3

Remote Operation

3.1 Introduction

The Series 8650A can be operated from a remote host over the General Purpose Interface Bus (GPIB) using either Standard Commands for Programmable Instruments (SCPI) or IEEE Standard 488-1978 (Digital Interface for Programmable Instruments) commands. Table 3-1 shows which functions of the IEEE 488 standards are implemented in the 8650A.

Table 3-1: Implemented IEEE Standards

Function	8650A Implementation
Source Handshake	SH1 (complete capability)
Acceptor Handshake	AH1 (complete capability)
Talker	T5 (basic talker, serial poll, talk only mode, unaddressed if MLA)
Extended Talker	TE0 (no capability)
Listener	L3 (basic listener, listen only mode, unaddressed if MTA)
Extended Listener	LE0 (no capability)
Service Request	SR1 (complete capability)
Remote/Local	RL1 (complete capability)
Parallel Poll	PP1 (remote configuration)
Device Clear	DC1 (complete capability)
Device Trigger	DT1 (complete capability)
Controller	C0 (no capability)

3.1.1 Sending Commands to the 8650A

The 8650A power meter uses standard protocols for communication over the GPIB. Commands conform to IEEE 488.1 or IEEE 488.2 guidelines. Four emulation modes (Giga-tronics 8540, HP436, HP437 and HP438) are available for users of power meters who cannot rewrite their application software.

The program examples in this chapter are written in HTBasic™ format (HTBasic is a trademark of TransEra Corporation). Other languages would use different commands but the string that is sent or received will always be the same. In HTBasic, the **OUTPUT** command sends a string to the GPIB. The number after **OUTPUT** is the GPIB address of the instrument.

The factory-set default address of the 8650A is 13 and the address of the GPIB is assumed to be 7; therefore, examples of command strings in this publication are preceded by **OUTPUT 713**.

3.1.1.1 Clear Device

The interface command **CLEAR 713** resets the GPIB and sets the 8650A to its preset condition.

3.1.1.2 Clear Interface

The interface command **ABORT 7** resets the GPIB without resetting the 8650A to its preset condition. The 8650A will not be addressed after the abort.

3.1.1.3 Local & Remote Control

The interface command **LOCAL 713** places the 8650A into the local control mode.

The interface command **REMOTE 713** places the 8650A into the remote control mode. Enter LOCAL 713 to return the instrument to local mode.

The interface command **LOCAL LOCKOUT 7** places the 8650A in the local lockout mode. This is a remote control mode in which all of the 8650A front panel keys are disabled. The **GPIB LOCAL** command must be issued to return the 8650A to local mode (Disconnecting the GPIB cable will also return the instrument to local mode).

3.1.1.4 Sensor Selection & Calibration

Power sensor selection data, specifications and calibration (local and remote) are contained in Appendix B of this publication.

3.1.2 Polling

The GPIB supports parallel and serial polling. The example programs below show how to use the parallel and serial poll capabilities of the 8650A to determine when a requested zeroing operation is completed.

3.1.2.1 Parallel Polling

```
Ppoll_zero                                  ! zero using parallel poll
PRINT entering parallel poll zero routine
PPOLL CONFIGURE 713;8                       ! configure response on bit zero
OUTPUT 713;CS  AEZE                         ! clear status byte, zero channel A
State=0                                     ! initialize variable
WHILE State 1                               ! stay here until zero done
      State=PPOLL(7)                        ! read the poll
      END WHILE PPOLL UNCONFIGURE 713       ! cancel parallel poll mode
PRINT parallel zero done RETURN
```

3.1.2.2 Serial Polling

```
Srq_zero:                                   ! zero with an srq interrupt
PRINT entering SRQ interrupt zero routine
ON INTR 7 GOSUB Srq_interrupt
OUTPUT 713;CS                               ! clear status byte
ENABLE INTR 7;2                             ! enable srq interrupts
OUTPUT 713;@1;CHR$(2)                       ! enable srq handshake
OUTPUT 713;AEZE                             ! execute zero command
Flag=0                                      ! test flag reset to false
      WHILE Flag=0                          ! stay here until test flag set true
      WAIT 1
      PRINT Still inside while loop
END WHILE
PRINT SRQ interrupt zero done
RETURN
Srq_interrupt:                              ! SRQ interrupts jump here
PRINT an SRQ interrupt has occurred
Example:OUTPUT 713;CS                       ! clear status byte
Flag=1                                      ! set control flag true
RETURN
```

3.1.3 Data Output Formats

The data output format for the standard measurement collection mode is:

±D.DDDDE±NNCRLF

±: Sign of the Mantissa
D.DDDD: Mantissa (5 digits)
E: Exponent (indicates that an exponent follows)
±: Sign of the Exponent
NN: Magnitude of the Exponent
CR: Carriage Return
LF: Line Feed

Data output formats for the swift and fast buffered modes are expressed in the form of a signed five-digit number with two digits to the right of the decimal and no exponents. In some cases multiple values are sent:

One sensor swift mode:±DDD.DD CRLF
Two sensor swift mode:±DDD.DD,±DDD.DD CRLF
Fast buffered mode:±DDD.DD, ...±DDD.DD CRLF

3.1.4 Power-On Default Conditions

The interface wake-up state is:

- GPIB Local Mode
- Unaddressed, Service Request Mask Cleared
- Status Byte Cleared
- TR3 Free Run Trigger Mode Set
- GT2 Group Execute Trigger Mode Set
- Parallel Poll Data Lined Unassigned
- Display Enabled
- Service Request Mask Cleared
- Event Status Register = 128
- Event Status Mask Clear

3.2 SCPI Command Interface

This section details operation of the 8650A power meter using the SCPI (Standard Communications for Programmable Instruments) interface commands. A SCPI command reference is presented in Table 3-3 and the sections that follow.

SCPI commands promote consistency in definition of a common instrument control and measurement command language. The structured approach of the SCPI standard offers test system design engineers a number of system integration advantages that achieve considerable efficiency gains during control program development.

SCPI compatible instrument commands are structured from a common functional organization or model of a test instrument (See Figure 3-1). Most of the power meter configuration and measurement functions fall within the Measurement Function Block and the Trigger Subsystem of the SCPI instrument model.

The 8650A uses the Sense Subsystem of the Measurement Function Block to implement commands that apply specifically to the individual power sensors; sensor 1 and sensor 2. For example, the SENSe2:CORRection:OFFSet command corrects for the attenuation of a signal that passes through an attenuator or coupler before it is measured by power sensor 2. Figure 3-1 illustrates the SCPI subsystem model.

☛ ***NOTE:*** *Throughout this publication, some commands will be in both upper- and lower-case, such as CALCulate and MEMory. The uppercase is the required input. The whole word can be used if desired. If the whole word is used, it must be used in its entirety. For example, CALCulat will not be recognized as a valid command.*

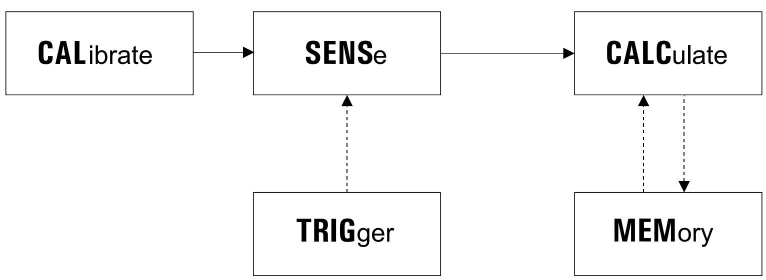

Figure 3-1: SCPI Subsystem Model

3.2.1 Sensor Calibration & Zeroing

The Calibration subsystem performs sensor calibration and zeroing. CALibrate:STATe*n* selects if the calibration data is applied or not. If STATe is ON, then the instrument uses the calibration data for correction. If STATe:OFF is selected, then no calibration using the calibration data will be made. The CALibrate:ZERO subsystem controls the autozero calibration of the sensor.

3.2.2 Sensor & Channel Configuration

The Calculate subsystem of the Measurement Function Block contains commands that define the form of the measured data from sensor 1 and sensor 2. Calculate commands define the configuration of the two Software Calculation Channels. For example, the CALC1:POW 1 command configures channel 1 to report the power level as measured by sensor 1. CALC2: RAT 2,1 configures channel 2 to report the ratio of power levels, sensor 2 over sensor 1.

3.2.3 Measurement Triggering

The power meter uses the Trigger Subsystem to trigger measurements in two different operational modes - a normal mode which maximizes the instrument's functionality, and swift and burst modes that maximize the power measurement rate.

3.2.4 Memory Functions

The MEMory commands control the configuration of the automated Voltage-Proportional-to-Frequency ($V_{PROP}F$ or VF), sensor Cal Factor correction, and the analog outputs on the rear panel. Each of these connectors is used with external devices. The $V_{PROP}F$ can be configured to match the $V_{PROP}F$ output of the microwave source. The analog outputs are used with a variety of devices including chart recorders, oscilloscopes, voltmeters and microwave source leveling inputs.

3.2.5 IEEE 488.2 Required Commands

Consistent with SCPI compliance criteria, the power meter implements all the common commands declared mandatory by IEEE 488.2 (See Table 3-2).

Table 3-2: IEEE Required Command Codes

Mnemonic	Name
*CLS	Clear Status Command
*ESE	Standard Event Status Enable Command
*ESR?	Standard Event Status Register Query
*IDN?	Identification Query
*OPC	Operation Complete Command
*RCL	Recall Command
*RST	Reset Command
*SAV	Save Command
*SRE	Service Request Enable Command
*STB?	Read Status Byte Query
*TRG	Trigger Command
*TST?	Self-Test Query
*WAI	Wait-to-Continue Command

3.2.6 Calculate Subsystem Commands

Calculate commands specify and query the configuration of power measurement channels, known in SCPI references as Software Configuration Channels, and in this publication as "channels". See Section 3.2.7 for sensor-specific configuration and measurement function control.

The query form of CALC#?, with # replaced by the channel modifier (1 or 2), returns the current configuration status for that channel. This verifies configuration commands or return current status information following data acquisition or power measurements.

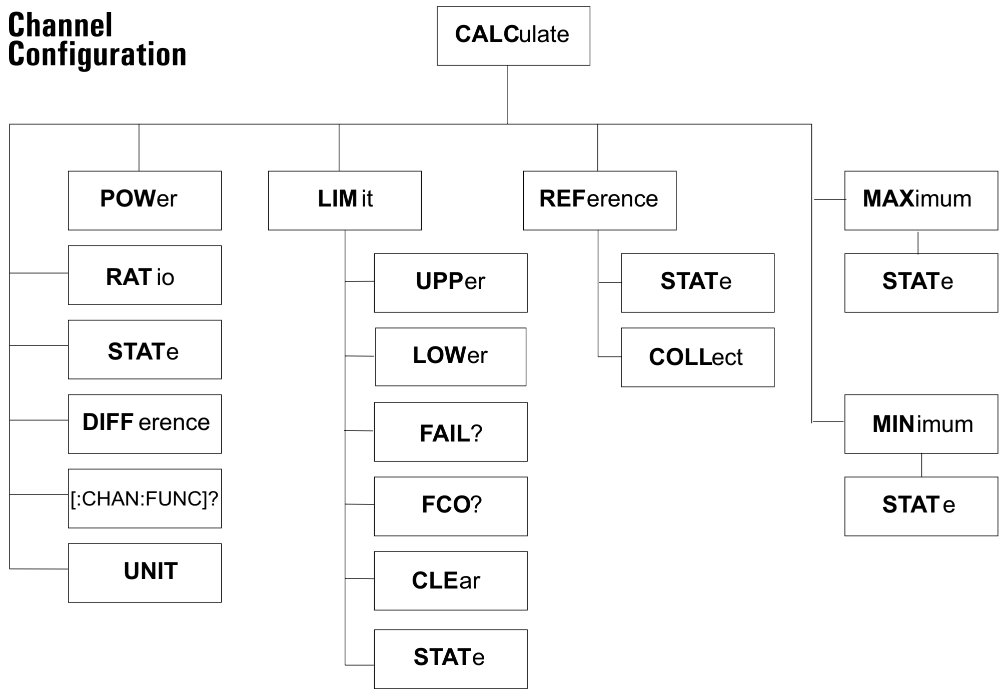

Figure 3-2: CALCulate Subsystem Commands

Limit Lines are set on a channel basis. These commands set limits, monitor the number of violations, and allow the violation counter to be cleared.

The REFerence command allows channel-based offset values. For example, using CALC#:REF:COLL automatically converts the inverse of the current channel measurement value to an offset — simplifying the 1 dB compression testing of amplifiers.

MIN and MAX commands monitor deviation of measured values over a user controllable time period.

The two software calculation channels can individually and simultaneously perform the internal instrument functions that calculate final measurement data. The final measurement data is calculated from Sense subsystem sensor data as well as the Calculate subsystem channel configuration data.

This means that only two measurement configurations can be obtained from the 8650A simultaneously. For example, the controller can obtain measurements for sensor 1 plus sensor 2/sensor 1 simultaneously, but not sensor 1 plus sensor 2 plus sensor 2/sensor 1 simultaneously.

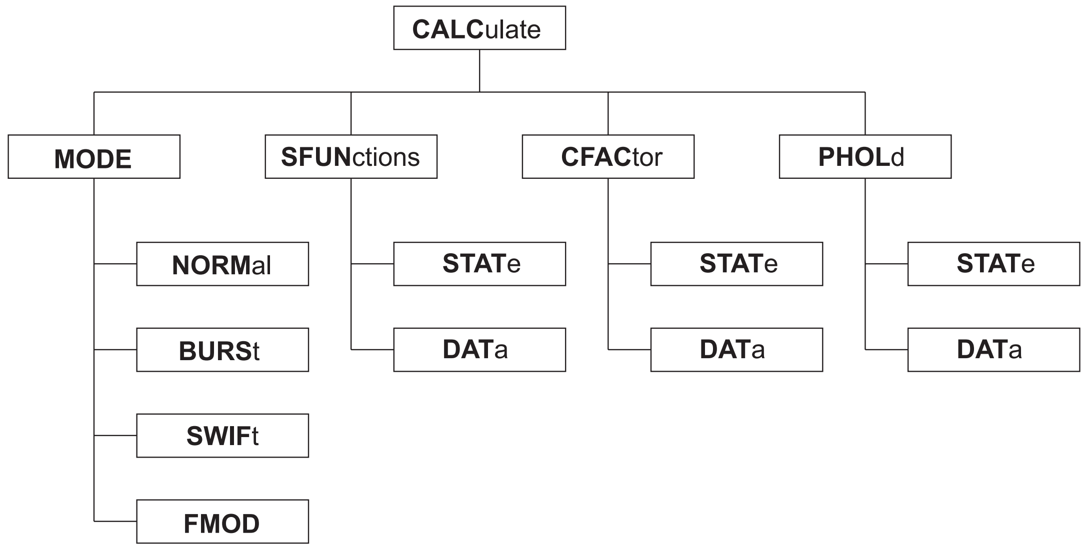

Figure 3-3: Additional CALCulate Subsystem Commands (Modulation Sensors)

3.2.7 Sense Subsystem Commands

The Sense subsystem configuration commands, illustrated in Figure 3-4, apply to individual sensors. These commands alter the value of the measured power level according to the sensor's characteristics. For example, measured power levels can be offset for attenuators or couplers in the measurement path so that the power data reading reflects the power level at the measurement point of interest.

Use the Sense subsystem commands for:

- Averaging power measurements in the 8650A
- Offsetting power measurements for attenuation or amplification
- Entering the operating frequency of the measured signal computes and applies sensor specific Calibration Factor corrections, which compensates for sensor frequency response characteristics
- Controlling Peak Power Sensor triggering
- Controlling Modulation Power Sensors

Sense subsystem commands control functions that are related directly to the individual power sensors. For example, these commands control items that would not apply to numerical alteration of a ratio measurement of Sensor1/Sensor2. Controls that would apply to that type of a configuration are channel functions, not sensor functions, and would therefore be located in the Calculate subsystem.

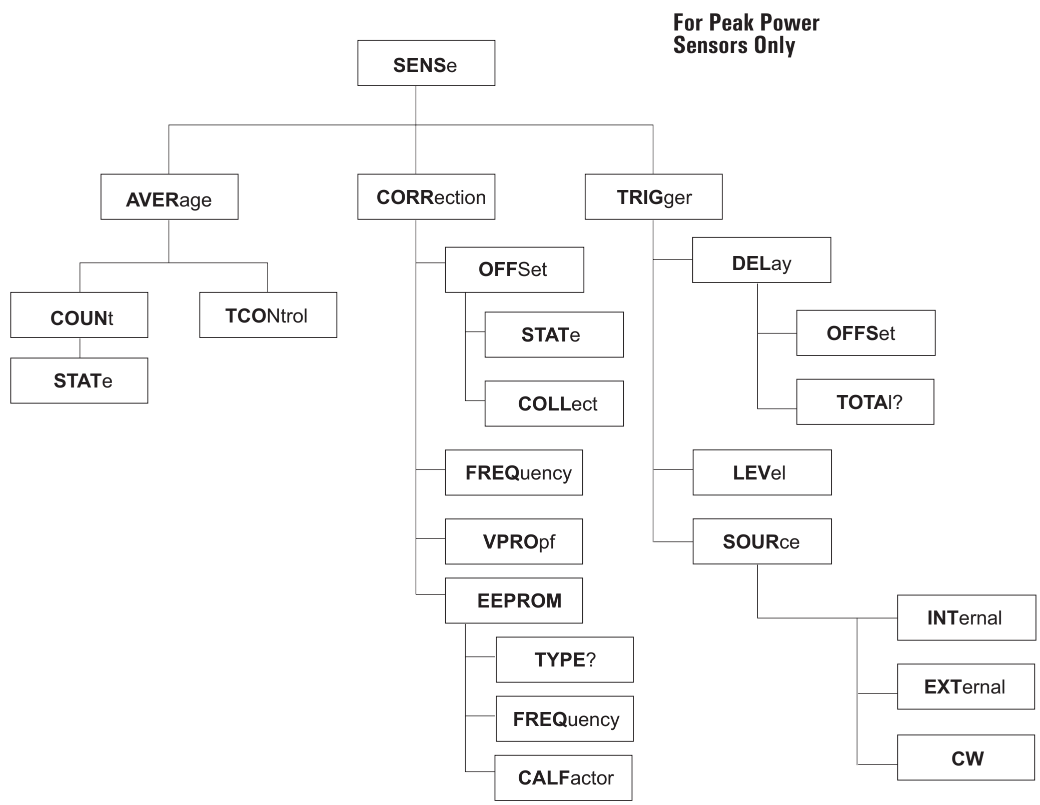

Figure 3-4: SENSe Subsystem Command Tree

SENSe:AVERage functions control the number of data samples for each measurement and the manner in which those numbers are accumulated. COUNt determines the averaging number or AUTO-averaging. TCONtrol determines whether each new sample is added to previous COUNt # of samples or if COUNt # of samples are taken each time the 8650A is triggered. Please note that the SENSe:TRIGger commands are not instrument triggers, but Peak Power Sensor configuration controls.

