" disk received with the network analyzer.

The purpose of this example is to show how to create a list-frequency table and transmit it to the analyzer.

The command sequence for entering a list-frequency table imitates the key sequence followed when entering a table from the front panel: there is a command for every key-press.

Editing a segment is also similar same as the front-panel key sequence, except that the analyzer automatically reorders each edited segment in order of increasing start frequency.

The list-frequency table is also carried as part of the learn string. While the table cannot be modified as part of the learn string, it can be stored and recalled with very little effort by storing and recalling the learn string. See the "Data-Processing Chain" section in Chapter 2 for additional information details on using learn strings.

This example takes advantage of the computer's ability to simplify:

- creating a list-frequency table
- editing a list-frequency table

The table is entered and completely edited before being transmitted to the analyzer. To simplify the programming task, options such as entering center frequency, frequency span, or step size are not included.

The list-frequency information may be acquired using the limit-test results array. The actual stimulus points are available as the first element in the array.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The existing list frequencies are edited and cleared.
- The number of segments to be defined is read in.
- An array for the list segments is defined.
- The parameters for each segment are requested.
- If the operator wants to edit, the segment parameters are re-entered.
- The new list is sent to the analyzer.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 6A: Setting Up a List-Frequency Sweep" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer's operating state to the default state (preset). This command is similar to pressing [PRESET] on the front panel, though it does not initiate an analyzer self-test. An operation-complete query (OPC?) asks for the analyzer to return an ASCII character one (1) to the system controller when the preset operation is complete.

Main Program

`"EDITLIST;"` **(Activate List-Frequency-Table Edit Mode)**

Enters the frequency-list edit mode. This function is the same as pressing the softkey under the list-frequency mode. This begins the interaction with the analyzer and displays the list-frequency table on the controller screen.

`"CLEL;"` **(Clear List)**

Clears any existing tables.

Caution Any existing segments are erased from the analyzer's internal registers. You may wish to comment-out this command line (in this case, the executable `CLES;` contained in LINE 190) and simply enter new segments.

For this example, a new list is generated by creating a table of values.

A table header that shows the entered segments is printed on the display. The operator is prompted for each entry in the segment.

If the operator chooses to edit a segment, the entire modified segment must be re-entered.

Once the list is complete the operator may choose to edit.

`"DO YOU WANT TO EDIT? Y OR N"` **(Operator Prompt)**

Choosing `Y` at the prompt activates the program subroutine "Loadpoin".

Subroutine: `Loadpoin:`

The subroutine `Loadpoin:` reads in each segment value for:

- start frequency
- stop frequency
- number of points in the segment

During the subroutine, the operator is prompted for the number of segments to define and a table is created to hold the defined segments.

After editing, the new segment is displayed as a table on the controller's display screen and the **subroutine ends.**

Main Program Resumes

`"SADD;"` **(Add Segment)**

Adds a segment by sending it to the analyzer. The segments are sent to the analyzer by adding segments to the list. The segment is defined by sending the table created.

`"STAR";Table(I,1);"MHZ;"` **(Select Start Frequency)**

Sends the start frequency using the numbers contained in the variable `Table(I,1)` (the first value in the list-frequency table) as the data value.

`"STOP";Table(I,2);"MHZ;"` **(Select Stop Frequency)**

Sends the stop frequency using the numbers contained in the variable `Table(I,2)` (the second value in the list-frequency table) as the data value.

`"POIN",Table(I,3),";"` **(Select Number of Points)**

Sends the number of points in the trace using the numbers contained in the variable `Table(I,3)` (the third value in the list- frequency table) as the data value.

`"SDON;"` **(Segment Done)**

Closes a segment after editing.

Indicates that the segment is done. The segments do not need to be in numerical order. The analyzer will reorder and display them.

`"EDITDONE;"` **(Edit Done)**

Done editing list frequency table. Ends the segment editing.

`"LISFREQ;"` **(Activate List-Frequency Mode)**

Enables the list-frequency mode.

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Caution This example program will delete any existing limit lines before entering the new limits. If this is not desired, omit the line(s) that clear the existing limits (in this case, the executable `CLEL;` contained in LINE 190). This program begins by presetting the analyzer. The programmer will have to add the necessary command lines to set the analyzer to the specific operating conditions required for testing. The example program will show the limit lines defined, but the limits will always fail without additional analyzer setup.

The program displays the frequency-list table as it is entered. During editing, the displayed table is updated as each line is edited. The table is not re-ordered. At the completion of editing, the table is entered into the analyzer, and list-frequency mode is switched ON. During editing, pressing [ENTER] leaves an entry at the old value.

If the analyzer display is within the range of the segments entered, then the number of points-per-segment may be observed on the analyzer's display.

Activate a marker and select the discrete-marker mode to observe the point spacing. Use an exaggerated scale with just a few points to find the list-frequency spacing between points.

Example 6B: Selecting a Single Segment from a Table of Segments

Note This program is stored as EXAMP6B on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example program demonstrates how to define a single segment as the operating-frequency range of the analyzer from a table of segments stored in the controller. The program assumes that a list-frequency table has already been entered into the analyzer, either manually, or using the program in Example 6A, "Setting Up a List-Frequency Sweep."

The program first loads the list-frequency table into the computer by reading the start and stop frequencies of each segment and the number of points for each segment. The segments' parameters are then displayed on the computer screen, and the operator can choose which segment is to be used by the analyzer. Note that only one segment can be chosen at a time.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The list-frequency segment is edited.
- The largest segment number is set.
- The highest segment number is requested.
- The number of actual segments in use is read in.
- A list-frequency table is defined and the segments are read in to the controller from the analyzer.
- The operator selects one of the segments of the sweep.
- The controller "zooms-in" and sweeps the defined segment.
- The operator ends the program by entering segment number (0).
- The analyzer returns to sweeping all the segments in the table.
- The activation loop is ended and the program ends.

Tutorial Program Explanation

The following section explains "Example 6B: Selecting a Single Segment from a Table of Segments" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

The program begins by opening the list-frequency segments, determining the number of segments that have been defined, and printing the data as a table on the controller display.

`"EDITLIST;"` **(Activate List-Frequency-Table-Edit Mode)**

Activates the list-frequency-table edit mode. This function is the same as pressing the softkey under the list-frequency mode. This begins the interaction with the analyzer and displays the list-frequency table on the controller screen.

`"SEDI30;"` **(Select Segment Editing Mode for Segment {30})**

Sets the analyzer to edit the largest segment number possible (30). The largest segment number, as defined in the active function area of the analyzer, is displayed.

`"SEDI?"` **(Select Segment Query)**

Sends the number of the highest segment defined.

A table is allocated in the controller to hold the number of segments defined in the analyzer. The segments are looped through to read the values into the segment table to display to the operator.

A subroutine is then activated by the program.

Subroutine `Readlist:`

This subroutine manages the controller functions related to editing the segment list.

`"EDITLIST;"` **(Activate List-Frequency-Table Edit Mode)**

Activates the list-frequency-table edit mode. This function is the same as pressing the softkey that activates the list-frequency mode. This command begins the interaction with the analyzer and displays the list-frequency table on the controller screen.

`"SEDI",I,";"` **(Select Segment to Edit)**

Reads the segment list into the controller by selecting a segment to edit. Each of the segment parameters are read by sending the parameter command and then reading the active area of the display.

For example:

`"STAR;"` **(Send Start Frequency for Display as a Variable)**

Puts start frequency value in active-display area.

`"OUTPACTI;"` **(Output Active-Function Area)**

Read value from active area. Place segment start frequency in table

`"STOP;"` **(Send Stop Frequency for Display as a Variable)**

Puts stop frequency value in active-display area.

`"OUTPACTI;"` **(Output Active-Function Area)**

Read value from active area. Place segment start frequency in table

`"POIN;"` **(Send Number of Points for Display as a Variable)**

Puts number of points value in active-display area.

When the operator presses ENTER on the controller keyboard, the display scrolls through the segments.

`"SELECT SEGMENT NUMBER: (0 TO EXIT)"` **(Operator Prompt)**

Type in the number of the segment, and the analyzer will then "zoom-in" on that segment by activating it in the analyzer.

`"EDITDONE;"` **(Done Editing)**

Terminates the segment selection and activates the selected segment. The selected segment is now the new frequency range of the analyzer. Prompt and read another segment number.

"SSEG;" **(Send Selected Segment)**

Sends the segment number. Once (0) has been selected, the subroutine ends by restoring by restoring the sweep to all segments.

"ASEG;" **(Set All Segment Sweep)**

Returns to sweep and display the entire list frequency and the controller displays "PROGRAM ENDED"

"OPC?;WAIT;" **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

The program will read the parameters for each list-frequency segment from the analyzer, and build a table containing all the segments. The parameters of each segment will be printed on the computer screen.

After all the segments are displayed, the program will prompt the operator for a specific segment to be used. Type in the number of the segment, and the analyzer will then "zoom-in" on that segment. The program will continue looping, allowing continuous selection of different segments. To exit the loop, type 0. This will restore all the segments (with the command ASEG), allowing the analyzer to sweep all of the segments, and the program will terminate.

Example 6C: Setting Up Limit Lines

Note This program is stored as EXAMP6C on the "HP 8753D Programming Examples" disk received with the network analyzer.

The purpose of this example is to show how to create a limit-test table and transmit it to the analyzer.

The command sequence for entering a limit-test table imitates the key sequence followed when entering a table from the analyzer's front panel: there is a command for every key-press. Editing a limit is also the same as the key sequence, but remember that the analyzer automatically re-orders the table in order of increasing start frequency.

The limit-test table is also carried as part of the learn string. While it cannot be modified as part of the learn string, the learn string can be stored and recalled with very little effort. See section, "Data-Processing Chain" in Chapter 2 for details on using learn strings.

This example takes advantage of the computer's capabilities to simplify creating and editing the table. The table is entered and completely edited before being transmitted to the analyzer. To simplify the programming task, options such as entering offsets are not included.

This example automates the front-panel operation of entering a limit-test table. Front-panel operation and limits are discussed in the "Application and Operation Concepts" in the *HP 8753D Network Analyzer User's Guide.*

The following is an outline of the program's processing sequence:

- The system is initialized.

- The limit lines are edited and cleared.
- The number of limits is requested.
- The limit table is created.
- The string array of limit types is created.
- The operator is prompted to enter the new limit values.
- The new limit table is sent back to the analyzer.
- The limit line is activated.
- The limit test is activated.
- The analyzer is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 6C: Setting Up Limit Lines" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer operating state to the default state (preset). This command is similar to pressing the front-panel PRESET, but it does not initiate an analyzer self-test. The analyzer sends an ASCII character one (1) to the instrument controller when the preset operation is complete.

Main Program

You must setup the analyzer to specific ranges to display and test the limits entered.

`"EDITLIML;"` **(Edit Limit Lines)**

Opens the limit table for editing.

`"CLEL;"` **(Clear Limits)**

Any existing limits are deleted.

"NUMBER OF LIMITS?" **(Operator Prompt)**

Requests the number of limits to be entered. A table is created in the controller to hold the following entries:

- stimulus value (frequency)
- upper limit value
- lower limit value
- limit type

A string array is created for the limits.

The controller display is cleared and the table header is printed.

"DO YOU WANT TO EDIT? Y OR N"

Prompts you to edit the limit table or to send the existing table to the analyzer. If you do make any changes to the entries, you must actually re-enter the entire limit.

"EDITLIML;"

Opens the limit table so you can begin to edit the table. If the limit table was not cleared as in this example, the new limits will just be added to the old ones.

"ENTRY NUMBER?" **(Operator Prompt)**

Prompts the operator to enter the number representing the desired limit. When the operator enters a value, the program activates a subroutine.

Subroutine: `Loadlimit:`

This subroutine manages the controller functions related to editing the limit table.

Cycle through the limits desired and enter the limit values.

The stimulus entry prompt: "STIMULUS VALUE? (MHz)" is displayed on the controller console. The operator then enters a value based on MHz as the units.

The upper limit entry prompt: "UPPER LIMIT VALUE? (DB)" is displayed on the controller console. The operator then enters a value based on DB as the units.

The lower limit entry prompt: "LOWER LIMIT VALUE? (DB)" is displayed on the controller console. The operator then enters a value based on DB as the units.

"ENTRY TYPE? (FL=FLAT, SL=SLOPE, SP=SINGPOINT)" **(Operator Prompt)**

Prompts the operator to enter the type representing the desired limit. The choices are FL=FLAT, SL=SLOPE, SP=SINGPOINT.

"SADD;LIMS"; stimulus entry;"MHZ;" **(Add Segment)**

Begins the sequence that allows you to add the segments. The stimulus value that was entered is included with the limit stimulus command.

"LIMU";(value);"DB;" **(Select Upper Limit Value)**

Sends the upper limit value from the table tto the analyzer.

`"LIML";(value);"DB;"` **(Select Upper Limit Value)**

Sends the upper limit value from the table to the analyzer

The type of limit point is sent to the analyzer by constructing a command based upon the string values entered in the table:

- `"FL"` sends `"LIMTFL;"` for a flat limit type.
- `"SL"` sends `"LIMTSL;"` for a sloped limit type.
- `"SP"` sends `"LIMTSP;"` for a point limit type.

`"SDON;"` **(Segment Done)**

Ends the complete segment definition. The segments are each sent in the same process. Then **the subroutine ends.**

Main Program Resumes

`"EDITDONE;"` **(Done Editing)**

Closes the edit list frequency or limit table.

`"LIMILINEON;"` **(Activate Limit Lines)**

Activates the limit lines.

`"LIMITESTON;"` **(Activate Limit Tests)**

Activates the limit testing.

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Caution This example program will delete any existing limit lines before entering the new limits. If this is not desired, omit the line(s) that clear the existing limits (in this case, the executable `"CLEL;"` contained in LINE 190). This program begins by presetting the analyzer. The programmer will have to add the necessary command lines to set the analyzer to the operating conditions required for testing. The example program will show the limit lines defined, but the limits will always fail without additional analyzer setup.

The program displays the limit table as it is entered. During editing, the displayed table is updated as each line is edited. The table is not reordered. At the completion of editing, the table is entered into the analyzer, and limit-testing mode switched ON. The analyzer will rearrange the table in ascending order starting with the lowest start frequency entry.

Example 6D: Performing PASS/FAIL Tests While Tuning

Note This program is stored as EXAMP6D on the "HP 8753D Programming Examples" disk received with the network analyzer.

The purpose of this example is to demonstrate the use of the limit/search-fail bits in event-status-register B, to determine whether a device passes the specified limits. Limits can be entered manually, or using the Example 5A.

The limit/search-fail bits are set and latched when limit testing or a marker search fails. There are four bits, one for each channel for both limit testing and marker search. See Figure 2-5 and Table 2-3 for additional information. Their purpose is to allow the computer to determine whether the test/search executed was successful. They are used in the following sequence:

1. Clear event-status-register B.
2. Trigger the limit test or marker search.
3. Check the appropriate fail bit.

When using limit testing, the best way to trigger the limit test is to trigger a single sweep. By the time the single sweep command `"SING;"` finishes, limit testing will have occurred.

Note If the device is tuned during the sweep, it may be tuned into and then out of limit, causing a limit test to qualify as "passed" when the device is not in fact within the specified limits.

When using marker searches (max, min, target, and widths), outputting marker or bandwidth values automatically triggers any related searches. Therefore, all that is required is to check the fail bit after reading the data.

In this example, several consecutive sweeps must qualify as "passing" in order to insure that the limit-test pass was not extraneous due to the device settling or operator tuning during the sweep. Upon running the program, the number of "passed" sweeps for qualification is entered. For very slow sweeps, a small number of sweeps such as two are appropriate. For relatively fast sweeps, where the device requires time to settle after tuning, as many sweeps as six or more sweeps may be more appropriate.

A limit-test table can be entered over HP-IB. The sequence is very similar to that used in entering a list-frequency table, as shown in Example 5D. The manual (front-panel entry) sequence is closely followed.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The pass counter is initialized on entry.
- The analyzer takes a sweep.
- The event-status register B byte is output and the channel-1 limit is tested.
- If the device fails the first sweep, the operator is prompted to insure it is tuned correctly and the device is measured again.
- Once the device passes, the operator is prompted not to touch the device as testing continues.
- If the device passes the required number of sweeps, the operator is prompted that the device has passed and to connect the next device for testing.

- The program initializes the pass counter and begins to measure the new device.

Tutorial Program Explanation

The following section explains "Example 6D: Performing PASS/FAIL Tests While Tuning" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

You must leave the limit lines and analyzer setup intact to perform this type of testing.

`"Number of consecutive passed sweeps for qualification?"` **(Operator Prompt)**

Prompts the operator to choose the number of sweeps to determine qualification; that is, how many sweeps must pass before the device is considered to have passed the limit test. The program's pass counter is initialized when the entry is made.

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Sets a single sweep and then queries the analyzer to insure that the sweep is complete before executing any further instructions. Once the sweep is complete (the analyzer has returned a one (1) to the controller), the limit test will be performed upon completion of the sweep cycle.

`"ESB?;"` **(Event-Status-Register B Query)**

Queries event-status-register B and returns the contents of the status byte to the program. The fail bit 4 (or numerical value 16), is read from the status byte to determine if the limit test failed.

- If the test passed, the "passed counter" is incremented and compared to the number of passes required to qualify the test as "passed." The controller creates a high pitch beep to warn you not to adjust the device under test and displays the message `"LEAVE DEVICE ALONE"`.
- If the fail bit for channel one is set, the number of sweeps in the "passed counter" is reset to zero. The controller creates a low pitch beep to indicate that the device is failing the test and displays the message `"TUNE DEVICE AS NECESSARY"`. The device under test may require adjustment by the operator.

If not enough sweeps have passed, the measurement cycle is repeated. The program issues `"OPC?;SING;"` to repeat the measurement process.

"DEVICE PASSED!"

Indicates that enough sweeps have passed to qualify the device. The controller creates a series of tones and displays a message that prompts you to change the device under test.

"PRESS RETURN FOR NEXT DEVICE" **(Operator Prompt)**

Prompts the operator to begin the testing cycle for the new device by pressing (ENTER).

Running the Program

Note This program assumes a response calibration (thru calibration) or a full 2-port measurement calibration has been performed prior to running the program.

Set up a limit-test table on channel 1 for a specific device either manually, or using the program in Example 5A.

Run the program, and enter the number of passed sweeps desired for qualification. After entering the qualification number, connect the device. When a sweep passes, the computer beeps. When enough consecutive sweeps qualify the device as "passing," the computer emits a dual-tone beep to attract the attention of the operator, and then prompts for a new device.

To test the program's pass/fail accuracy, try causing the DUT to fail by loosening the cables connecting the DUT to the analyzer and running the program again.

Example 7: Report Generation

The analyzer has three operating modes with respect to HP-IB. These modes can be changed by accessing softkeys in the (LOCAL) menu. System-controller mode is used when no computer is present. This mode allows the analyzer to control the system. The other two modes allow a remote system controller to coordinate certain actions: in talker/listener mode the remote system controller can control the analyzer, as well as coordinate plotting and printing; and in pass-control mode the remote system controller can pass active control to the analyzer so that the analyzer can plot, print, control a power meter, or load/store to disk. The amount of peripheral interaction is the main difference between talker/listener and pass-control mode. See Chapter 2 for more details.

Example 7A1: Operation Using Talker/Listener Mode

Note This program is stored as EXAMP7A1 on the "HP 8753D Programming Examples" disk received with the network analyzer.

The commands OUTPPLOT and OUTPPRIN allow talker/listener mode plotting and printing via a one way data path from the analyzer to the plotter or printer. The computer sets up the path by addressing the analyzer to talk, the plotter to listen, and then releases control of the analyzer in order to transfer the data. The analyzer will then make the plot or print. When it is finished, it asserts the End or Identify (EOI) control line on HP-IB. The controller detects the presence of EOI and re-asserts control of the HP-IB. This example program makes a plot using the talker/listener mode.

Note One of the attributes of the OUTPPLOT command is that the plot can include the current softkey menu. The plotting of the softkeys is enabled with the `PSOFTON;` command and disabled with the `PSOFTOFF;` command.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The selected frequency span is swept once.
- The plot command is sent to the analyzer.
- The analyzer is set to talker mode and the plotter is set to listener mode.
- The plot is spooled to the plotter.
- The analyzer is set to listener mode when the controller detects an EOI from the analyzer.
- The controller puts the analyzer back in continuous-sweep mode.
- The analyzer is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 7A1: Analyzer Operation Using Talker/Listener Mode" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

`"OUTPPLOT;"` **(Output Plot Data in ASCII Format)**

Commands the analyzer to plot. (For printing, substitute the `"OUTPPRIN;"` executable.)

7;UNL LISTEN 5 TALK 16 DATA'' **(Create Data Path)**

A data path is established from the analyzer to the plotter. Before the plotter operation can take place, the talker/listener assignments must be made by the instrument controller.

- The plotter is assigned as a listener at address 5 (default address for a plotter).
- The printer is assigned as a listener at address 1 (default address for a printer).
- The analyzer is addressed to talk as device 16; that is, transmit the contents of its output queue.

After the addressing is complete, the instrument controller releases the ATN line and the bus changes from command to data mode. The analyzer becomes a talker, the plotter a listener, and the plot begins.

Note The `"OUTPPLOT;"` command allows you to include the current softkey menu on the hardcopy. The plotting of the softkeys is enabled with the command `"PSOFTON;"` and disabled with `"PSOFTOFF;"`.

`"Plotting and waiting for EOI"` **(Program Message)**

Prompts the operator to wait for the analyzer to assert the EOI line, indicating the end of transmission. The HP-IB interface status registers for the interface installed on the instrument controller are read and tested for the presence of the active EOI line.

`"End of plot"` **(Program Message)**

Indicates the computer has detected the active EOI line on the interface and the system controller returns the analyzers device mode to listener. The analyzer will go on asserting EOI until some other activity on the bus causes it to clear the line.

`"CONT;"` **(Select Continuous-Sweep Mode)**

Places the analyzer in continuous-sweep mode.

"OPC?;WAIT;" (Wait for Reply with Operation Complete Query)

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

If a problem arises with the plotter, such as no pen or paper, the analyzer cannot detect the situation because it only has a one-way path of communication. Therefore, the analyzer attempts to continue plotting until you intervene and abort the plot by pressing the (LOCAL) key. This key aborts the plot, causes the warning message "CAUTION: PLOT ABORTED" asserts EOI, and frees the computer.

Because of possible peripheral malfunctions, it is generally advisable to use pass-control mode, which allows two way communication between the peripherals and the analyzer.

Example 7A2: Controlling Peripherals Using Pass-Control Mode

If the analyzer is in pass-control mode and it receives a command telling it to plot, print, control a power meter, or store/load to disk, it sets bit 1 in the event-status register to indicate that it requires control of the bus. If the computer then uses the HP-IB pass-control command to pass control to the analyzer, the analyzer will take control of the bus and access the peripheral. When the analyzer no longer requires control, it will pass control back to the computer. For a discussion on the pass-control mode, see Chapter 2.

In this example, the pass-control mode is used to allow the network analyzer to dump a screen display to a printer.

Pass-control mode allows the analyzer to control the printer while sending the screen display to be printed. Once the printer-dump operation is complete, the analyzer passes control back to the controller and the controller continues programming the analyzer. The analyzer requests control from the instrument controller and the controller allows the analyzer to take control of the HP-IB and dump the plot. The instrument controller must not interact with the HP-IB while this remote analyzer control is taking place.

Note The analyzer assumes that the address of the computer is correctly stored in its HP-IB addresses menu under (LOCAL) ADDRESS: CONTROLLER. If this address is incorrect, control will not return to the computer. Similarly, if control is passed to the analyzer while it is in talker/listener mode, control will not return to the computer.

Control should not be passed to the analyzer before it has set event-status-register bit 1 making it Request Active Control. If the analyzer receives control before the bit is set, control is passed immediately back to the controller.

When the analyzer becomes the active system controller, it is free to address devices to talk and listen as required. The only functions denied the analyzer are the ability to assert the interface clear line (IFC), and the remote line (REN). These are reserved for the master system controller. As the active system controller, the analyzer can send and receive messages from printers, plotters, and disk drives.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The status registers are cleared.

- Bit 1 of ESR request control is set.
- The ESR interrupt for SRQ is enabled.
- The pass-control mode is enabled.
- The data is dumped to the printer.
- The program loops until the SRQ bit is set.
- The status byte is read with a serial poll.
- The program tests for bit 6, SRQ.
- If SRQ is detected, the program tests for pass control (bit 5 of the status byte).
- If the analyzer requests control, the system controller gives the analyzer control of the bus.
- The program loops and waits for the analyzer to complete the print dump.
- The controller reads the HP-IB interface status register.
- If bit 6 is active in the controller, control has returned from the analyzer to the controller.
- The status-byte assignments are cleared.
- The analyzer returns to continuous-sweep mode.
- The analyzer is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 7A2: Controlling Peripherals in Pass-Control Mode" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Sets a single sweep and then queries the analyzer to insure that the sweep is complete. The analyzer is now in the state used to dump its display to the printer.

`"CLES;` **(Clear Status)**

Clears the status register, the event-status registers, and the enable registers.

"ESE2;" (Event-Status Enable {Control-Summary Bit 1})

Enables the request-control bit when the analyzer is ready to perform a the print function over the HP-IB. This is part of the pass-control operation synchronization. The control-summary bit is enabled in the event-status register as bit 1, decimal 2.

When this bit is set, the event-status-register bit is set in the analyzer-status register as bit 5, decimal 32.

Bit 5, decimal 32 will indicate that a request control is active.

"SRE 32;" (Service-Request Enable {Bit 5 of the Status Byte})

Service request enable. Enables bit 5 of the analyzer's status byte so that an SRQ is generated.

"USEPASC;" (Enable Pass-Control Mode)

The analyzer sets bit 2 in the event-status register to request pass control from the controller. The controller recognizes the presence of the bit 2 by reading the status byte from the analyzer. In this example, a serial poll is performed by the analyzer to read the status byte and to test for bit 5 which indicates that an event has occurred in the event-status register.

The pass control operation has now been established. When the analyzer requests control, bit 1in the event status register will be set. This bit will cause bit 5 in the status byte to be set and generate an SQR. The analyzer's status byte is read by performing a serial poll until SRQ is detected. Once SRQ occurs, bit 5 is tested. If bit 5 is set the analyzer is requesting control.

"PRINALL;" (Print Display)

Copies the display, in raster graphics mode, to a printer.

If bit 2 has been set by the analyzer, pass control is being requested by the analyzer.

In this case, the executable "CLES;" must be entered from the controller to clear the status byte and continue the program.

If the controller finds a status byte bit other than status byte bit 5 has been enabled, the controller displays the message: "SRQ but not request pass control" and the program halts.

The system controller will pass control by addressing the analyzer to talk and sending the TCT message (take control). The controller relinquishes control to the analyzer when the ATN line is released. The analyzer sends the printer dump to the printer.

The analyzer will very briefly flash the message WAITING FOR CONTROL before actually receiving control and generating the printer output.

"Printing from analyzer and waiting for control" (Program Message)

Indicates that the system is waiting for the printing to complete. Once the printer dump is complete, the analyzer passes control back to the address stored as the controller address under the (LOCAL) SET ADDRESSES menu. The computer recognizes that the transfer has taken place by monitoring the active controller status bit in the HP-IB interface registers. The computer then exits the wait loop.

"Control returned from analyzer" (Program Message)

Indicates that the control has been passed from the analyzer to the controller. The program can now continue to interact with the analyzer. The computer releases remote control of the analyzer.

`"TALKLIST;"` **(Select Talker/Listener Mode)**

Enables the talker/listener mode in the analyzer.

`"CLES;"` **(Clear Status)**

Clears the status register, the event-status registers, and the enable registers.

`"CONT;"` **(Select Continuous-Sweep Mode)**

Places the analyzer in continuous-sweep mode.

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

The analyzer will briefly flash the message `WAITING FOR CONTROL`, before actually receiving control and generating the printer output. The computer will display the `Printing from analyzer and waiting for control` message.

When the printer output is complete, the analyzer passes control back to the address stored as the controller address under the (LOCAL) `SET ADDRESSES` menu. The computer will detect the return of active control and exit the wait loop. The controller will display the message `Control returned from analyzer` and then release the analyzer from remote control.

Note Because the program waits for the analyzer's request for control, it can be used to respond to front-panel requests as well. Remove the `"PRINALL;"` command from the program and run the program. Nothing will happen until the operator requests a print, plot, or disk access from the front panel of the analyzer. For example, press (LOCAL) (COPY) and `PRINT MONOCHROME`.

Example 7A3: Printing with the Serial Port

Note This program is stored as EXAMP7A3 on the "HP 8753D Programming Examples" disk received with the network analyzer.

This program will select the serial port and program the analyzer to copy its display to a printer. There are a number of commands associated with the serial and parallel ports which allow the programmer to configure the output modes, for example the baud rate and the handshake type used by the port and the printer. In this example, the serial port is configured by the program. The interface may also be configured from the analyzer's front-panel keys by pressing (LOCAL) `SET ADDRESSES` `PRINTER PORT`. This menu allows manual selection of the serial-interface parameters.

Since the HP-IB port is not being used for the copy operation, programming of the analyzer and measurement operations may continue once the copy operation has been initiated. An internal spooler in the analyzer's memory provides buffering of the printer operation. In the

example which follows, the status byte of the analyzer is checked to determine when the print operation is complete.

- The analyzer is initialized.
- A single sweep is taken and the analyzer is placed in hold mode.
- The status registers are cleared.
- The copy-complete bit is set and enabled.
- The printer operation and communication modes are set.
- The print command is sent.
- The analyzer is released from remote control and placed in continuous-sweep mode.
- The analyzer is polled until the status bit representing copy complete is detected.
- The analyzer is released from remote control and the program ends.

Tutorial Program Explanation

The following section explains "Example 7A3: Printing with the Serial Port" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

`"CLES;"` **(Clear Status)**

Clears the status register, the event-status registers, and the enable registers.

`"ESNB128;"` **(Enable Copy Complete)**

Enables the copy complete bit in event-status-register B.

`"SRE4;"` **(Enable Event-Status-Register B)**

Enables the copy complete bit in event-status-register B.

`"PRNTRAUTF OFF;"` **(Deactivate Printer Autofeed Function)**

Deactivates the printer's auto feed function.

`"PRNTYPTJ;"` **(Select Print Device {ThinkJet})**

An HP ThinkJet printer is selected as the print device.

"PRNPRTSERI;" **(Select Output Port {Serial})**

The display data is routed to the serial port.

"PRNTRBAUD 9600;" **(Select Baud Rate)**

The ThinkJet's baud rate is set to 9600 bits-per-second.

"PRNHNDSHK XON;" **(Select Handshake {Xon-Xoff})**

The Xon-Xoff printer handshake mode is selected

"PRINALL;" **(Print All)**

Prints all list values or operating and marker parameters in ASCII.

The entire screen display is output to the printer.

"PRINTING" **(Program Message)**

Acknowledges that the display information has been output to the printer.

"CONT;" **(Select Continuous-Sweep Mode)**

Places the analyzer in continuous-sweep mode.

The controller then performs a serial poll to test for bit 2 (data-entry-complete bit) enabled in event-status-register B. The serial poll is repeated until the bit is set at the end of the printing process.

The controller then displays the message: "DONE" on its display console.

The analyzer is then addressed and returned to local control with the HP-IB Meta- Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

Run the program. The analyzer is initialized, set to single-sweep mode, and a sweep is taken. The program sets the system up to print the analyzer's display to an HP ThinkJet printer connected to the interface. At this time, the analyzer can continue making measurements as the ThinkJet prints the display. When the analyzer display has finished printing, the controller displays the message: "DONE", the analyzer is released from HP-IB control, and the program ends.

Example 7B: Plotting to a File and Transferring the File Data to a Plotter

Note This program is stored as EXAMP7B on the "HP 8753D Programming Examples" disk received with the network analyzer.

Another report-generation technique is to transfer the plotter string to a disk file, and retrieve and plot the disk file at another time. Test time is increased when a plot occurs during the process. It may be more convenient to plot the data at another site or time. One solution to this problem is to capture the plot data using the controller and store it to a disk file. This disk file may then be read from the controller and the contents transferred to a plotter. This next example shows a method of accomplishing this task.

The analyzer is initialized without presetting the analyzer. The data that is in place on the analyzer is not disturbed by the program operation. A large string is dimensioned to hold the plotter commands as they are received from the analyzer. The length of this string depends upon the complexity of the analyzer's display. The analyzer is placed in the single-sweep mode and `OPC?;SING;` is used to make sure that operation is complete before plotting. The plotting begins with the `OUTPPLOT;` command.

The string transfer is ended by the controller detecting the EOI line which the analyzer pulls at the end of the transfer. The string transfer terminates and the plot data is now stored in a string in the analyzer.

These strings contain ASCII characters which represent the plotter commands in HP-GL (Hewlett-Packard Graphics Language). A disk file is created and the string is written into the file containing the display-plot commands.

Once the strings are transferred to the disk file, the file pointer is rewound and the data read out into a string for plotting. The string is sent to the plotter which uses the commands to generate a plot.

The following is an outline of the program's processing sequence:

- The system is initialized.
- The string for plotter commands is defined.
- The frequency span is swept once.
- The plotter output is requested and read into the plot string.
- A plot file is created in the controller.
- The plot string is stored into the disk file.
- The plot string is read from the disk file and sent to the plotter.
- The analyzer returns to continuous-sweep mode.
- The analyzer is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 7B: Plotting to a File and Transferring the File Data to a Plotter" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

Main Program

A large string is dimensioned to hold the plotter commands as they are received from the analyzer. The length of this string depends upon the complexity of the display on the analyzer.

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Sets a single sweep and then queries the analyzer to insure that the sweep is complete. The analyzer is now in the state used to dump its display to the printer.

`"OUTPPLOT;"` **(Output Plot Data in ASCII Format)**

Requests the analyzer to plot using the talker/listener mode. Outputs the plot string in ASCII format to the HP-IB port.

The plotter output is read into the string buffer as one large string. Once the plotter caoomands have been received, the prompt is displayed.

`"Plotter output complete. Press RETURN to store on disk."` **(Prompt)**

A data file is created and opened to receive the plotter commands. Once the file transfer takes place, another prompt is displayed.

A file is assigned to transmit the plot string.

Pressing ENTER will output the data to the plot file.

`"Plot to file is complete. Press RETURN to plot."` **(Prompt)**

When the operator presses ENTER, the file pointer is reset, the plot string is read from the file and sent to the plotter, and the controller displays the message: "Plot is complete. End of program."

`"CONT;"` **(Select Continuous-Sweep Mode)**

The analyzer is returned to continuous-sweep mode.

`"OPC?;WAIT;"` **(Wait for Reply with Operation Complete Query)**

The controller waits for the analyzer to return a one (1), signifying that the last command received from the controller has completed. Upon reading the one (1), the analyzer is addressed and returned to local control with the HP-IB Meta-Message: Go To Local (GTL) on the HP-IB and the program ends.

Running the Program

The program begins by initializing the analyzer and placing it into single-sweep mode. The plotter commands are captured into strings in the controller. The controller display prompts `Plotter output complete. Press RETURN to store on disk.` Pressing ENTER causes the data to be stored to disk. Once this task is complete, the program prompts once more, `Plot to file is complete. Press Return to plot.` After pressing ENTER again, the string output is sent to the plotter and the plot begins. Once the plot is complete, the program prompts `Plot is complete. End of program.` and the analyzer begins sweeping and returns to local control.

Utilizing PC-Graphics Applications Using the Plot File

You can use this Example 7B to generate a plot that can be read into a PC and used in several different graphics generation programs. HP-GL is a commonly recognized graphic format and may be used to transfer information to PC application programs such as CorelDRAW!®, Lotus Freelance® and other graphics packages. By importing the graphics data into these application packages, you can generate reports in many work-processors.

You can then use graphic-data files to generate the following:

- test results documentation
- data sheets from testing results
- archival information for a digital-storage medium

If you would like to create a disk file for graphics processing, modify the previous program to only store the plotter commands to the disk file. The PC will now have a DOS-format file (.hpg) file that can be imported and examined by the graphics package.

Example 7C: Reading ASCII Disk Files to the Instrument Controller's Disk File

---

Note This program is stored as EXAMP7C on the "HP 8753D Programming Examples" disk received with the network analyzer.

Another approach to accessing the analyzer's test results is to store the data onto a disk file from the analyzer. This operation generates an ASCII file of the analyzer data in a CITIFILE format. A typical file is generated by the next example and is shown below:

```
CITIFILE A.01.00
#NA VERSION HP8753C.04.13

NAME DATA
VAR FREQ MAG 11
DATA S[1,1] RI
SEG_LIST_BEGIN
SEG 100000000 200000000 11
SEG_LIST_END
BEGIN
8.30566E-1,-1.36749E-1
8.27392E-1,-1.43676E-1
8.26080E-1,-1.52069E-1
8.25653E-1,-1.60003E-1
8.26385E-1,-1.68029E-1
8.26507E-1,-1.77154E-1
8.26263E-1,-1.87316E-1
8.26721E-1,-1.97265E-1
8.2724E-1,-2.07611E-1
8.28552E-1,-2.19940E-1
8.29620E-1,-2.31109E-1
END
```

This data file is stored by the analyzer under remote control or manually from the front panel. See "Printing, Plotting, or Saving Measurement Results" of the *HP 8753D Network Analyzer User's Guide* for more details on manual operation. This program performs the same steps that are required to manually store a file from front panel.

Note

If a PC is being used as the system controller, the disk format will have to be transformed from LIF to PC-DOS format.

To accomplish this task, a file-transformation utility, "LIF2DOS.EXE" has been provided on the "HP 8753D Programming Examples" disk (HP part number 08753-10028).

Once transformed, the PC will now have a DOS-format file to read and interpret.

A more extensive utility called LIFUTIL allows file transformation in both directions and is available as a commercial product. Call the nearest HP sales and service office for ordering information.

This program stores a file in the same manner as an operator would store a file on to the analyzer's internal disk drive from the front panel.

This example explains the process of storing the data from the analyzer to a file on the internal disk drive. There is also a program to read the data from the file into a data array for further processing or reformatting to another file type. The internal drive will store in the same format that is present on the disk. A new disk may be formatted in either LIF or DOS. For the example, the assumption has been made that the format transformation has already taken place, and there is a file that can be read record by record, from which data can be retrieved.

The goal of this example is to recover an array of stimulus frequency along with the trace-data values. CITIFILES contain the real and imaginary values of each data point. Some further transformation will be required to obtain magnitude values, for example.

The disk file contents for this example are shown above. This file contains more information than will be used in this example. The file is accessed and the records read from the file and printed on the controller display to observe the actual file contents. The file pointer is reset and the records are then read and interpreted for their data contents.

The first six records are skipped for this example. The seventh record contains the stimulus-frequency values and the number of points in the trace. These values are read from the record. The frequency increment, or point spacing, is calculated and used later the frequency-data calculations for a point. Two more records are skipped and the next is the first record representing data values. The data values are read in a loop until the values for the number of points have been recovered from the file. The data values are tabulated and printed out on the controller display.

The following is an outline of the program's processing sequence:

- An I/O path is assigned to the analyzer.
- The system is initialized.
- A string is dimensioned to hold a file record.
- The analyzer operating state is set.
- The internal drive is selected for storage (only ASCII data is stored).
- A file name is entered and the data stored into it.
- The operator is prompted to move the disk to the controller disk drive.
- The disk file is read and the contents displayed.
- The file pointer is rewound.
- The file contents are converted to trace data.
- The frequency and complex-data pair is displayed for each point.

- The analyzer is restored to continuous-sweep mode.
- The analyzer is returned to local control and the program ends.

Tutorial Program Explanation

The following section explains "Example 7C: Reading ASCII Disk Files to the Instrument Controller's Disk File" at the instrument-command level. Individual commands used in the program and their affects on the system are explained. For a list of all of the network analyzer's remote commands, see "Command Reference."

Initializing the System

IFC **(HP-IB Meta-Message: Interface Clear)**

Resets the HP-IB interface by toggling the state of the IFC line and taking control of the bus, halting all operations for all connected instruments.

SDC **(HP-IB Meta-Message: Selected Device Clear)**

Resets the particular HP-IB interface within the addressed device. When addressed to the analyzer, this command will reset the bus driver chip, but not the analyzer state. Other HP-IB instruments may be affected differently by this command. In many cases, the instrument settings are reset to a power-on condition.

`"OPC?;PRES;"` **(Instrument Preset with Operation Complete Query)**

Resets the network analyzer operating state to the default state (preset). This command is similar to pressing the front-panel PRESET, but it does not initiate an analyzer self-test. The analyzer sends an ASCII character one (1) to the instrument controller when the preset operation is complete.

Main Program

`"STAR100MHZ;"` **(Select Start Frequency at 100 MHz)**

Selects 100 MHz as the analyzer's start frequency.

`"STOP200MHZ;"` **(Select Stop Frequency at 200 MHz)**

Selects 200 MHz as the analyzer's stop frequency.

`"POIN 11;"` **(Edit Segment Points {11})**

Sets the number of points in the trace to 11.

This reduced span and number of points are chosen to keep the example simple and reduce the effort to decode the data file. In actual use, the task could be more involved with list frequency sweeps or log sweeps.

`"OPC?;SING;"` **(Select Single-Sweep Mode with Operation Complete Query)**

Sets a single sweep and then queries the analyzer to insure that the sweep is complete before executing any further instructions. Once the sweep is complete (the analyzer has returned a one (1) to the controller), the limit test will be performed upon completion of the sweep cycle.

`"INTD;"` **(Select Storage Internal Disk)**

Selects the analyzer's internal disk as the data storage destination.

`"EXTMFORMON;"` **(Activate Store-Formatted-Data-Only Mode)**

Activates the analyzer's store-formatted-data-only mode.

`"EXTMDATOON;"` **(Activate Store-Data-Only Mode)**

Sets the store-data-only mode. This mode allows you to store only the data, not the accompanying files of instrument state and calibration data. This mode will not allow the analyzer to recall the data without the analyzer setups.

`"Enter data file name (5 chars)"` **(Operator Prompt)**

Prompts the operator to enter a 5-character name for the file. After entering, the program changes the file name characters to upper case.

`"TITF1"` **(Title Save-Register 1)**

Save-register 1 is titled and encoded with double-quotation marks using the file name input by the operator.

`"SAVUASCI;"` **(Select ASCII/CITIFILE Save Format)**

Saves the file as ASCII/CITIFILE data and completes the command sequence for file storage.

`"STOR1;"` **(Store to Disk)**

Stores the ASCII data to the disk file.

`"Place disk in controller disk drive, then press Return"` **(Operator Prompt)**

Note If you are using a PC as the instrument controller, the disk format will have to be transformed from LIF to PC-DOS format. Once transformed, the PC now will have a DOS-formatted file to read and interpret.

Remove the disk from the analyzer's disk file and place it into the controller disk drive.

The contents of this disk file are shown at the beginning of this example. The file records are read and then printed on the controller display. The file pointer is reset and the records are interpreted for their data contents.

The first 6 records are skipped for this example. The seventh record contains the stimulus frequency values and the number of points in the trace. These values are read from the record. The frequency increment, or point spacing, is calculated and used later to calculate the frequency data for a point. Two more records are skipped. The next record is the first of the data values. The data values are read in a loop until the number of points have been recovered from the file. The data values are tabulated and printed out on the controller display.

Index

Contents

Figures

Tables

3

HP BASIC Programming Examples

Introduction

This is an introduction to the remote operation of the HP 8753D network analyzer using an HP 9000 Series 300 computer. It is a tutorial introduction using BASIC programming examples to demonstrate the remote operation of the network analyzer. The examples used in this chapter are on the "HP 8753D HP BASIC Programming Examples" disk.

The user should be familiar with the operation of the analyzer before attempting to remotely control the analyzer via the Hewlett-Packard Interface Bus (HP-IB). See the *The HP 8753D Network Analyzer User's Guide* for analyzer operating information.

The Hewlett-Packard computers specifically addressed in these examples are the HP 9000 Series 300 computers, operating with BASIC 6.2.

This document is not intended to teach BASIC programming or to discuss HP-IB theory except at an introductory level. For more information concerning BASIC, see the Table 3-1 for a list of manuals supporting the BASIC revision being used. For more information concerning the Hewlett-Packard Interface Bus, see Table 3-2.

Table 3-1. Additional BASIC 6.2 Programming Information

Description	HP Part Number
HP BASIC 6.2 Programming Guide	98616-90010
HP BASIC 6.2 Language Reference (2 Volumes)	98616-90004
Using HP BASIC for Instrument Control, Volume I	82303-90001
Using HP BASIC for Instrument Control, Volume II	82303-90002

Table 3-2. Additional HP-IB Information

Description	HP Part Number
HP BASIC 6.2 Interface Reference	98616-90013
Tutorial Description of the Hewlett-Packard Interface Bus	5021-1927

Required Equipment

To run the examples in this chapter, in addition to the HP 8753D network analyzer, the following equipment is required:

Computer ..... HP 9000 Series 300
BASIC operating system ..... BASIC 6.2
"HP 8753D HP BASIC Programming Examples" disk 08753-10028
HP-IB interconnect cables ..... HP 10833A/B/C/D
Test device ..... such as a 125 MHz bandpass filter

Note

The HP 9000 Series 300 computer must have enough memory to store:

- BASIC 6.2 (4 MBytes of memory is required)
- the required binaries

Upon receipt, make copies of the "HP 8753D Programming Examples" disks. Label them "HP 8753D Programming Examples BACKUP". These disks will act as reserves in the event of loss or damage to the original disks.

Optional Equipment

50Ω type-N calibration kit	HP 85032B
Test port return cables	HP 11852D
Plotter	HP 7440A ColorPro
Printer	HP 2225A Thinkjet
Disk drive	HP 9122 or HP 9153 CS80

See the "Compatible Peripherals" chapter in the *HP 8753D Network Analyzer User's Guide* for a more complete list of compatible peripherals.

System Setup and HP-IB Verification

This section describes how to:

- Connect the test system.
- Set the test system addresses.
- Set the network analyzer's control mode.
- Verify the operation of the system's interface bus (HP-IB).

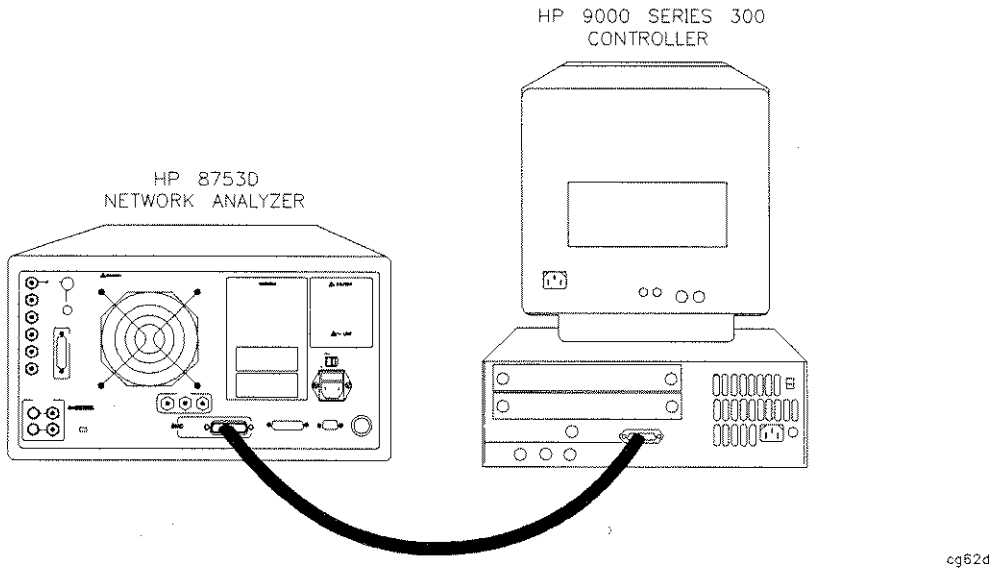


Figure 3-1. The HP 8753D Network Analyzer System with Controller

1. Connect the analyzer to the computer with an HP-IB cable as shown in Figure 3-1.
2. Switch on the computer.

3. Load the BASIC 6.2 operating system.
4. Switch on the analyzer.
   a. To verify the analyzer's address, press:

      (LOCAL) SET ADDRESSES ADDRESS: 8753

      The analyzer has only one HP-IB interface, though it occupies two addresses: one for the instrument and one for the display. The display address is equal to the instrument address with the least-significant bit incremented. The display address is automatically set each time the instrument address is set.