SENSe:TRIGger functions apply only to Giga-tronics Peak Power Sensors. The DELay and LEVel functions of these Peak Power Sensor controls apply to the 80350A Peak Power Sensors. The AVERage and CORRection commands apply to all 803XXA Series CW & Peak Power Sensors.

STATe ON OFF controls for COUNt and DELay, previous page, are not shown.

3.2.8 Trigger Subsystem Commands

The Trigger Subsystem is divided into two sections; Instrument Measurement Event Triggering and Special Triggering Configuration commands for the fast-reading buffered data modes (Burst and Swift modes). The Trigger command tree is illustrated in Figure 3-5.

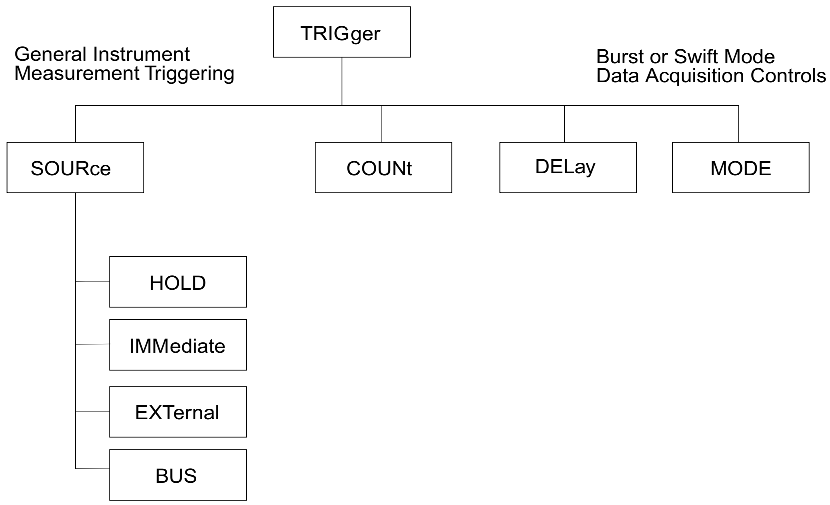

Figure 3-5: TRIGger Subsystem Command Tree

The query form of these commands, TRIGger? with the appropriate modifier inserted ahead of the ?, will return the instrument's current configuration status. This can be used to verify triggering configuration or return status information following command errors which are commonly caused by using illegal configuration commands during Swift or Burst Modes.

SOURce:IMMediate triggering allows the 8650A to control measurement triggering; this is the default configuration. External triggering is performed using a TTL signal input. BUS allows software controlled triggering.

COUNt refers to the number of data points to store in the meter's 5000 reading buffer (128,000 with option 02) before the measurement data is requested by the controller.

DELay controls the time interval between Burst Mode data samples, and the MODE command controls whether the data is taken after receipt of the instrument trigger or if data is collected (in a FIFO buffer) immediately preceding receipt of an instrument trigger.

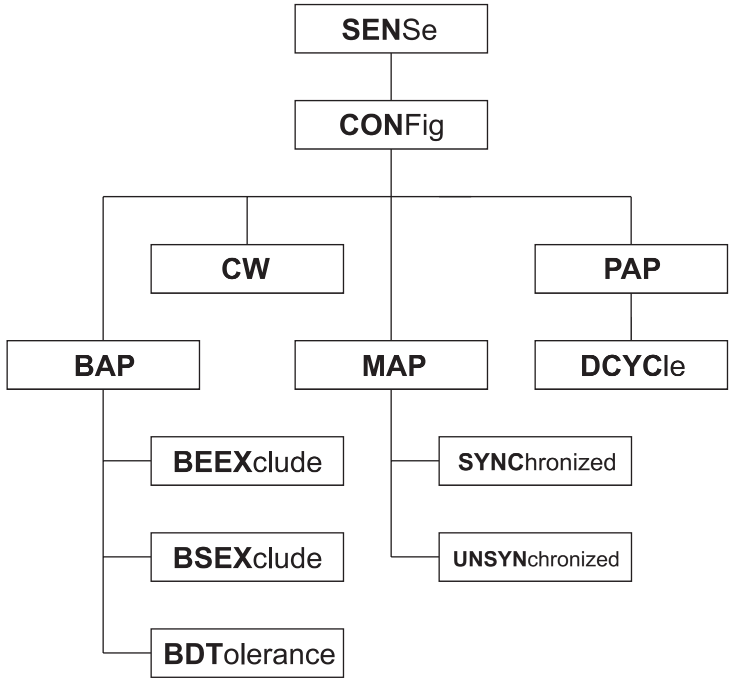

Figure 3-6: SENSe Subsystem Command Tree for Configuring Modulation Sensors

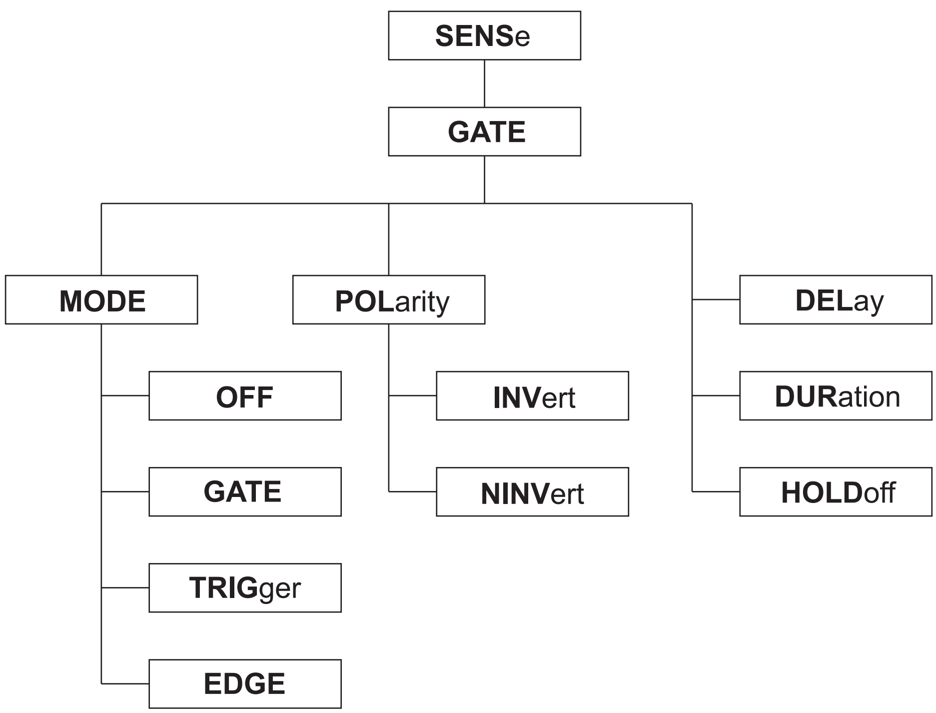

Figure 3-7: SENSe Subsystem Command Tree for Configuring Modulation Sensor Gate Setting

3.2.9 GPIB Command Syntax

The following conventions are used with the GPIB commands in this publication. Some commands will be in both upper- and lower-case, such as CALCulate and MEMory. The uppercase is the required input. The whole word can be used if desired. Table 3-3 lists in alphabetical order all of the GPIB commands supported by the power meter. The basic function performed by that command is given, and the page number of this chapter where a description of each command can be located.

3.2.9.1 Commands in Brackets

Commands and command separators within brackets, such as [COMMand:], are optional. These portions of the commands may be used in the program command strings, but are not required for proper operation of the power meter.

3.2.9.2 Programmer Selective Parameters

Command descriptions enclosed in angle brackets (< and >) show the syntax placement of configurable parameters. A description of the necessary parameter and the range of values or mnemonics valid for that parameter are enclosed in angle brackets.

3.2.9.3 Italics in Syntax Descriptions

Some command syntax descriptions show certain words in italics such as *space* and *comma* to indicate where the character must be included within a command string.

3.2.9.4 Query Format

Except where specifically noted, all query commands are formed by adding a question mark (?) to the command header. Be sure to omit command parameters when using the query format. Some commands have only a query format. With the exception of the Calibration queries, query commands will not change the status of the power meter. The CALibration1? and CALibration1:ZERO? commands query respectively to automatically begin the calibration and zeroing process.

3.2.9.5 Linking Command Strings

The 8650A uses ASCII strings for commands. When sending more than one command in a single string, a semicolon must be used as a delimiter between commands. No spaces or other characters are necessary. Use only a semicolon to link commands in a string.

3.2.9.6 Measurement Data Output Format

The examples in this chapter are written in HTBasic™ format. Different languages will use different commands, but the string sent or received will always be the same. In HTBasic, the OUTPUT command sends a string to the GPIB bus. The number or variable after the word OUTPUT is the GPIB address of the power meter.

™HTBasic is a trademark of TransEra Corporation.

Table 3-3: SCPI Command Syntax

Command Syntax	Function	Page
SCPI Required Codes		
*CLS	Clear SRQ and status byte registers	3-46
*ESE	Enable bit 4 of event status register mask	3-46
*ESR?	Returns event status register value	3-46
*IDN?	Query instrument mfgr and model number	3-59
*OPC	Allows a one-time SRQ enable	3-27 3-46
*OPC?	Returns a value of 1 after an operation is completed	3-27 3-46
*RCL	Recall 8650A register *n*	3-55
*RST	Reset 8650A configuration	3-57
*SAV	Save at 8650A register *n*	3-55
*SRE	Service Request Enable	3-47
*STB?	Query status byte register	3-47
*TRG	BUS Trigger Event	3-29
*TST?	Query self-test result	3-61
*WAI	Wait, following command completion	3-30
SCPI Function Codes		
ABORt	Halts measurement & triggering	3-55
CALCulate<channel 1 to 4>:CFACtor:STATe*space*<ON | OFF>	Enable/Disable crest factor measurements.	3-33
CALCulate<channel 1 to 4>:CFACtor:DATa?	Reads the crest factor measurements.	3-34
CALCulate<channel 1 to 4>[:CHANnel]: difference*space*<sensor 1 or 2*comma*sensor 2 or 1>	Configures channel 1 to measure Sensor 2 minus Sensor 1 power	3-32
CALCulate<channel 1 to 4>:DATA?	Query channel 1 & 2 burst mode data	3-37
CALCulate<channel 1 to 4>[:FUNCtion]?	Query channel 2 sensor mode	3-33
CALCulate<channel 1 to 4>:LIMit:CLEar[:IMMediate]	Reset channel 1 limit violation indicator to 0	3-51
CALCulate<channel 1 to 4>:LIMit:FCOunt?	Query number of channel 1 limit failures	3-51
CALCulate<channel 1 to 4>:LIMit:FAIL?	Check for channel 1 limit line violation (Output 0 = OK; 1 = fail)	3-51
CALCulate<channel 1 to 4>:LIMit:LOWer*space*<numeric value in dB from -299.99 to 299.99>	Set channel 1 lower limit line to -50 dBm	3-52
CALCulate<channel 1 to 4>:LIMit:STATe*space*<ON | OFF>	Enable channel 1 upper and lower limit checking. (0 = off; 1 = on)	3-53
CALCulate<channel 1 to 4>:LIMit:UPPer*space*<numeric value in dB from -299.99 to 299.99>	Set channel 1 upper limit line to 17 dBm	3-53
CALCulate<channel 1 to 4>:PHOLd:STATe*space*< ON | OFF>	Enable/disable peak hold measurements	3-34
CALCulate<channel 1 to 4>:PHOLd:DATa?	Reads the peak hold measurements	3-34
CALCulate<channel 1 to 4>[:CHANnel]:POWer*space*<sensor 1 or 2>	Configures channel 2 to measure Sensor 1 power	3-32

Table 3-3: SCPI Command Syntax (Continued)

Command Syntax	Function	Page
CALCulate<channel 1 to 4>[:CHANnel]:RATio*space*<sensor 1 or 2*comma*sensor 2 or 1>	Configures channel 1 to measure Sensor 2 over Sensor 1 power	3-32
CALCulate<channel 1 to 4>:MODE*space*<NORMal, BURst, SWIFt or FMOD Selection>	Set up time gating mode on the specified input	3-37
CALCulate<channel 1 to 4>:MAXimum:STATe*space*<ON | OFF>	Enable channel 2 maximum value monitoring	3-50
CALCulate<channel 1 to 4>:MAXimum[:MAGnitude]?	Query channel 2 maximum value in dBm	3-50
CALCulate<channel 1 to 4>:MINimum:STATe*space*<ON | OFF>	Enable channel 1 min. value monitoring	3-50
CALCulate<channel 1 to 4>:MINimum[:MAGnitude]?	Query channel minimum value in dBm	3-50
CALCulate<channel 1 to 4>:REFerence:STATe*space*<ON | OFF>	Activate level reference for relative measurements	3-43
CALCulate<channel 1 to 4>:REFerence[:MAGnitude]<dB Offset value from -299.999 to 299.999>	Set channel 2 reference offset value in dB Reset channel 2 Relative meas. operation	3-43
CALCulate<channel 1 to 4>:REFerence:COLLect	Take channel 2 reading as reference value. Reset with CALC:REF 0.0	3-43
CALCulate<channel 1 to 4>:SFUNctions:STATe*space*<ON |OFF>	Enables/disables statistical functions (means and standard)	3-34
CALCulate<channel 1 to 4>:SFUNctions:DATa?	Reads the result produced by the statistics function. The response is in the form <means>, <standard deviation>	3-34
CALCulate<channel 1 to 4>:STATe*space*<ON | OFF>	Disable channel 1 measurement	3-33
CALCulate<channel 1 to 4>:UNIT[:POWer]*space*<data units selection, dBm or Watt>	Selects channel 1 linear units in Watts Selects channel 1 Log Units in dBm	3-33
CALibrate<sensor 1 or 2>[:SENSor]	Calibrate Sensor 1	3-18
CALibrate<sensor 1 or 2>:STATe?	Query sensor for calibration status	3-18
CALibrate<sensor 1 or 2>:ZERO	Zero Sensor 1	3-19
FETCh<channel 1 to 4>?	Read previously triggered channel 1 power, Normal, Swift, Burst or FMOD mode data	3-26
INITiate[:IMMediate]	Initiate instrument triggering cycle	3-31
INITiate:CONTinuous*space*<ON | OFF>	8650A self-triggers continuously	3-31
MEASure<channel 1 to 4>[:SCALar:POWer]?	Configure triggering and measure channel 2 power in NORMal mode	3-26
MEMory[:TABLe]:CHANnel*space*<channel 1 to 4>	Set analog out to channel 2	3-53
MEMory[:TABLe]:FREQuency< Start frequency>	Set V_{PROP}F start frequency to 1 GHz	3-36
MEMory[:TABLe]:POWer*space*<start value from -80 to +20 dBm>*comma*<stop value from -80 to +20 dBm>	Set analog out power range (in dBm) from -80 to 20 dBm	3-53
MEMory[:TABLe]:POWer*space*<start value from 0 to 0.01W>*comma*<stop value from 0 to 0.01W>	Set analog out power range (in Watts) from 0 to 0.01 Watts	3-53
MEMory[:TABLe]:SELect*space*<ANALOGout,VPROPF1, VPROPF2>	Select memory table V_{PROP}F1 for editing in following lines	3-35 3-53
MEMory[:TABLe]:SLOPe*space*<Volts per Hz>	Set V_{PROP}F slope in 1V/GHz	3-35
MEMory[:TABLe]:UNIT*space*<Units for analog output configuration, dBm or Watt>	Set analog out power unit	3-53
MEMory[:TABLe]:VOLTage*space*<start value from -10 to +10 V>*comma*<stop value from -10 to +10 V>	Set analog out voltage range from -8 to 2 volts corresponding to power	3-54

Table 3-3: SCPI Command Syntax (Continued)