      The default analyzer addresses are:

      - 16 for the instrument
      - 17 for the display

Caution Other devices connected to the bus cannot occupy the same address as the analyzer.

The analyzer displays the instrument's address in the upper right section of the display. If the address is not 16, return the address to its default setting (16) by pressing:

(1) (6) (x1) (PRESET)

   b. Set the system control mode to either "pass-control" or "talker/listener" mode. These are the only control modes in which the analyzer will accept commands over HP-IB. For more information on control modes, see Chapter 2, "HP-IB Programming Reference." To set the system-control mode, press:

      (LOCAL) TALKER/LISTENER

      or

      (LOCAL) USE PASS CONTROL

5. Check the interface bus by performing a simple command from the computer controller. Type the following command on the controller:

   OUTPUT 716;"PRES;" (EXECUTE) or (RETURN)

Note HP 9000 Series 300 computers use the (RETURN) key as both execute and enter. Some other computers may have an (ENTER), (EXECUTE), or (EXEC) key that performs the same function. For reasons of simplicity, the notation (RETURN) is used throughout this chapter.

This command should preset the analyzer. If an instrument preset does not occur, there is a problem. Check all HP-IB addresses and connections. Most HP-IB problems are caused by an incorrect address and faulty/loose HP-IB cables.

HP 8753D Network Analyzer Instrument Control Using BASIC

A remote controller can manipulate the functions of the analyzer by sending commands to the analyzer via the Hewlett-Packard Interface Bus (HP-IB). The commands used are specific to the analyzer. Remote commands executed over the bus take precedence over manual commands executed from the instrument's front panel. Remote commands are executed as soon as they are received by the analyzer. A command only applies to the active channel (except in cases where functions are coupled between channels). Most commands are equivalent to front-panel hardkeys and softkeys.

Command Structure in BASIC

Consider the BASIC command for setting the analyzer's start frequency to 10 MHz:

```
OUTPUT 716;"STAR 10 MHZ;"
```

The command structure in BASIC has several different elements:

the BASIC command statement	`OUTPUT` - The BASIC data-output statement.
the appendage	`716` - The data is directed to interface 7 (HP-IB), and on to the device at address 16 (the HP 8753D). This appendage is terminated with a semicolon. The next appendage is `STAR`, the instrument mnemonic for setting the analyzer's start frequency.
data	`10` - a single operand used by the root mnemonic `STAR` to set the value.
unit	`MHZ` - the units that the operand is expressed in.
terminator	`;` - indicates the end of a command, enters the data, and deactivates the active-entry area.

The `"STAR 10 MHZ;"` command performs the same function as pressing the following keys on the anlyzer's front panel:

START 1 0 M/u

STAR is the root mnemonic for the start key, 10 is the data, and MHZ are the units. Where possible, the analyzer's root mnemonics are derived from the equivalent key label. Otherwise they are derived from the common name for the function. Chapter 1, "Command Reference," lists all the root mnemonics and all the different units accepted.

The semicolon (`;`) following `MHZ` terminates the command within the analyzer. It removes start frequency from the active-entry area, and prepares the analyzer for the next command. If there is a syntax error in a command, the analyzer will ignore the command and look for the next terminator. When it finds the next terminator, it starts processing incoming commands normally. Characters between the syntax error and the next terminator are lost. A line feed also acts as a terminator. The BASIC OUTPUT statement transmits a carriage return/line feed following the data. This can be suppressed by putting a semicolon at the end of the statement.

The `OUTPUT 716;` statement will transmit all items listed (as long as they are separated by commas or semicolons) including:

literal information enclosed in quotes,

numeric variables,

string variables,

and arrays.

A carriage return/line feed is transmitted after each item. Again, this can be suppressed by terminating the commands with a semicolon. The analyzer automatically goes into remote mode when it receives an OUTPUT command from the controller. When this happens, the front-panel remote (R) and listen (L) HP-IB status indicators illuminate. In remote mode, the analyzer ignores any data that is input with the front-panel keys, with the exception of (LOCAL). Pressing (LOCAL) returns the analyzer to manual operation, unless the universal HP-IB command LOCAL LOCKOUT 7 has been issued. There are two ways to exit from a local lockout. Either issue the LOCAL 7 command from the controller or cycle the line power on the analyzer.

Setting a parameter such as start frequency is just one form of command the analyzer will accept. It will also accept simple commands that require no operand at all. For example, execute:

```
OUTPUT 716;"AUTO;"
```

In response, the analyzer autoscales the active channel. Autoscale only applies to the active channel, unlike start frequency, which applies to both channels as long as the channels are stimulus-coupled.

The analyzer will also accept commands that switch various functions ON and OFF. For example, to switch on dual-channel display, execute:

```
OUTPUT 716;"DUACON;"
```

DUACON is the analyzer root mnemonic for "dual-channel display on". This causes the analyzer to display both channels. To go back to single-channel display mode, for example, switch off dual-channel display, execute:

```
OUTPUT 716;"DUACOFF;"
```

The construction of the command starts with the root mnemonic DUAC (dual-channel display,) and ON or OFF is appended to the root to form the entire command.

The analyzer does not distinguish between upper- and lower-case letters. For example, execute:

```
OUTPUT 716;"auto;"
```

Note The analyzer also has a debug mode to aid in troubleshooting systems. When the debug mode is ON, the analyzer scrolls incoming HP-IB commands across the display. To manually activate the debug mode, press (LOCAL) HP-IB DIAG ON. To deactivate the debug mode from the controller, execute:

```
OUTPUT 716;"DEBUOFF;"
```

Command Interrogate

Suppose the operator has changed the power level from the front panel. The computer can find the new power level using the analyzer's command-interrogate function. If a question mark is appended to the root of a command, the analyzer will output the value of that function.

For instance, POWE 7 DB; sets the analyzer's output power to 7 dB, and POWE?; outputs the current RF output power at the test port to the system controller. For example:

Type SCRATCH and press (RETURN) to clear old programs.

Type EDIT and press (RETURN) to access the edit mode.

Then type in:

```
10 OUTPUT 716;"POWE?;"
20 ENTER 716;Reply
30 DISP Reply
40 END
```

Running the Program

The computer will display the source-power level in dBm. The preset source-power level is 0 dBm. Change the power level by pressing (LOCAL) (MENU) POWER (1) (5) (x1). Now run the program again.

When the analyzer receives POWE?, it prepares to transmit the current RF source-power level. The BASIC statement ENTER 716 allows the analyzer to transmit information to the computer by addressing the analyzer to talk. This illuminates the analyzer front-panel talk (T) light. The computer places the data transmitted by the analyzer into the variables listed in the ENTER statement. In this case, the analyzer transmits the output power, which gets placed in the variable Reply.

The ENTER statement takes the stream of binary-data output from the analyzer and reformats it back into numbers and ASCII strings. With the formatting set to its default state, the ENTER statement will format the data into real variables, integers, or ASCII strings, depending on the variable being filled. The variable list must match the data the analyzer has to transmit. If there are not enough variables, data is lost. If there are too many variables for the data available, a BASIC error is generated.

The formatting done by the ENTER statement can be changed. For more information on data formatting, see Chapter 2, "HP-IB Programming Reference" under the section titled "Array Data Formats." The formatting can be deactivated to allow binary transfers of data. Also, the ENTER USING statement can be used to selectively control the formatting.

ON/OFF commands can be also be interrogated. The reply is a one (1) if the function is active, a zero (0) if it is not active. Similarly, if a command controls a function that is underlined on the analyzer softkey menu when active, interrogating that command yields a one (1) if the command is underlined, a zero (0) if it is not. For example, press (MEAS). Though there are seven options on the measurement menu, only one is underlined at a time. The underlined option will return a one (1) when interrogated.

For instance, rewrite line 10 as:

```
10  OUTPUT 716;"DUAC?;"
```

Run the program once and note the result. Then press (LOCAL) (DISPLAY) DUAL CHAN to toggle the display mode, and run the program again.

Another example is to rewrite line 10 as:

```
10  OUTPUT 716;"PHAS?;"
```

In this case, the program will display a one (1) if phase is currently being displayed. Since the command only applies to the active channel, the response to the PHAS? inquiry depends on which channel is active.

Held Commands

When the analyzer is executing a command that cannot be interrupted, it will not process any new HP-IB commands until the previous command has completed. These "uninterruptable" commands are called "held" commands. During the processing of a held command, the analyzer will fill the 16-character input buffer, and then halt all HP-IB operation until the held command has completed execution.

Note This action will be transparent to a program unless HP-IB timeouts have been set with the ON TIMEOUT statement.

While a held command is executing, the analyzer will still service the HP-IB interface commands, such as SPOLL(716), CLEAR 716, and ABORT 7. Executing CLEAR 716 or CLEAR 7 will abort a hold off, leaving the held command to complete execution as if it had been input from the front panel. These commands also clear the input buffer, destroying any commands received after the held command. If the analyzer has halted the bus because its input buffer was full, `ABORT 7` will release the bus.

Operation Complete

Occasionally, there is a need to query the analyzer as to when certain analyzer operations have completed. For instance, a program should not have the operator connect the next calibration standard while the analyzer is still measuring the current one. To provide such information, the analyzer has an "operation complete" reporting mechanism, or OPC command, that will indicate when certain key commands have completed operation. The mechanism is activated by sending either OPC or OPC? immediately before an OPC-compatible command. When the command completes execution, bit 0 of the event-status register will be set. If OPC was interrogated with OPC?, the analyzer will also output a one (1) when the command completes execution.

As an example, type `SCRATCH` and press (RETURN).

Type `EDIT` and press (RETURN).

Type in the following program:

```
10 OUTPUT 716;"SWET 3 S;OPC?;SING;"
20 DISP "SWEEPING"
30 ENTER 716;Reply
40 DISP "DONE"
50 END
```

Set the sweep time to 3 seconds, and OPC a single sweep.

The program will halt at this point until the analyzer completes the sweep and issues a one (1).

Running the Program

Running this program causes the computer to display the sweeping message as the instrument executes the sweep. The computer will display `DONE` just as the instrument goes into hold. When `DONE` appears, the program could then continue on, being assured that there is a valid data trace in the instrument.

Preparing for Remote (HP-IB) Control

At the beginning of a program, the analyzer is taken from an unknown state and brought under remote control. This is done with an abort/clear sequence. ABORT 7 is used to halt bus activity and return control to the computer. CLEAR 716 will then prepare the analyzer to receive commands by:

- clearing syntax errors
- clearing the input-command buffer
- clearing any messages waiting to be output

The abort/clear sequence readies the analyzer to receive HP-IB commands. The next step involves programming a known state into the analyzer. The most convenient way to do this is to preset the analyzer by sending the PRES (preset) command. If preset cannot be used, the status-reporting mechanism may be employed. When using the status-reporting register, CLES (Clear Status) can be transmitted to the analyzer to clear all of the status-reporting registers and their enables.

Type SCRATCH and press RETURN.

Type EDIT and press RETURN.

Type in the following program:

`10 ABORT 7`	*This halts all bus action and gives active control to the computer.*
`20 CLEAR 716`	*This clears all HP-IB errors, resets the HP-IB interface, and clears the syntax errors. It does not affect the status-reporting system.*
`30 OUTPUT 716;"PRES;"`	*Presets the instrument. This clears the status-reporting system, as well as resets all of the front-panel settings, except for the HP-IB mode and the HP-IB addresses.*
`40 END`	*Running this program brings the analyzer to a known state, ready to respond to HP-IB control.*

The analyzer will not respond to HP-IB commands unless the remote line is asserted. When the remote line is asserted, the analyzer is addressed to listen for commands from the controller. In remote mode, all the front-panel keys are disabled (with the exception of LOCAL and the line-power switch). ABORT 7 asserts the remote line, which remains asserted until a LOCAL 7 statement is executed. Another way to assert the remote line is to execute:

```
REMOTE 716
```

This statement asserts the analyzer's remote-operation mode and addresses the analyzer to listen for commands from the controller. Press any front-panel key except LOCAL. Note that none of the front-panel keys will respond until LOCAL has been pressed.

LOCAL can also be disabled with the sequence:

```
REMOTE 716
LOCAL LOCKOUT 7
```

After executing the code above, none of front-panel keys will respond. The analyzer can be returned to local mode temporarily with:

```
LOCAL 716
```

As soon as the analyzer is addressed to listen, it goes back into local-lockout mode. The only way to clear the local- lockout mode, aside from cycling line power, is to execute:

```
LOCAL 7
```

This command un-asserts the remote line on the interface. This puts the instrument into local mode and clears the local-lockout command. Return the instrument to remote mode by pressing:

(LOCAL) `TALKER/LISTENER`

or

(LOCAL) `USE PASS CONTROL`

I/O Paths

One of the features of HP BASIC is the use of input/output paths. The instrument may be addressed directly by the instrument's device number as shown in the previous examples. However, a more sophisticated approach is to declare I/O paths such as: `ASSIGN @Nwa TO 716`. Assigning an I/O path builds a look-up table in the computer's memory that contains the device-address codes and several other parameters. It is easy to quickly change addresses throughout the entire program at one location. I/O operation is more efficient because it uses a table, in place of calculating or searching for values related to I/O. In the more elaborate examples where file I/O is discussed, the look-up table contains all the information about the file. Execution speed is increased, because the computer no longer has to calculate a device's address each time that device is addressed.

For example:

Type `SCRATCH` and press (RETURN).

Type `EDIT` and press (RETURN).

Type in the following program:

```
10 ASSIGN @Nwa TO 716          Assigns the analyzer to ADDRESS 716.
20 OUTPUT @Nwa;"STAR 10 MHZ;"  Sets the analyzer's start frequency to 10 MHz.
```

Note The use of I/O paths in binary-format transfers allows the user to quickly distinguish the type of transfer taking place. I/O paths are used throughout the examples and are highly recommended for use in device input/output.

BASIC Programming Examples

The following sample programs provide the user with factory-tested solutions for several remotely-controlled analyzer processes. The programs can be used in their present state or modified to suit specific needs. The programs discussed in this section can be found on the "HP 8753D HP BASIC Programming Examples" disk received with the analyzer.

Example 1: Measurement Setup

The programs included in Example 1 provide the user the option to perform instrument-setup functions for the analyzer from a remote controller. Example 1A is a program designed to setup the analyzer's measurement parameters. Example 1B is a program designed to verify the measurement parameters.

Example 1A: Setting Parameters

Note This program is stored as EXAMP1A on the "HP 8753D Programming Examples" disk received with the network analyzer.

In general, the procedure for setting up measurements on the network analyzer via HP-IB follows the same sequence as if the setup was performed manually. There is no required order, as long as the desired frequency range, number of points, and power level are set prior to performing the calibration first, and the measurement second.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The analyzer is adjusted to measure return loss on channel 1 and display it in log magnitude.
- The analyzer is adjusted to measure return loss on channel 2 and display the phase.
- The dual-channel display mode is activated.
- The system operator is prompted to enter the frequency range of the measurement.
- The displays are autoscaled.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program selects the S-parameter to be measured, the display
20  ! format and then sets the specified start and stop frequencies.
30  ! The analyzer display is then autoscaled.
40  !
50  ! EXAMP1A
60  !
70  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
80  !
90  CLEAR SCREEN
100 ! Initialize the system
110 ABORT 7                               ! Generate an IFC (Interface Clear)
120 CLEAR @Nwa                            ! SDC (Selected Device Clear) analyzer
130 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
140 ENTER @Nwa;Reply                      ! Read in the 1 returned
150 !
160 ! Set up measurement and display
170 OUTPUT @Nwa;"CHAN1;"                  ! Channel 1
180 OUTPUT @Nwa;"S11;"                    ! Return Loss measurement
190 OUTPUT @Nwa;"LOGM;"                   ! Log magnitude display
200 !
210 OUTPUT @Nwa;"CHAN2;"                  ! Channel 2
220 OUTPUT @Nwa;"S11;"                    ! Return Loss measurement
230 OUTPUT @Nwa;"PHAS;"                   ! Phase display
240 !
250 OUTPUT @Nwa;"DUACON;"                 ! Dual channel display
260 !
270 ! Request start and stop frequency
280 INPUT "ENTER START FREQUENCY (MHz):",F_start
290 INPUT "ENTER STOP FREQUENCY (MHz):",F_stop
300 !
310 ! Program the analyzer settings
320 OUTPUT @Nwa;"STAR";F_start;"MHZ;"     ! Set the start frequency
330 OUTPUT @Nwa;"STOP";F_stop;"MHZ;"      ! Set the stop frequency
340 !
350 ! Autoscale the displays
360 OUTPUT @Nwa;"CHAN1;AUTO;"             ! Autoscale channel 1 display
370 OUTPUT @Nwa;"CHAN2;AUTO;"             ! Autoscale channel 2 display
380 !
390 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for the analyzer to finish
400 ENTER @Nwa;Reply                      ! Read the 1 when complete
410 LOCAL @Nwa                            ! Release HP-IB control
420 END
```

Running the Program

The analyzer is initialized and the operator is queried for the measurement's start and stop frequencies. The analyzer is setup to display the S11 reflection measurement as a function of log magnitude and phase over the selected frequency range. The displays are autoscaled and the program ends.

Example 1B: Verifying Parameters

Note This program is stored as EXAMP1B on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example shows how to read analyzer settings into your program. Chapter 1, "Command Reference," contains additional information on the command formats and operations. Appending a "?" to a command that sets an analyzer parameter will return the value of that setting. Parameters that are set as ON or OFF when queried will return a zero (0) if OFF or a one (1) if active. Parameters are returned in ASCII format, FORM 4. This format is varying in length from 1 to 24 characters-per-value. In the case of marker or other multiple responses, the values are separated by commas.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The number of points in the trace is queried and dumped to a printer.
- The start frequency is queried and output to a printer.
- The averaging is queried and output to a printer.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program performs some example queries of network analyzer
20  ! settings. The number of points in a trace, the start frequency
30  ! and if averaging is turned on, are determined and displayed.
40  !
50  ! EXAMP1B
60  !
70  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
80  !
90  CLEAR SCREEN
100 ! Initialize the system
110 ABORT 7                               ! Generate an IFC (Interface Clear)
120 CLEAR @Nwa                            ! SDC (Selected Device Clear)
130 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
140 ENTER @Nwa;Reply                      ! Read in the 1 returned
150 !
160 ! Query network analyzer parameters
170 OUTPUT @Nwa;"POIN?;"                  ! Read in the default trace length
180 ENTER @Nwa;Num_points
190 PRINT "Number of points ";Num_points
200 PRINT
210 !
220 OUTPUT @Nwa;"STAR?;"                  ! Read in the start frequency
230 ENTER @Nwa;Start_f
240 PRINT "Start Frequency ";Start_f
250 PRINT
260 !
270 OUTPUT @Nwa;"AVERO?;"                 ! Averaging on?
280 ENTER @Nwa;Flag
```

```
290 PRINT "Flag =";Flag;"   ";
300 IF Flag=1 THEN                          ! Test flag and print analyzer state
310   PRINT "Averaging ON"
320 ELSE
330   PRINT "Averaging OFF"
340 END IF
350 !
360 OUTPUT @Nwa;"OPC?;WAIT;"                ! Wait for the analyzer to finish
370 ENTER @Nwa;Reply                        ! Read the 1 when complete
380 LOCAL @Nwa                              ! Release HP-IB control
390 END
```

Running the Program

The analyzer is preset. The preset values are returned and printed out for: the number of points, the start frequency, and the state of the averaging function. The analyzer is released from remote control and the program ends.

Example 2: Calibration

Performing a Measurement Calibration

This section will demonstrate how to perform a measurement calibration over HP-IB. The HP-IB command sequence follows the key sequence required to perform a front-panel calibration. There is a command for every step of the sequence.

The remote measurement calibration sequence has three steps:

- selecting the calibration
- measuring the calibration standards
- declaring the calibration finished

The actual sequence depends on the calibration kit and changes slightly for 2-port calibrations, which are divided into three calibration sub-sequences.

Example 2A is a program designed to perform an S11 1-port calibration. Example 2B is a program designed to perform a full 2-port measurement calibration.

Example 2A: S11 1-Port Calibration

Note This program is stored as EXAMP2A on the "HP 8753D Programming Examples" disk received with the network analyzer.

The following program performs an S11 1-port calibration, using the HP 85032B 50Ω type-N calibration kit. This program simplifies the calibration by providing explicit directions on the analyzer display while allowing the user to run the program from the controller keyboard. More information on selecting calibration standards can be found in the Optimizing Measurement Results chapter of the *HP 8753D Network Analyzer User's Guide*.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The appropriate calibration kit is selected.
- The softkey menu is deactivated.
- The S11-calibration sequence is run.
- The S11-calibration data is saved.
- The softkey menu is activated.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program performs a 1-port calibration on the HP 8753.LOAD "EXAMP2A"
20  ! It guides the operator through a 1-port calibration
30  ! using the HP 85032B 50 ohm type N calibration kit.
40  !
50  ! The routine Waitforkey displays a message on the instrument's
60  ! display and the console, to prompt the operator to connect the
70  ! calibration standard. Once the standard is connected, the
80  ! ENTER key on the computer keyboard is pressed to continue.
90  !
100 ! EXAMP2A
110 !
120 ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
130 !
140 CLEAR SCREEN
150 ! Initialize the system
160 ABORT 7                               ! Generate an IFC (Interface Clear)
170 CLEAR @Nwa                            ! SDC (Selected Device Clear)
180 !
190 OUTPUT @Nwa;"CALKN50;"                ! Select CAL kit type
200 OUTPUT @Nwa;"MENUOFF;"                ! Turn softkey menu off.
210 !
220 OUTPUT @Nwa;"CALIS111;"               ! S11 1 port CAL initiated
230 !
240 CALL Waitforkey("CONNECT OPEN AT PORT 1")
250 OUTPUT @Nwa;"CLASS11A;"               ! Open reflection CAL
260 OUTPUT @Nwa;"OPC?;STANB;"             ! Select the second standard, B
270 ENTER @Nwa;Reply                      ! Read in the 1 returned
280 !
290 CALL Waitforkey("CONNECT SHORT AT PORT 1")
300 OUTPUT @Nwa;"CLASS11B;"               ! Short reflection CAL
310 OUTPUT @Nwa;"OPC?;STANB;"             ! Select the second standard, B
320 ENTER @Nwa;Reply                      ! Read in the 1 returned
330 !
340 CALL Waitforkey("CONNECT LOAD AT PORT 1")
350 OUTPUT @Nwa;"OPC?;CLASS11C;"          ! Reflection load CAL
360 ENTER @Nwa;Reply                      ! Read in the 1 returned
370 !
380 OUTPUT 717;"PG;"                      ! Clear the analyzer display
390 !
400 DISP "COMPUTING CALIBRATION COEFFICIENTS"
410 !
420 OUTPUT @Nwa;"DONE;"                   ! Finished with the CAL cycle
430 OUTPUT @Nwa;"OPC?;SAV1;"              ! Save the ONE PORT CAL
440 ENTER @Nwa;Reply                      ! Read in the 1 returned
450 !
460 DISP "S11 1-PORT CAL COMPLETED. CONNECT TEST DEVICE."
470 OUTPUT @Nwa;"MENUON;"                 ! Turn on the softkey menu
480 !
490 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for the analyzer to finish
500 ENTER @Nwa;Reply                      ! Read the 1 when complete
510 LOCAL @Nwa                            ! Release HP-IB control
520 !
530 END
```

```
540 !
550 ! ************************** Subroutines ******************************
560 !
570 Waitforkey:  ! Prompt routine to read a keypress on the controller
580 SUB Waitforkey(Lab$)
590   !   Position and display text on the analyzer display
600   OUTPUT 717;"PG;PU;PA390,3700;PD;LB";Lab$;", PRESS ENTER WHEN READY;"
610   !
620   DISP Lab$&"  Press ENTER when ready"; ! Display prompt on console
630   INPUT A$                              ! Read ENTER key press
640   !
650   OUTPUT 717;"PG;"                      ! Clear analyzer display
660 SUBEND
```

Running the Program

Note This program does not modify the instrument state in any way. Before running the program, set up the desired instrument state.

Run the program and connect the standards as prompted. When a standard is connected, press the (ENTER) on the controller keyboard to measure it.

The program assumes that the port being calibrated is a 50Ω type-N female test port. The prompts appear just above the message line on the analyzer display. Pressing (RETURN) on the controller keyboard continues the program and measures the standard. The program will display a message when the measurement calibration is complete.

Example 2B: Full 2-Port Measurement Calibration

Note This program is stored as EXAMP2B on the "HP 8753D Programming Examples" disk received with the network analyzer.

The following example program performs a full 2-port measurement calibration using the HP 85032B 50Ω type-N calibration kit. A full 2-port calibration removes both the forward- and reverse-error terms so all four S-parameters of the device under test can be measured. PORT 1 is a female test port and PORT 2 is a male test port.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The appropriate calibration kit is selected.
- The softkey menu is deactivated.
- The 2-port calibration sequence is run.
- The operator is prompted to choose or skip the isolation calibration.
- The softkey menu is activated.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program performs a full 2-port measurement calibration.
20  ! It guides the operator through a full 2-port calibration
30  ! using the HP 85032B 50 ohm type N calibration kit.
40  ! The routine Waitforkey displays a message on the instrument's
50  ! display and the console to prompt the operator to connect the
60  ! calibration standard. Once the standard is connected, the
70  ! ENTER key on the computer keyboard is pressed to continue.
80  !
90  ! EXAMP2B
100 !
110 ASSIGN @Nwa TO 716                    ! Assign an I/O path to the analyzer
120 !
130 CLEAR SCREEN
140 ! Initialize the analyzer
150 ABORT 7                               ! Generate an IFC (Interface Clear)
160 CLEAR @Nwa                            ! SDC (Selected Device Clear)
170 !
180 OUTPUT @Nwa;"CALKN50;MENUOFF;"     ! Select CAL kit type and turn off menu
190 !
200 OUTPUT @Nwa;"CALIFUL2;"               ! Full 2 port CAL
210 !
220 OUTPUT @Nwa;"REFL;"                   ! Reflection CAL
230 !
240 CALL Waitforkey("CONNECT OPEN AT PORT 1")
250 OUTPUT @Nwa;"CLASS11A;"               ! S11 open CAL
260 OUTPUT @Nwa;"OPC?;STANB;"             ! Select the second standard, B
270 ENTER @Nwa;Reply                      ! Read in the 1 returned
280 !
290 CALL Waitforkey("CONNECT SHORT AT PORT 1")
300 OUTPUT @Nwa;"CLASS11B;"               ! S11 short CAL
310 OUTPUT @Nwa;"OPC?;STANB;"             ! Select the second standard, B
320 ENTER @Nwa;Reply                      ! Read in the 1 returned
330 !
340 CALL Waitforkey("CONNECT LOAD AT PORT 1")
350 OUTPUT @Nwa;"OPC?;CLASS11C;"          ! S11 load CAL
360 ENTER @Nwa;Reply                      ! Read in the 1 returned
370 !
380 CALL Waitforkey("CONNECT OPEN AT PORT 2")
390 OUTPUT @Nwa;"CLASS22A;"               ! S22 open CAL
400 OUTPUT @Nwa;"OPC?;STANA;"             ! Select the first standard, A
410 ENTER @Nwa;Reply                      ! Read in the 1 returned
420 !
430 CALL Waitforkey("CONNECT SHORT AT PORT 2")
440 OUTPUT @Nwa;"CLASS22B;"               ! S22 short CAL
450 OUTPUT @Nwa;"OPC?;STANA;"             ! Select the first standard, A
460 ENTER @Nwa;Reply                      ! Read in the 1 returned
470 !
480 CALL Waitforkey("CONNECT LOAD AT PORT 2")
490 OUTPUT @Nwa;"OPC?;CLASS22C;"          ! S22 load CAL
500 ENTER @Nwa;Reply
510 !
520 DISP "COMPUTING REFLECTION CALIBRATION COEFFICIENTS"
530 !
```

```
540 OUTPUT @Nwa;"REFD;"                    ! Reflelction portion complete
550 !
560 OUTPUT @Nwa;"TRAN;"                    ! Transmission portion begins
570 !
580 CALL Waitforkey("CONNECT THRU [PORT1 TO PORT 2]")
590 DISP "MEASURING FORWARD TRANSMISSION"
600 OUTPUT @Nwa;"OPC?;FWDT;"               ! Measure forward transmission
610 ENTER @Nwa;Reply                       ! Read in the 1 returned
620 !
630 OUTPUT @Nwa;"OPC?;FWDM;"               ! Measure forward load match
640 ENTER @Nwa;Reply                       ! Read in the 1 returned
650 !
660 DISP "MEASURING REVERSE TRANSMISSION"
670 OUTPUT @Nwa;"OPC?;REVT;"               ! Measure reverse transmission
680 ENTER @Nwa;Reply                       ! Read in the 1 returned
690 !
700 OUTPUT @Nwa;"OPC?;REVM;"               ! Measure reverse load match
710 ENTER @Nwa;Reply                       ! Read in the 1 returned
720 !
730 OUTPUT @Nwa;"TRAD;"                    ! Transmission CAL complete
740 !
750 INPUT "SKIP ISOLATION CAL? Y OR N.",An$
760 IF An$="Y" THEN
770   OUTPUT @Nwa;"OMII;"                  ! Skip isolation cal
780   GOTO 940
790 END IF
800 !
810 CALL Waitforkey("ISOLATE TEST PORTS")
820 !
830 OUTPUT @Nwa;"ISOL;"                    ! Isolation CAL
840 OUTPUT @Nwa;"AVERFACT10;"              ! Average for 10 sweeps
850 OUTPUT @Nwa;"AVEROON;"                 ! Turn on averaging
860 DISP "MEASURING REVERSE ISOLATION"
870 OUTPUT @Nwa;"OPC?;REVI;"               ! Measure reverse isolation
880 ENTER @Nwa;Reply                       ! Read in the 1 returned
890 !
900 DISP "MEASURING FORWARD ISOLATION"
910 OUTPUT @Nwa;"OPC?;FWDI;"               ! Measure forward isolation
920 ENTER @Nwa;Reply                       ! Read in the 1 returned
930 !
940 OUTPUT @Nwa;"ISOD;AVEROOFF;"           ! Isolation complete averaging off
950 OUTPUT 717;"PG;"                       ! Clear analyzer display prompt
960 !
970 DISP "COMPUTING CALIBRATION COEFFICIENTS"
980 OUTPUT @Nwa;"DONE;"                    ! End the CAL sequence
990 OUTPUT @Nwa;"OPC?;SAV2;"               ! Save THE TWO PORT CAL
1000 ENTER @Nwa;Reply                      ! Read in the 1 returned
1010!
1020 DISP "DONE WITH FULL 2-PORT CAL. CONNECT TEST DEVICE."
1030 OUTPUT @Nwa;"MENUON;"                 ! Turn softkey menu on
1040 !
1050 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for the analyzer to finish
1060 ENTER @Nwa;Reply                      ! Read the 1 when complete
1070 LOCAL @Nwa                            ! Release HP-IB control
1080!
```

```
1090 END
1100!
1110! ************************ Subroutines ******************************
1120!
1130 SUB Waitforkey(Lab$)
1140   ! Position and display prompt on the analyzer display
1150  OUTPUT 717;"PG;PU;PA390,3700;PD;LB";Lab$;", PRESS ANY KEY WHEN READYA;"
1160   !
1170   DISP Lab$&"   Press ENTER when ready";   ! Display prompt on console
1180   INPUT A$                                 ! Read ENTER keypress on controller
1190   OUTPUT 717;"PG;"                         ! Clear analyzer display
1200 SUBEND
```

Running the Program

Note Before running the program, set the desired instrument state. This program does not modify the instrument state in any way.

Run the program and connect the standards as prompted. After the standard is connected, press (ENTER) on the controller keyboard to continue the program. The program assumes that the test ports being calibrated are 50Ω type-N, PORT 1 being a female test port and PORT 2 being a male test port. The HP 85032B 50Ω type-N Calibration Kit is used. The prompts appear just above the message line on the analyzer display. After the prompt is displayed, pressing (ENTER) on the computer console continues the program and measures the standard. The operator has the option of omitting the isolation calibration. If the isolation calibration is performed, averaging is automatically employed to insure a good calibration. The program will display a message when the measurement calibration is complete.

Example 3: Measurement Data Transfer

Trace information can be read out of the analyzer in several ways. Data can be read from the trace selectively using the markers, or the entire trace can be read out. If only specific information such as a single point on the trace or the result of a marker search is required, the marker output command can be used to read the information. If all the trace data is required, see Examples 3B through 3E.

Example 3A: Data Transfer Using Markers

Note This program is stored as EXAMP3A on the "HP 8753D Programming Examples" disk received with the network analyzer.

Markers are the simplest form of trace-data transfer. A marker may be positioned using one of three methods:

- by a frequency location
- by an actual data point location
- by a trace-data value

In the following example, the marker is positioned on the trace's maximum value. Once positioned on the trace, the trace data at that point can be read into the controller. The marker data is always returned in FORM 3, ASCII format. Each number is sent as a 24-character string. Characters can be digits, signs, or decimal points. All characters should be separated by commas. In the case of markers, three numbers are sent. The display format determines the values of the marker responses. See Table 3-3, "HP 8753D Network Analyzer Array-Data Formats."

When using trace data, it is important to control the network analyzer's sweep function (and therefore the trace data) from the computer. Using the computer to control the instrument's sweep insures that the data you read into the controller is in a quiescent or steady state. It also insures that the measurement is complete.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The selected frequency span is swept once.
- The marker is activated and placed on the maximum trace value.
- The three marker values are output to the controller and displayed.
- The instrument is returned to local control and the program ends.

The program is written as follows:

```
10  ! This program takes a sweep on the analyzer and turns on a marker.
20  ! The marker is positioned on the trace maximum and the marker data
30  ! is output in ASCII format.
40  !
50  ! EXAMP3A
60  !
70  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
80  !
90  CLEAR SCREEN
100 ! Initialize the analyzer
110 ABORT 7                               ! Generate an IFC (Interface Clear)
120 CLEAR @Nwa                            ! SDC (Selective Device Clear)
130 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
140 ENTER @Nwa;Reply                      ! Read in the 1 returned
150 !
160 OUTPUT @Nwa;"OPC?;SING"               ! Single sweep mode and wait
170 ENTER @Nwa;Reply                      ! Read 1 when sweep complete
180 !
190 OUTPUT @Nwa;"MARK1;"                  ! Turn on marker 1
200 OUTPUT @Nwa;"SEAMAX;"                 ! Find the maximum
210 !
220 OUTPUT @Nwa;"OUTPMARK;"               ! Request the current marker value
230 ENTER @Nwa;Value1,Value2,Stim         ! Read three marker values
240 !
250 ! Show the marker data received.
260 PRINT " Value 1"," Value 2","     Stimulus (Hz)"
270 PRINT Value1,Value2,Stim              ! Print the received values
280 PRINT
290 PRINT " Compare the active marker block with the received values"
300 !
310 LOCAL @Nwa                            ! Release HP-IB control
320 END
```

Running the Program

Run the program. The analyzer is preset and a sweep is taken. Marker 1 is enabled and positioned on the largest value in the trace. The marker is output to the controller and printed on the controller display. The analyzer is returned to local control. Position the marker using the RPG or data-entry keys, and compare the displayed value on the analyzer with the value that was transmitted to the controller.

Example 3B: Data Transfer Using FORM 4 (ASCII Transfer)

Note This program is stored as EXAMP3B on the "HP 8753D Programming Examples" disk received with the network analyzer.

Table 3-3. HP 8753D Network Analyzer Array-Data Formats

Format type	Type of Data	Bytes per Data Value	Bytes-per-point 2 data values	201 Bytes-per-trace	Total Bytes add header
FORM 1	Internal Binary	3	6	1206	1210
FORM 2	IEEE 32-bit Floating-Point	4	8	1608	1612
FORM 3	IEEE 64-bit Floating-Point	8	16	3216	3220
FORM 4	ASCII Numbers	24	50	10,050	10,050*
FORM 5	PC-DOS 32-bit Floating-Point	4	8	1206	1220

*No header is used in FORM 4.

The next most common data transfer is to transfer a trace array from the analyzer. Table 3-3 shows the relationship of the two values-per-point that are transferred to the analyzer. When FORM 4 is used, each number is sent as a 24-character string, each character represented by a digit, sign, or decimal point. Each number is separated from the previous number with a comma. Since there are two numbers-per-point, a 201-point transfer in FORM 4 takes 10,050 bytes. This form is useful only when input-data formatting is difficult with the instrument controller. Refer to Table 3-3 for a comparison with the other formats.

An example of a simple data transfer using FORM 4 (ASCII data transfer) is shown in this program. A fairly common requirement is to create frequency-amplitude data pairs from the trace data. No frequency information is included with the trace data transfer, because the frequency data must be calculated. Relating the data from a linear frequency sweep to frequency can be done by interrogating the analyzer start frequency, the frequency span, and the number of points in the sweep. Given that information, the frequency of point N in a linear frequency sweep is:

$$F = Start\_frequency + (N-1) \times Span/(Points-1)$$

Example 3B illustrates this technique. It is a straight-forward solution for linear uniform sweeps. For other sweep types, frequency data is more difficult to construct and may best be read directly from the analyzer's limit-test array. See Example 3D for an explanation of this technique.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The trace-data array is allocated.
- The trace length is set to 11.
- The selected frequency span is swept once.
- The FORM 4, ASCII format is set.
- The formatted trace is read from the analyzer and displayed.
- The frequency increments between the points are calculated.
- The marker is activated and placed at 30 kHz.
- The instrument is returned to local control and the program ends.

The program is written as follows:

```
10  ! This program shows an ASCII format trace data transfer using form 4.
20  ! The data is received as a string of ASCII characters, 24 characters
30  ! per data point and transferred into a real array in the controller. The
40  ! corresponding frequency data is calculated from the analyzer settings.
50  !
60  ! EXAMP3B
70  !
80  ASSIGN @Nwa TO 716                    ! Assign an I/O path to the analyzer
90  !
100 CLEAR SCREEN
110 ! Initialize
120 ABORT 7                               ! Generate an IFC (Interface Clear)
130 CLEAR @Nwa                            ! SDC (Selective Device Clear)
140 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer
150 ENTER @Nwa;Reply                      ! Read the 1 when complete
160 !
170 ! Trace values are two elements per point, display format dependent
180 DIM Dat(1:11,1:2)                     ! Trace data array
190 !
200 OUTPUT @Nwa;"POIN 11;"                ! Set trace length to 11 points
210 OUTPUT @Nwa;"OPC?;SING;"              ! Single sweep mode and wait
220 ENTER @Nwa;Reply                      ! Read reply
230 !
240 OUTPUT @Nwa;"FORM4;"                  ! Set form 4 ASCII format
250 OUTPUT @Nwa;"OUTPFORM;"               ! Send formatted trace to controller
260 ENTER @Nwa;Dat(*)                     ! Read in data array from analyzer
270 !
280 ! Now to calculate the frequency increments between points
290 OUTPUT @Nwa;"POIN?;"                  ! Read number of points in the trace
300 ENTER @Nwa;Num_points
310 OUTPUT @Nwa;"STAR?;"                  ! Read the start frequency
320 ENTER @Nwa;Startf
330 OUTPUT @Nwa;"SPAN?;"                  ! Read the span
340 ENTER @Nwa;Span
350 !
360 F_inc=Span/(Num_points-1)            ! Calculate fixed frequency increment
370 !
380 PRINT "Point","Freq (MHz)","  Value 1","  Value 2"
390 IMAGE 3D,7X,5D.3D,3X,3D.3D,3X,3D.3D   ! Formatting for controller display
400 !
410 FOR I=1 TO Num_points                 ! Loop through data points
420   Freq=Startf+(I-1)*F_inc             ! Calculate frequency of data point
430   PRINT USING 390;I,Freq/1.E+6,Dat(I,1),Dat(I,2)! Print analyzer data
440 NEXT I
450 !
460 OUTPUT @Nwa;"MARKDISC;"               ! Discrete marker mode
470 OUTPUT @Nwa;"MARK1 .3E+6;"            ! Position marker at 30 KHz
480 !
490 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for the analyzer to finish
500 ENTER @Nwa;Reply                      ! Read the 1 when complete
510 LOCAL 7                               ! Release HP-IB control
520 !
530 PRINT
```

```
540 PRINT "Position the marker with the knob and compare the values"
550 !
560 END
```

Running the Program

Run the program and watch the controller console. The analyzer will perform an instrument preset. The program will then print out the data values received from the analyzer. The marker is activated and placed at the left-hand edge of the analyzer's display. Position the marker with the knob and compare the values read with the active marker with the results printed on the controller console. The data points should agree exactly. Keep in mind that no matter how many digits are displayed, the analyzer is specified to measure:

- magnitude to a resolution of 0.001 dB
- phase to a resolution of 0.01 degrees
- group delay to a resolution of 0.01 ps

Example 3C: Data Transfer Using Floating-Point Numbers

Note This program is stored as EXAMP3C on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example program illustrates data transfer using FORM 3 in which data is transmitted in the floating-point formats. FORM 2 is nearly identical except for the IEEE 32-bit format of 4 bytes-per-value. FORM 5 reverses the order of the bytes to conform with the PC conventions for defining a real number.

The block-data formats have a four-byte header. The first two bytes are the ASCII characters "#A" that indicate that a fixed-length block transfer follows, and the next two bytes form an integer containing the number of bytes in the block to follow. The header must be read in so that data order is maintained.

This transfer is more than twice as fast than a FORM 4 transfer. With the FORM 4 transfer, 10,050 bytes are sent (201 points × 2 values-per-point × 24 bytes-per-value). Using FORM 2 to transfer the data, only 1612 bytes are sent (201 points × 2 values-per-point × 4 bytes-per-value). See "Array-Data Formats."

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The integer variables are defined to contain the header information.
- The number of points in the trace is set to 11.
- The selected frequency span is swept once.
- Data-transfer format 3 is set.
- The headers are read from the trace.
- The array size is calculated and allocated.
- The trace data is read in and printed.
- The marker is activated and placed at 30 kHz.
- The instrument is returned to local control and the program ends.

The program is written as follows:

```
10  ! This program shows how to read in a data trace in IEEE 64 bit
20  ! format. The array header is used to determine the length of the
30  ! array and to allocate the array size.
40  !
50  ! Program Example 3C
60  !
70  CLEAR SCREEN
80  ! Initialize the analyzer
90  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
100 ASSIGN @Nwadat TO 716;FORMAT OFF      ! Binary data path definition
110 !
120 ABORT 7                               ! Generate an IFC ( Interface Clear)
130 CLEAR @Nwa                            ! SDC (Selected Device Clear)
140 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
150 ENTER @Nwa;Reply                      ! Read the 1 when completed
160 !
170 INTEGER Dheader,Dlength               ! Integer variables for header info
180 Numpoints=11                          ! Number of points in the trace
190 OUTPUT @Nwa;"POIN";Numpoints;";"      ! Set number of points in trace
200 !
210 !  Set up data transfer
220 OUTPUT @Nwa;"OPC?;SING"               ! Single sweep and wait
230 ENTER @Nwa;Reply                      ! Read the 1 when completed
240 !
250 OUTPUT @Nwa;"FORM3;"                  ! Select form 3 format
260 OUTPUT @Nwa;"OUTPFORM;"               ! Send formatted output trace
270 !
280 ENTER @Nwadat;Dheader,Dlength         ! Read headers from trace data
290 !
300 ALLOCATE Dat(1:Dlength/16,1:2)        ! Use length to determine array size
310 ENTER @Nwadat;Dat(*)                  ! Read in trace data
320 !
330 PRINT "Size of array ";Dlength/16;" elements"
340 PRINT "Number of bytes ";Dlength
350 !
360 ! Print out the data array
370 PRINT "Element","Value 1","        Value 2"
380 IMAGE 3D,6X,3D.6D,6X,3D.6D
390 FOR I=1 TO Numpoints                  ! Loop through the data points
400   PRINT USING 380;I,Dat(I,1),Dat(I,2)
410 NEXT I
420 !
430 OUTPUT @Nwa;"MARKDISC;"               ! Discrete marker mode
440 OUTPUT @Nwa;"MARK1 .3E+6;"            ! Position marker at 30 KHz
450 !
460 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for the analyzer to finish
470 ENTER @Nwa;Reply                      ! Read the 1 when complete
480 LOCAL @Nwa                            ! Release HP-IB control
490 !
500 PRINT
510 PRINT "Position the marker with the knob and compare the values."
520 !
530 END
```

Running the Program

Run the program. The computer displays the number of elements and bytes associated with the transfer of the trace, as well as the first 10 data points. Position the marker and examine the data values. Compare the displayed values with the analyzer's marker values.

Example 3D: Data Transfer Using Frequency-Array Information

Note This program is stored as EXAMP3D on the "HP 8753D Programming Examples" disk received with the network analyzer.

Example 3C was used to read in the trace-data array. Example 3D explains how to use the limit-test array to read in the when corresponding frequency values for the completed trace array. The analyzer is set to sweep from 10 MHz to 200 MHz in log-frequency mode with the number of points in the trace set to 11. This makes it very difficult to compute the frequency-point spacing in the trace. The points are equally spaced across the trace, but not equally spaced in relation to frequency (because the frequency span is displayed in a logarithmic scale, as opposed to a linear scale). The limit-test data array may be read from the analyzer to provide the frequency values for each data point. All four values-per-trace location must be read from the analyzer. The test results and limit values are not used in this example. Only the frequency values are used. This technique is the only method of obtaining the non-linear frequency data from the analyzer's display. The test data and frequencies are printed on the controller display and the marker enabled to allow the operator to examine the actual locations on the analyzer's display.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The integer variables for the header information are defined.
- The number of points in the trace is set to 11.
- The frequency span (10 MHz to 200 MHz) is selected.
- The log-frequency sweep is selected.
- The data-transfer format 3 is set.
- The headers are read from the trace.
- The array size is calculated and allocated.
- The trace data is read in.
- The limit-test array is calculated and allocated.
- The limit-line test array is read in.
- The table header is printed.
- The program cycles through the trace values.
- The trace data and frequency are printed.
- The discrete-marker mode is activated.
- The marker is activated and placed at 10 MHz.
- The instrument is returned to local control and the program ends.