<table>
<tr><th>Command Syntax</th><th>Function</th><th>Page</th></tr>
<tr><td>OUTPut[:BNC]:ANAlog[:STATe]space<ON | OFF></td><td>Enable analog out function</td><td>3-53</td></tr>
<tr><td>OUTPut:ROSCillator[:STATe]space<ON | OFF></td><td>Turn ON 0.0 dBm Calibrator Oscillator</td><td>3-60</td></tr>
<tr><td>READ<channel 1 to 4>[:POWer]?</td><td>Trigger measurement and read channel 2</td><td>3-26</td></tr>
<tr><td>SENSe<sensor 1 or 2>:AVERage:COUNtspace<averaging value 1,2,4,8,16,32,64,128,256,or512></td><td>Set sensor 1 average number to 16</td><td>3-42</td></tr>
<tr><td>SENSe<sensor 1 or 2>:AVERage:COUNt:AUTOspace<ON | OFF></td><td>Select sensor 2 Auto-average mode</td><td>3-42</td></tr>
<tr><td>SENSe<sensor 1 or 2>:AVERage:TCONtrolspace<data acquisition averaging method, MOVing or REPeat></td><td>Set sensor 1 to acquire fresh measurement data before averaging</td><td>3-42</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg?</td><td>Query sensor configuration on specified input. Response is character data</td><td>3-20</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:BAP</td><td>Query sensor configuration on specified input. Response is character data</td><td rowspan="8">Starts 3-20</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:BAP:BEEXcludespace<samples></td><td>Enables BAP mode on the specified input</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:BAP:BEEXclude?</td><td>Reads the number of burst end exclude samples</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:BAP:BSEXclude?space<samples></td><td>Set number of burst start exclude samples</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:BAP:BSEXclude?</td><td>Reads the number of burst end exclude samples</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:BAP:BDTolerancespace<samples></td><td>Set the burst tolerance duration in milliseconds</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:BAP:BDTolerance?</td><td>Reads the current burst dropout tolerance duration</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:CW</td><td>Enables CW mode on the specified input</td><td>3-21</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:MAP[SYNChronized]</td><td>Enables synchronized MAP mode on the specified input</td><td rowspan="2">Starts 3-21</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:MAP[USYnchronized]</td><td>Enables unsynchronized MAP mode on the specified input</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:PAP</td><td>Enables PAP mode on the specified input</td><td rowspan="3">Starts 3-22</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:PAP:DCYClespace<Duty Cycle Value></td><td>Sets the duty cycle value that will be used when PAP is enabled</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CONFIg:PAP:DCYCle?</td><td>Query the duty cycle</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CORRection:EEPROM:TYPE?</td><td>Query sensor 1 eeprom type</td><td>3-60</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CORRection:EEPROM:FREQuency?</td><td>Query sensor 1 eeprom frequency table</td><td>3-60</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CORRection:EEPROM:CALFactor?</td><td>Query sensor 1 eeprom cal factor table</td><td>3-60</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CORRection:FREQuency[:CW-FIXed] 5e7</td><td>Set sensor 2 frequency to 50 MHz</td><td>3-35</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CORRection:OFFSet:COLLect</td><td>Take sensor 1 current reading as offset value</td><td>3-44</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CORRection:OFFSet [:MAGnitude]space<offset value in dB -99.99 to 99.99></td><td>Compensate sensor 2 for 10.2 dB attenuation</td><td>3-44</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CORRection:OFFSet:STATespace<ON | OFF></td><td>Enable sensor 2 offset correction</td><td>3-44</td></tr>
<tr><td>SENSe<sensor 1 or 2>:CORRection:VPROpf[:STATe]space<ON | OFF></td><td>Enable sensor 1 v_prop_f function</td><td>3-35</td></tr>
<tr><td>SENSe<sensor 1 or 2>:GATE:MODEspace<OFF, GATE, TRIGger or EDGE></td><td>Sets up time gating mode on the specified input</td><td>3-23</td></tr>
<tr><td>SENSe<sensor 1 or 2>:GATE:POLarityspace<INVert or NINVert></td><td>Sets the polarity of the trigger input signal</td><td>3-24</td></tr>
<tr><td>SENSe<sensor 1 or 2>:GATE:DELayspace<time></td><td>Sets the delay time from the trigger input edge to the start of the gating period</td><td>3-24</td></tr>
</table>

Table 3-3: SCPI Command Syntax (Continued)

Command Syntax	Function	Page
SENSe<sensor 1 or 2>:GATE:DURation*space*<time>	Sets the duration of the gating period. This command is relevant only in the External gating period	3-24
SENSe<sensor 1 or 2>:GATE:HOLDoff<time>	Sets the hold-off time from the end of the gating period to the time when the circuit will accept another trigger input edge. This command is relevant only in the External Trigger mode	3-24
SENSe<sensor 1 or 2>:TEMPerature?	Query sensor 1 temperature (in centigrade)	3-19
SENSe<sensor 1 or 2>:TRIGger:SOURce*space*<peak sensor triggering INTernal, EXTernal or CW>	Set sensor 1 peak trigger mode to CW mode	3-40
SENSe<sensor 1 or2>:TRIGger:DELay[:MAGnitude]:*space*<Peak Sensor sample delay in seconds from -20e-9 to 104e-3>	Set sensor 2 peak delay value to 1e6 seconds	3-40
SENSe<sensor 1 or 2>:TRIGger:DELay:STATe*space*<ON | OFF>	Enable peak sensor 1 delay function	3-41
SENSe<sensor 1or2>:TRIGger:LEVel[:MAGnitude]*space*<trigger level of peak power sensor>	Set peak sensor 1 trigger level to 1.7 volts	3-41
SENSe<sensor 1 or 2>:TRIGger:OFFSet[:MAGnitude]	Set peak sensor 1 trigger offset time to 0 second	3-41
SENSe<sensor 1 or 2>:TRIGger:TOTAl[:MAGnitude]?	Query peak sensor 1 total trigger delay time	3-41
*SRE*space*<event status register value, 128, 64, 32, 16, 8, 4, 2, 1>	Enable bit 5 of Status Byte register	3-47
STATus:OPERation[:EVENt]?	Query operation event register result	3-47
STATus:OPERation[:ENABle]	Set Status Enable Mask	3-47
STATus:PRESet	Clears all the status register value	3-47 3-57
SYSTem:COMMunicate:SERial:RECeive:BAUD*space*<baud rate>	Set the receive baud rate	3-45
SYSTem:COMMunicate:SERial:TRANsmit:BAUD*space*<baud rate>	Set the transmit baud rate	3-45
SYSTem:ERRor?	Query system error message	3-61
SYSTem:LANGuage NATIVE	Switch to native language	3-56
SYSTem:PRESet	Reset 8650A configuration	3-57
SYSTem:VERSion?	Query SCPI version	3-59
TRIGger[:IMMediate]	Trigger measurement cycle	3-29
TRIGger:SOURce*space*<instrument triggering source IMMediateBUS, HOLD, or EXTernal>	Set instrument measurement trigger source to IMMediate	3-29
TRIGger:MODE*space*<burst mode data gathering POST trigger receipt or PRE trigger receipt>	Set Burst or Swift mode data collection relative to Timing of event trigger post	3-28
TRIGger:DELay*space*<delay time between buffered readings in sec 0.000 to 5.000>	Set Burst mode measurement time to 5 ms between measurements (resolution, one msec)	3-29
TRIGger:COUNt*space*<number of data values to buffer in memory from 1 to 5000 for standard 8650s from1 to 128,000 on one channel with option 02; from 1 to 64000 for two channels with option 02>	Set Burst or Swift mode buffer reading number to 100	3-30

3.2.10 Sensor Calibration & Zeroing

- **CALibrate<sensor 1 or 2>**
- **CALibrate<sensor 1 or 2>STATe?**
- **CALibrate<sensor 1 or 2>ZERO**
- **SENSe<sensor 1 or 2>:TEMPerature?**

3.2.10.1 Sensor Calibration

The CALibration commands for sensor calibration and zeroing are important for accurate power measurement results. Be sure to perform the sensor calibration prior to beginning measurement operation or channel configuration. Sensors must be calibrated to the meter before performing measurements.

Zeroing of all active sensors should always be performed whenever a second sensor (whether calibrated or not) is added or removed. Zeroing should also be performed prior to measurement of low signal levels, generally within the lower 15 dB of a sensor's dynamic range. For standard sensors, this is -55 dBm.

CAL#

Syntax:	CALibration<sensor 1 or 2>[:SENSor]	
Example:	OUTPUT @PWR_MTR;CAL1	! CALIBRATE SENSOR 1
Description:	This command begins the sensor power sweep calibration process. Power sweep calibration can be performed only during Normal mode. The sensor must be attached to the front panel CALIBRATOR (use an adapter for high frequency sensors with Type K connectors). If the sensor is not connected or if the sensor is disconnected during the power sweep calibration procedure, the calibration automatically fails and the sensor power sweep calibration table is restored to its previous values.	

CALibrate:STATE?

Syntax:	CALibrate<sensor 1 or 2>:STATe?	
Example:	OUTPUT @PWR_MTR;CAL1:STAT?	! QUERIES SENSOR FOR CALIBRATION STATUS
Response:	1 if sensor is calibrated, or 0 if sensor is not calibrated.	
Description:	This command queries whether or not a sensor has been calibrated.	

3.2.10.2 Sensor Zero

Zeroing automatically accounts for ground noise and other noise in the measurement system. Measurements will be sensitive to noise-induced errors only in the lowest 15 dB of the sensor dynamic range. *Be sure to turn off the signal going into the sensor during zeroing, otherwise a failure will be indicated.*

CAL#:ZERO

Syntax:	CALibration <sensor 1 or 2>:ZERO	
Example:	OUTPUT @PWR_MTR;CAL1:ZERO	! ZERO SENSOR 1
Description:	This command begins the zeroing process. To offset for measurement circuit noise during zeroing, The sensor needs to remain attached to the measurement circuit during zeroing. Disable the signal source into the sensor. If disabling the signal source cannot occur, connect the sensor to a grounded connector (preferably RF grounded) or leave the sensor disconnected. Do not connect the sensor to the power meter front panel calibrator port. If the meter detects excessive RF power at the start of the zeroing process (generally above -50 dBm for standard sensors), zeroing will automatically fail.	

SENS#:TEMP?

Syntax:	SENSe <sensor 1 or 2>:TEMPerature?	
Example:	OUTPUT @PWR_MTR;SENS1:TEMP?	! QUERIES SENSOR 1 TEMPERATURE (IN CENTIGRADE)
Response:	Current sensor temperature in degrees centigrade	
Description:	This command reports sensor temperature. If the temperature varies in the operating environment, monitor the relative temperature variations. If the variation since the previous power sweep calibration exceeds ±5 °C (±9 °F), perform the power sweep calibration for that sensor.	

Example programs for Sensor Calibration and Zeroing are contained in Appendix A.

3.2.11 Sensor Configuration Commands

- **SENSe:CONFig?**
- **SENSe:CONFig:BAP**
- **SENSe:CONFig:MAP**
- **SENSe:CONFig:CW**
- **SENSe:CONFig:PAP**

The following commands are for sensor configuration.

SENSe:CONFig?

Syntax:	SENSe<sensor 1 or 2>:CONFig?	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG?	! QUERIES SENSOR 1
Description:	Queries sensor configuration on specified input. Response is character response data: CW | MAP | MAPU | BAP | PAP | UNCAL | NOSENSOR.	

SENSe:CONFig:BAP

Syntax:	SENSe<sensor 1 or 2>:CONFig:BAP	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:BAP	! ENABLES SENSOR 1 FOR BAP MODE
Description:	Enables BAP mode on the specified input. This command does not affect the burst star or end exclude or dropout tolerance parameters. The default values, set by *RST, correspond to the “automatic” settings in manual mode.	

Syntax:	SENSe<sensor 1 or 2>:CONFig:BAP:BEEXclude*space*<samples> (Where <samples> is an integer in the range of 0 to 123)	
Example:	OUTPUT @PWR_MTR;CONFIG:BAP:BEEX 2	! SETS NUMBER OF BURST END EXCLUDE SAMPLES
Response:	The timing for each sample is approximately 27 µs.	
Description:	Sets the burst end exclude to 2, i.e., 54 µs.	

Syntax:	SENSe<sensor 1 or 2>:CONFig:BAP:BEEXclude?	
Example:	OUTPUT @PWR_MTR;CONFIG:BAP:BEEX?	! READS THE NUMBER OF BURST END EXCLUDE SAMPLES
Description:	Query form for burst end exclude.	

Syntax:	SENSe<sensor 1 or 2>:CONFig:BAP:BSEXclude*space*<samples> (Where <samples> is an integer in the range of 0 to 1565)	
Example:	OUTPUT @PWR_MTR;CONFIG:BAP:BSEX 1	! SETS NUMBER OF BURST START EXCLUDE SAMPLES
Response:	The timing for each sample is approximately 27 µs.	
Description:	Sets the burst start exclude to 1, i.e., 27 µs.	

Syntax:	SENSe<sensor 1 or 2>:CONFig:BAP:BSEXclude?	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:BAP:BSEX?	! READS THE NUMBER OF BURST START EXCLUDE SAMPLES
Description:	Query form for burst start exclude.	

Syntax:	SENSe<sensor 1 or 2>:CONFig:BAP:BDTolerance*space*<dropout> (Where <dropout> is the dropout time in milliseconds in the range of 0 to 74 and with a resolution of 0.1 ms. The system selects the nearest value from a series of discrete values.)	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:BAP:BDT:0.054	! SETS THE BURST DROPOUT TOLERANCE DURATION IN MILLISECONDS
Description:	Sets the burst dropout tolerance to 54 µs.	

Syntax:	SENSe<sensor 1 or 2>:CONFig:BAP:BDTolerance?	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:BAP:BDT?	! READS THE CURRENT BURST DROPOUT TOLERANCE DURATION
Description:	Query form for burst dropout tolerance.	

SENSe:CONFig:CW

Syntax:	SENSe<sensor 1 or 2>:CONFig:CW	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:CW	! ENABLES CW MODE ON THE SPECIFIED INPUT
Description:	Configures CW or Modulation sensor for CW mode.	

SENSe:CONFig:MAP

Syntax:	SENSe<sensor 1 or 2>:CONFig:MAP[SYNChronized]	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:MAP	! ENABLES SYNCHRONIZED MAP MODE ON THE SPECIFIED INPUT
Description:	Configures modulation sensor for MAP (Modulated Average Power) Synchronized mode.	

Syntax:	SENSe<sensor 1 or 2>:CONFig:MAP[UNSYnchronized]	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:UNSYMAP	! ENABLES UNSYNCHRONIZED MAP MODE ON THE SPECIFIED INPUT
Description:	Configures modulation sensor for MAP (Modulated Average Power) Unsynchronized mode or when performing measurements using high average values where the signal exhibits very large power swings. If an irregularly modulated signal is measured in MAP mode, measurement settling time will vary as the power meter attempts to synchronize to the modulation, disabling synchronization achieves faster measurement.	

SENSe:CONFig:PAP

Syntax:	SENSe <sensor 1 or 2> :CONFig:PAP	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:PAP	! ENABLES PAP MODE ON THE SPECIFIED INPUT
Description:	Configures modulation sensor for PAP (Pulse Average Power) Synchronized mode. This mode requires duty cycle configuration.	

Syntax:	SENSe <sensor 1 or 2> :CONFig:PAP:DCYCle *space* <Duty Cycle Value> (Where <Duty Cycle Value> is in the range 0.001 to 99.999% with a resolution of 0.001.)	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:PAP:DCYCLE 0.001	! SETS THE DUTY CYCLE VALUE THAT WILL BE USED WHEN PAP IS ENABLED
Description:	Configures duty cycle setting for PAP mode.	

Syntax:	SENSe <sensor 1 or 2> :CONFig:PAP:DCYCle?	
Example:	OUTPUT @PWR_MTR;SENS1:CONFIG:PAP:DCYCLE?	! QUERIES THE DUTY CYCLE
Description:	Query form for PAP duty cycle command.	

3.2.12 Sensor Gate Commands

- **SENSe:GATE:MODE**
- **SENSe:GATE:MODE?**
- **SENSe:GATE:POLarity**
- **SENSe:GATE:POLarity?**
- **SENSe:GATE:DELay**
- **SENSe:GATE:DELay?**
- **SENSe:GATE:DURation**
- **SENSe:GATE:DURation?**
- **SENSe:GATE:HOLDoff**
- **SENSe:GATE:HOLDoff?**

The following commands are for sensor gate settings.

SENSe:GATE:MODE

Syntax:	SENSe<sensor 1 or 2>:GATE:MODE*space*<OFF, GATE, TRIGger or EDGE>	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:MODE GATE	! SETS SENSOR 1 TO GATE TIME GATING MODE
Description:	Configures sensor for Time Gating operation. Selection is GATE, TRIGGER and EDGE. In GATE mode, the meters measures the average power of a signal while the system is triggered. The duration of the measurement is determined by the duration of the trigger signal. In TRIGGER mode, the meter measures the average of a signal for the duration specified in by the Gate Duration command after the gate delay time has been satisfied. A TTL signal supplied at the trigger input initiates the trigger cycle. IN EDGE mode, the meter internally detects the rising edge of a pulsed signal to trigger a gated measurement. TRIGGER and EDGE mode require the delay and duration to be set.	

SENSe:GATE:MODE?

Syntax:	SENSe<sensor 1 or 2>:GATE:MODE?	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:MODE GATE?	! REQUESTS THE CURRENT SENSOR GATE MODE FROM THE METER
Description:	Query form for sensor gate mode. Response can be OFF, GATE, TRIG or EDGE.	

SENSe:GATE:POLarity

Syntax:	SENSe<sensor 1 or 2>:GATE:POLarity*space*<INVert or NINVert)	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:POL NINV	! SETS NON-INVERTED POLARITY FOR THE TRIGGER INPUT
Description:	Configures the Time Gating trigger input for positive or negative polarity. INV configures the meter to trigger from the falling edge of the trigger input.	

SENSe:GATE:POLarity?

Syntax:	SENSe<sensor 1 or 2>:GATE:POLarity?	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:SENS1:GATE:POL?	! QUERIES THE METER FOR POLARITY OF THE TRIGGER INPUT SIGNAL
Description:	Query form for sensor gate polarity. Response can be INV or NINV.	

SENSe:GATE:DELay

Syntax:	SENSe<sensor 1 or 2>:GATE:DELay*space*<time> (Where <time> is in the range 0 to 327.675 ms with a resolution of 5 ms, where 0 represents some minimum non-zero delay time.)	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:DELAY 20E-3	! SETS THE GATE DELAY TO 20 MS
Description:	Sets the delay time from the trigger input edge to the start of the gating period.	

SENSe:GATE:DELay?

Syntax:	SENSe<sensor 1 or 2>:GATE:DELay?	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:DEL?	! REQUESTS SENSOR 1 GATE DELAY SETTING
Description:	Query form for sensor gate delay command.	

SENSe:GATE:DURation

Syntax:	SENSe<sensor 1 or 2>:GATE:DURation*space*<time> (Where <time> is in the range 0 to 327.675 µs with a resolution of 5 µs)	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:DUR 250E-3	! SETS THE DURATION TO 250 USEC
Description:	Sets the time gate duration for TRIGGER and EDGE mode.	

SENSe:GATE:DURation?

Syntax:	SENSe<sensor 1 or 2>:GATE:DURation?	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:DUR?	! REQUESTS SENSOR 1 TIME GATE DURING SETTING
Description:	Query form for sensor gate duration command.	