The program is written as follows:

```
10  ! This program shows how to read in a trace and create the frequency
20  ! value associated with the trace data value. EXAMP3C is used to
30  ! read in the data from the analyzer. The start and stop
40  ! frequencies are set to provide two decades of log range. Log sweep
50  ! is set and the frequency data points are read from the limit test
60  ! array and displayed with the data points.
70  !
80  ! EXAMP3D
90  !
100 ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
110 ASSIGN @Nwadat TO 716;FORMAT OFF      ! Binary path for data transfer
120 !
130 CLEAR SCREEN
140 ! Initialize the analyzer
150 ABORT 7                               ! Generate an IFC ( Interface Clear)
160 CLEAR @Nwa                            ! SDC (Selective Device Clear)
170 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
180 ENTER @Nwa;Reply                      ! Read the 1 when completed
190 !
200 INTEGER Dheader,Dlength               ! Integer variables for header info
210 !
220 OUTPUT @Nwa;"POIN 11;"                ! Set trace length to 11 points
230 OUTPUT @Nwa;"STAR 10.E+6;"            ! Start frequency 10 MHz
240 OUTPUT @Nwa;"STOP 200.E+6;"           ! Stop frequency 200 MHz
250 OUTPUT @Nwa;"LOGFREQ;"                ! Set log frequency sweep
260 !
270 !  Set up data transfer
280 OUTPUT @Nwa;"OPC?;SING"               ! Single sweep and wait
290 ENTER @Nwa;Reply                      ! Read the 1 when completed
300 !
310 OUTPUT @Nwa;"FORM3;"                  ! Select form 3 trace format
320 OUTPUT @Nwa;"OUTPFORM;"               ! Output formatted trace
330 !
340 ENTER @Nwadat;Dheader,Dlength         ! Read headers from trace data
350 !
360 ALLOCATE Dat(1:Dlength/16,1:2)        ! Use length to determine array size
370 ENTER @Nwadat;Dat(*)                  ! Read in trace data
380 !
390 ! Create the corresponding frequency values for the array
400 !
410 ! Read the frequency values using the limit test array
420 ALLOCATE Freq(1:Dlength/16,1:4)       ! Limit line results array
430 ! Limit line values are frequency, test results, upper and lower limits
440 !
450 OUTPUT @Nwa;"OUTPLIML;"               ! Request limit line test results
460 ENTER @Nwa;Freq(*)                    ! Read 4 values per point
470 !
480 ! Display table of freq and data
490 !
500 PRINT " Freq (MHz)","Mag (dB)"        ! Print table header
510 FOR I=1 TO 11                         ! Cycle through the trace values
520   Freqm=Freq(I,1)/1.E+6               ! Convert frequency to MHz
530   PRINT USING "4D.6D,9X,3D.3D";Freqm,Dat(I,1)  ! Print trace data
```

```
540 NEXT I
550 !
560 ! Set up marker to examine frequency values
570 OUTPUT @Nwa;"MARKDISC;"              ! Discrete marker mode
580 OUTPUT @Nwa;"MARK1 10.E+6;"          ! Turn on marker and place at 10 MHz
590 !
600 OUTPUT @Nwa;"OPC?;WAIT;"             ! Wait for the analyzer to finish
610 ENTER @Nwa;Reply                     ! Read the 1 when complete
620 LOCAL @Nwa                           ! Release HP-IB control
630 PRINT                                ! Blank line
640 PRINT "Position marker and observe frequency point spacing"
650 !
660 END
```

Running the Program

Run the program. Observe the controller display. The corresponding frequency values are shown with the trace-data values. Position the marker and observe the relationship between the frequency values and the point spacing on the trace. Compare the trace-data values on the analyzer with those shown on the controller display.

Example 3E: Data Transfer Using FORM 1, Internal-Binary Format

Note This program is stored as EXAMP3E on the "HP 8753D Programming Examples" disk received with the network analyzer.

FORM 1 is used for rapid I/O transfer of analyzer data. It contains the least number of bytes-per-trace and does not require re-formatting in the analyzer. This format is more difficult to convert into a numeric array in the controller. Analyzer-state information, such as learn strings and calibration arrays, may be easily transferred in this format because data conversion is not required. Recalling an instrument state that has been stored in a file and transferring instrument state information to the analyzer are excellent applications of a FORM 1 data transfer.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The integer variables for the header information are defined.
- The string variable for the header is defined.
- The selected frequency span is swept once.
- The internal-binary format is selected.
- The error-corrected data is output from the analyzer.
- The two data-header characters and the two length bytes are read in.
- The string buffer is allocated for data.
- The trace data is read into the string buffer.
- The analyzer is restored to continuous-sweep mode and queried for command completion.
- The instrument is returned to local control and the program ends.

The program is written as follows:

```
10  ! This program is an example of a form 1, internal format data
20  ! transfer. The data is stored in a string dimensioned to the
30  ! length of the data being transferred.
40  !
50  ! EXAMP3E
60  !
70  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
80  ASSIGN @Nwa_bin TO 716;FORMAT OFF     ! Binary path for data transfer
90  !
100 CLEAR SCREEN
110 ! Initialize the analyzer
120 ABORT 7                               ! Send IFC Interface Clear
130 CLEAR @Nwa                            ! SDC (Selective Device Clear)
140 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
150 ENTER @Nwa;Reply                      ! Read the 1 when completed
160 !
170 INTEGER Length                        ! Header length 2 bytes
180 DIM Header$[2]                        ! Header string 2 bytes
190 !
200 OUTPUT @Nwa;"OPC?;SING;"              ! Single sweep and wait
210 ENTER @Nwa;Reply                      ! Read the 1 when completed
220 !
230 OUTPUT @Nwa;"FORM1;"                  ! Select internal binary format
240 OUTPUT @Nwa;"OUTPDATA;"               ! Output error corrected data
250 !
260 ! Read in the data header two characters and two bytes for length
270 ! "#,2A"
280 !       # no early termination, terminate when ENTER is complete
290 !       2A read two chars
300 !
310 ENTER @Nwa_bin USING "#,2A";Header$   ! Read header as 2 byte string
320 ENTER @Nwa_bin;Length                 ! Read length as 2 byte integer
330 PRINT "Header ";Header$,"Array length";Length
340 !
350 ALLOCATE Data$[Length]                ! String buffer for data bytes
360 ! "+,-K" format statement
370 ! + EOI as a terminator LF is suppressed and read as data
380 ! -K All characters are read and not interpreted LF is included
390 ENTER @Nwa_bin USING "+,-K";Data$     ! Read trace into string array
400 !
410 PRINT "Number of bytes received ";LEN(Data$)
420 !
430 OUTPUT @Nwa;"CONT;"                   ! Restore continuous sweep
440 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for the analyzer to finish
450 ENTER @Nwa;Reply                      ! Read the 1 when complete
460 !
470 LOCAL @Nwa                            ! Release HP-IB control
480 END
```

Running the Program

The analyzer is initialized. The header and the number of bytes in the block transfer are printed on the controller display. Once the transfer is complete, the number of bytes in the data string is printed. Compare the two numbers to be sure that the transfer was completed.

Example 4: Measurement Process Synchronization

Status Reporting

The analyzer has a status-reporting mechanism that provides information about specific internal analyzer functions and events. The status byte is an 8-bit register with each bit summarizing the state of one aspect of the instrument. For example, the error-queue summary bit will always be set if there are any errors in the queue. The value of the status byte can be read with the HP-IB serial-poll operation. Serial Poll is described in Chapter 2, "HP-IB Programming Reference" under the section titled "Response to HP-IB Meta-Message (IEEE-488 Universal Commands)." This command does not automatically put the instrument in remote mode, thus giving the operator access to the analyzer front-panel functions. The status byte can also be read by sending the command OUTPSTAT. Reading the status byte does not affect its value.

The status byte also summarizes two event-status registers that monitor specific conditions within the instrument. The status byte also has a bit that is set when the instrument is issuing a service request over HP-IB and a bit that is set when the analyzer has data to send out over HP-IB. See "Error Reporting" in Chapter 2 of this guide for a discussion of the event-status registers.

Example 4A: Using the Error Queue

Note This program is stored as EXAMP4A on the "HP 8753D Programming Examples" disk received with the network analyzer.

The error queue holds up to 20 instrument errors and warnings in the order that they occurred. Each time the analyzer detects an error condition, the analyzer displays a message on the CRT, and puts the error in the error queue. If there are any errors in the queue, bit 3 of the status byte will be set. The errors can be read from the queue with the OUTPERRO command. OUTPERRO causes the analyzer to transmit the error number and message of the oldest error in the queue.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The error-message string is allocated.
- The analyzer is released from remote control.
- The program begins an endless loop to read the error queue.
- The status byte is read with a serial poll.
- The program tests to see if an error is present in the queue.
- The error-queue bit is set.
- The program requests the content of the error queue.
- The error number and string are read.
- The error messages are printed until there are no more errors in the queue.
- The instrument is returned to local control.
- The controller emits a beep to attach the attention of the operator and resumes searching for errors.

The program is written as follows:

```
10  ! This program is an example of using the error queue to detect
20  ! errors generated by the analyzer. The status byte is read and
30  ! bit 3 is tested to determine if an error exists. The error queue
40  ! is printed out and emptied.
50  !
60  ! EXAMP4A
70  !
80  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
90  !
100 CLEAR SCREEN
110 ! Initialize the analyzer
120 ABORT 7                               ! Generate an IFC (Interface Clear)
130 CLEAR @Nwa                            ! SDC (Selective Device Clear)
140 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
150 ENTER @Nwa;Reply                      ! Read the 1 when complete
160 !
170 DIM Error$[50]                        ! String for analyzer error message
180 !
190 LOCAL @Nwa                            ! Release analyzer from remote control
200 !
210 LOOP                                  ! Endless loop to read error queue
220   REPEAT
230     Stat=SPOLL(@Nwa)                  ! Read status byte with serial poll
240   UNTIL BIT(Stat,3)                   ! Test for error queue present
250   !
260 !                 Error queue bit is set
270   REPEAT                              ! Loop until error number is 0
280     OUTPUT @Nwa;"OUTPERRO;"           ! Request error queue contents
290     ENTER @Nwa;Err,Error$             ! Read error number and string
300     PRINT Err,Error$                  ! Print error messages
310   UNTIL Err=0                         ! No more errors in queue
320 !
330   LOCAL @Nwa                          ! Release analyzer from remote
340   BEEP 600,.2                         ! Beep to attract attention
350 END LOOP                              ! Repeat error search
360 !
370 END
```

Running the Program

Run the program. The analyzer goes through the preset cycle. Nothing will happen at first. The program is waiting for an error condition to activate the error queue. To cause an error, press a blank softkey. The message `CAUTION: INVALID KEY` will appear on the analyzer. The computer will beep and print out two error messages. The first line will be the invalid key error message, and the second line will be the `NO ERRORS` message. To clear the error queue, you can either loop until the `NO ERRORS` message is received, or until the bit in the status register is cleared. In this case, we wait until the status bit in the status register is clear. Note that while the program is running, the analyzer remains in the local mode and the front-panel keys may be accessed.

The error queue will hold up to 20 errors until all the errors are read out or the instrument is preset. It is important to clear the error queue whenever errors are detected. Otherwise, old errors may be mistakenly associated with the current instrument state. Press the unlabeled key

several times quickly and watch the display. The number of errors observed should correspond to the number of times you pressed the key.

As another example, press (CAL). Then the CALIBRATE MENU key. Select the RESPONSE calibration. Press DONE RESPONSE without performing any calibrations. Note the error message on the analyzer and on the controller display. Push the THROUGH key and then DONE RESPONSE. We are not concerned with the validity of the calibration, just setting a simple calibration on the analyzer. Note that COR is displayed in the upper left-hand section of the graticule. Now, press START and (↑). This will generate an error because the start frequency has been changed, invalidating the calibration. This error is reported on the controller display as well. A complete list of error messages and their descriptions can be found in Chapter 10 of the *HP 8753D Network Analyzer User's Guide.*

The program is in an infinite loop waiting for errors to occur. End the program by halting it with RESET or BREAK.

Note Not all messages displayed by the analyzer are put in the error queue: operator prompts and cautions are not included.

Example 4B: Generating Interrupts

Note This program is stored as EXAMP4B on the "HP 8753D Programming Examples" disk received with the network analyzer.

It is also possible to generate interrupts using the status-reporting mechanism. The status-byte bits can be enabled to generate a service request (SRQ) when set. In turn, the instrument controller can be set up to generate an interrupt on the SRQ and respond to the condition which caused the SRQ.

To generate an SRQ, a bit in the status byte is enabled using the command SREn. A one (1) in a bit position enables that bit in the status byte. Hence, SRE 8 enables an SRQ on bit 3, the check-error queue, since the decimal value 8 equals 00001000 in binary representation. Whenever an error is put into the error queue and bit 3 is set, the SRQ line is asserted, illuminating the (S) indicator in the HP-IB status block on the front panel of the analyzer. The only way to clear the SRQ is to disable bit 3, re-enable bit 3, or read out all the errors from the queue.

A bit in the event-status register can be enabled so that it is summarized by bit 5 of the status byte. If any enabled bit in the event-status register is set, bit 5 of the status byte will also be set. For example ESE 66 enables bits 1 and 6 of the event-status register, since in binary, the decimal number 66 equals 01000010. Hence, whenever active control is requested or a front-panel key is pressed, bit 5 of the status byte will be set. Similarly, ESNBn enables bits in event-status register B so that they will be summarized by bit 2 in the status byte.

To generate an SRQ from an event-status register, enable the desired event-status register bit. Then enable the status byte to generate an SRQ. For instance, ESE 32;SRE 32; enables the syntax-error bit. When the syntax-error bit is set, the summary bit in the status byte will be set. This will, in turn, enable an SRQ on bit 5 of the status byte, the summary bit for the event-status register.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.

- The status registers are cleared.
- The event-status register bit 5 is enabled.
- The status-register bit 5 is enabled.
- The interrupt pointer is enabled and points to a subroutine.
- Two bad commands are set to the analyzer to generate errors.
- The controller reads a serial-poll byte from HP-IB in the event of an interrupt.
- The program tests for an SRQ.
- If the SRQ is not generated by the analyzer, the subroutine stops and displays `SRQ FROM OTHER DEVICE`.
- If the SRQ was generated by the analyzer, the program reads the status byte and event-status register.
- If bit 5 in the event-status register is set, program prints: `SYNTAX ERROR FROM ANALYZER`.
- If bit 5 in the event-status register is NOT set, program prints: `SYNTAX ERROR BIT NOT SET`.
- The SRQ interrupt is re-enabled on the bus.
- At the finish, the interrupt is deactivated.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program is an example of using an SRQ based interrupt to
20  ! detect an error condition in the analyzer. In this example, a
30  ! syntax error is generated with an invalid command. The status byte
40  ! is read in and tested. The error queue is read, printed out and
50  ! then cleared.
60  !
70  ! EXAMP4B
80  !
90  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
100 !
110 CLEAR SCREEN
120 ! Initialize the analyzer
130 ABORT 7                               ! Generate and IFC (Interface Clear)
140 CLEAR @Nwa                            ! SDC (Selective Device Clear)
150 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
160 ENTER @Nwa;Reply                      ! Read the one from the analyzer
170 !
180 DIM Error$[50]                        ! String for analyzer error message
190 ! Set up syntax error interrupt
200 OUTPUT @Nwa;"CLES;"                   ! Clear the status registers
210 !
220 ! Generate SRQ when bit 5 is set
230 OUTPUT @Nwa;"ESE 32;"                 ! Event status register bit 5 enabled
240 !
250 ! Generate bit 5 in status register when syntax error occurs
260 OUTPUT @Nwa;"SRE 32;"                 ! Status register bit 5 enabled
270 !
280 ! Setup the interrupt pointer to a subroutine
290 ON INTR 7 GOSUB Srq_det               ! When interrupt occurs go to Srq_det
300 Stat=SPOLL(@Nwa)                      ! Clear any pending SRQs
```

```
310 ENABLE INTR 7;2                          ! Set interrupt on HP-IB bit 2 (SRQ)
320 !
330 DISP "Waiting for bad syntax"
340 WAIT 2                                   ! Pause for 2 seconds
350 !
360 OUTPUT @Nwa;"STIP 2GHZ;;"                ! Send bad STOP command syntax
370 !
380 WAIT 2                                   ! Pause for 2 seconds
390 DISP ""                                  ! Clear display line
400 GOTO Finish                              ! Exit program example
410 !
420 !************************ Subroutines *****************************
430 !
440 Srq_det:                                 ! SRQ handler
450 Stat=SPOLL(@Nwa)                         ! Read serial poll byte from HP-IB
460 PRINT "Stat from Serial Poll";Stat
470 IF BIT(Stat,6) THEN                      ! Test for SRQ
480   PRINT "SRQ received from analyzer"
490 ELSE                                     ! No SRQ from analyzer
500   PRINT "SRQ from other device"
510   STOP                                   ! Stop if not from analyzer
520 END IF
530 !
540 IF BIT(Stat,5) THEN                      ! Event status register bit set
550   PRINT "Event Status Register caused SRQ"
560 ELSE                                     ! Some other bit set
570   PRINT "Some other bit caused the SRQ"
580   STOP                                   ! Stop if bit not set
590 END IF
600   !
610 REPEAT
620   OUTPUT @Nwa;"OUTPERRO;"                ! Read analyzer error queue
630   ENTER @Nwa;Err,Error$                  ! Read error number and string
640   PRINT Err,Error$                       ! Print error message
650 UNTIL Err=0                              ! No more errors in queue
660 !
670 PRINT                                    ! White space
680 ENABLE INTR 7;2                          ! Re-enable SRQ interrupt on HP-IB
690 RETURN
700 !
710 !************************ End Subroutines *****************************
720 !
730 Finish:                                  ! End of program and exit
740 DISP "Finished"
750 OFF INTR 7                               ! Turn off interrupt
760 LOCAL @Nwa                               ! Release HP-IB control
770 END
```

Running the Program

Run the program. The computer will preset the analyzer, then pause for a second or two. After pausing, the program sends an invalid command string "STIP 2 GHZ;" to cause a syntax error. This command is intended to be "STOP 2 GHZ;". The computer will display a series of messages from the SRQ-handler routine. The analyzer will display CAUTION: SYNTAX ERROR and the incorrect command, pointing to the first character it did not understand.

The SRQ can be cleared by reading the event-status register and clearing the latched bit, or by clearing the enable registers with CLES. The syntax-error message on the analyzer display can only be cleared by the HP-IB Device Clear (DCL) message or Selected Device Clear (SDC) message. Device Clear is not commonly used because it clears every device on the bus. Selected Device Clear can be used to reset the input and output queue and the registers of a specific instrument on the bus. This will also clear all the interrupt definitions.

After a 4-second pause, the out-of-range command, "STAR 10 HZ;", is sent to the analyzer. Note that an impossible data condition does not generate a syntax error. Sending the out-of-range command will not generate an error on the display, but it will generate a syntax error. The analyzer simply sets the start frequency to 30 kHz, the minimum start frequency of the analyzer. Syntax errors are created when a command cannot be interpreted by the input parser of the analyzer. The controller releases the analyzer from remote control and the program ends.

Example 4C: Power Meter Calibration

Note This program is stored as EXAMP4C on the "HP 8753D Programming Examples" disk received with the network analyzer.

For increased accuracy during calibration of the 8753D's PORT 1-output power, a power meter calibration is available. This measurement-accuracy enhancement technique is described in the Optimizing Measurement Results chapter of the *HP 8753D Network Analyzer User's Guide*. The example described will perform the sample and sweep calibration under HP-IB remote control.

The power meter is usually connected to PORT 1 for the forward measurements. Its address must be set correctly and it must be connected to the HP-IB. The power meter address can be set by pressing: (LOCAL) SET ADDRESSES ADDRESS P MTR/HPIB and using the ⇧ and ⇩ keys to complete the process. The appropriate command must be selected for the model number of power meter being used. Press POWER MTR: [ ] until the model being used is displayed between the brackets. All of these steps are explained in the Compatible Peripherals chapter in the *HP 8753D Network Analyzer User's Guide*.

The correction factors for the power sensor are entered into the analyzer. All of these steps are explained in the Optimizing Measurement Results chapter of the *HP 8753D Network Analyzer User's Guide*.

The number of readings-per-point must also be selected before starting. The number of points directly affects the measurement time of the calibration sequence. The power meter must interact with the analyzer for each of the selected points and read the number of values specified for each trace point. Typically, two readings per point is considered appropriate. More than two readings-per-point could lead to unacceptable processing time.

Controlling a power meter calibration via HP-IB requires that the analyzer to be set to pass-control mode. The analyzer must position the local oscillator to a point in the sweep and read the power present at the power meter sensor. For this operation to take place, the system controller must set up the measurement and then pass control to the analyzer to read in each point in the sweep. After reading the data point from the power meter, the analyzer passes control back to the system controller. The analyzer then sets up to measure the next point and again requests control from the system controller. This process continues until the analyzer signals that the entire sweep has been measured point by point.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The number of points in the trace is set.
- The number of readings-per-point is set.
- The frequency span is set.
- The reference channel is measured.
- The power meter calibration array is allocated.
- The power meter model is chosen.
- The status registers are cleared.
- The request-control summary bit is enabled.
- The pass-control mode is enabled.
- A calibration sweep is taken to begin the sequence.
- The status byte is read until control is requested.
- The computer passes control to the analyzer.
- The display is cleared and the analyzer is set to talker/listener mode.
- The HP-IB interface status is read until control is returned.
- The program loops until all the points have been measured.
- The power meter calibration is enabled.
- The calibration data is output to the controller in FORM 4, ASCII format.
- The power meter-calibration factors are read into the controller.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This routine does a power meter cal using pass control.
20  ! A measurement cycle takes place on each point of the trace. The
30  ! point is measured by the power meter and the measured value read
40  ! into the analyzer. The command TAKCS; arms this measurement mode.
50  ! The number of measurements is determined by the number of points in
60  ! the trace, the number of readings per point and an extra measurement
70  ! cycle to release the powr meter.
80  ! Control is passed to the analyzer, the point is measured and
90  ! the data is transferred to the analyzer. Control is passed back to
100 ! the controller and the cycle begins again. Serial poll is used to
110 ! read the status byte of the analyzer and test the logic.
120 ! The HP-IB interface status register is monitored to determine when
130 ! control is returned to the interface from the analyzer.
140 !
150 ! EXAMP4C
160 !
170 ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
180 !
190 CLEAR SCREEN
200 ! Initialize the analyzer
```

```
210 ABORT 7                                 ! Generate an IFC (Interface Clear)
220 CLEAR @Nwa                              ! SDC (Selective Device Clear)
230 OUTPUT @Nwa;"OPC?;PRES;"                ! Preset the analyzer and wait
240 ENTER @Nwa;Reply                        ! Read the 1 when complete
250 !
260 INTEGER Stat
270 !
280 ! Set up the analyzer parameters
290 Numpoints=11                            ! Number of points in the trace
300 Numreads=2                              ! Number of readings per point
310 Startf=1.00E+8                          ! Start frequency
320 Stopf=5.0E+8                            ! Stop frequency
330 !
340 OUTPUT @Nwa;"POIN";Numpoints;";"        ! Set trace length to numpoints
350 OUTPUT @Nwa;"NUMR";Numreads;";"         ! Set number of readings per point
360 OUTPUT @Nwa;"STAR";Startf               ! Set start frequency
370 OUTPUT @Nwa;"STOP";Stopf                ! Set stop frequency
380 OUTPUT @Nwa;"MEASR;"                    ! Measure the reference channel
390 !
400 ALLOCATE Pmcal(1:Numpoints)             ! Create power meter cal array
410 !
420 ! Store the original trace for comparison
430 OUTPUT @Nwa;"DATI;"
440 OUTPUT @Nwa;"DISPDATM;"
450 OUTPUT @Nwa;"AUTO;"
460 !
470 ! Select the power meter being used for cal
480 ! OUTPUT @Nwa;"POWM ON;"                 ! Select 436A power meter
490 OUTPUT @Nwa;"POWMOFF;DEBUON;"           ! Select 437B/438A power meter
500 !
510 ! Set analyzer HP-IB, status regs to interrupt on pass control
520 OUTPUT @Nwa;"CLES;"                     ! Clear status registers
530 OUTPUT @Nwa;"ESE2;"                     ! Enable request control summary bit
540 OUTPUT @Nwa;"SRE32;"                    ! SRQ on events status register
550 !
560 PRINT "Beginning Power Meter CAL"
570 OUTPUT @Nwa;"USEPASC;"                  ! Enable pass control operation
580 OUTPUT @Nwa;"TAKCS;"                    ! Take Cal Sweep
590 !
600 FOR I=1 TO Numpoints*Numreads+1         ! Points * Number of readings plus 1
610   ! Serial poll does not place analyzer in remote operation
620   ! and does not require the analyzer to process the command.
630   !
640   REPEAT                                ! Repeat until SRQ detected
650     Stat=SPOLL(@Nwa)                    ! Serial poll to read status byte
660     DISP "Stat ";Stat;" Waiting for request"
670   UNTIL BIT(Stat,6)                     ! SRQ detected for request control
680   OUTPUT @Nwa;"ESR?;"                   ! Read status register to clear
690   ENTER @Nwa;Reply                      ! Read and discard register value
700   !
710   PRINT "Passing Control"               ! status read and pasing control
720   PASS CONTROL @Nwa                     ! Pass control to analyzer
730   !
740   REPEAT
750   ! Read HP-IB interface state information register.
```

```
760     STATUS 7,6;Hpib                 ! Test HP-IB register for control
770   !
780   ! Reading the interface status register does not interact with the
790   ! analyzer. Bit 6 is set when control is returned.
800   !
810     DISP "Waiting for control"
820   UNTIL BIT(Hpib,6)                 ! Loop until control is returned
830 NEXT I
840 !
850 PRINT "Finished with Power meter Cal"
860 DISP ""                             ! Clear display message
870 !
880 OUTPUT @Nwa;"TALKLIST;"             ! Restore Talker/Listener operation
890 OUTPUT @Nwa;"CLES;"                 ! Clear and reset status byte operation
900 !
910 OUTPUT @Nwa;"PWMCONES;"             ! Power meter cal correct one sweep
920 OUTPUT @Nwa;"OPC?;WAIT;"            ! Wait for the analyzer to finish
930 ENTER @Nwa;Reply                    ! Read the 1 when complete
940 !
950 ! Read the power meter cal correction factors
960 OUTPUT @Nwa;"FORM4;"                ! ASCII data format to read cal data
970 OUTPUT @Nwa;"OUTPPMCAL1;"           ! Request the power meter cal factors
980 ENTER @Nwa;Pmcal(*)                 ! Read the factors
990 !
1000! Display the power meter cal factors
1010 PRINT "Point","Factor"
1020 FOR I=1 TO Numpoints               ! Cycle throught the factors
1030 PRINT I,Pmcal(I)
1040 NEXT I
1050!
1060 LOCAL @Nwa                         ! Release HP-IB control
1070 END
```

Running the Program

The analyzer is preset and the power meter-calibration routine begins. The analyzer displays the message "WAITING FOR HP-IB CONTROL"' when it is requesting control. The system controller display prints "Passing Control" when control is passed to the analyzer. The interaction of the messages and the movement of the cursor allow observation of the calibration process. Once the calibration is complete, the analyzer displays "POWER METER CAL IS COMPLETE" and the system controller displays "Finished with Power meter Cal". The power meter-calibration mode (with one sweep of correction data) is enabled and the calibration is switched ON. At the completion of the program, talker/listener mode is restored, the event-status registers are cleared (to halt the status-byte interaction), the sweep is placed in continuous-sweep mode, the analyzer is released from HP-IB control, and the program ends.