SENSe:GATE:HOLDoff

Syntax:	SENSe<sensor 1 or 2>:GATE:HOLDoff*space*<time> (Where <time> is in the range 0 to 327.675 µs with a resolution of 5 µs)	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:HOLD 10E-6	! SETS THE GATE HOLD-OFF TO 1 USEC
Description:	Specifies the hold-off time from the end of the gating period to the time when the circuit will accept another trigger.	

SENSe:GATE:HOLDoff?

Syntax:	SENSe<sensor 1 or 2>:GATE:HOLDoff?	
Example:	OUTPUT @PWR_MTR;SENS1:GATE:HOLD?	! REQUESTS SENSOR 1 TIME GATE HOLD-OFF SETTING
Description:	Query form for sensor gate hold-off command. Response is trigger hold-off time in seconds.	

3.2.13 Reading Power Measurements

- **FETCh?**
- **MEASure?**
- **READ?**
- ***OPC**
- ***OPC?**

These commands return measurement data from the 8650A. During Normal mode the data will be single measurement values. During Swift or Burst modes, the data will be an array of values. Generally, it is a single array if one sensor is connected and calibrated, and a dual array if two sensors are connected and calibrated.

FETC?

Syntax:	FETCh<channel 1 to 4>?	
Example:	OUTPUT @PWR_MTR;FETC1?	! READS PREVIOUSLY TRIGGERED CHANNEL 1 POWER ! NORMAL, SWIFT, BURST OR FMOD MODE DATA
Response:	Whatever is in the output buffer whether it is a new measurement or an old one.	
Description:	The FETCh? query command returns post-processed measurement data from the active channels of the 8650A. After receiving FETCh? the 8650A will output the contents of its active data output buffer. The data size and output format of the buffer are dependent upon channel and measurement mode configuration. For example, if channel 1 is defined as sensor 1 and channel 2 is defined as sensor 2/1, the FETCh? query command will return the power level incident upon sensor 1 and the ratio of the power levels incident upon sensor 2 and 1. FETCh? is used with NORMal mode measurements when INITiate:IMMediate is active, and for reading SWIFt, BURSt and FMOD Mode measurement data from the meter's data buffer.	

MEAS?

Syntax:	MEASure<channel 1 to 4>[:SCALar:POWer]?	
Example:	OUTPUT @PWR_MTR;MEAS2?	! ARMS, TRIGGERS AND MEASURES CHANNEL 2 ! POWER IN NORMAL MODE
Response:	A power measurement.	
Description:	The MEAS? command returns measured data from the active software calculation channels of the 8650A. The MEAS? command will also initiate the trigger cycle and will turn on the Auto Averaging mode; that is, measurement will be triggered and the data will be transmitted from the power meter. This is in contrast to the FETCh? command, which is capable of causing immediate output of the measurement data but not initiating the triggering cycle.	

READ?

Syntax:	READ<channel 1 to 4> <channel 1 to 4> <channel 1 to 4>[:POWer]?	
Example:	OUTPUT @PWR_MTR;READ2?	! TRIGGER MEASUREMENT AND READ CHANNEL 2
Response:	A power measurement.	
Description:	The READ[:POWer]? query command triggers measurement and sends the data to the controller. The READ? command enables the power meter to acquire data at the next instrument trigger and return post-processed measurement data from the active channels of the 8650A. READ uses the current sensor averaging setup.	

*OPC

Example:	OUTPUT @PWR_MTR;*OPC	! *OPC ALLOWS ONE-TIME SRQ ENABLE
Description:	The *OPC command determines when an operation is completed. This command is generally used to monitor the completion of long measurement sequences. It sets the operation complete bit in the event status register upon completion of operation.	

*OPC?

Example:	OUTPUT@PWR_MTR;*OPC?	! RETURNS A VALUE OF 1 AFTER AN OPERATION IS COMPLETED
Description:	The *OPC? operation complete query allows synchronization between the controller and the test set using either the message available (MAV) bit in the status byte, or a read of the output OPC?.	

3.2.13.1 Special Errors

An unusual or non-sense numeric response, such as 9e+40, indicates an error response. For instance, not performing the power sweep calibration procedure to calibrate the sensor to the power meter, the response to a MEAS#? command will be 9.0000e+40.

Selected basic SCPI syntax and execution errors apply to these commands.

If using the READ#? measurement query and INIT:CONT is ON, a bad value is returned, 9e+40. Sending SYST:ERR? error query, -213, Init ignored will be returned. READ#? contains the low level function INIT, since INIT:CONT is ON the INIT within READ#? generates an error. Set INIT:CONT to OFF when using READ#?.

There are no device-specific errors for the preset configuration or status reset commands except the high level -300, Device-specific error response.

Example programs for Reading Power Measurements are contained in Appendix A.

3.2.14 Instrument Triggering

- ***TRG**
- **TRIGger[:IMMediate]**
- **TRIGger:SOURce**
- **TRIGger:MODE**
- **TRIGger:DELay**
- **TRIGger:COUNt**
- **INITiate:CONTinuous**
- **INITiate[:IMMediate]**
- ***WAI**

These SCPI commands trigger the measurement cycle. They do not configure or provide triggering for Peak Power Sensors. Those commands are defined in Section 3.2.19. For power meter operation, the TRIGger Subsystem is divided into two sections; Instrument Measurement Event Triggering, and Special Triggering Configuration commands for the fast reading buffered data, Burst, and Swift modes.

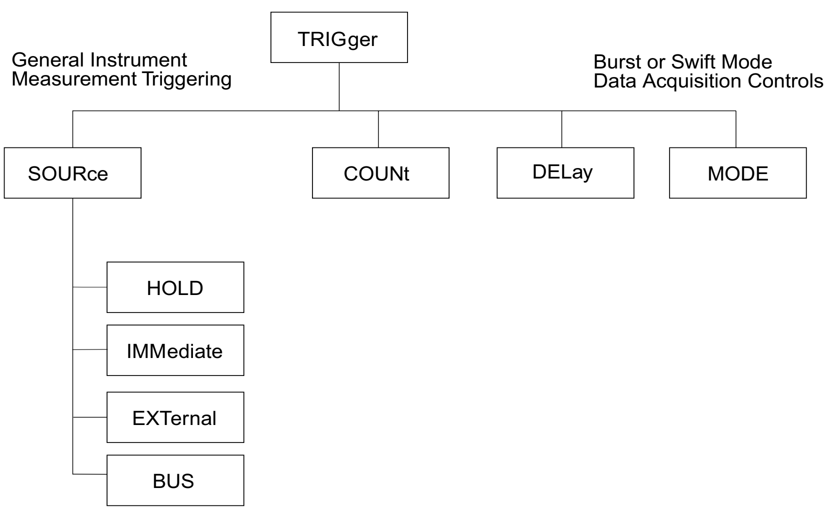

Figure 3-8: TRIGger Subsystem Command Tree

EXT triggering is performed with the rear panel BNC connector and functions only in the BURSt and SWIFt Modes. EXT triggering is not available in the NORMal Mode. Provision has been made for a hardware ready for new trigger type handshaking capability using the analog output.

BUS triggering is available for all operating modes, BURSt, SWIFt and NORMal.

IMMediate triggering allows the 8650A to free run and perform continuous measurements. This is the default setting. To INITiate on IMMediate to increase measurement speed. During Normal Mode, with both of these controls set to IMM, power measurements can be read with MEAS, READ, or FETCh. IMMediate triggering is not compatible with the BURSt Mode. To send the CALC#:MODE BURS command to enter the BURSt Mode, IMM instrument triggering for NORMal Mode will automatically be switched to TRIG:SOUR BUS. To send the CALC#: MODE SWIF command to enter the SWIFt Mode, IMM instrument triggering will remain IMM but a device-specific error will be generated whenever specifying a TRIG:COUN# higher than 1.

*TRG

Syntax:	*TRG	
Example:	OUTPUT @PWR_MTR;*TRG	! BUS TRIGGER EVENT
Description:	*TRG is an IEEE 488.2 compatible programming command to initiate BUS triggered measurements. The power meter interprets *TRG as a BUS source for instrument triggering events. This command will not trigger Peak Power Sensors. It is the same as the TRIG command.	

TRIG

Syntax:	TRIGger[:IMMediate]	
Example:	OUTPUT @PWR_MTR;TRIG	! TRIGGER MEASUREMENT CYCLE
Description:	TRIG is the SCPI command form of *TRG. The power meter interprets TRIG as a BUS source for instrument triggering events. If INIT:CONT is OFF, measurements using FETCh#? will not proceed until triggering is armed and instrument triggering is actuated.	

TRIG:SOUR

Syntax:	TRIGger:SOURce*space* <instrument triggering sourc IMMediate, BUS, HOLD, or EXTernal>	
Example:	OUTPUT @PWR_MTR;TRIG:SOUR IMM	! SETS INSTRUMENT MEASUREMENT TRIGGER ! SOURCE TO IMMEDIATE
Description:	TRIG:SOUR IMM sets trigger control to the 8650A. BUS triggering sets the triggering control to the controller software using TRIG or *TRG commands. HOLD halts triggering sequences. EXTernal sets triggering control to the front panel BNC connector. The following data shows triggering and operating mode compatibility.	

Operating Mode	IMM	BUS	HOLD	EXT
NORMal	X	X	X,	
SWIFt	X	X		X
BURSt		X		X

If the power meter has been configured for EXT triggering in the BURSt or SWIFt modes and the operating mode is switched to NORMal, the triggering configuration remains EXT. However, if the configuration is IMM in NORMal mode and the meter is switched to BURSt, it automatically switches to BUS. As a general rule, when a measurement subroutine switches operating modes, send the triggering configuration.

TRIG:MODE

Syntax:	TRIGger:MODE*space* <burst mode data gathering POST trigger receipt or PRE trigger receipt>	
Example:	OUTPUT @PWR_MTR;TRIG:MODE POST	! SETS BURST OR SWIFT MODE DATA COLLECTION ! RELATIVE TO TIMING OF EVENT TRIGGER POST
Description:	This command is only for BURSt Mode measurements. Send this command only immediately after CALC#:MODE BURS. TRIG:MODE controls the operation of the FIFO data buffer inside the power meter. When set to POST, the burst of data points is taken after the receipt of a valid instrument trigger. When set to PRE, the burst of data is assumed to be those data points that have arrived immediately preceding the valid instrument trigger event. Using PRE requires some caution: The single processor in the 8650A is constantly performing data acquisition. To send any configuration commands while it is collecting the PRE trigger data, the timing of the data point collection can be interrupted.	

TRIG:DEL

Syntax:	TRIGger:DELay*space*<delay time between buffered readings in sec, 0.000 to 5.000>	
Example:	OUTPUT @PWR_MTR;TRIG:DEL 5E-3	! SETS BURST MODE MEASUREMENT TIME TO ! 5 MS BETWEEN MEASUREMENTS.
Example 2:	OUTPUT @PWR_MTR;TRIG:DEL 0	! MAX. MEASUREMENT RATE (RESOLUTION, ! 1 MSEC)
Description:	This command applies only to the BURSt Mode. It sets the time interval between the measurements and is accurate to about 5%. This is the only way to control the data acquisition of individual data points in a burst measurement. Burst measurement data is taken following or just prior to a single instrument trigger event. The number of measurements taken is controlled by TRIG:COUNt. Timing can be set in 0.001 second increments from 0.000 to 5.000 seconds, inclusive. Setting this control to 0.000 activates the highest reading rate (5100 rdgs/sec) possible with the 8650A. Triggering of the Peak Power Sensor occurs independently from instrument triggering. Therefore, there is a moderate ability to control triggering at the Peak Power Sensors. Generally, if controlling is required of individual data point triggering, the SWIFt mode will provide the highest measurement rates, with or without SRQ or hardware handshaking.	

TRIG:COUN

Syntax:	TRIGger:COUNt*space*<number of data values to buffer in memory from 1 to 5000 for the standard 8650; from 1 to 128,000 on one channel with option 02; and 1 to 64000 for two channels with option 02>	
Example:	OUTPUT @PWR_MTR;TRIG:COUN 100	! SETS BURST OR SWIFT MODE BUFFER READING ! NUMBER TO 100
Description:	The power meter has an internal and expandable data buffer for storing measurement data until the slot 0 controller/resource manager is ready to read the data. During the SWIFt or BURSt modes, this command controls the number of power readings that are stored in the 8650A data buffer. The value specified here applies to each individual channel. That is, if two sensors are attached and calibrated, the 8650A will perform the specified measurement and buffering for both channels. In the example shown below for this command, that would be 100 points per channel. Option 02 adds additional memory for 128,000 data point buffering. When two sensors are attached and calibrated, the maximum is 64,000 readings per channel. During BURSt mode, the COUNt# of readings will be taken with just one instrument trigger. During the SWIFt Mode however, to control the triggering of individual data points. set triggering to EXT, the power meter will perform one measurement each time it detects a TTL level signal. The value is then stored in the data buffer. This continues until the meter has received and processed a COUNt# of triggers and also read in a COUNt# of data points to the buffer.	

*WAI

Example:	OUTPUT @PWR_MTR;*WAI	! WAIT, FOLLOWING COMMAND COMPLETION
Description:	The *WAI command causes the power meter to wait until all previous commands and queries are completed before continuing operation. It functions only when using the Normal mode of operation.	

Example programs for Instrument Triggering are contained in Appendix A.

3.2.15 Arming the Triggering Cycle

■ INITiate:CONTinuous
■ INITiate[:IMMediate]

The Initiate commands enable the power meter to acquire measurement data at the next instrument trigger. In the absence of an instrument signal, the 8650A is placed in the waiting-for-trigger-state. The default configuration is continuous initiation, INIT:CONT ON.

INITiate[:IMMediate] causes the power meter to exit the idle state and causes the trigger system to initiate and complete one full trigger cycle, returning to the idle state upon completion. For example, INITiate[:IMMediate] can be used with Peak Power Sensors to measure the power level of transient or one-shot pulsed microwave signals. After execution of the triggering sequence, send the FETCh? query command to return the measurement data from the 8650A.

Perform triggering configuration with the TRIGger Subsystem commands. When TRIGger:SOURce is IMMediate, the measurement will start as soon as INITiate is sent to the 8650A and executed (or INITiate:CONTinuous ON sent and executed).

INIT:CONT

Syntax:	INITiate:CONTinuous*space* < ON | OFF >	
Example:	OUTPUT @PWR_MTR;INIT:CONT ON	! CONTINUOUS TRIGGERING
Description:	This allows user control of measurement cycle arming. When INIT:CONT is set to ON, measurement cycle arming occurs automatically. The default setting is OFF. To take control of arming the triggering of the measurement cycle, set this control to OFF and use MEAS#? or INIT with READ#? to perform the measurement. Another legal measurement control option would be INIT with TRIG: IMM and FETC#? When INITiate:CONTinuous is ON, the instrument triggering cycle resets continuously. That is, upon instrument triggering, data is constantly being acquired and updated. Upon completion of a trigger cycle, the meter will immediately enter the wait-for-trigger state rather than the idle state, as is the case with INITiate:CONT OFF.	

INIT:IMM

Syntax:	INITiate[:IMMediate]	
Example:	OUTPUT @PWR_MTR;INIT	! INITIATE INSTRUMENT TRIGGERING CYCLE
Description:	This command enables the triggering cycle. Instrument triggering will not occur unless INIT has been enabled, even when sending BUS or EXT triggers to the 8650A. Either this command or INIT:CONT ON performs measurements with FETCh? and READ?. MEAS? is a high level function which contains the INITiation enabling function. Arming occurs automatically when using the MEAS#? command to automatically INITiate, TRIGger and FETCh the measurement data. However, if using READ#? or FETCh#? to read data, this command arms the 8650A for triggering upon the occurrence of the next valid trigger. The INITiate[:IMMediate] command places the power meter in the wait-for-trigger-state. When used with EXT (TTL levels) instrument triggering, this command is useful for measuring signals that are asynchronous with programming (BUS triggered) sequences. When INITiate:CONTinuous is set to OFF, INITiate:[IMMediate] enables the instrument triggering cycle. Sending INITiate:[IMMediate] when INITiate:CONTinuous is set to ON will cause a -123 command error. The INITiate [:IMMediate] command is an event; no query form exists for this command.	

3.2.16 Channel Configuration

- **CALCulate:POWer**
- **CALCulate:RATio**
- **CALCulate:DIFFerence**
- **CALCulate?**
- **CALCulate:UNIT**
- **CALCulate:STATe**
- **CALCulate:STATe?**
- **CALCulate:CFACtor:STATe**
- **CALCulate:CFACtor:DATa?**
- **CALCulate:PHOLd:STATe**
- **CALCulate:PHOLd:STATe?**
- **CALCulate:PHOLd:DATa?**
- **CALCulate:MODE**
- **CALCulate:SFUNctions:STATE**
- **CALCulate:SFUNctions:DATa?**

CALC#:POW

Syntax:	CALCulate <channel 1 to 4>[:CHANnel]:POWer*space*<sensor 1 or 2>	
Example:	OUTPUT @PWR_MTR;CALC2:POW 1	! CONFIGURES CHANNEL 2 TO ! MEASURES SENSOR 1 POWER
Description:	This command configures a sensor to an individual channel, and the channel measures the sensor power level. This command does not have a query format. For the example above, the response to a CALC2? query is POW 1. This command should be executed only after the designated sensor has been calibrated.	

CALC#:RAT

Syntax:	CALCulate <channel 1 to 4>[:CHANnel]:RATio*space*<sensor 1 or 2*comma*sensor 2 or 1>	
Example:	OUTPUT @PWR_MTR;CALC1:RAT 2,1	! CONFIGURES CHANNEL 1 TO MEASURE ! SENSOR 2 OVER SENSOR 1 POWER
Description:	This command configures a channel as a ratio of the power levels present at the two sensors. When UNITs are dBm, RATio configuration produces measurements in dB. When UNITS are Watts, RATio configuration measurements are percentage. This command does not have a query format. For the above example, the response to a CALC1? query is RAT 2,1. This command should be executed only after the designated sensor has been calibrated.	

CALC#:DIFF

Syntax:	CALCulate <channel 1 to 4>[:CHANnel]:DIFFerence*space*<sensor 1 or 2*comma*sensor 2 or 1>	
Example:	OUTPUT @PWR_MTR;INIT:CONT ON	! CONFIGURES CHANNEL 1 TO MEASURE ! SENSOR 2 MINUS SENSOR 1 POWER
Description:	This command configures a channel as the difference between the power levels present at the two sensors. UNITs are dBm or Watts. This command does not have a query format. For the above example, the response to a CALC1? query is DIFF 2,1. This command should be executed only after the designated sensor has been calibrated.	

CALC#?

Syntax:	CALCulate<channel 1 to 4>[:FUNCtion]?	
Example:	OUTPUT @PWR_MTR;CALC2?	! QUERIES CHANNEL 2 SENSOR CONFIGURATION
Response:	POW #, RAT #,#, OR DIFF #,#	
Description:	This command returns the current channel configuration. This command is a query format only.	

CALC#:UNIT

Syntax:	CALCulate<channel 1 to 4>:UNIT[:POWer]*space*<data units selection, dBm or Watt>	
Example:	OUTPUT @PWR_MTR;CALC1:UNIT W	! SELECTS CHANNEL 1 LINEAR UNITS, WATTS
Example 2:	OUTPUT @PWR_MTR;CALC1:UNIT DBM	! SELECTS CHANNEL 1 LOG UNITS, DBM
Description:	This command configures a channel to report power measurements in either linear Watts units or logarithmic dBm units. When UNIT is dBm, the RATio configuration produces measurements in dB, and POWer and DIFF measurements in dBm. When UNIT is Watts, the RATio configuration measurements are percentages, and POWer and DIFF measurements are in Watts.	

CALC#:STAT

Syntax:	CALCulate<channel 1 to 4>:STATe*space*<ON | OFF>	
Example:	OUTPUT @PWR_MTR;CALC1:STAT OFF	! DISABLES CHANNEL 1 MEASUREMENT
Description:	This command turns a channel off or on. Default is ON. During Normal and Swift modes, measurement speed increases when only one sensor (Channel Configuration is CALC#:POW) is performing measurements.	

CALC#:STAT?

Syntax:	CALCulate<channel 1 to 4>:STATe?	
Example:	OUTPUT @PWR_MTR;CALC1:STAT?	! QUERIES CHANNEL 1 MEASUREMENT STATUS
Response:	0 = off, 1 = on	
Description:	This command is the query format of the command above.	

CALC:CFACtor:STATe

Syntax:	CALCulate<channel 1 to 4>CFACtor:STATe*space*<ON | OFF>	
Example:	OUTPUT @PWR_MTR;CALC1:STAT ON	! ENABLE/DISABLE CREST FACTOR MEASUREMENTS
Description:	Enables crest factor measurements. Sending ’ON’ when crest factor is already enabled causes the value to be reset. The crest factor function holds on to the highest instantaneous power measured from the time the function is enabled until it is reset.	

CALC:CFACtor:DATa?

Syntax:	CALCulate<channel 1 to 4>CFACtor:DATa?	
Example:	OUTPUT @PWR_MTR;CALC3:CFAC:DAT?	! READS THE CREST FACTOR MEASUREMENT
Description:	Reads the crest factor measurement from display channel line 3.	

CALC:PHOLd:STATe

Syntax:	CALCulate<channel 1 to 4>PHOLd:STATe*space*<ON | OFF>	
Example:	OUTPUT @PWR_MTR;CALC1:PHOL:STAT ON	! ENABLES/DISABLE PEAK HOLD FUNCTION
Description:	The Peak Hold feature causes the measured value to hold at the highest instantaneous power measured from the time the feature is enabled until it is reset (the measured value changes only when it is rising to a new maximum, or when it is reset).	

CALC:PHOLd:STAT?

Syntax:	CALCulate<channel 1 to 4>PHOLd:STAT?	
Example:	OUTPUT @PWR_MTR;CALC1:PHOL:STAT?	! ENABLES/DISABLE PEAK HOLD FUNCTION
Description:	This command is the query format of the command above.	

CALC:PHOLd:DATa?

Syntax:	CALCulate<channel 1 to 4>PHOLd:DATa?	
Example:	OUTPUT @PWR_MTR;CALC1:PHOL:DAT?	! READS THE PEAK HOLD MEASUREMENT
Description:	The Peak Hold function monitors the maximum power as it is measured; but does not provide any feedback to the controller until a Peak Hold command or CALC#:PHOL:DAT? is received.	

CALC:SFUNctions:STATe

Syntax:	CALCulate<channel 1 to 4>:SFUNctions:STATe*space*<ON | OFF>	
Example:	OUTPUT @PWR_MTR;CALC3:SFUN:STAT ON	! ENABLES/DISABLE STATISTICAL FUNCTIONS
Description:	This command enables or disables the statistic computation function of mean and standard deviation over a period of time.	

CALC:SFUNctions:DATa?

Syntax:	CALCulate<channel 1 to 4>:SFUNctions:DATa?	
Example:	OUTPUT @PWR_MTR;CALC3:SFUN:DAT?	! READS STATISTICAL MEASUREMENT
Description:	The command queries statistics data of the current active GPIB measurement. Error measurement is returned if statistics is not active.	

Example programs for Channel Configuration are contained in Appendix A.

3.2.17 Cal Factor Correction

- **SENS#:CORR:FREQ**
- **MEMory:FREQuency**
- **MEMory:SELect**
- **MEMory:SLOPe**

Power Sensors have a measurable frequency response. During manufacture, this response is calibrated at 1 GHz intervals. Instead of printing the data on the sensor label, each Giga-tronics power sensor includes a built-in EEPROM which has been programmed with the frequency calibration factor data for that particular power sensor.

The following SCPI commands tell the 8650A the operating frequency of the measured microwave or RF signal. Thus, the meter automatically interpolates the correct Cal Factor and applies that value to the measurement data. By performing this automatically, there is no need to read Cal Factor data from the side of sensor housings and program the data into tables within the ATE programming. Except for advanced measurement techniques using Burst Mode, there is no need to write cal factor interpolation routines; this is all done by the power meter. As shown below, sensor Cal Factor data can be read into the computer. Also see the Sensor EEPROM Commands in Section 3.2.34 for Cal Factor programming.

SENS#:CORR:FREQ

Syntax:	SENSe < sensor 1 or 2 > :CORRection:FREQuency[:CW-FIXed] 5e7	
Example:	OUTPUT @PWR_MTR;SENS2:CORR:FREQ 5E7	! SETS SENSOR 2 FREQUENCY TO 50 MHZ
Response:	0 = off, 1 = on	
Description:	This command allows correcting for sensor frequency response characteristics. The power meter uses this frequency to automatically interpolate and apply the Cal Factor for that sensor (computed from Cal Factor data points stored within each sensor's EEPROM). Unless the EEPROM has been modified subsequent to shipment, the Cal Factor is always 0.0 dB at 50 MHz, 5e7. Be sure to correct for sensor cal factor by using this command, using automated V_{PROP}F correction, sending cal factor REFerence value, or correcting for Cal Factor in the ATE system by downloading Cal Factor points for the 8650A.	

SENS#:CORR:FREQ

Syntax:	SENSe < sensor 1 or 2 > :CORRection:VPROpf[:STATe]*space* < ON-OFF >	
Example:	OUTPUT @PWR_MTR;SENS1:CORR:VPRO ON	! ENABLES SENSOR 1 V_{PROP}F FUNCTION
Description:	Setting this control to ON activates V_{PROP}F and deactivates SENS#:CORR:FREQ.	

☛ ***NOTE:*** *Example programs for Cal Factor Corrections are contained in Appendix A of this publication.*

MEM:FREQ

Syntax:	MEMory[:TABLe]:FREQuency*space* < Start frequency >	
Example:	OUTPUT @PWR_MTR;MEM:FREQ 1E9	! SETS V_{PROP}F FREQUENCY TO 1 GHZ
Description:	This command sets the automated V_{PROP}F correction start frequency. The frequency at which the voltage input into the 8650A is 0.0 V.	

MEM:SEL

Syntax:	MEMory[:TABLe]:SELect*space* < ANALOGout,VPROpf1,VPROpf2 >	
Example:	OUTPUT @PWR_MTR;MEM:SEL VPROPF1	! SELECTS MEMORY TABLE V_{PROP}F1 ! FOR EDITING IN FOLLOWING LINES
Description:	This command selects between one of three editable tables. Using the meter's automated V_{PROP}F capability to correct for Cal Factor requires configuring the V_{PROP}F IN connector to match the signal that source will output. This is performed by setting the frequency so that source's V_{PROP}F output is 0.0 V. Then the slope relationship is entered in the form of a value representing the V/Hz relationship. If source is a Giga-tronics GT 9000 Series Signal Generator, this relationship is 0.5 V/GHz or 5e-8 V/Hz.	

MEM:SLOP

Syntax:	MEMory[:TABLe]:SLOPe*space* < Volts per Hz >	
Example:	OUTPUT @PWR_MTR;MEM:SLOP 1E-9	! SETS V_{PROP}F SLOPE TO 1 V/GHZ
Description:	This command sets the V/Hz relationship for automated Cal Factor correction.	

3.2.17.1 Cal Factor Error Control

- See Error Messages in Section 3.2.36
- Selected basic SCPI syntax and execution errors apply to these commands
- Device-specific errors include the following and other -300 level errors
- A common device-specific error occurs when the frequency sent to the 8650A in the **SENS#:CORR:FREQ** ### command is outside the sensor operating frequency range. For example, sending SENS1:CORR:FREQ 18.4e9 when an 18 GHz (max) 80301A CW Power Sensor is attached will yield a device-specific error

3.2.18 High Speed Measurements

- **CALCulate:DATA?**
- **CALCulate:MODE**
- **TRIGger:MODE**
- **TRIGger:DELay**
- **TRIGger:COUNt**

Measurements in Normal mode are fastest with only one sensor attached. When two sensors are attached, the Normal mode measurement rate is reduced. This applies for all three major measurement commands, FETCh, READ and MEASure. Both of the averaging types, MOVing and REPeat, slow down when in the Normal mode. MOVing provides faster averaging; the speed is equivalent to REPeat whenever the Averaging number is 1.

When performing measurements in either Swift or Burst modes, measurement rates are the same with two sensors as with one sensor. Approximate measuring speeds are listed in Table 1-1.

CALC:DATA?

Syntax:	CALCulate<channel 1 to 4>:DATA?	
Example:	OUTPUT @PWR_MTR;CALC1:DATA?	! QUERIES CHANNEL 1 & 2 BURST MODE DATA
Description:	This command interrupts the burst measurement and dumps the measurement buffer. All empty locations will be filled with -300 before the data dump. The Burst mode remains active after this command so that a trigger will start a new measurement reference. The command returns measurement data (while the measurement is completing) during BURSt mode only. This command is not used in NORMal or SWIFt Modes. Operation is similar to using FETCh? Both are instrument (not channel) commands. That is, when both sensors are connected, calibrated, and the channel set to ON, this command will return power level data for both channels.	

3.2.18.1 Operating Mode Control

The Operating mode is controlled through the CALC#:MODE command. The choices are NORMal, BURSt, SWIFt and FMOD. Sending CALC1:MODE will set the operating mode for the entire instrument, not just for channel 1. The fast mode permits more frequent return of measurement data to the host during operation in the modulated measurement modes (MAP, PAP or BAP).

CALC:MODE

Syntax:	CALCulate<channel 1 to 4>:MODE *space* <NORMal, BURSt, SWIFt or FMOD selection>	
Example:	OUTPUT @PWR_MTR;CALC2:MODE FMOD	! SETS THE MEASUREMENT TO FMOD
Description:	This command sets the measurement mode. This is NOT a channel-specific command; it is an instrument configuration command. Selecting one channel changes both channels to the desired mode. Use this command to select the level of functionality and performance for unique applications. NORMal Mode is full functioned. SWIFt Mode allows individual data point triggering. BURSt Mode allows the fastest measurement rates. FMOD added to the list of available modes.	

The following commands are duplicates of those given in the Instrument Triggering section of this chapter and are shown here for convenience. These commands must be used with BURSt mode operation.

TRIG:MODE

Syntax:	TRIGger:MODE*space* < burst mode data gathering POST trigger receipt or PRE trigger receipt >	
Example:	OUTPUT @PWR_MTR;TRIG:MODE POST	! SETS BURST OR SWIFT MODE DATA COLLECTION ! RELATIVE TO TIMING OF EVENT TRIGGER POST
Description:	This command is only for BURSt Mode measurements. Send this command only after CALC#:MODE BURS. TRIG:MODE controls the operation of the FIFO data buffer in the 8650A. When set to POST, the burst of data points is taken after the receipt of a valid instrument trigger. When set to PRE, the burst of data is assumed to be those data points that have arrived immediately preceding the valid instrument trigger event. Using PRE requires some caution: the single processor inside the 8650A is performing data acquisition constantly. Sending any configuration commands while the 8650A is collecting the PRE trigger data, interrupting the timing of the data point collection can be done.	

TRIG:DEL

Syntax:	TRIGger:DELay*space* < delay time between buffered readings in sec, 0.000 to 5.000 >	
Example:	OUTPUT @PWR_MTR;TRIG:DEL 5E-3	! SETS BURST MODE MEASUREMENT TIME TO ! 5 MS BETWEEN MEASUREMENTS
Example 2:	OUTPUT @PWR_MTR;TRIG:DEL 0	! MAX. MEASUREMENT RATE, 5100/SEC. (RESOLUTION, ONE MSEC)
Description:	This command applies only to BURSt Mode. It sets the time interval between the measurements; it is accurate to about 5%. This is the only way to control the meter's data acquisition of individual data points in a burst measurement. Burst measurement data is taken following or just prior to a single instrument trigger event. The number of measurements taken is controlled by TRIG:COUNt. Timing can be set in 0.001 second increments from 0.000 to 5.000, inclusive. Setting this control to 0.000 activates the highest reading rate possible with 8650; 5100 readings per second. Triggering of the Peak Power Sensor occurs independently from instrument triggering. There is a moderate ability to control triggering at the Peak Power Sensors. Generally, if requiring control of individual data point triggering, SWIFt mode will provide the highest measurement rates, with or without SRQ or hardware handshaking. The 8650A controls its gain ranges transparently by verifying internal range changes just before and just after individual measurement points. If the internal range does not remain the same, the data point will be discarded as invalid.	

TRIG:COUN

Syntax:	TRIGger:COUNt*space* < number of data values to buffer in memory from 1 to 5000 >	
Example:	OUTPUT @PWR_MTR;TRIG:COUN 100	! SETS BURST OR SWIFT MODE BUFFER READING ! NUMBER TO 100
Description:	The 8650A has an internal data buffer for storing measurement data until the slot 0 controller/ resource manager is ready to read the data. During SWIFt or BURSt modes, this command controls the number of power readings that are stored in the 8650A data buffer. The value specified here applies to each channel. If two sensors are attached and calibrated, the 8650A will perform the specified measurement and buffering for both channels. In the example above, that would be 100 points per channel. During BURSt mode, the COUNt # of readings will be taken with just one instrument trigger. During SWIFt Mode however, control the triggering of individual data points. Setting triggering to EXT, the 8650A will perform one measurement each time a TTL level signal is detected by the 8650A. Then the value is stored in the data buffer. This continues until the 8650A has received and operated the COUNt # of triggers and also read in the COUNt # of data points to the buffer.	

3.2.18.2 Triggering Notes

Refer to Instrument Triggering in Section 3.2.14 for more information.

EXT triggering is performed with the front panel BNC connector and will work only in BURSt and SWIFt Modes. EXT triggering is not available in NORMal Mode. Provision has been made for a hardware ready type handshaking capability using the analog output. This feature can be used as a loop back test when performing troubleshooting procedures.

BUS triggering is available for all operating modes, BURSt, SWIFt and NORMal.

IMMediate triggering is not compatible with BURSt and SWIFt Modes. If sending the CALC#:MODE BURS command to enter the BURSt Mode, IMM instrument triggering for NORMal Mode will automatically be switched to TRIG:SOUR BUS. If using the CALC#:MODE SWIF command to enter the SWIFt Mode, IMM instrument triggering will remain IMM, but a device-specific error will be generated whenever specifying a TRIG:COUN # higher than 1. Using the FETCh#? measurement query, taking accurate measurements can be done anyway; do not set TRIG:COUN.

Using FETCh#? when having time-dependent measurement processes can be a little tricky unless using SRQs. If the 8650A has not had enough time to process the measurement or has not received a trigger, it will return an abnormally large number — 9.e40 is common, but other obviously invalid readings can occur. Not using SRQs is fastest for measurement speed, manage the measurement/triggering timing problem closely.

3.2.18.3 Measurement Level Notes

The SWIFt Mode should not be used for measuring power in the bottom 10 dB of the CW dynamic range. At low power levels, the NORMal mode reduces the measurement speed to account for the effects of noise.

Example programs for High Speed Measurements are contained in Appendix A.

3.2.19 Peak Power Sensor Triggering

■ SENSe:TRIGger

The Peak Power Sensors sample power levels almost instantaneously. Since there is a sampler built into the Peak Power sensor housing, there are several controls to configure the source of the sensor trigger signal. These include the delay time from triggering to when the sample is to be taken (Sample Delay), and the trigger level. All Peak Power Sensors will operate in the SWIFt and BURSt modes when in the CW measurement mode (SOURce) configuration (not when using INT or EXT trigger).

SENSe:TRIGger

Syntax:	SENSe<sensor 1 or 2>:TRIGger:SOURce*space*<INTernal, EXTernal, or CW>	
Example:	OUTPUT @PWR_MTR;SENS1:TRIG:SOUR CW	! SETS SENSOR 1 PEAK TRIGGER MODE TO CW MODE
Description:	This command controls the trigger source for the sensors. Both the 80350A Series respond to this command. INTernal triggering is performed automatically by the 80350A Sensor; however, the trigger level must be set to a value of power that is lower than the power level of the pulsed signal incident upon the sensor. EXTernal triggering is performed by inputting a baseband pulse into the rear connector on the sensor housing. BNC to SMB cables have been provided for this purpose. For high speed trigger pulses, SMB to SMA adapters are available from Giga-tronics. CW triggering sets the power sensors to measure CW power and triggering is essentially disabled. CW is automatically set by the 8650A during power sweep calibration; do not apply any triggering to the peak power sensor during power sweep calibration. Be sure to set the trigger level, SENS:TRIG:LEV ##.# after the INTernal or EXTernal triggering is selected.	