Example 5: Network Analyzer System Setups

Saving and Recalling Instrument States

Note The most efficient option for storing and recalling analyzer states is using the analyzer's internal registers to save the CAL data. Recalling these registers is the fastest solution to restoring analyzer setups. See the Printing, Plotting, and Saving Measurement Results chapter in the *HP 8753D Network Analyzer User's Guide* for detailed information on the analyzer's internal storage registers.

In the event that all the registers have been used, or if internal memory limitations exist, then these external solutions become viable.

The purpose of this example is to demonstrate several programming options for storing and recalling entire instrument states over HP-IB. The examples describe two different processes for storing and recalling instrument states. The first example accomplishes the task using the learn string. The second example involves reading both the learn string and the calibration arrays out of the analyzer and storing them to disk or storing them in the system controller itself.

Using the learn string is a very rapid way of saving the instrument state, but using direct disk access has the advantage of automatically storing calibrations, cal kits, and data along with the instrument state.

A complete analyzer setup requires sending the learn string and a calibration array to set the analyzer parameters. The CAL array may also be placed in the analyzer, just as if a calibration was performed. By sending both sets of data, the analyzer may be quickly setup for a measurement.

Several different measurements may be required in the course of testing a device. An efficient way of performing multiple measurements is to send both the calibration array and the learn string, and then perform the measurements.

Example 5A: Using the Learn String

Note This program is stored as EXAMP5A on the "HP 8753D Programming Examples" disk received with the network analyzer.

The learn string is a very fast and easy way to read an instrument state. The learn string includes all front-panel settings, the limit table for each channel, and the list-frequency table. It can be read out of the analyzer with the command OUTPLEAS, and input to the analyzer with the command INPULEAS. This array is transmitted in FORM 1, the internal format for the analyzer. It cannot be longer than 3000 bytes. The example for a FORM 1 transfer could also have been used. However, Example 5A is the simplest solution for reading the learn string from the analyzer.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The string storage is allocated.
- The learn string is requested.
- The string is read without any processing.

- The analyzer is released from remote control.
- The instrument state is changed by the operator.
- The learn string is sent back to the analyzer.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program shows how to retrieve a learn string from the analyzer
20  ! into a string array. The state of the analyzer is then changed and the
30  ! learn string re-loaded to return the analyzer to the previous settings.
40  !
50  ! EXAMP5A
60  !
70  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
80  !
90  CLEAR SCREEN
100 ! Initialize the analyzer
110 ABORT 7                               ! Generate an IFC (Interface Clear)
120 CLEAR @Nwa                            ! SDC (Selected Device Clear)
130 !
140 DIM State$[3000]                      ! Define a string for contents
150 !
160 OUTPUT @Nwa;"OUTPLEAS;"               ! Output the learn string
170 ENTER @Nwa USING "+,-K";State$        ! Read the string with no processing
180              ! + Terminate on EOI only
190              ! -K ignore LF as terminator treat as data
200              !
210 LOCAL @Nwa                            ! Release HP-IB control
220 !
230 INPUT "Change state and press ENTER",A$
240 !
250 OUTPUT @Nwa;"INPULEAS;";State$;       ! Send the learnstring to analyzer
260 DISP "Analyzer state has been restored!"
270 !
280 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for the analzyer to finish
290 ENTER @Nwa;Reply                      ! Read the 1 when complete
300 LOCAL @Nwa                            ! Release HP-IB control
310 END
```

Running the Program

Run the program. When the program stops, change the instrument state and press ENTER on the controller. The analyzer will be returned to its original state by using the learn string.

Example 5B: Reading Calibration Data

Note This program is stored as EXAMP5B on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example demonstrates:

- How to read measurement calibration data out of the analyzer.
- How to read it back into the analyzer.
- How to determine which calibration is active.

The data used to perform measurement-error correction is stored inside the analyzer in one (or more) of twelve calibration-coefficient arrays. Each array is a specific error coefficient, and is stored and transmitted as an error-corrected data array. Each point is a real/imaginary pair, and the number of points in the array is the same as the number of points in the sweep. The four data formats also apply to the transfer of calibration-coefficient arrays. The Preset State and Memory Allocation chapter in the *HP 8753D Network Analyzer User's Guide* contains information on the storage locations for calibration coefficients and different calibration types.

A computer can read out the error coefficients using the commands OUTPCALC01, OUTPCALC02, . . . through OUTPCALC12. Each calibration type uses only as many arrays as required, beginning with array 1. Hence, it is necessary to know the type of calibration about to be read out: attempting to read an array not being used in the current calibration causes the `"REQUESTED DATA NOT CURRENTLY AVAILABLE"` warning.

A computer can also store calibration coefficients in the analyzer. To do this, declare the type of calibration data about to be stored in the analyzer just as if you were about to perform that calibration. Then, instead of calling up different classes, transfer the calibration coefficients using the INPUCALCnn commands. When all the coefficients are stored in the analyzer, activate the calibration by issuing the mnemonic SAVC, and trigger a sweep on the analyzer.

This example reads the calibration coefficients into a very large array, from which they can be examined, modified, stored, or put back into the instrument. If the data is to be directly stored on to disk, it is usually more efficient to use FORM 1 (analyzer's internal-binary format), and to store each coefficient array as it is read in.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- A binary path is assigned.
- The system is initialized.
- The calibration types and number of arrays are defined.
- The integer variables for reading the headers are defined.
- The calibration type and number of arrays are read by the controller.
- The output is formatted in FORM 3.
- The number of points in the trace is read.
- The memory is allocated for the calibration arrays.
- Each calibration array is requested from the analyzer.
- Header information is read with a binary I/O path.
- The elements from each calibration array are read in.

- The next calibration array is requested until all the arrays have been read.
- The calibration type is sent to the analyzer.
- Each calibration array is sent.
- The calibration is activated.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program shows how to manipulate calibration data from the analyzer.
20  ! It demonstrates how to read calibration data from the analyzer, and
30  ! how to replace it. The type of calibration active is determined and
40  ! the program reads in the correct number of arrays. The number of points
50  ! in the trace, and in the cal array, is determined and used to dimension
60  ! storage arrays.
70  !
80  ! EXAMP5B
90  !
100 ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
110 ASSIGN @Nwa_bin TO 716;FORMAT OFF     ! Assign binary path
120 !
130 CLEAR SCREEN
140 ! Initialize the analyzer
150 ABORT 7                               ! Generate an IFC (Interface Clear)
160 CLEAR @Nwa                            ! SDC (Selected Device Clear)
170 !
180 ! Data for determining CAL type and number of arrays
190 DATA "CALIRESP",1,"CALIRAI",2,"CALIS111",3
200 DATA "CALIS221",3,"CALIFUL2",12
210 DATA "NOOP",0
220 !
230 INTEGER Hdr,Lgth,I,J                  ! Integers for reading headers
240 !
250 READ Calt$,Numb                       ! Read CAL type from data statement
260 IF Numb=0 THEN GOTO 690               ! If no CAL type is present Exit
270 OUTPUT @Nwa;Calt$;"?;"                ! Query if CAL type is active
280 ENTER @Nwa;Active                     ! Read 1 if active
290 IF NOT Active THEN GOTO 250           ! Load another CAL type and re-try
300 !
310 PRINT Calt$,Numb                      ! Active CAL and number of arrays
320 !
330 OUTPUT @Nwa;"FORM3;"                  ! Form 3 IEEE 64 bit floating point
340 OUTPUT @Nwa;"POIN?;"                  ! Request trace length
350 ENTER @Nwa;Poin                       ! Read number of points
360 ALLOCATE Cal(1:Numb,1:Poin,1:2)       ! Arrays for CAL arrays
370 !         Number of arrays, number of points real and imag value per point
380 !
390 FOR I=1 TO Numb                          ! Read arrays
400   OUTPUT @Nwa USING "K,ZZ";"OUTPCALC",I  ! Format I to add 0 in command
410   ENTER @Nwa_bin;Hdr,Lgth                ! Read header & length from array
420   FOR J=1 TO Poin                        ! Read elements for CAL array
430     ENTER @Nwa_bin;Cal(I,J,1),Cal(I,J,2)  ! Read real & imag pair elements
440   NEXT J                                 ! Next location in array
450 NEXT I                                   ! Next CAL array
```

```
460 !
470 ! All CAL arrays have been read
480 !
490 INPUT "PRESS RETURN TO RE-TRANSMIT CALIBRATION",Dum$
500 !
510 OUTPUT @Nwa;"FORM3;"                          ! Use same format as read
520 OUTPUT @Nwa;Calt$;";"                         ! Send CAL type to analyzer
530 !
540 FOR I=1 TO Numb                               ! Send each array in CAL
550   DISP "TRANSMITTING ARRAY: ",I               ! Show array number
560   OUTPUT @Nwa USING "K,ZZ";"INPUCALC",I       ! Send array number 0 format
570   OUTPUT @Nwa_bin;Hdr,Lgth                    ! Send header & array length
580   FOR J=1 TO Poin                             ! Send each array element
590     OUTPUT @Nwa_bin;Cal(I,J,1),Cal(I,J,2)     ! Real and Imag pair
600   NEXT J                                      ! Next element in array
610 NEXT I                                        ! Next array
620 !
630 OUTPUT @Nwa;"SAVC;"                           ! Activate CAL
640 !
650 OUTPUT @Nwa;"CONT;"                           ! Restore continuous sweep
660 OUTPUT @Nwa;"OPC?;WAIT;"                      ! Wait for analyzer to finish
670 ENTER @Nwa;Reply                              ! Read the 1 when complete
680 !
690 DISP "Finished with CAL transfer"
700 LOCAL @Nwa                                    ! Release HP-IB control
710 END
```

Running the Program

Before executing the program, perform a calibration.

The program is able to detect which type of calibration is active. With that information, it predicts how many arrays to read out. When all the arrays have been sent to the computer, the program prompts the user. The operator then turns the calibration OFF or performs a completely different calibration on the analyzer and continues the program. The computer reloads the old calibration. The operator should not preset the analyzer because the instrument settings must be the same as those that were present when the calibration was taken.

Note The re-transmitted calibration is associated with the current instrument state: the instrument has no way of knowing the original state associated with the calibration data. For this reason, it is recommended that the learn string be used to store the instrument state whenever calibration data is stored. The next example demonstrates how to reload the analyzer state with both the learn string and the calibration arrays.

Example 5C: Saving and Restoring the Analyzer Instrument State

Note This program is stored as EXAMP5C on the "HP 8753D Programming Examples" disk received with the network analyzer.

Note The instrument state may also be stored in the analyzer's internal registers. This is the fastest and most efficient method for toggling between instrument states. This example is for use when the analyzer's internal memory is full, or when the are other internal-memory limitations.

This example demonstrates using both the learn string and the calibration arrays to completely re-program the analyzer state. If you were performing two entirely different measurements on a device and wanted to quickly change between instrument states and perform the measurements, this example program is a potential solution.

The example will request the learn string and calibration array from the analyzer and store them in a disk file on the system controller. Once the storage is complete, the operator will be prompted to change the state of the analyzer and then re-load the state that was previously stored in the disk file. Once the file is created on the disk, the state information can be retrieved from the controller and restored on the analyzer.

Note The disk file can only be created once. Errors will occur if the operator repeatedly tries to re-create the file.

For this example, only a through calibration will be performed and transferred. This means only one calibration array will be read from the analyzer and written to the disk file with the instrument state. To work with more elaborate calibrations, additional arrays will need to be defined and transferred to the disk file. This is not difficult, but requires some further programming steps which were omitted in the interest of presenting a simple example.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- A binary path is assigned.
- The integers for reading the headers are defined.
- The system is initialized.
- An array is created to hold the learn string.
- The learn string is requested by the controller.
- The number of points in the trace is read.
- The controller allocates an array for the calibration data.
- The calibration data is read into the controller.
- The controller creates and assigns a data file for the calibration array and the learn string.
- The learn string and calibration array are stored in the disk file.
- The operator presses (ENTER) on the controller to read the calibration data back into the analyzer.
- The learn string is read from the disk file and output to the analyzer.
- The calibration array is read in from the disk file and stored in the analyzer.

- The analyzer is returned to continuous-sweep mode.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program reads an instrument state and stores it in a disk file.
20  ! The learn string and CAL array are both read into the controller and
30  ! then transferred to a disk file for storage. The file contents are
40  ! then restored to the analyzer. The analyzer is preset to the default
50  ! settings before the instrument state is transferred back.
60  !
70  ! EXAMP5C
80  !
90  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
100 ASSIGN @Nwa_bin TO 716;FORMAT OFF     ! Assign a binary path
110 !
120 INTEGER Head,Length                   ! Integer 2 byte format for headers
130 !
140 CLEAR SCREEN
150 ! Initialize the analyzer
160 ABORT 7                               ! Generate an IFC (Interface Clear)
170 CLEAR @Nwa                            ! SDC (Selected Device Clear)
180 !
190 ! Read in the learn string as a form 1 binary data trace
200 DIM Learn$[3000]                      ! Array to hold learn string
210 !
220 OUTPUT @Nwa;"OPC?;SING;"              ! Place analyzer in single sweep
230 ENTER @Nwa;Reply                      ! Read the 1 when complete
240 !
250 OUTPUT @Nwa;"OUTPLEAS;"               ! Request learn string
260 ENTER @Nwa USING "+,-K";Learn$
270 !
280 ! Allocate an array for storing the CAL data
290 OUTPUT @Nwa;"POIN?;"                  ! Find number of points in trace
300 ENTER @Nwa;Num_points                 ! Read number to allocate array
310 ALLOCATE Cal_array(1:Num_points,1:2)   ! Real and Imag for each point
320 !
330 ! Read Cal array
340 OUTPUT @Nwa;"FORM3;"                  ! Form 3 64 bit floating point data
350 OUTPUT @Nwa;"OUTPCALC01;"             ! Request the cal array
360 !
370 ! Read the #A and 2 byte length as integers
380 ENTER @Nwa_bin;Head,Length,Cal_array(*)   ! Read cal array data
390 !
400 ! Write instrument state data to disk file
410 ! CREATE BDAT "DATA_FILE:,1406",1,Length+3000  ! Create data file once!
420 ASSIGN @File TO "DATA_FILE:,1406          "    ! Assign I/O path to file
430 OUTPUT @File;Learn$                   ! Send learn string to disk file
440 OUTPUT @File;Head,Length,Cal_array(*)  ! Send CAL arrays to disk file
450 ASSIGN @File TO *                             ! Close file
460 !
470 INPUT "Cal data received. Press ENTER to send it back.",A$
480 !
490 ! Read arrays from file
500 !
```

```
510 DIM Learn2$[3000]                    ! String for learn string storage
520 ASSIGN @File TO "DATA_FILE:,1406"    ! Open file for reading arrays
530 ENTER @File;Learn2$                  ! Read learn string from file
540 !
550 ENTER @File;Head,Length              ! Read CAL data headers from file
560 Size=Length/16                     ! Array is 2 numbers, 8 bytes per number
570 ALLOCATE Cal_array2(1:Size,1:2)      ! new cal array from file record
580 ENTER @File;Cal_array2(*)            ! Read cal array from disk file
590 !
600 ! Send Learn string back
610 OUTPUT @Nwa;"INPULEAS;",Learn2$      ! Send learn string array
620 !
630 ! Send Cal array back
640 OUTPUT @Nwa;"CALIRESP;"              ! Send CAL type (Response)
650 OUTPUT @Nwa;"INPUCALC01;"            ! Output CAL array to analyzer
660 OUTPUT @Nwa_bin;Head,Length,Cal_array2(*)
670 OUTPUT @Nwa;"OPC?;SAVC;"             ! Save the CAL array
680 ENTER @Nwa;Reply                     ! Read the 1 when complete
690 !
700 OUTPUT @Nwa;";CONT;"                 ! Start the analzyer sweeping
710 OUTPUT @Nwa;"OPC?;WAIT;"             ! Wait for the analyzer to finish
720 ENTER @Nwa;Reply
730 LOCAL @Nwa                           ! Release HP-IB control
740 END
```

Running the Program

Setup the analyzer and perform a thorough calibration.

Run the program. The program prompts the operator to change the state of the analyzer and then press (ENTER) to continue. At this point, the analyzer state is stored on the disk file in the controller. Pressing (ENTER) will begin the transfer from the disk file to internal arrays within the controller and then on to the analyzer.

Once completed:

- The original state will be restored.
- The analyzer will be sweeping.
- The analyzer will be calibrated.
- COR will be displayed on the analyzer's CRT.

Example 6: Limit-Line Testing

Using List-Frequency Mode

The analyzer normally takes data points spaced at regular intervals across the overall frequency range of the measurement. For example, for a 2 GHz frequency span using 201 points, data will be taken at intervals of 10 MHz. The list-frequency mode allows the operator to select the specific points, or frequency spacing between points, at which measurements are to be made. This mode of operation allows flexibility in setting up tests to insure device performance in an efficient manner. By only sampling specific points, measurement time is reduced, since additional time is not spent measuring device performance at frequencies which are of no concern. List-frequency sweeps are also discussed in the Application and Operation Concepts chapter of the *HP 8753D Network Analyzer User's Guide*. These programs emulate operation from the analyzer's front panel when using list sweeps.

The following two examples illustrate the use of the analyzer's list-frequency mode to perform arbitrary frequency testing. Example 6A lets the operator construct a table of list-frequency segments which is then loaded into the analyzer's list-frequency table. There are a maximum of 30 segments available. Each segment stipulates a start and stop frequency, and the number of data points to be taken over that frequency range. Example 6B lets the operator select a specific segment to "zoom-in" on. A single instrument can programmed to measure several different devices, each with its own frequency range, using a single calibration performed with all of the segments active. When a specific device is connected, the operator selects the appropriate segment for that device. Note that list-frequency segments can be overlapped, but the total number of points in all the segments must not exceed 1632.

Example 6A: Setting Up a List-Frequency Sweep

Note This program is stored as EXAMP6A on the "HP 8753D Programming Examples" disk received with the network analyzer.

The purpose of this example is to show how to create a list-frequency table and transmit it to the analyzer.

The command sequence for entering a list-frequency table imitates the key sequence followed when entering a table from the front panel: there is a command for every key-press.

Editing a segment is also the same as the front-panel key sequence, but remember the analyzer automatically reorders each edited segment in order of increasing start frequency.

The list-frequency table is also carried as part of the learn string. While the table cannot be modified as part of the learn string, it can be stored and recalled with very little effort by storing and recalling the learn string. See Chapter 2 under the section titled "Data Processing Chain" for details on using learn strings.

This example takes advantage of the computer's capabilities to simplify:

- creating a list-frequency table
- editing a list-frequency table

The table is entered and completely edited before being transmitted to the analyzer. To simplify the programming task, options such as entering center frequency, frequency span, or step size are not included.

The list-frequency information may be acquired using the limit-test results array. The actual stimulus points are available as the first element in the array.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The existing list frequencies are edited and cleared.
- The number of segments to define is read in.
- An array for the list segments is defined.
- The parameters for each segment are requested.
- If the operator wants to edit, the segment parameters are re-entered.
- The new list is sent to the analyzer.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program shows how to enter and edit a list frequency table.
20  ! Any existing table is deleted and a new table is defined and
30  ! edited. This list is then sent to the analyzer. Any number of
40  ! segments or points may be entered. Be sure not to enter more than
50  ! 1632 points or 30 segments.
60  !
70  !  EXAMP6A
80  !
90  ASSIGN @Nwa TO 716                  ! Assign an I/O path for the analyzer
100 !
110 CLEAR SCREEN
120 ! Initialize the analyzer
130 ABORT 7                             ! Generate an IFC (Interface Clear)
140 CLEAR @Nwa                          ! SDC (Selective Device Clear)
150 OUTPUT @Nwa;"OPC?;PRES;"            ! Preset the analyzer and wait
160 ENTER @Nwa;Reply                    ! Read the 1 when complete
170 !
180 OUTPUT @Nwa;"EDITLIST;"             ! Begin editing the frequency list
190 OUTPUT @Nwa;"CLEL;"                 ! Clear the existing list frequencies
200 !
210 INPUT "Number of segments?",Numb    ! Read number of segments to define
220 ALLOCATE Table(1:Numb,1:3)          ! Define an array for the list segments
230 !
240 PRINT USING "10A,15A,15A,20A";"SEGMENT","START(MHZ)","STOP(MHZ)","NUMBER OF
    POINTS"
250 !
260 FOR I=1 TO Numb         ! Cycle through the segments and read in the values
270   GOSUB Loadpoin
280 NEXT I
290 !
300 LOOP
310   INPUT "DO YOU WANT TO EDIT? Y OR N",An$
320 EXIT IF An$="N"
330   INPUT "ENTRY NUMBER?",I           ! Get an entry number
340   GOSUB Loadpoin                    ! Go load point
350 END LOOP
360 !
370 OUTPUT @Nwa;"EDITLIST"              ! Send the new list to the analyzer
```

```
380 FOR I=1 TO Numb                              ! Send one segment at a time
390   OUTPUT @Nwa;"SADD;"                        ! Add a segment
400   OUTPUT @Nwa;"STAR";Table(I,1);"MHZ;"       ! Start frequency
410   OUTPUT @Nwa;"STOP";Table(I,2);"MHZ;"       ! Stop frequency
420   OUTPUT @Nwa;"POIN",Table(I,3),";"          ! Number of points
430   OUTPUT @Nwa;"SDON;"                        ! Segment done
440 NEXT I                                 ! Next segment to send to the analyzer
450 !
460 OUTPUT @Nwa;"EDITDONE;"                ! Done with list
470 OUTPUT @Nwa;"LISFREQ;"                 ! Set list frequency mode
480 !
490 OUTPUT @Nwa;"OPC?;WAIT;"               ! Wait for analyzer to finish
500 ENTER @Nwa;Reply                       ! Read the 1 when complete
510 LOCAL @Nwa                             ! Release HP-IB control
520 STOP                                   ! End of main program
530 !
540 !   **************************Subroutines ******************************
550 !
560 Loadpoin:                              ! Sub to read in each segment value
570 INPUT "START FREQUENCY? (MHZ)",Table(I,1)   ! Read start frequency
580 INPUT "STOP FREQUENCY? (MHZ)",Table(I,2)    ! Read stop frequency
590 INPUT "NUMBER OF POINTS?",Table(I,3)        ! Read number of points in seg
600 IF Table(I,3)=1 THEN Table(I,2)=Table(I,1)  ! Single point same start stop
610 !
620 ! Print new segment into table on display
630 PRINT TABXY(0,I+1);I;TAB(10);Table(I,1);TAB(25);
640 PRINT Table(I,2);TAB(40),Table(I,3)
650 RETURN
660 END
```

Running the Program

Caution This example program will delete any existing limit lines before entering the new limits. If this is not desired, omit the line(s) that clear the existing limits (in this case LINE 190). This program begins by presetting the analyzer. The programmer will have to add the necessary command lines to set the analyzer to the specific operating conditions required for testing. The example program will show the limit lines defined, but the limits will always fail without additional analyzer setup.