Syntax:	SENSe<sensor 1 or 2>:TRIGger:DELay[:MAGnitude]:*space*<delay in seconds from -20e-9 to 105e-3>	
Example:	OUTPUT @PWR_MTR;SENS1:TRIG:DEL 1E-/6	! SETS SENSOR 2 PEAK DELAY VALUE TO 1E-6 ! SECONDS
Description:	The SENS:TRIG:DEL # is the time in seconds between the 80350A Series Peak Power Sensor's receipt of a sensor trigger (INT is auto, EXT is a trigger input) and the time that the sampler is fired. This delay is typically accurate to about 0.5%; repeatability is an order of magnitude better. There is a delay line built into the peak power sensor. Therefore, measuring the power on the leading edge of the pulsed signal even if it is the same edge on which the sensor was triggered. This also allows limited look ahead capability as identified by the allowed negative time values for this command. The other reason negative time values are allowed is due to the sample delay offset capability.	

Syntax:	SENSe<sensor 1 or 2>:TRIGger:DELay:STATe*space*<ON | OFF>	
Example:	OUTPUT @PWR_MTR;SENS1:TRIG:DEL:STAT ON	! ENABLES PEAK SENSOR 1 DELAY FUNCTION
Description:	This command activates the sample delay in the 80350A Series Peak Power Sensors.	

Syntax:	SENSe<sensor 1 or 2>:TRIGger:LEVel[:MAGnitude]*space*<R In EXT triggering use -0.1 to 5.000 V. In INT triggering use -30 to +20 dBm>	
Example:	OUTPUT @PWR_MTR;SENS1:TRIG:LEV 1.7	! SETS PEAK SENSOR 1 TRIGGER LEVEL TO1.7 VOLTS
Description:	This command sets the trigger level for the 80350A Series Peak Power Sensors. Under INTernal triggering the value is in dBm (-30 to +20 dBm, with default of -20 dBm). When EXTernal triggering is used the value is in Volts (-0.100 to 5.000 V, with a default of 1.700 V). The trigger level must be lower than the signal that is triggering the peak power sensor. Under Internal triggering, this means that a -16 dBm signal must be triggered with a setting that is less than -16 dBm. EXT responds similarly. The input impedance to the EXT trigger port on the back of the sensor housing is not 50 ohms: it is a high impedance to prevent damage from triggering inputs such as 5.0 V TTL signals. Using a high speed 50 Ω trigger source, this may cause unwanted signal reflections and noise on the trigger line. An SMB to SMB attenuator is available from Giga-tronics to alleviate this condition. The attenuation reduces the reflected noise, and reduces/eliminates the noisy trigger characteristics. Be sure to set the trigger level after the INTernal or EXTernal triggering is selected.	

Syntax:	SENSe<sensor 1 or 2>:TRIGger:OFFSet[:MAGnitude]	
Example:	OUTPUT @PWR_MTR;SENS1:TRIG:OFFS 0	! SETS PEAK SENSOR 1 TRIGGER OFFSET TIME TO 0 ! SECOND
Description:	This command sets the peak power sensor trigger offset time.	

Syntax:	SENSe<sensor 1 or 2>:TRIGger:TOTAl[:MAGnitude]?	
Example:	OUTPUT @PWR_MTR;SENS1:TRIG:TOTA?	! QUERIES PEAK SENSOR 1 TOTAL TRIGGER DELAY ! TIME
Response:	Current peak power sensor total trigger delay setting; a single value in seconds.	
Description:	This query reports the peak power sensor total trigger delay time.	

3.2.20 Averaging

■ SENSe:AVERage

Averaging is applied during normal operating mode. In the normal mode, Avg $_n$ results in $_n$ x 192 samples taken. In the Swift or Burst modes, Avg $_n$ results in 2_n samples taken. Auto averaging takes 4 samples all the time. For auto averaging in the normal mode, the averaging number is level dependent with very low averaging at +20 dBm, and many samples taken at very low levels.

SENSe:Average

Syntax:	SENSe<sensor 1 or 2>:AVERage:COUNt*space*<1,2,4,8,16,32,64,128,256,or 512>	
Example:	OUTPUT @PWR_MTR;SENS1:AVER:COUN 16	! SETS SENSOR 1 AVERAGE NUMBER TO 16
Description:	This command averages successive sensor readings in a digital averaging filter. As the measured signal level approaches the noise floor of the power sensor, measurement data values begin to fluctuate. To measure very low signal levels (within a few dB of the sensor noise floor), set a large value for averaging such as 128, 256, or 512. Be sure to zero the sensor, CALibration:ZERO, prior to critical measurements.	

Syntax:	SENSe<sensor 1 or 2>:AVERage:COUNt:AUTO*space*<ON | OFF>	
Example:	OUTPUT @PWR_MTR;SENS2:AVER:COUN:AUTO ON	! SELECTS SENSOR 2 AUTO-AVERAGING MODE
Description:	Auto-Averaging optimizes the measurement averaging value for reading stability and update rate when the measurement data is being repeatedly updated. At high power levels (measured), minimum averaging is applied to optimize measurement speed. As the power level drops, additional averaging is applied to steady the measured data reading.	

Syntax:	SENSe<sensor 1 or 2>:AVERage:TCONtrol*space*<data acquisition averaging method, MOVing or REPeat>	
Example:	OUTPUT @PWR_MTR;SENS1:AVER:TCON REP	! SETS SENSOR 1 TO ACQUIRE FRESH ! MEASUREMENT DATA BEFORE AVERAGING
Description:	The SENSe:AVERage:TCONtrol commands are instrument level commands that control the internal measurement data averaging of the power meter. Two modes are selectable, MOVing and REPeat. The MOVing control applies the current measurement to the previously measured data. That is, even if the averaging number is 16, a measurement data point every time a sample is taken will occur, not just after every 16 samples. There is some limited built in level sensing capability. For example, when Auto-Averaging is used and the power is suddenly changed from a low value to a high, MOVing averaging throws out the very low level previous data as the gain range changes and begins averaging fresh data. Thus, wide abrupt changes in power level are not masked by the MOVing averaging. The 8650A can fully range in about 4 ms. MOVing averaging is similar to TR1 mode in typical benchtop power meters. REPeat averaging performs full averaging every time the 8650A is triggering. Unless the averaging number is set to 1, REPeat is slower than MOVing averaging. REPeat control is similar to TR2 mode in typical benchtop power meters. Fresh data is measured and averaged with each complete measurement cycle.	

3.2.21 Relative or Referenced Measurements

■ CALCulate:REFerence

Relative and referenced measurements are used when one measured value needs to be compared to another measured value on the same channel. For example, this function is used when it is desired to monitor the power level variation around an initial turn on or reference set value.

Referenced measurements are performed when finding the 1 dB compression power of an amplifier or using a return loss bridge. When a stable source is used, a relative measurement is useful for measuring the loss through an attenuator with a single sensor or channel.

Due to the ability to set a specific value of reference using the CALC:REF[:MAG] # command, this reference measurement function can also be used setting personal calibration factors. Thus the channel and sensor offset functions can remain dedicated to other setup dependent controls in programming.

CALCulate:REFerence

<table>
<tr><td>Syntax:</td><td colspan="2">CALCulate<channel 1 to 4>:REFerence:STATespace<ON | OFF></td></tr>
<tr><td>Example:</td><td>OUTPUT @PWR_MTR;CALC2:REF:STAT ON</td><td>! ACTIVATES LEVEL REFERENCE FOR RELATIVE
! MEASUREMENTS</td></tr>
<tr><td>Description:</td><td colspan="2">This command activates the meter’s level reference to give it the ability to perform relative measurements.</td></tr>
</table>

Syntax:	CALCulate<channel 1 to 4>:REFerence [:MAGnitude]<dB Offset value from -299.999 to 299.999>	
Example:	OUTPUT @PWR_MTR;CAL2:REF -30.11	! SETS CHANNEL 2 REFERENCE OFFSET VALUE IN DB
Description:	This command sets the value to subtract from the channel measurement value. This command applies to the channel (1, or 2, or 1/2, ...), not the sensor (1 or 2, only).	

Syntax:	CALCulate<channel 1 to 4>:REFerence:COLLect	
Example:	OUTPUT @PWR_MTR;CALC2:REF:COLL	! CURRENT RDG. = = > REF LEVEL. ! TAKE CHANNEL 2 READING AS REFERENCE VALUE
Description:	The CALCulate:REFerence:COLLect command will cause the level of the current power measurement to become the reference level. There is no query form of this command. After sending this command, the channel measurement will be zero, assuming the power levels at the sensor(s) did not change and REF:STAT is ON.	

3.2.21.1 Error Control

- Selected basic SCPI syntax and execution error apply to these commands.
- Device-specific errors include the following and other -300 level errors.
- Use **CALC:REF** commands only in the NORMal Mode. Attempts to use these commands in the SWIFt and BURSt modes will be ignored and the error, -300, Device-specific error; Normal mode is off, will be generated.

3.2.22 Offsets

■ SENSe:CORRection

Sensor power offsets apply to the individual sensor. Use sensor offsets to account for the loss of attenuators, which are commonly used during measurement to reduce standing waves to improve measurement accuracy. If measuring high power signals (>100 mW), but are not using Giga-tronics high power sensors, use a power attenuator between the high power output and the sensor input to prevent damage. Enter the value of attenuation as a sensor offset, and the 8650A will automatically respond with the actual power level output in its measurement data.

SENSe:CORRection

<table>
<tr><td>Syntax:</td><td colspan="2">SENSe<sensor 1 or 2>:CORRection:OFFSet:COLLect</td></tr>
<tr><td>Example:</td><td>OUTPUT @PWR_MTR;SENS1:CORR:OFFS:COLL</td><td>! TAKE SENSOR 1 CURRENT READING AS OFFSET
! VALUE</td></tr>
<tr><td>Description:</td><td colspan="2">This command enters the current power level reading as the offset value.</td></tr>
</table>

<table>
<tr><td>Syntax:</td><td colspan="2">SENSe<sensor 1 or 2>:CORRection:OFFSet[:MAGnitude]space<offset value in dB, -99.99 to 99.99></td></tr>
<tr><td>Example:</td><td>OUTPUT @PWR_MTR;SENS2:CORR:OFFS 10.2</td><td>! COMPENSATE SENSOR 2 FOR 10.2 DB
! ATTENUATION</td></tr>
<tr><td>Description:</td><td colspan="2">This command enters a specific sensor offset value.</td></tr>
</table>

<table>
<tr><td>Syntax:</td><td colspan="2">SENSe<sensor 1 or 2>:CORRection:OFFSet:STATEspace<ON | OFF></td></tr>
<tr><td>Example:</td><td>OUTPUT @PWR_MTR;SENS2:CORR:STAT ON</td><td>! ENABLES SENSOR 2 OFFSET CORRECTION</td></tr>
<tr><td>Description:</td><td colspan="2">This command activates and deactivates the sensor offset function.</td></tr>
</table>

3.2.23 Communication Configurations

- **STATus:COMMunicate:SERial:RECeive**
- **STATus:COMMunicate:SERial:TRANsmit**

STAT:COMMunicate:SERial:RECeive

Syntax:	SYST:COMMunicate:SERial:RECeive:BAUD*space* < baud rate > (Where baud rate values are: 300, 1200, 2400, 4800, 9600, 19200, 38400)	
Example:	OUTPUT@PWR_MTR; “SYST:COMM:SER:REC:BAUD 38400”	! SETS THE RECEIVE BAUD RATE TO 38400
Example 2:	OUTPUT@PWR_MTR; “SYST:COMM:SER:REC:BAUD?” (Command is available for GPIB and RS-232 connection)	
Description:	Command modifies the receive baud rate for the RS-232 communications port to the specified baud rate. Query form is available.	

STAT:COMMunicate:SERial:TRANsmit

Syntax:	SYST:COMMunicate:SERial:TRANsmit:BAUDspace < baud rate > (Where baud rate values are: 300, 1200, 2400, 4800, 9600, 19200, 38400)	
Example:	OUTPUT@PWR_MTR; “SYST:COMM:SER:TRAN:BAUD 38400”	! SETS THE RECEIVE BAUD RATE TO 38400
Example 2:	OUTPUT@PWR_MTR; “SYST:COMM:SER:TRAN:BAUD?” (Command is available for GPIB and RS-232 connection)	
Description:	Command modifies the transmit baud rate for the RS-232 communications port to the specified baud rate. Query form is available.	

3.2.24 SRQ & Status Monitoring

- *CLS
- *ESE
- *ESR?
- *OPC
- *OPC?
- *SRE
- STATus:OPERation
- STATus:PREset
- *STB?

*CLS

Example:	OUTPUT @PWR_MTR;*CLS	! CLEARS SRQ AND STATUS BYTE REGISTERS
Description:	*CLS is the clear status command defined by IEEE 488.2. This command clears all of the status bytes to the value 0. After a service request interrupt is transmitted from the 8650A to the controller, use the *STB command to read the status byte from the 8650A. Then reset the SRQ and use *CLS clear status command to reset the numeric status indication of the status byte/registers to 0 (all bits will be 0). *CLS does not affect the enabled or disabled status of the status byte/register masks. For example, having bit 8 of the operation register enabled before sending *CLS, it will remain enabled afterward. *CLS also clears the output queue (takes 0.3 sec to complete this function).	

*ESE

Syntax:	ESE*space*<event status register value, 0 to 255>	
Example:	OUTPUT @PWR_MTR;*ESE 16	! ENABLES BIT 4 OF EVENT STATUS REGISTER MASK
Description:	This command sets/clears the event status register enable mask.	

*ESR?

Example:	OUTPUT @PWR_MTR;*ESR?	! RETURNS EVENT STATUS REGISTER VALUE
Description:	This query returns the current status register value.	

*OPC

Example:	OUTPUT @PWR_MTR;*OPC	! *OPC ALLOWS ONE TIME SRQ ENABLE
Description:	*OPC determines when an operation is completed. This command is generally used to monitor the completion of long measurement sequences. It sets the operation complete bit in the event status register upon completion of operation.	

*OPC?

Example:	OUTPUT @PWR_MTR;*OPC?	! RETURNS A VALUE OF 1 AFTER AN OPERATION IS COMPLETED
Description:	The *OPC? operation complete query allows synchronization between the controller and the test set using either the message available (MAV) bit in the status byte, or a read of the output OPC?.	

*SRE

Syntax:	*SRE*space* < event status register value, 128, 64, 32, 16, 8, 4, 2, 1 >	
Example:	OUTPUT @PWR_MTR;*SRE 32	! ENABLES BIT 5 OF STATUS BYTE REGISTER
Description:	This command sets the mask of the Status Byte register.	

STAT:OPER

Syntax:	STATus:OPERation[:EVENt]?	
Example:	OUTPUT @PWR_MTR;STAT:OPER?	! QUERIES OPERATION EVENT REGISTER RESULT
Description:	This query returns the operation event register result.	

Syntax:	STATus:OPERation:ENABle*space* < event register value 0 through 65535 >	
Example:	OUTPUT @PWR_MTR;STAT:OPER:ENAB 1	! SETS THE EVENT REGISTER ENABLE MASK
Description:	This command sets the enable mask, which allows true conditions in the event register to be reported in the summary bit.	

STAT:PRES

Syntax:	STATus:PRESet	
Example:	OUTPUT @PWR_MTR;STAT:PRES	! CLEARS ALL THE STATUS REGISTER VALUE
Description:	This command clears all status registers.	

*STB?

Example:	OUTPUT @PWR_MTR;*STB?	! QUERIES STATUS BYTE REGISTER
Description:	This query returns status byte register result. The SCPI status reporting structure includes the IEEE 488.2 Status Registers. Please note that bit 5 of the OPERation Status Register is only used during BURSt and SWIFt modes. NORMal mode does not use this function.	

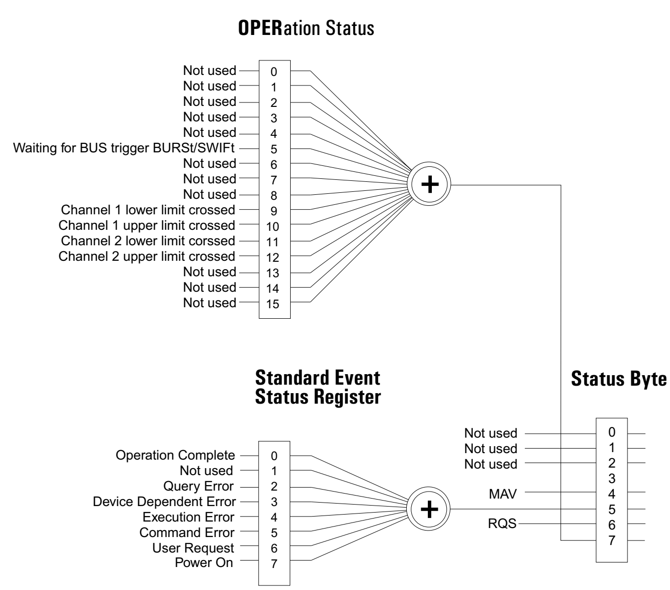

Figure 3-9: SCPI Status Structure Registers

3.2.24.1 Event Status Register

The *ESE command is used in combination with group commands that form the meter's service request system. These commands and their responses are almost identical to the IEEE 488 (GPIB) SRQ service request command structures. The *ESE command is used to enable bits of the event status register mask.

The *ESE command is one of the commands that can used to monitor the status of the power meter. Together with the status byte (also see commands *STB? or *SRE?) and the operation status register (STATus:OPERation), the event status register provides information on several critical 8650A functions and error conditions.

7	6	5	4	3	2	1	0
Power On	0	Command Error	Execution Error	Device-Dependent Error	Query Error	0	Operation Complete
Value = 128	Value = 64	Value = 32	Value =16	Value = 8	Value = 4	Value = 2	Value = 1

☛ ***NOTE:*** *The logical value in the register is used both to enable a bit's event function and to enable that bit of the event status register mask to report the event status (Following the *ESR? command).*

The event status register is 8 bits long and is structured as follows:

Bits 1 and 6 are not used. When bit 3 is a 1, a device-dependent error has occurred. When bit 4 is a 1, an execution error has occurred. When bit 5 is a 1, a command error has occurred. Bit 7 is a 1 when the 8650A is turned ON. The 8650A does not have a standby mode; therefore, it is probably not useful to enable bit 7 of the event status register mask (by sending the command *ESE 128).

In the default state, the meter's event status register is masked, or set to off. None of the bits are enabled. If an execution error occurs, a service request will not be sent to the controller and the event status register will remain set to 0. The *ESE command is used to enable individual bits. These bits must be enabled individually. They can be cleared or turned off as a group by sending the *CLS command.

To use the event status register, bit 5 of the status byte must also be enabled by sending the command *SRE 32.

The *ESR? query returns the current value of the event status register.

☛ ***NOTE:*** *The logical value in the register is used both to enable a bit's event function and to enable that bit of the event status register mask to report the event status (Following the *ESR? command).*

3.2.24.2 Status Byte Register

7	6	5	4	3	2	1	0
Operational Status	Require Service (RQS)	Event Status	Message Available	0	0	0	0

Notes:

1. The condition indicated in bits 1 through 5 must be enabled by the Service Request Mask to cause a Service Request Condition. The mask is set with the *SRE command followed by ASCII characters. The value of the byte is determined by summing the weight of each bit to be checked.
2. The RQS (bit 6) is true when any of the conditions of bits 4, 5 and 7 are enabled and occur.
3. Bits remain set until the Status Byte is cleared.

3.2.25 Min/Max Configuration & Monitoring

■ CALCulate:MAXimum
■ CALCulate:MINimum

MIN and MAX channel monitoring records the minimum and maximum variation of channel amplitude over time. Time zero is set by setting the MIN or MAX STATe to ON. The default Preset and turn on configuration is ON. After setting to ON, the minimum and maximum value of the measured value on that channel will be recorded. To reset the MIN and MAX value back to the current value, set the STATe to ON again. The STAT OFF does not need to be sent during reset of MIN and MAX values. Individual sensor minimum and maximum power level variations can be monitored only if that sensor is defined directly to a channel with POW 1 or POW 2. MIN and MAX channel monitoring is only available in Normal Mode. BURSt and SWIFt modes do not perform MIN and MAX monitoring.

CALC:MAX

Syntax:	CALCulate<channel 1 to 4>:MAXimum:STATe*space*<ON | OFF>	
Example:	OUTPUT @PWR_MTR;CALC2:MAX:STAT ON	! ENABLES CHANNEL 2 MAXIMUM TRACKING
Description:	ON activates the MAX monitoring function. The maximum value of the channel (not sensor) is monitored until the value is reset by CALC#:MAX:STAT ON or CALC#:MAX:STAT OFF is sent.	

Syntax:	CALCulate<channel 1 to 4>:MAXimum[:MAGnitude]?	
Example:	OUTPUT @PWR_MTR;CALC2:MAX?	! QUERIES CHANNEL 2 MAXIMUM VALUE IN DBM
Response:	Highest power reading since CALC2:MAX:STAT ON was sent.	
Description:	This command reports the value maintained as the maximum in the 8650A max monitor.	

CALC:MIN

Syntax:	CALCulate<channel 1 to 4>:MINimum:STATe*space*ON | OFF	
Example:	OUTPUT @PWR_MTR;CALC1:MIN:STAT ON	! ENABLES CHANNEL 1 MINIMUM TRACKING
Description:	This command activates the minimum channel value monitor.	

Syntax:	CALCulate<channel 1 to 4>:MINimum [:MAGnitude]?	
Example:	OUTPUT @PWR_MTR;CALC1:MIN?	! QUERIES CHANNEL MINIMUM VALUE IN DBM
Response:	Lowest power reading since CALC2:MIN:STAT ON was sent.	
Description:	This command reports the value maintained as the minimum in the 8650A min monitor.	

3.2.26 Limit Line Configuration & Monitoring

■ CALCulate:LIMit

The CALCulate:LIMit commands specify and query the status of power measurement limit values and limit line pass/fail checking. This allows the 8650A to monitor measured values and determine if the values are outside certain limits or above/below a single limit. The upper limit cannot be specified any lower than the lower limit; meaning that an exclusion zone of values cannot be specified.

Limit values can be specified separately for either of the two software calculation channels - 1 and 2. These channels can be specified to correspond to power sensors 1 and 2 (default) or as a combination of the power sensors. For example, set limit lines to monitor overload or under range conditions on an amplifier's output power and gain by specifying channel 1 as sensor 1 limits checking and channel 2 as Sensor 2/Sensor 1. Power sensor assignments to channel measurement definition is also part of the CALCulate Subsystem. Limit monitoring can not be performed for individual sensors unless a channel is configured for single sensor, POW 1 or POW 2 operation.

Upper and Lower limits can be active simultaneously.

Unless serial requests are enabled, the CALCulate:LIMit commands can not be configured to automatically alert the controller directly during a limit violation; the 8650A must be queried to receive information regarding pass/fail status of limit line violations.

Automatic notification of limit line violations is accomplished using the status byte and the operation register. A controller can be notified of a limit line violation via the request service, *SRE command. After the controller receives the request service, query the event status register. Check the status register for a limit line violation or send CALCulate:LIMit:FAIL? to check if a limit line is being or has been violated. 1 indicates a limit line violation. 0 indicates the channel measurement is within the limit lines.

CALCulate:LIMit

Syntax:	CALCulate<channel 1 to 4>:LIMit:CLEar[:IMMediate]	
Example:	OUTPUT @PWR_MTR;CALC1:LIM:CLE	! RESET CHANNEL 1 LIMIT VIOLATION INDICATOR TO 0
Description:	This command resets the limit line pass/fail indicator to 0.	

Syntax:	CALCulate<channel 1 to 4>:LIMit:FCOunt?	
Example:	OUTPUT @PWR_MTR;CALC1:LIM:FCO?	! QUERIES NUMBER OF CHANNEL 1 LIMIT FAILURES
Response:	Single number; the number of times the limit lines were exceeded.	
Description:	This command reports the number of times a limit line has been exceeded since the limit checking function was set to ON, or the limit line monitor was cleared with CALC#:LIM:CLE.	

Syntax:	CALCulate<channel 1 to 4>:LIMit:FAIL?	
Example:	OUTPUT @PWR_MTR;CALC1:LIM:FAIL?	! CHECK FOR CHANNEL 1 LIMIT LINE VIOLATION
Response:	0 = OK; 1 = fail	
Description:	This command reports whether or not a limit line has been exceeded since the limit checking function was set to ON, or the limit line monitor was cleared with CALC#:LIM:CLE.	

Syntax:	CALCulate<channel 1 to 4>:LIMit:LOWer*space*<numeric value in dB from -299.99 to 299.99>	
Example:	OUTPUT @PWR_MTR;CALC1:LIM:LOW -50.0	! SETS CHANNEL 1 LOWER LIMIT LINE TO -50 DBM
Description:	This command specifies the lower limit line power level. The value should allow for any offset values currently in use. The default value is -299.99 dB or dBm.	

Syntax:	CALCulate<channel 1 to 4>:LIMit:STATe*space*<ON | OFF>	
Example:	OUTPUT @PWR_MTR;CALC1:LIM:STAT ON	! ENABLES CHANNEL 1 UPPER AND LOWER LIMIT ! CHECKING
Response:	0 = off; 1 = on	
Description:	This command activates and deactivates the limit line checking. The default condition is OFF.	

Syntax:	CALCulate<channel 1 to 4>:LIMit:UPPer*space*<numeric value in dBm or dB from -299.99 to 299.99>	
Example:	OUTPUT @PWR_MTR;CALC1:LIM:UPP 17.0	! SETS CHANNEL 1 UPPER LIMIT LINE TO 17 DBM
Description:	This command specifies the upper limit line power level. The value should allow for any offset values currently in use. The default value is 299.99 dB or dBm.	

3.2.27 Analog Output

- **MEMory:CHANnel**
- **MEMory:UNIT**
- **MEMory:POWer**
- **MEMory:VOLTage**
- **OUTPut:ANAlog**

The ANALOG OUT BNC connector on the front panel can be configured to output a voltage that corresponds to the power levels on one of the channels. This is useful for applications such as source leveling or printing to a chart recorder.

The Analog Output operates only in NORMal Mode. It is automatically deactivated during SWIFt or BURSt modes, and comes back when operation is returned to the NORMal Mode.

MEM:CHANnel

Syntax:	MEMory[:TABLe]:CHANnel*space*<channel 1 to 4>	
Example:	OUTPUT @PWR_MTR;MEM:CHAN 2	! SETS ANALOG OUT TO CHANNEL 2
Description:	This command selects which channel will be present on the analog output connector.	

Syntax:	MEMory[:TABLe]:SELect*space*<ANALOGout,VPROPF1,VPROPF2>	
Example:	OUTPUT @PWR_MTR;MEM:SEL VPROPF1	! SELECTS MEMORY TABLE VPROPF1 ! FOR EDITING IN FOLLOWING LINES
Description:	This command selects between one of three editable tables. Using the meter's Analog Output capability requires configuring the numerically linear power measurement to voltage output relationship by defining the corresponding end points of power and voltage.	

MEM:UNIT

Syntax:	MEMory[:TABLe]:UNIT*space*<choice of units dBm or Watt>	
Example:	OUTPUT @PWR_MTR;MEM:UNIT DBM	! SETS ANALOG OUT POWER UNIT
Description:	This command selects between one of two configurations, log units or linear units. In either case the voltage output will be numerically linear. Be sure to set the units control properly (dBm or W) before trying to set the numeric values.	

MEM:POWer

Syntax:	MEMory[:TABLe]:POWer*space*<start value from -80 to +20 dBm>*comma*<stop value from -80 to +20 dBm>	
Example:	OUTPUT @PWR_MTR;MEM:POW -70,20	! SETS ANALOG OUT POWER RANGE (IN DBM) ! FROM -70 TO 20 DBM
Description:	If log units are selected for analog output control, this command selects the beginning and end points of the power levels which will be assigned corresponding voltages at the analog output.	

Syntax:	MEMory[:TABLe]:POWer*space* < start value from 0 to 0.01 W > *comma* < stop value from 0 to 0.01 W >	
Example:	OUTPUT @PWR_MTR;MEM:POW 0,1E2	! SETS ANALOG OUT POWER RANGE (IN WATTS) ! FROM 0 TO 0.01 WATTS
Description:	If linear units are selected for analog output control, this command selects the beginning and end points of the power levels which will be assigned corresponding voltages at the analog output. Also see MEM:VOLT below.	

MEM:VOLTage

Syntax:	MEMory[:TABLe:VOLTage*space* < start value from -10 to +10 V > *comma* < stop value from -10 to +10 V >	
Example:	OUTPUT @PWR_MTR;MEM:VOLT -7,2	! SETS ANALOG OUT VOLTAGE RANGE FROM ! -8 TO 2 VOLTS CORRESPONDING TO POWER
Description:	This command is not units dependent. It selects the beginning and end points of the voltage levels desired to be present at the analog output. These voltages correspond to the power end points that are defined with either MEM:VOLT or MEM:POW.	

OUTP:ANAlog

Syntax:	OUTPut[:BNC]:ANAlog[:STATe]*space* < ON | OFF >	
Example:	OUTPUT @PWR_MTR;OUTP:ANA ON	! ENABLES ANALOG OUT FUNCTION
Description:	This command activates and deactivates the analog output. The analog output will not operate unless this control is set to ON.	

3.2.28 Saving & Recalling Configurations

- ***RCL**
- ***SAV**

The 8650A has 21 instrument state memory registers. Registers 1 through 20 are available for store and recall. Register 0 contains the previous state of the instrument and can be used to toggle between two different instrument configuration states.

Instrument configuration can be saved to registers 1 through 20.

CAUTION

Any configuration items which are not listed under the *RST or PRESet conditions are not savable. Make sure all aspects of the configuration are savable. For example, sensor power sweep calibration curves can not be saved in the configuration memory registers. Sensors must be calibrated to the 8650A power meter each time a new sensor is attached.

*RCL

Syntax:	*RCL*space* <memory location number 0 to 20>	
Example:	OUTPUT @PWR_MTR;*RCL 19	! RECALL 8650A REGISTER 19
Description:	Recalls instrument configuration. 0 is the PRESet configuration.	

*SAV

Syntax:	*SAV*space* <memory location number 1 to 20>	
Example:	OUTPUT @PWR_MTR;*SAV 20	! SAVES AT 8650A REGISTER 20
Description:	Saves current configuration to memory. Saving a configuration to memory position 0 cannot be done.	

3.2.29 Halting Operation

- **ABORt**

ABORt

Example:	OUTPUT @PWR_MTR;ABOR	! HALTS MEASUREMENT & TRIGGERING
Description:	This command stops operation, but it does not interrupt the completion of the current action. For example, sensor calibration is not interrupted. Burst mode data collection is not interrupted.	

3.2.30 Language Configuration

■ SYSTem:LANGuage NATIVE

SYST:LANG NATIVE

Example:	OUTPUT@PWR_MTR;SYST:LANG NATIVE	! SWITCHES ACTIVE GPIB COMMAND SET TO NATIVE LANGUAGE
Description:	Command modifies the active GPIB command set to native (8600) from SCPI. Query from is not available for this command.	

3.2.31 Preset Configuration

- ***RST**
- **STATus:PRESet**
- **SYSTem:PRESet**

*RST

Example:	OUTPUT @PWR_MTR;*RST	! RESET 8650A CONFIGURATION
Description:	This command resets the 8650A configuration to a known condition (See Table 3-4). These are not the power ON conditions. The 8650A has an internal battery which powers a non-volatile memory chip to retain configuration information. The only configuration that will change between power OFF and power ON is noted at the end of the table.	

STATus:PRESet

Example:	OUTPUT @PWR_MTR;STAT:PRES	! CLEARS ALL THE STATUS REGISTER VALUE
Description:	This command resets the 8650A Status information buffers. See Table 3-4. Note that these are not the power ON conditions.	

SYSTem:PRESet

Example:	OUTPUT @PWR_MTR;SYST:PRES	! RESET 8650A CONFIGURATION
Description:	This command resets the 8650A configuration to a known condition. SYST:PRES is identical in function to *RST (Command Default, Minimum, Maximum).	

Table 3-4: Reset & Power on Default Commands

Command	Default	Minimum	Maximum
CALCulate1[:CHANnel]: CALCulate2[:CHANnel]:	POWer 1 POWer 2	,	
CALCulate1-2:			
LIMit			
STATe	OFF		
UPPer	299.999	-299.999	299.999
LOWer	-299.999	-299.999	299.999
FAIL?	0		
FCOunt?	0		
MAXimum:			
STATe	OFF		
[MAGnitude]	299.999	-299.999	299.999
MINimum:			
STATe	OFF		
[MAGnitude]	-299.99	-29.999	29.99
MODE	NORmal		
REFerence:			
STATe	OFF		
[MAGnitude]	0	-299.99	299.99
STATe	ON		
UNIT	dBm		
INITiate:CONTinuous	OFF	,	

Table 3-4: Reset & Power on Default Commands (Continued)

Command	Default	Minimum	Maximum
MEASure1-2[:SCALar:POWer]	With Auto average on	,	
MEMory[:TABLe]: CHANnel FREQuency POWer SELect ANALOGout SLOPe VOLTage UNIT	 1 0 -80,20 0 -10,10 dBm	 1 0 -80,-80 0 -10,-10	 2 40e9 20,20 10e9 10,10
OUTput: ROScillator[:STATe] [BNC:]ANAlog[:STATe]	 OFF OFF	,	
SENSe1-2: AVERage: COUNt: AUTO TCONtrol CORRection: FREQ OFFset: STATe [MAGnitude] VPROpf[:STATe] TRIGger: DELay: STATe [MAGnitude] LEVel (if INT trig) LEVel (if EXT trig) SOURce	 1 ON MOVing 5e7 OFF 0 OFF OFF 1e-6 -20.0 1.7 IMMediate	 1 0 -99.999 -5e-8 -30.0 0.1	 512 4e10 99.999 104e-3 20.0 5.000
TRIGger: COUNt DELay MODE SOURce	 1 0.000 POST IMMediate	 0 0.000	 128000 5.000
UNIT1-2[:POWer]	dBm	,	
POWER ON will override the internal battery back up memory and reset only the following configuration items as shown below. All other configuration items will remain the same as was true before power OFF.			
CALCulate1-2:MODE	NORMal	,	
INITiate:CONTinuous	OFF	,	
MEMory[:TABLe]:SELect	ANALOGout	,	
OUTput:ROSCillator[:STATe]	OFF	,	
TRIGger: COUNt DELay MODE SOURce	 1 0.000 POST IMMediate	 0 0.000	 128000 5.000

3.2.32 Identification Commands

- ***IDN**
- **SENSe:CORRection:EEPROM:TYPE?**
- **SYSTem:VERSion**

Identification commands ensure that the power meter and sensor are appropriate for the test program. The sensor identification commands allow monitoring what model of sensor is attached and ensure that the sensor has been properly calibrated to the power meter.

☛ ***NOTE:*** *Example programs for Identification Commands are contained in Appendix A of this publication.*

*IDN?

Example:	OUTPUT @PWR_MTR;*IDN?	QUERY INST. MFGR & MODEL #
Response:	Giga-tronics, 8650,0,1.09 where 1.09 is software version	
Description:	The *IDN? is the identify query command defined by IEEE 488.2. Upon receipt of this command, the power meter will output a string that identifies itself as the Giga-tronics 8650A and indicates the firmware version number.	

SENS:CORR:EEPROM:TYPE?

Syntax:	SENSe<sensor 1 or 2>:CORRection:EEPROM:TYPE?	
Example:	OUTPUT @PWR_MTR;SENS1:CORR:EEPROM:TYPE?	! QUERIES SENSOR 1 EEPROM TYPE
Response:	80301,1818436 which is model number and serial number respectively	
Description:	This command returns sensor model and serial number information.	