The program displays the frequency-list table as it is entered. During editing, the displayed table is updated as each line is edited. The table is not re-ordered. At the completion of editing, the table is entered into the analyzer, and list-frequency mode is switched ON. During editing, simply pressing (RETURN) leaves an entry at the old value.

The table will be sorted by the analyzer and displayed.

Example 6B: Selecting a Single Segment from a Table of Segments

Note This program is stored as EXAMP6B on the "HP 8753D Programming Examples" disk received with the network analyzer.

This example program demonstrates how to define a single segment as the operating-frequency range of the analyzer from a table of segments stored in the controller. The program assumes that a list-frequency table has already been entered into the analyzer, either manually, or using the program in Example 6A, "Setting Up a List-Frequency Sweep."

The program first loads the list-frequency table into the computer by reading the start and stop frequencies of each segment and the number of points for each segment. The segment's parameters are then displayed on the computer screen, and the operator can choose which segment is to be used by the analyzer. Note that only one segment can be chosen at a time.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The list-frequency segment is edited.
- The largest segment number is set.
- The highest segment number is requested.
- The number of actual segments is read in.
- A list-frequency table is defined and the segments are read in to the controller from the analyzer.
- The operator selects one of the segments of the sweep.
- The controller "zooms-in" and sweeps the defined segment.
- The operator presses [0] and the analyzer returns to sweeping all the segments in the table.
- The activation loop is ended and the program ends.

The program is written as follows:

```
10  !  This program shows how to select a single segment from a list
20  ! frequency sweep and activate it as the sweep. The list frequency
30  ! table is read from the analyzer and displayed on the computer
40  ! screen. The operator is prompted to select a segment and the
50  ! program then activates it. All the segments are activated upon
60  ! completion.
70  !
80  ! EXAMP6B
90  !
100 ASSIGN @Nwa TO 716                  !  Assign an I/O path for the analyzer
110 !
120 CLEAR SCREEN
130 ! Initialize the analyzer
140 ABORT 7                             ! Generate an IFC (Interface Clear)
150 CLEAR @Nwa                          ! SDC (Selected Device Clear)
160 !
170 ! Print header for table of existing segments
180 PRINT USING "10A,15A,15A,20A";"SEGMENT","START(MHZ)","STOP(MHZ)","NUMBER OF
```

```
    POINTS"
190 OUTPUT @Nwa;"EDITLIST;"              ! Edit list frequency segment
200 OUTPUT @Nwa;"SEDI30;"                ! Set largest segment number
210 OUTPUT @Nwa;"SEDI?;"                 ! Request number of highest segment
220 ENTER @Nwa;Numsegs                   ! Read number of actual segments
230 !
240 ! Setup table and read segments from analyzer
250 ALLOCATE Table(1:Numsegs,1:3)        ! Allocate table of segments
260 FOR I=1 TO Numsegs                   ! Cycle through segments
270   GOSUB Readlist                     ! Read in segment definitions
280 NEXT I                               ! Next segment
290 !
300 ! Loop and read segment to be activated
310 LOOP                                 ! Request operator to enter segment
320   INPUT "SELECT SEGMENT NUMBER: (0 TO EXIT)",Segment
330 EXIT IF Segment=0                    ! Exit point
340   OUTPUT @Nwa;"EDITDONE;";"SSEG";Segment;";" ! Set active segment to sweep
350 END LOOP                             ! End activation loop
360 !
370 OUTPUT @Nwa;"ASEG;"                  ! Set all segment sweep
380 DISP "PROGRAM ENDED"
390 !
400 OUTPUT @Nwa;"OPC?;WAIT;"             ! Wait for analyzer to finish
410 ENTER @Nwa;Reply                     ! Read the 1 when complete
420 LOCAL @Nwa                           ! Release HP-IB control
430 STOP                                 ! End of main program
440 !
450 ! ************************* Subroutines ******************************
460 !
470 Readlist:                            ! Read segment list from analyzer
480 OUTPUT @Nwa;"EDITLIST;"              ! Edit segment list
490 OUTPUT @Nwa;"SEDI",I,";"             ! Select segment to edit
500 OUTPUT @Nwa;"STAR;"                  ! Send start freq to display value
510 OUTPUT @Nwa;"OUTPACTI;"              ! Output active function value
520 ENTER @Nwa;Table(I,1)                ! Read start frequency
530 OUTPUT @Nwa;"STOP;"                  ! Send stop freq to display value
540 OUTPUT @Nwa;"OUTPACTI;"              ! Output active function value
550 ENTER @Nwa;Table(I,2)                ! Read stop frequency
560 OUTPUT @Nwa;"POIN;"                  ! Send number of points to display
570 OUTPUT @Nwa;"OUTPACTI;"              ! Output active function value
580 ENTER @Nwa;Table(I,3)                ! Read number of points
590 !
600 IF I=18 THEN                         ! Pause if more than 17 segments
610 INPUT "PRESS RETURN FOR MORE",A$     ! Read Return to continue
620 END IF
630 ! Print new header for segment data
640 IMAGE 4D,6X,4D.6D,3X,4D.6D,3X,4D     ! Format image to disp segment data
650 PRINT USING 640;I;Table(I,1)/1.E+6;Table(I,2)/1.E+6;Table(I,3)
660 RETURN
670 !
680 END
```

Running the Program

The program will read the parameters for each list-frequency segment from the analyzer, and build a table containing all the segments. The parameters of each segment will be printed on the computer screen. If there are more than 17 segments, the program will pause. Press RETURN to see more segments. The maximum number of segments that can be read is 30 (the maximum number of segments the analyzer can hold). Use the computer's Page Up and Page Down keys to scroll through the list of segments if there are more than 17.

After all the segments are displayed, the program will prompt the operator for a specific segment to be used. Type in the number of the segment, and the analyzer will then "zoom-in" on that segment. The program will continue looping, allowing continuous selection of different segments. To exit the loop, type `0`. This will restore all the segments (with the command `ASEG`), allowing the analyzer to sweep all of the segments, and the program will terminate.

Using Limit Lines to Perform PASS/FAIL Tests

There are two steps to performing limit testing on the analyzer via HP-IB. First, limit specifications must be defined and loaded into the analyzer. Second, the limits are activated, the device is measured, and its performance to the specified limits is signaled by a pass or fail message on the analyzer's display.

Example 6C illustrates the first step, setting up limits. Example 6D performs the actual limit testing.

Example 6C: Setting Up Limit Lines

Note This program is stored as EXAMP6C on the "HP 8753D Programming Examples" disk received with the network analyzer.

The purpose of this example is to show how to create a limit-test table and transmit it to the analyzer.

The command sequence for entering a limit-test table imitates the key sequence followed when entering a table from the analyzer's front panel: there is a command for every key-press. Editing a limit is also the same as the key sequence, but remember that the analyzer automatically re-orders the table in order of increasing start frequency.

The limit-test table is also carried as part of the learn string. While it cannot be modified as part of the learn string, the learn string can be stored and recalled with very little effort. See the section titled "Data Processing Chain" in Chapter 2 for details on using learn strings.

This example takes advantage of the computer's capabilities to simplify creating and editing the table. The table is entered and completely edited before being transmitted to the analyzer. To simplify the programming task, options such as entering offsets are not included.

This example automates the front-panel operation of entering a limit-test table. Front-panel operation and limits are discussed in the Application and Operation Concepts chapter of the *HP 8753D Network Analyzer User's Guide*. The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The limit lines are edited and cleared.
- The number of limits is requested.

- The limit table is created.
- The string array of limit types is created.
- The operator is prompted to enter the new limit values.
- The new limit table is sent back to the analyzer.
- The limit line is activated.
- The limit test is activated.
- The analyzer is returned to local control and the program ends.

The program is written as follows:

```
10  ! This program shows how to create a limit table and send it to the
20  ! analyzer. The operator enters the desired limits when prompted for
30  ! the stimulus value, upper and lower value and type of limit
40  ! desired. Once the table is created, the limits are sent to the
50  ! analyzer and activated.
60  !
70  ! EXAMP6C
80  !
90  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
100 !
110 CLEAR SCREEN
120 ! Initialize the analyzer
130 ABORT 7                               ! Generate an IFC (Interface clear)
140 CLEAR @Nwa                            ! SDC (Selected Device Clear)
150 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
160 ENTER @Nwa;Reply                      ! Read the 1 when completed
170 !
180 OUTPUT @Nwa;"EDITLIML;"               ! Edit limit lines
190 OUTPUT @Nwa;"CLEL;"                   ! Clear any existing limits
200 INPUT "NUMBER OF LIMITS?",Numb        ! Request the number of limits
210 ALLOCATE Table(1:Numb,1:3)            ! Create a table
220 ALLOCATE Limtype$(Numb)[2]            ! Create string array of limit types
230 !
240 ! Print out the header for the table
250 PRINT USING "10A,20A,15A,20A";"SEG","STIMULUS (MHz)","UPPER (dB)","
    LOWER (dB)", "TYPE"
260 !
270 ! Prompt the operator to enter the limit values
280 FOR I=1 TO Numb                       ! Cycle through the limits
290   GOSUB Loadlimit                     ! Go read limit values
300 NEXT I                                ! Next limit value
310 !
320 ! Allow the operator to edit the array entered
330 LOOP                                  ! Cycle to edit limit lines
340   INPUT "DO YOU WANT TO EDIT? Y OR N",An$
350 EXIT IF An$="N"                       ! Exit loop on N and send to analyzer
360   INPUT "ENTRY NUMBER?",I             ! Read limit number to edit
370   GOSUB Loadlimit                     ! Go read limit values
380 END LOOP                              ! Next edit entry
390 !
400 ! Send the limit line array segments to the analyzer
410 OUTPUT @Nwa;"EDITLIML;"               ! Edit the limit
420 FOR I=1 TO Numb                       ! Each segment of the limit
430   OUTPUT @Nwa;"SADD;"                   ! Add segment
440   OUTPUT @Nwa;"LIMS";Table(I,1);"MHZ"   ! Send segment stimulus value
450   OUTPUT @Nwa;"LIMU";Table(I,2);"DB"    ! Upper limit value
460   OUTPUT @Nwa;"LIML";Table(I,3);"DB"    ! Lower limit value
470   IF Limtype$(I)="FL" THEN OUTPUT @Nwa;"LIMTFL;" ! Flat limit
480   IF Limtype$(I)="SL" THEN OUTPUT @Nwa;"LIMTSL;" ! Sloping limit
490   IF Limtype$(I)="SP" THEN OUTPUT @Nwa;"LIMTSP;" ! Point limit
500   OUTPUT @Nwa;"SDON;"                 ! Segment done
510 NEXT I                                ! next segment
520 !
```

```
530 OUTPUT @Nwa;"EDITDONE;"              ! Edit complete
540 OUTPUT @Nwa;"LIMILINEON;"            ! Turn limit line on
550 OUTPUT @Nwa;"LIMITESTON;"            ! Turn limit test on
560 !
570 OUTPUT @Nwa;"OPC?;WAIT;"             ! Wait for the analyzer to finish
580 ENTER @Nwa;Reply                     ! Read the 1 when complete
590 !
600 LOCAL @Nwa                           ! Release HP-IB control
610 STOP                                 ! End of main program
620 !
630 !************************ Subroutines ******************************
640 !
650 Loadlimit:                           ! Sub to interact to load data
660 INPUT "STIMULUS VALUE? (MHz)",Table(I,1)  ! and print table created
670 INPUT "UPPER LIMIT VALUE? (DB)",Table(I,2)
680 INPUT "LOWER LIMIT VALUE? (DB)",Table(I,3)
690 INPUT "LIMIT TYPE? (FL=FLAT, SL=SLOPED, SP=SINGLE POINT)",Limtype$(I)
700 !
710                      ! Format and display table values
720 PRINT TABXY(0,I+1);I;TAB(10);Table(I,1);TAB(30);Table(I,2);TAB(45),
    Table(I,3),TAB(67);Limtype$(I)
730 RETURN                                ! Next limit value
740 !
750 END
```

Running the Program

Caution This example program will delete any existing limit lines before entering the new limits. If this is not desired, omit the line(s) that clear the existing limits (in this case LINE 190). This program begins by presetting the analyzer. The programmer will have to add the necessary command lines to set the analyzer to the operating conditions required for testing. The example program will show the limit lines defined, but the limits will always fail without additional analyzer setup.

The program displays the limit table as it is entered. During editing, the displayed table is updated as each line is edited. The table is not reordered. At the completion of editing, the table is entered into the analyzer, and limit-testing mode switched ON. The analyzer will rearrange the table in ascending order starting with the lowest start frequency entry. During editing, simply pressing (RETURN) leaves an entry at the old value.

Example 6D: Performing PASS/FAIL Tests While Tuning

Note This program is stored as EXAMP6D on the "HP 8753D Programming Examples" disk received with the network analyzer.

The purpose of this example is to demonstrate the use of the limit/search-fail bits in event-status register B, to determine whether a device passes the specified limits. Limits can be entered manually, or using the Example 5A.

The limit/search-fail bits are set and latched when limit testing or a marker search fails. There are four bits, one for each channel for both limit testing and marker search. See Figure 2-5 "Status Reporting Structure" and Table 2-4 "Units as a Function of Display Format" for additional information. Their purpose is to allow the computer to determine whether the test/search executed was successful. They are used in the following sequence:

1. Clear event-status register B.
2. Trigger the limit test or marker search.
3. Check the appropriate fail bit.

When using limit testing, the best way to trigger the limit test is to trigger a single sweep. By the time the single sweep command (`SING`) finishes, limit testing will have occurred.

Note If the device is tuned during the sweep, it may be tuned into and then out of limit, causing a limit test to qualify as "passed" when the device is not in fact within the specified limits.

When using marker searches (max, min, target, and widths), outputting marker or bandwidth values automatically triggers any related searches. Therefore, all that is required is to check the fail bit after reading the data.

In this example, several consecutive sweeps must qualify as "passing" in order to insure that the limit-test pass was not extraneous due to the device settling or operator tuning during the sweep. Upon running the program, the number of "passed" sweeps for qualification is entered. For very slow sweeps, a small number of sweeps such as two are appropriate. For relatively fast sweeps, where the device requires time to settle after tuning, as many sweeps as six or more sweeps may be more appropriate.

A limit-test table can be entered over HP-IB. The sequence is very similar to that used in entering a list-frequency table, as shown in Example 5D. The manual (front-panel entry) sequence is closely followed.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The pass counter is initialized on entry.
- The analyzer takes a sweep.
- The event-status register B byte is output and the channel-1 limit is tested.
- If the device fails the first sweep, the operator is prompted to insure it is tuned correctly and the device is measured again.
- If the device passes the first sweep, the operator is prompted not to touch the device as testing continues.

- If the device passes the required number of sweeps, the operator is prompted that the device has passed and to connect the next device for testing.
- The program initializes the pass counter and begins to measure the new device.

The program is written as follows:

```
10  ! This program demonstrates Pass/Fail tests using limit lines. The
20  ! program uses the latch-on-fail limit bits in event status register
30  ! B to determine if the device performance passes the specified test
40  ! limit lines.  It then requires that the device passes for multiple
50  ! consecutive sweeps in order to ensure that the device is static in
60  ! the response and not varying. The operator specifies how many sweeps
70  ! are required to pass the test.
80  !
90  ! EXAMP6D
100 !
110 ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
120 !
130 CLEAR SCREEN
140 ! Initialize the analyzer  No preset to retain settings for testing
150 ABORT 7                               ! Generate an IFC (Interface Clear)
160 CLEAR @Nwa                            ! SDC (Selected Device Clear)
170 !
180 INPUT "Number of consecutive passed sweeps for qualification?",Qual
190 Pass=0                                ! Initialize pass counter on entry
200 !
210 Tune: DISP "TUNE DEVICE AS NECESSARY"  ! Device is not passing warning
220 !
230 Measure:OUTPUT @Nwa;"OPC?;SING;"      ! Single sweep and wait
240 ENTER @Nwa;Reply                      ! Read the 1 when completed
250 !
260 OUTPUT @Nwa;"ESB?;"                   ! Event status register B byte
270 ENTER @Nwa;Estat                      ! Reading byte clears the register
280 !
290 IF BIT(Estat,4) THEN                  ! Bit 4 is failed limit on channel 1
300      IF Pass>0 THEN BEEP 1200,.05     ! passed before? Now not passing beep
310      Pass=0                           ! Reset pass to 0
320      GOTO Tune                        ! Adjust and measure again
330 END IF
340   !
350 BEEP 2500,.01                         ! Limit test passed passing beep
360 Pass=Pass+1                           ! Increment number of passes
370 DISP "LEAVE DEVICE ALONE"             ! Warn not to adjust as it passed
380 !
390 IF Pass<Qual THEN GOTO Measure        ! If not enough passes to qualify
400 !
410 ! Device passed
420 DISP "DEVICE PASSED!"                 ! Number of passes enough to qualify
430 FOR I=1 TO 10                         ! Announce the device passed and
440      BEEP 1000,.05                    ! prompt operator to connect new
450      BEEP 2000,.01                    ! device to test.
```

```
460 NEXT I
470 !
480 INPUT "PRESS RETURN FOR NEXT DEVICE",Dum$
490 Pass=0                                  ! Initialize pass counter
500 GOTO Measure                            ! Begin measurement
510 !
520 END
```

Running the Program

Note This program assumes a response calibration (through calibration) or a full 2-port calibration has been performed prior to running the program.

Set up a limit-test table on channel 1 for a specific device either manually, or using the program in Example 5A.

Run the program, and enter the number of passed sweeps desired for qualification. After entering the qualification number, connect the device. When a sweep passes, the computer beeps. When enough consecutive sweeps qualify the device as "passing," the computer emits a dual-tone beep to attract the attention of the operator, and then prompts for a new device.

To test the program's pass/fail accuracy, try causing the DUT to fail by loosening the cables connecting the DUT to the analyzer and running the program again.

Example 7: Report Generation

The analyzer has three operating modes with respect to HP-IB. These modes can be changed by accessing softkeys in the (LOCAL) menu. System-controller mode is used when no computer is present. This mode allows the analyzer to control the system. The other two modes allow a remote system controller to coordinate certain actions: in talker/listener mode the remote system controller can control the analyzer, as well as coordinate plotting and printing; and in pass-control mode the remote system controller can pass active control to the analyzer so that the analyzer can plot, print, control a power meter, or load/store to disk. Peripheral control is the main difference between talker/listener and pass-control mode. See Chapter 2 for more details.

Example 7A1: Operation Using Talker/Listener Mode

Note This program is stored as EXAMP7A1 on the "HP 8753D Programming Examples" disk received with the network analyzer.

The commands OUTPPLOT and OUTPPRIN allow talker/listener mode plotting and printing via a one way data path from the analyzer to the plotter or printer. The computer sets up the path by addressing the analyzer to talk, the plotter to listen, and then releasing control of the analyzer in order to transfer the data. The analyzer will then make the plot or print. When it is finished, it asserts the End or Identify (EOI) control line on HP-IB. The controller detects the presence of EOI and re-asserts control of the HP-IB. This example program makes a plot using the talker/listener mode.

Note One of the attributes of the OUTPPLOT; command is that the plot can include the current softkey menu. The plotting of the softkeys is enabled with the command PSOFTON; and disabled with PSOFTOFF;.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The selected frequency span is swept once.
- The plot command is sent to the analyzer.
- The analyzer is set to talker mode and the plotter is set to listener mode.
- The plot is spooled to the plotter.
- The analyzer is set to listener mode when the controller detects an EOI from the analyzer.
- The controller puts the analyzer back in continuous-sweep mode.
- The analyzer is returned to local control and the program ends.

The program is written as follows:

```
10  ! This example shows a plot operation under the control of the
20  ! analyzer. The analyzer is commanded to output plot data, the
30  ! plotter is addressed to listen, and the analyzer to talk. The
40  ! controller watches for EOI at the end of the plot sequence and
50  ! then regains control of the HP-IB operations.
60  !
70  ! EXAMP7A1
80  !
90  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
100 !
110 CLEAR SCREEN
120 ! Initialize analzyer without preset to preserve data
130 ABORT 7                               ! Generate an IFC ( Interface Clear)
140 CLEAR @Nwa                            ! SDC (Selected Device Clear)
150 !
160 OUTPUT @Nwa;"OPC?;SING;"              ! Stop sweep and prepare for plot
170 ENTER @Nwa;Reply                      ! Read in "1" when completed
180 !
190 OUTPUT @Nwa;"OUTPPLOT;"               ! Send plot command
200 SEND 7;UNL LISTEN 5 TALK 16 DATA      ! Unlisten address devices and plot
210 DISP "Plotting and waiting for EOI"
220 WAIT .5                               ! Pause 500 mS to start process
230 !
240 REPEAT                                ! Loop until EOI detected bit is set
250   STATUS 7,7;Stat                     ! Read HP-IB interface register 7
260 UNTIL BIT(Stat,11)                    ! Test bit 11 EOI on HP-IB
270 !
280 End_plot:DISP "End of plot"
290 !
300 OUTPUT @Nwa;"CONT;"                   ! Restore continuous sweep
310 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for analyzer to finish
320 ENTER @Nwa;Reply                      ! Read the 1 when complete
330 LOCAL @Nwa                            ! Release remote control
340 END
```

Running the Program

The analyzer will go into remote, and make the plot. During the plot, the computer will display the message `Plotting and waiting for EOI`. When the plot is completed, the analyzer asserts the EOI line on the HP-IB. The computer detects this and displays the `End of plot` message.

If a problem arises with the plotter, such as no pen or paper, the analyzer cannot detect the situation because it only has a one-way path of communication. Hence, the analyzer will attempt to continue plotting until the operator intervenes and aborts the plot by pressing the analyzer's [LOCAL] key.

Pressing [LOCAL] will do the following:

- Aborts the plot.
- Causes the warning message `CAUTION: PLOT ABORTED`
- Asserts EOI to return control of the bus to the system controller.

Because of possible peripheral malfunctions, it is generally advisable to use pass-control mode, which allows two way communication between the peripherals and the analyzer.

Example 7A2: Controlling Peripherals Using Pass-Control Mode

Note This program is stored as EXAMP7A2 on the "HP 8753D Programming Examples" disk received with the network analyzer.