SYST:VERSion

Syntax:	SYSTem:VERSion?	
Example:	OUTPUT @PWR_MTR;SYST:VERS?	! QUERIES SCPI VERSION
Response:	1995.0	
Description:	This query returns the compiled SCPI version.	

3.2.33 Calibrator Controls

■ OUTPut:Reference:OSCillator

The reference oscillator is the RF calibration source for the sensors.

■ OUTP:ROSC

OUTP:ROSC

Syntax:	OUTPut:ROSCillator[:STATe]*space* < ON | OFF >	
Example:	OUTPUT @PWR_MTR;OUTP:ROSC ON	QUERY INST. MFGR & MODEL #
Description:	The reference oscillator (Calibrator port) connection is on the front panel of the power meter. The OUTPut:ROSCillator ON command turns on this 0 dBm, 50 MHz output. The reference oscillator power level should be calibrated annually. Please refer to the Calibration Procedures in Chapter 4 for additional details.	

3.2.34 Sensor EEPROM Commands

■ SENSe:CORRection:EEPROM:TYPE?
■ SENSe:CORRection:EEPROM:CALFactor?
■ SENSe:CORRection:EEPROM:FREQuency?

SENSe:CORRection:EEPROM:TYPE?

Syntax:	SENSe < sensor 1 or 2 > :CORRection:EEPROM:TYPE?	
Example:	OUTPUT @PWR_MTR;SENS1:CORR:EEPROM:TYPE?	! QUERIES SENSOR 1 EEPROM TYPE
Response:	Typically, 80301,1818436 which is model number and serial number respectively	
Description:	This command allows programming to verify correct sensor usage and monitor whether power sweep calibration has been performed or not.	

SENSe:CORRection:EEPROM:CALFactor?

Syntax:	SENSe < sensor 1 or 2 > :CORRection:EEPROM:CALFactor?	
Example:	OUTPUT @PWR_MTR;SENS1:CORR:EEPROM:CALF?	! QUERIES SENSOR 1 EEPROM CAL FACTOR TABLE
Response:	Typically, 5.000e7,2.000e9,......	
Description:	Provides a listing of calibration factors in the sensor, sorted in frequency order, to match previous command. The number of items matches the sum of the number at standard frequencies, and the number of special frequencies reported by DIAG:SENS1:EEPROM:CALFR?	

SENSe:CORRection:EEPROM:FREQuency?

Syntax:	SENSe < sensor 1 or 2 > :CORRection:EEPROM:FREQuency?	
Example:	OUTPUT @PWR_MTR;SENS1:CORR:EEPROM:FREQ?	! QUERIES SENSOR 1 EEPROM FREQUENCY TABLE
Response:	Typically, 5.000e7,2.000e9,......	
Description:	Provides a listing of calibration frequencies in the sensor, sorted by frequency, so that both standard and special frequencies are listed. The number of items matches the sum of the number at standard frequencies, and the number of special frequencies.	

3.2.35 Self-Test

■ *TST?

*TST?

Syntax:	*TST?	
Example:	OUTPUT @PWR_MTR;*TST?	! QUERIES SELF-TEST RESULT
Response:	0 if all is OK, otherwise, it returns a value of 1.	

3.2.36 Error Messages

■ SYSTem:ERRor?

SYSTem:ERRor?

Syntax:	SYSTem:ERRor?	
Example:	OUTPUT @PWR_MTR;SYST:ERR?	! QUERIES SYSTEM ERROR MESSAGE
Description:	This command reads error messages from the error buffer. Use *CLS and CLEAR @PM_address to clear the SYST:ERR buffer just prior to entering measurement configurations and measurement routines. A listing of compatible standard SCPI error and device-specific errors follows.	

Table 3-5: SCPI Standard Error Messages

Error Number	Description
0	No Error
-7	Invalid Error Number
-108	Parameter Not Allowed
-111	Header Separator Error
-113	Undefined Header;- - - -
-120	Numeric Data Error
-200	Execution Error
-211	Trigger Ignored
-213	Init Ignored
-214	Trigger Deadlock
-230	Data Corrupt or Stale
-300	Device-specific errors (See Table 3-6)

Table 3-6: Device Specific Error Messages

Command	Error Message	Example of Problem
ABORt		
CALCulate1:DATA?	Burst mode is off	Not burst mode; Command requires BURSt mode
CALCulate1:DIFFerence 1,1 CALCulate1:RATio 1,1	Conflict in channel configuration	Same sensor
CALCulate1:LIMit:LOWer:UPPer	Conflict between upper and lower limits	Upper is smaller than lower or lower is greater than upper
CALCulate1:MAXimum:STATe	Channel is not valid	No sensor or sensor not calibrated
CALCulate1:MINimum:STATe	Channel is not valid	No sensor or sensor not calibrated
CALCulate1:MODE	No valid sensor Channel is not valid	No sensor No sensor or sensor not calibrated
CALCulate1:REFerence:COLLect	Normal mode is off Channel is not valid	Burst mode setup No sensor or sensor not calibrated
CALibrate1	No sensor Sensor not connected to calibrator Sensor calibration error	No sensor Calibration error
CALibrate1:ZERO	No sensor Sensor zeroing error	Zeroing error
DIAGnostic:SENSe1:EEPROM:CORRection	No valid sensor sensor eeprom data:correction factor number does not match	No sensor
DIAGnostic:SENSe1:EEPROM:READ	No valid sensor	No sensor
DIAGnostic:SENSe1:EEPROM:CALFSPecial DIAGnostic:SENSe1:EEPROM:CALFSTandard	No valid sensor Sensor eeprom data is over space Sensor eeprom data:cal factor number does not match	No sensor Too much data
DIAGnostic:SENSe:EEPROM:FREQSTandard?	No valid sensor Sensor eeprom data is over space	No sensor Too much data
DIAGnostic:SENSe1:EEPROM:CALFRange DIAGnostic:SENSe1:EEPROM:TYPE	No valid sensor	No sensor
DIAGnostic:SENSe1:EEPROM:FREQSPecial	No valid sensor Sensor eeprom data is over space Sensor eeprom data:freq is above upper freq limit Sensor eeprom data:freq is below lower freq limit Sensor eeprom data:freq number do not match	No sensor Too much data
DIAGnostic:SENSe1:EEPROM:TYPE	No valid sensor Sensor eeprom data:upper freq should be larger than lower freq	No sensor
DIAGnosticSENSe1:EEPROM:WRITe	No valid sensor Sensor eeprom data:write error Password is incorrect	No sensor Sensor eeprom error Password is activated

Table 3-6: Device Specific Error Messages (Continued)

Command	Error Message	Example of Problem
FETCh1?	Data corrupt or stale TTL trigger mode not applicable	No valid sensor Normal mode
INITiate:IMMediate	Init ignored	Continuous is on
MEASure1?	Data corrupt or stale Normal mode is off	No valid sensor Burst or swift mode
MEMory:TABLe:FREQuency	No valid sensor V_{PROP}F Table is not selected V_{PROP}F Table freq data error	No sensor Select others Start freq is greater than sensor maximum
MEMory:TABLe:POWer	Analog out Table is not selected Analog out Table data input error	Other table Lower power is greater than upper power
MEMory:TABLe:SLOPe	No valid sensor V_{PROP}F Table is not selected	Select others No sensor
MEMory:TABLe:UNIT	Analog out Table is not selected	Other table
MEMory:TABLe:VOLTage	Analog out Table is not selected Analog out Table data input error	Other table Lower voltage is greater than upper voltage
READ1?	Data corrupt or stale Normal mode is off Trigger deadlock	No valid sensor Burst or swift mode is on TRIG:SOUR is not in IMM mode
SENSe1:EEPROM:CALFactor? SENSe1:EEPROM:FREQuency?	No valid sensor Sensor eeprom data is over space	No sensor Too much data
SENSe1:CORRection:VPROpf:STATe	No valid sensor	No sensor
SENSe1:TRIGger:DELay SENSe1:TRIGger:LEVel	No valid sensor Sensor is in CW source mode	No pulse sensor
SENSe1:TRIGger:DELay:STATe ON	No valid sensor	No pulse sensor
TRIGger:COUNter	Normal mode is on Counter has to be one in Swift immediate source No valid sensor Channel is not valid	Not in burst or swift mode Swift mode, imm source No sensor Ratio mode setup
TRIGger:IMMediate	Init ignored	Source is in immediate mode
TRIGger:MODE PRE	No PRE mode for trigger immediate source	
TRIGger:SOURce:IMMediate	Counter has to be one in Swift immediate source No valid sensor Channel is not valid	Swift mode is on No sensor Ratio mode setup

3.3 IEEE 488.2 Interface Command Codes

3.3.1 Command Syntax

The elements of the 8650A interface commands are introduced in this section. Because some commands are included only for compatibility with earlier models, there are some variations in syntax from one command to another, which must be carefully observed.

3.3.1.1 Function Codes

An interface command includes a function code to indicate the nature and purpose of the command. Some commands contain a function code and nothing else. For example, the function AP, which causes the 8650A to measure power using the A sensor, stands alone as a command. Commands consisting only of a function code are referred to in this manual as *simple* commands. However, most commands consist of a function code combined with other elements.

The CH *n* EN (where *n* = display line number) command must precede commands for setting units and resolution of power measurement and for setting relative mode, limits, min/max, and statistics. All subsequent commands will apply to the same display line number.

Function codes are listed alphabetically in the Command Code Sets in this section.

3.3.1.2 Prefixes

Some commands must begin with a prefix that identifies the sensor to which the command applies. For example, function code ZE (which causes a sensor to be zeroed) must be combined with a prefix to specify which sensor is zero'ed. The full command is either AE ZE (for sensor A) or BE ZE (for sensor B).

Many of the commands described in this chapter require an AE or BE prefix, which specifies the sensor that will be affected by the command. When the 8650A receives a command containing a sensor-specific prefix, it assumes that all subsequent commands refer to the same sensor until a command is received to specify the other sensor. Therefore, if a command prefixed by AE is received, subsequent commands can omit the prefix provided that they are intended for Sensor A.

It does no harm to include the prefix even when it is not required; some users may find that the most convenient approach is to include the prefix in all applicable commands.

3.3.1.3 Variables

Some commands must include one or more variables to specify quantities or options for the command. For example, the function code ANALOG (used in commands to configure the analog output) is combined with several variables to specify different aspects of the analog output. In the command

```
ANALOG LOG -80.0, 20.0, 0.0, 10.0
```

the variables are interpreted as follows:

LOG Specifies that power is to be measured in logarithmic units (that is, dB or dBm).

-80.0 Specifies that the low end of the analog output voltage range represents -80 dBm in.

+20.0 Specifies that the high end of the analog output voltage range represents +20 dBm in.

0.00 Specifies that the low end of the analog output range is 0 volts.

10.0 Specifies that the high end of the analog output range is 10 volts.

In the above example, the numeric variables are strung together with separator characters between them (See *Separators*, Section 3.3.1.5 below). However, in some commands, numeric variables are preceded in the command string by the variable name. For example, in the command FBUF PRE TTL BUFFER 200 TIME 1300, the numeric variables known as buffer and time are identified by name within the string.

Many variables are qualitative rather than quantitative; they select from among the various modes or options available for a particular function.

3.3.1.4 Suffixes

Some commands require a terminating suffix. For example, the function code DY specifies a duty cycle. It requires an AE or BE prefix to identify the sensor, and a numeric variable to indicate the duty cycle as a percentage. Finally, the command must include a terminating suffix (the choices of suffix in this case are EN, PCT and %). The command AE DY 50 % sets the duty cycle for sensor A to 50 percent.

☛ ***NOTE:** Some commands that include numeric variables require a terminating suffix while other commands do not, and interface problems will occur if the suffixes are used in commands which do not need them. Each command must be used so that its particular syntax requirements are met.*

3.3.1.5 Separators

Spaces, commas, colons, and semicolons can be used as separators between the various elements of a command (function codes, variables, etc.). For readability, commands in this manual are usually spelled out with spaces inserted between the elements (for example, SWIFT PRE GET BUFFER 100). Although separators within a command are permitted, they are usually not required. In the command descriptions in this chapter (beginning with Section 3.4), required separators are noted.

3.3.1.6 Command Format Illustrations

A command format is used in this manual to show the possible elements of a command, as illustrated below:

[AE or BE] DY [*n*] [EN or PCT or %]

Variables are shown within brackets. In this example, the prefix can be AE or BE, the function is DY, a numerical variable [*n*] follows the function, and the suffix at the end can be EN, PCT, or %.

Possible commands which use this example format include AE DY 42 % and BE DY 29.5 EN.

Because the Model 8651A supports only one sensor, the AE and BE prefixes can be omitted from any command issued to that model.

3.3.2 8650A Command Code Set

Table 3-7 lists the IEEE 488.2 common commands that are implemented in the 8650A. Most of these codes do not stand alone; commands; prefixes, variables, and suffixes must be combined with them. For further information refer to the sections cited in Table 3-7.

Table 3-7: 8650A IEEE 488.2 Command Set

Command	Description	Section
	Required Commands	
*CLS	Clear status byte	3.19.13
*ESE	Set Event Status Enable Register	3.19.13
*ESE?	Ask for current status of Event Status Enable Register	3.19.13
*ESR?	Ask for and clear Event Status Register bits	3.19.13
*IDN?	Ask for instrument ID	3.13
*OPC	Operation complete command	3.2.13 3.2.24
*OPC?	Operation complete query	3.2.13 3.2.24
*RST	Software reset	3.19.9
*SRE	Set the service request mask	3.19.13
*SRE?	Ask for service request mask	3.19.13
*STB?	Ask for status byte	3.19.13
	Function Commands	
@1	Set service request mask	3.19.13
@2	Set learn mode 2 data	3.14.3
?ID	Ask for instrument ID	3.13
AD	Measure A-B	3.19.12
AE	Select sensor A or analog output A for subsequent settings	3.3.1
ANALOG	Configure analog output	3.4
AP	Measure sensor A	3.19.12
AR	Measure A/B	3.19.12
BAP	BAP mode	3.18.4
BD	Measure B-A	3.19.12
BE	Select sensor B or analog output B for subsequent settings	3.3.1
BP	Measure sensor B	3.19.12
BR	Measure B/A	3.19.12
BSPE	Burst end exclude	3.19
BSTE	Burst start exclude	3.19
BTDP	Burst dropout	3.19.3
BURST	Fast buffered mode (see also FBUF in this table)	3.17.3
CH	Select active measurement line for subsequent commands	3.3.1
CL	Calibrate sensor	3.7
CRF	Ask for crest factor value	3.8

Table 3-7: 8650A IEEE 488.2 Command Set (Continued)

Command	Description	Section
CR	Crest factor	3.8
CS	Clear status byte	3.19.3
CW	CW mode	3.18
DA	Test LCD display	3.9
DC0	Duty cycle disable	3.10
DC1	Duty cycle enable	3.10
DD	Display disable	3.9
DE	Display enable	3.9
DU	Display user message	3.9
DY	Set duty cycle	3.10
EEPROM	Sensor EEPROM query	3.12
FA	Auto averaging	3.5
FBUF	Fast buffered mode	3.17.3
FH	Hold current averaging number	3.5.1
FM	Set averaging number	3.5.2
FMOD	Fast modulated mode	3.17.5
FR	Frequency	3.12.2
GATE	Time gating function	3.11
GT0	Cancel GET	3.16.2
GT1	GET single measurement	3.16.2
GT2	GET full measurement with settling	3.16.2
HIST	Histogram display	3.19.19
ID	Ask for instrument ID	3.13
KB	Enter cal factor	3.6
LG	Log units (dB or dBm)	3.19.6
LH	Set high limit	3.15
LL	Set low limit	3.15
LM0	Disable limit checking	3.15
LM1	Enable limit checking	3.15
LN	Linear units (Watts or %)	3.19.16
LP1	Ask for learn mode #1 string	3.14.1
LP2	Ask for learn mode #2 output	3.14.3
MAP	MAP mode	3.18
MAX	Ask for max value	3.19.4
MEAS	Ask for measurement mode	3.18.6
MIN	Ask for minimum value	3.19.4
MN0	Min/max disable	3.19.4
MN1	Min/max enable	3.19.4
OC0	Disable calibrator source	3.7.2
OC1	Enable calibrator source	3.7.2

Table 3-7: 8650A IEEE 488.2 Command Set (Continued)

Command	Description	Section
OF0	Offset disable	3.19.5
OF1	Offset enable	3.19.5
OS	Set offset value	3.19.5 3.19.6
PAP	PAP mode	3.18
PEAK	Peak sensor settings	3.19.8
PH0	Peak hold off	3.19.7
PH1	Peak hold on	3.19.7
PKH	Ask for peak hold value	3.19.7
PR	Preset the 8650A	3.19.9
PULSE	Peak sensor settings	3.19.8
RE	Display resolution	3.19.11
RL0	Disable relative measurement	3.19.10
RL1	Enable relative measurement	3.19.10
RL2	Use old reference for relative measurement	3.19.10
RV	Ask for service request mask	3.19.13
SETUP	Setup Configuration	3.19.14
SM	Ask for status message	3.19.13
ST	Store instrument state	3.19.14.2 3.19.15
STAT	Statistical analysis	3.19.21
STRIP	Strip Chart pause/resume and query	3.19.20
SWIFT	Swift mode	3.17.4
TEST	Test calibrator power output level	3.7.3
TEST CALIB	Test calibrator output power level	3.7.3
TR0	Trigger hold mode	3.16
TR1	Trigger single measurement	3.16
TR2	Trigger full measure with settling	3.16
TR3	Free run trigger mode	3.16
VPROPF	Configure $V_{PROP}F$ feature	3.19.17
ZE	Sensor zeroing	3.19.18

The 8650A is always able to measure over its entire dynamic range; there is no need to specify the range. Therefore, range-related commands have no effect on the measurement capability of the 8650A. The auto range, range hold, and set range commands only offset the analog output voltage, and only in HP436, HP437 or HP438 GPIB emulation modes. In these emulation modes (when using a single sensor, and not measuring in a relative mode), the power will be scaled to a range of 0 to 1 volts, representing the relative power within the current 10 dB range of the 8650A. The range hold and set range commands will simulate locking the range of power represented by the output voltage.