If the analyzer is in pass-control mode and it receives a command telling it to plot, print, control a power meter, or store/load to disk, it sets bit 1 in the event-status register to indicate that it requires control of the bus. If the computer then uses the HP-IB pass-control command to pass control to the analyzer, the analyzer will take control of the bus and access the peripheral. When the analyzer no longer requires control, it will pass control back to the computer. For a discussion on the pass-control mode, see Chapter 2.

In this example, the pass-control mode is used to allow the network analyzer to dump a screen display to a printer.

Pass-control mode allows the analyzer to control the printer while sending the screen display to be printed. Once the printer-dump operation is complete, the analyzer passes control back to the controller and the controller continues programming the analyzer. The analyzer requests control from the instrument controller and the controller allows the analyzer to take control of the HP-IB and dump the plot. The instrument controller must not interact with the HP-IB while this remote analyzer control is taking place.

Note The analyzer assumes that the address of the computer is correctly stored in its HP-IB addresses menu under (LOCAL) ADDRESS: CONTROLLER . If this address is incorrect, control will not return to the computer. Similarly, if control is passed to the analyzer while it is in talker/listener mode, control will not return to the computer.

Control should not be passed to the analyzer before it has set event-status-register bit 1 making it Request Active Control. If the analyzer receives control before the bit is set, control is passed immediately back to the controller.

When the analyzer becomes the active system controller, it is free to address devices to talk and listen as required. The only functions denied the analyzer are the ability to assert the interface clear line (IFC), and the remote line (REN). These are reserved for the master system controller. As the active system controller, the analyzer can send and receive messages from printers, plotters, and disk drives.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer.
- The system is initialized.
- The status registers are cleared.
- Bit 1 of ESR request control is set.
- The ESR interrupt for SRQ is enabled.
- The pass-control mode is enabled.
- The data is dumped to the printer.
- The program loops until the SRQ bit is set.
- The status byte is read with a serial poll.
- The program tests for bit 6, SRQ.

- If SRQ is detected, the program tests for pass control (bit 5 of the status byte).
- If the analyzer requests control, the system controller gives the analyzer control of the bus.
- The program loops and waits for the analyzer to complete the print dump.
- The analyzer reads the interface.
- If bit 6 is active in the controller, control is returned from the analyzer to the controller.
- The status-byte assignments are cleared.
- The analyzer returns to continuous-sweep mode.
- The analyzer is returned to local control and the program ends.

The program is written as follows:

```
10  !  This example shows a pass-control operation to print the display
20  !  under the analyzer HP-IB control. The controller reads the status
30  ! of the analyzer looking for SRQ to indicate that the analyzer is
40  ! requesting control. Once control is passed to the analyzer, the
50  ! controller monitors the status of its interface registers to detect
60  ! when the interface is again the active controller. The analyzer will
70  ! pass control back to the controller when finished.
80  !
90  ! EXAMP7A2
100 !
110 ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
120 !
130 CLEAR SCREEN
140 ! Initialize the analyzer without preset to preserve data
150 ABORT 7                               ! Generate an IFC ( Interface Clear)
160 CLEAR @Nwa                            ! SDC (Selected Device Clear)
170 !
180 OUTPUT @Nwa;"OPC?;SING;"              ! Single sweep and stop for print
190 ENTER @Nwa;Reply                      ! Read in "1" when complete
200 !
210 OUTPUT @Nwa;"CLES;"                   ! Clear status registers
220 OUTPUT @Nwa;"ESE2;"                   ! Enable bit 1 of ESR request control
230 OUTPUT @Nwa;"SRE32;"                  ! Enable ESR interrupt for SRQ
240 !
250 OUTPUT @Nwa;"USEPASC;"                ! Enable pass control mode
260 OUTPUT @Nwa;"PRINALL;"                ! Begin printer dump
270 !
280 REPEAT                                ! Loop until SRQ bit is set
290   Stat=SPOLL(@Nwa)                    ! Read status byte with serial poll
300 UNTIL BIT(Stat,6)                     ! Test for bit 6, SRQ
310 !
320 Pass_control:                         ! SRQ detected. Test for pass control
330 IF BIT(Stat,5) THEN                   ! Requested pass control
340   PASS CONTROL @Nwa                   ! Send take control message
350 ELSE                                  ! Not bit 5, some other event
360   DISP "SRQ but not request pass control"
370   STOP                                ! Halt program
380 END IF
390 !
400 DISP "Printing from analyzer and waiting for control"
410 !
```

```
420 REPEAT                                  ! Loop and wait for completion
430   STATUS 7,6;Hpib                       ! Read HP-IB interface register
440 UNTIL BIT(Hpib,6)                       ! Bit 6 is active controller
450 !
460 DISP "Control returned from analyzer"
470 OUTPUT @Nwa;"TALKLIST;"                 ! Set talker/listener mode again
480 OUTPUT @Nwa;"CLES;"                     ! Clear status byte assignments
490 !
500 OUTPUT @Nwa;"CONT;"                     ! Start analyzer sweeping again
510 OUTPUT @Nwa;"OPC?;WAIT;"                ! Wait for analyzer to finish
520 ENTER @Nwa;Reply                        ! Read the 1 when complete
530 !
540 LOCAL @Nwa                              ! Release HP-IB control
550 END
```

Running the Program

The analyzer will briefly flash the message `WAITING FOR CONTROL`, before actually receiving control and generating the printer output. The computer will display the `Printing from analyzer and waiting for control` message.

When the printer output is complete, the analyzer passes control back to the address stored as the controller address under the (LOCAL) `SET ADDRESSES` menu. The computer will detect the return of active control and exit the wait loop. The controller will display the message `Control returned from analyzer` and then release the analyzer from remote control.

Because the program waits for the analyzer's request for control, it can be used to responding to front-panel requests as well. Remove the `PRINALL;` command from the program and run the program again. Nothing will happen until a print, plot, or disk access is requested from the analyzer's front panel. For example, press (LOCAL) (COPY) and (PRINT).

Example 7A3: Printing with the Serial Port

Note This program is stored as EXAMP7A3 on the "HP 8753D Programming Examples" disk received with the network analyzer.

This program will select the serial port and program the analyzer to copy its display to a printer. There are a number of commands associated with the serial and parallel ports which allow the programmer to configure the output modes, for example the baud rate and the handshake type used by the port and the printer. In this example, the serial port is configured by the program. The interface may also be configured from the analyzer's front-panel keys by pressing (LOCAL) `SET ADDRESSES PRINTER PORT`. This menu allows manual selection of the serial-interface parameters.

Since the HP-IB port is not being used for the copy operation, programming of the analyzer and measurement operations may continue once the copy operation has been initiated. An internal spooler in the analyzer's memory provides buffering of the printer operation. In the example which follows, the status byte of the analyzer is checked to determine when the print operation is complete.

- An I/O path is assigned to the analyzer.
- The analyzer is initialized.
- A single sweep is taken and the analyzer is placed in hold mode.
- The status registers are cleared.
- The copy-complete bit is set and enabled.

- The printer operation and communication modes are set.
- The print command is sent.
- The analyzer is released from remote control and placed in continuous-sweep mode.
- The analyzer is polled until the status bit representing copy complete is detected.
- The analyzer is released from remote control and the program ends.

The program is written as follows:

```
10  ! This program shows how to set up and print the display through the
20  ! serial printer port.
30  !
40  ! EXAMP7A3
50  !
60  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
70  !
80  CLEAR SCREEN
90  ! Initialize the analzyer without preset to preserve the data
100 ABORT 7                               ! Generate an IFC (Interface Clear)
110 CLEAR @Nwa                            ! SDC (Selected Device Clear)
120 !
130 OUTPUT @Nwa;"OPC?;SING;"              ! Single sweep and stop for print
140 ENTER @Nwa;Reply                      ! Read the 1 when complete
150 !
160 OUTPUT @Nwa;"CLES;"                   ! Clear status registers
170 OUTPUT @Nwa;"ESNB128;"                ! Enable copy complete
180 OUTPUT @Nwa;"SRE4;"                   ! Enable Event Status Register B
190 OUTPUT @Nwa;"PRNTRAUTF OFF;"          ! Set printer auto feed off
200 OUTPUT @Nwa;"PRNTYPTJ;"               ! Select ThinkJet printer
210 OUTPUT @Nwa;"PRNPRTSERI;"             ! Select serial port for output
220 OUTPUT @Nwa;"PRNTRBAUD 9600;"         ! Set baud rate to 9600 bps
230 OUTPUT @Nwa;"PRNHNDSHK XON;"          ! Use Xon-Xoff handshake
240 !
250 OUTPUT @Nwa;"PRINALL;"                ! Print screen
260 !
270 DISP "PRINTING"
280 !
290 !  Set up next measurement over HP-IB
300 OUTPUT @Nwa;"CONT;"                   ! Restore continuous sweep
310 !
320 ! Measurements can continue but wait for print to finish
330 REPEAT                                ! Test for bit 2 (4) ESRB
340 Stat=SPOLL(@Nwa)
350 UNTIL BIT(Stat,2)                     ! Wait for printer to complete
360 !
370 DISP "DONE"
380 LOCAL @Nwa                            ! Release HP-IB control
390 END
```

Example 7B: Plotting to a File and Transferring File Data to a Plotter

Note This program is stored as EXAMP7B on the "HP 8753D Programming Examples" disk received with the network analyzer.

Another report-generation technique is to transfer the plotter string to a disk file, and retrieve and plot the disk file at another time. Test time is increased when a plot occurs during the process. It may be more convenient to plot the data at another site or time. One solution to this problem is to capture the plot data using the controller and store it to a disk file. This disk file may then be read from the controller and the contents transferred to a plotter. This next example shows a method of accomplishing this task.

The analyzer is initialized without presetting the analyzer. The data that is in place on the analyzer is not disturbed by the program operation. A large string is dimensioned to hold the plotter commands as they are received from the analyzer. The length of this string depends upon the complexity of the analyzer's display. The analyzer is placed in the single-sweep mode and `OPC?;SING;` is used to make sure that operation is complete before plotting. The plotting begins with the `OUTPPLOT;` command.

The string transfer is ended by the controller detecting the EOI line which the analyzer pulls at the end of the transfer. The string transfer terminates and the plot data is now stored in a string in the analyzer.

These strings contain ASCII characters which represent the plotter commands in HP-GL (Hewlett-Packard Graphics Language). A disk file is created and the string is written into the file containing the display-plot commands.

Once the strings are transferred to the disk file, the file pointer is rewound and the data read out into a string for plotting. The string is sent to the plotter which uses the commands to generate a plot.

The following is an outline of the program's processing sequence:

- An I/O path is assigned for the analyzer
- An I/O path is assigned for the plotter
- The system is initialized
- The string for plotter commands is defined
- The frequency span is swept once
- The plotter output is requested and read into the plot string
- A plot file is created in the controller
- The plot string is stored into the disk file
- The plot string is read from the disk file and sent to the plotter
- The analyzer returns to continuous-sweep mode
- The analyzer is returned to local control and the program ends

The program is written as follows:

```
10  ! This program shows how to read the plotter output from the analyzer
20  ! and store it in a disk file as an ASCII file. The disk file is then
30  ! read back into the controller and the plot commands sent to a
40  ! plotter to generate the plot of the analyzer display. This allows
50  ! plotting at a different time than data collection.
60  !
70  ! EXAMP7B
80  !
90  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
100 ASSIGN @Plt TO 705                    ! Assign an I/O path for the plotter
110 !
120 CLEAR SCREEN
130 ! Initialize the analyzer without preset to preserve data
140 ABORT 7                               ! Generate an IFC (Interface Clear)
150 CLEAR @Nwa                            ! SDC (Selected Device Clear)
160 !
170 DIM Plot$[32000]                      ! Define string for plotter commands
180 !
190 OUTPUT @Nwa;"OPC?;SING;"              ! Stop sweep for plot and wait
200 ENTER @Nwa;Reply                      ! Read the 1 when complete
210 OUTPUT @Nwa;"OUTPPLOT;"               ! Request plotter output
220 !
230 ENTER @Nwa;Plot$                      ! Plotter output of analyzer display
240 !
250 INPUT "Plotter output complete. Press RETURN to store on disk.",Reply$
260 !
270 ! Disk file operations
280 ! Create data file on disk  32000/256 = 125 records
290 !CREATE ASCII "PLOTFILE:,1400",125    ! Use only once to generate file
300 ASSIGN @File TO "PLOTFILE:,1400"      ! Assign file I/O path
310 OUTPUT @File;Plot$                    ! Write plot string to file
320 !
330 INPUT "Plot to file is complete. Press Return to plot.",A$
340 !
350 ! Read plotter commands from file and send to plotter
360 RESET @File                           ! Reset file pointer to beginning
370 ENTER @File;Plot$                     ! Read plot string from file
380 OUTPUT @Plt;Plot$                     ! Send plot string to plotter
390 !
400 !
410 DISP "Plot is complete. End of program."
420 OUTPUT @Nwa;"CONT;"                   ! Restore continuous sweep
430 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for analzyer to finish
440 ENTER @Nwa;Reply                      ! Read the 1 when complete
450 LOCAL @Nwa                            ! Release HP-IB control
460 END
```

Running the Program

The program begins by initializing the analyzer and placing it into single-sweep mode. The plotter commands are captured into strings in the controller. The controller display prompts `Plotter output complete. Press RETURN to store on disk.` Pressing (RETURN) causes the data to be stored to disk. Once this task is complete, the program prompts once more, `Plot to file is complete. Press Return to plot.` After pressing (RETURN) again, the string output is sent to the plotter and the plot begins. Once the plot is complete, the program prompts `Plot is complete. End of program.` and the analyzer begins sweeping and returns to local control.

Plotting to a File and Utilizing PC-Graphics Applications

A slightly modified version of the previous example may be used to merely generate a file on the controller disk. This disk file can then be read into the PC and used in several different graphics-generation programs. HP-GL is a commonly recognized graphic format and may be used to transfer information to PC application programs such as CorelDRAW!®, Lotus Freelance® and other graphics packages. Importing the graphics data into these application packages allows report generation in a variety of word-processors. Graphic-data files may then be used to document test results, generate data sheets from testing results, or for simply archiving on a digital-storage medium that can be recalled for later use.

When working with HP BASIC, a utility is available to convert the HP LIF (Logical Information Format, used in the Series 9000 workstations) to a PC-DOS format file. The utility is called "LIF2DOS". This utility is available on the "HP BASIC Programming Examples" disk (HP part number 08753-10028) that was received with the analyzer. A more extensive utility called "LIFUTIL" allows file transformation in both directions and is available as a commercial product. Contact the nearest HP sales and service office for ordering information.

Once the HP-GL file is present in the DOS file system, the HP-GL file is imported and examined with the graphics package. The text labels may need to be rescaled, but on the whole, the graphics results are quite usable.

Example 7C: Reading ASCII Disk Files to the System Controller Disk File

Note This program is stored as EXAMP7C on the "HP 8753D Programming Examples" disk received with the network analyzer.

Another way to access the analyzer's test results is to store the data onto a disk file from the analyzer. This operation generates an ASCII file of the analyzer data in a CITIFILE format. A typical file generated by Example 7C is shown below:

```
CITIFILE A.01.00
#NA VERSION HP8753C.04.13
NAME DATA
VAR FREQ MAG 11
DATA S[1,1] RI
SEG_LIST_BEGIN
SEG 100000000 200000000 11
SEG_LIST_END
BEGIN
8.30566E-1,-1.36749E-1
8.27392E-1,-1.43676E-1
8.26080E-1,-1.52069E-1
8.25653E-1,-1.60003E-1
8.26385E-1,-1.68029E-1
8.26507E-1,-1.77154E-1
8.26263E-1,-1.87316E-1
8.26721E-1,-1.97265E-1
8.2724E-1,-2.07611E-1
8.28552E-1,-2.19940E-1
8.29620E-1,-2.31109E-1
END
```

This data file can be stored from the analyzer by remote control or by front-panel operations. See "Saving Instrument States" in the Printing, Plotting, and the Saving Measurement Results chapter in the *HP 8753D Network Analyzer User's Guide* for more details on manual operation.

This program stores a file in the same manner as an operator would store a file on to the analyzer's internal disk drive from the front panel. If a PC is being used as the system controller, the disk format will have to be transformed from LIF to PC-DOS format. A file utility is available to perform this format translation. The utility is called "LIF2DOS". This utility is available on the "HP BASIC Programming Examples" disk (HP part number 08753-10028) that was received with the analyzer. Once transformed, the PC will now have a DOS-format file to read and interpret.

This example explains the process of storing the data from the analyzer to a file on the internal disk drive. There is also a program to read the data from the file into a data array for further processing or reformatting to another file type. The internal drive will store in the same format that is present on the disk. A new disk may be formatted in either LIF or DOS. For the example, the assumption has been made that the format transformation has already taken place, and there is a file that can be read record by record, from which data can be retrieved.

The goal of this example is to recover an array of stimulus frequency along with the trace-data values. CITIFILES contain the real and imaginary values of each data point. Some further transformation will be required to obtain magnitude values, for example.

The disk file contents for this example are shown above. This file contains more information than will be used in this example. The file is accessed and the records read from the file and

printed on the controller display to observe the actual file contents. The file pointer is reset and the records are then read and interpreted for their data contents.

The first six records are skipped for this example. The seventh record contains the stimulus-frequency values and the number of points in the trace. These values are read from the record. The frequency increment, or point spacing, is calculated and used later the frequency-data calculations for a point. Two more records are skipped and the next is the first record representing data values. The data values are read in a loop until the values for the number of points have been recovered from the file. The data values are tabulated and printed out on the controller display.

The following is an outline of the program's processing sequence:

- An I/O path is assigned to the analyzer.
- The system is initialized.
- A string is dimensioned to hold a file record.
- The analyzer operating state is set.
- The internal drive is selected for storage (only ASCII data is stored).
- A file name is entered and the data stored into it.
- The operator is prompted to move the disk to the controller disk drive.
- The disk file is read and the contents displayed.
- The file pointer is rewound.
- The file contents are converted to trace data.
- The frequency and complex-data pair is displayed for each point.
- The analyzer is restored to continuous-sweep mode.
- The analyzer is returned to local control and the program ends.

The program is written as follows:

```
10  ! This program shows how to store an ASCII data file in CITIFILE format
20  ! and retrieve the data with the controller. The disk is written in the
30  ! analyzer system and then moved to the controller disk and the data
40  ! accessed.
50  !
60  ! EXAMP7C
70  !
80  ASSIGN @Nwa TO 716                    ! Assign an I/O path for the analyzer
90  !
100 CLEAR SCREEN
110 ABORT 7                               ! Generate an IFC (Interface Clear)
120 CLEAR @Nwa                            ! SDC (Selected Device Clear)
130 OUTPUT @Nwa;"OPC?;PRES;"              ! Preset the analyzer and wait
140 ENTER @Nwa;Reply                      ! Read the 1 when complete
150 !
160 DIM Record$[80]                       ! String to read the disk records
170 !
180 ! Set up analyzer
190 OUTPUT @Nwa;"STAR100MHZ;"             ! Start frequency 100 MHz
200 OUTPUT @Nwa;"STOP 200MHZ"             ! Stop frequency 200 MHz
210 OUTPUT @Nwa;"POIN11;"                 ! Trace length 11 points
220 OUTPUT @Nwa;"OPC?;SING;"              ! Single sweep and wait
230 ENTER @Nwa;Reply                      ! Read in the 1 when complete
240 !
250 ! Program disk storage operation
260 !
270 OUTPUT @Nwa;"INTD;"                   ! Select internal disk file
280 OUTPUT @Nwa;"EXTMFORMON;"             ! Store formated data
290 OUTPUT @Nwa;"EXTMDATOON;"             ! Store data file only
300 INPUT "Enter data file name (5 chars)",File_name$ ! Get file name
310 File_name$=UPC$(File_name$)           ! File names are uppercase
320 OUTPUT @Nwa;"TITF1""";File_name$;""";" ! Title for save reg 1
330 OUTPUT @Nwa;"SAVUASCI;"               ! Save as ASCII file
340 !
350 OUTPUT @Nwa;"STOR1;"                  ! Store data to disk file
360 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait until store is complete
370 ENTER @Nwa;Reply
380 !
390 ! File storage is complete
400 !
410 INPUT "Place disk in controller disk drive, then press Return",A$
420 !
430 ! Read data file information
440 !
450 ASSIGN @File TO File_name$&"D1:,1400"  ! Open an I/O path for file
460 Record_cnt=1                          ! Counter to count records
470 !
480 PRINT CHR$(12);                       ! Formfeed to clear display
490 PRINT "Contents of data file"         ! Show contents of the data file
500 Readfile:   !
510 ON END @File GOTO End_file            ! Test for end of file and exit
520 ENTER @File;Record$                   ! Read ASCII record
530 PRINT Record_cnt,Record$              ! print record on display
```

```
540 Record_cnt=Record_cnt+1                ! Increment record counter
550 GOTO Readfile                          ! Read next record
560 !
570 End_file: !                            ! Reached the end of file
580 PRINT "End of File. ";Record_cnt-1;" Records found"
590 INPUT "Press Return to continue",A$
600 PRINT CHR$(12);                        ! Formfeed to clear display
610 !
620 ! Read file data into arrays
630 !
640 RESET @File                            ! Rewind file pointer to begining
650 FOR I=1 TO 6
660   ENTER @File;Record$                  ! Skip first six records
670 NEXT I
680 ENTER @File;Record$                    ! Read frequency data record
690 Record$=Record$[POS(Record$," ")+1]    ! skip SEG to first space + 1
700 Startf=VAL(Record$)                    ! Read start frequency
710 Record$=Record$[POS(Record$," ")+1]    ! Skip to next space + 1
720 Stopf=VAL(Record$)                     ! Read stop frequency
730 Record$=Record$[POS(Record$," ")+1]    ! Skip to next space +1
740 Num_points=VAL(Record$)                ! Read the number of points
750 PRINT " Number of points in file ";Num_points
760 PRINT                                  ! White space
770 !
780 Freq_inc=(Stopf-Startf)/(Num_points-1) ! Compute frequency increment
790 !
800 ALLOCATE Array(Num_points,2)           ! Allocate array from Num_points
810 ENTER @File;Record$                    ! Skip SEG_LIST_END record
820 ENTER @File;Record$                    ! Skip BEGIN record
830 !
840 ! Read in the data array
850 PRINT "Freq (MHz)  Data 1    Data 2"  ! Table header for data array
860 FOR I=1 TO Num_points                  ! Read in array entries
870   ENTER @File;Record$                  ! Read in the record of 2 entries
880   !
890   Array(I,1)=VAL(Record$)              ! Read first data value
900   Data$=Record$[POS(Record$,",")+1]    ! Skip to comma and next value
910   Array(I,2)=VAL(Data$)                ! Read second data value
920   !
930   Freq=Startf+(Freq_inc*(I-1))         ! Compute stimulus value for array
940   Freq=Freq/1.E+6                      ! Convert frequency to MHz
950   !
960   PRINT Freq,Array(I,1),Array(I,2)     ! Print data array values
970 NEXT I                                 ! Read next array data points
980 !
990  OUTPUT @Nwa;"CONT;"                   ! Restore continuous sweep
1000 OUTPUT @Nwa;"OPC?;WAIT;"              ! Wait for analyzer to finish
1010 ENTER @Nwa;Reply                      ! Read the 1 when complete
1020 LOCAL @Nwa                            ! Release HP-IB control
1030 END
```

Running the Program

The analyzer is initialized and the operating range re-defined to an 11-point trace from 100 to 200 MHz. This setup gives a restricted range to be evaluated when the ASCII data file (CITIFILE) is read in from the controller. The operator is prompted for a 5-character filename to use for storing the data. The analyzer is setup for external storage and stores the data file. Once the "pass control/storage/return control" operation is complete, the operator is prompted to place the disk in the controller disk drive and press RETURN. The disk is then read and the records contained in the file are printed on the controller display. A prompt appears, `Press return to continue`, which allows viewing of the file contents. Once RETURN is pressed, the data records are read and decoded and a table of the stimulus frequency and the data values are printed.

Index

Index

D

K

L

M

P

S

T