Willtek 4300

Mobile Service Tester

SCPI reference guide

CDMA version 5.1
TDMA version 7.5

Notice Every effort was made to ensure that the information in this document was accurate at the time of printing. However, information is subject to change without notice, and Willtek reserves the right to provide an addendum to this document with information not available at the time that this document was created.

Ordering information This guide is issued as part of the **4300 Series Mobile Service Tester**. The ordering number for a published guide is M 293 043.

The ordering number for the product depends on the exact model:

Table 1 4300 Series Mobile Service Tester models

Ordering number	Model	Supported formats
M 104 301	4301	AMPS
M 104 302	4302	AMPS, CDMA (cellular)
M 104 303	4303	AMPS, CDMA (cellular, PCS)
M 104 304	4304	AMPS, TDMA (cellular)
M 104 305	4305	AMPS, TDMA (cellular, PCS)

Federal Communications Commission (FCC) Notice

This product was tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This product generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this product in a residential area is likely to cause harmful interference, in which case you will be required to correct the interference at your own expense.

The authority to operate this product is conditioned by the requirements that no modifications be made to the equipment unless the changes or modifications are expressly approved by Willtek.

Industry Canada Requirements

This Class A digital apparatus complies with Canadian ICES-003.

Cet appareil numérique de la classe A est conforme à la norme NMB-003 du Canada.

Table of Contents

Chapter 3 Program Sequence Operation 17

About This Guide

Purpose and scope

The purpose of this guide is to help you successfully use the 4300 Series Mobile Service Tester features and capabilities. This guide includes task-based instructions that describe how to install, configure, use, and troubleshoot the 4300 Series Mobile Service Tester. Additionally, this guide provides a description of Willtek's warranty and repair information.

Assumptions

This guide is intended for intermediate and experienced users who want to use the 4300 Series Mobile Service Tester effectively and efficiently. We are assuming that you have basic computer experience and are familiar with basic telecommunication concepts and terminology.

Related information

Use this guide in conjunction with the following information:

Table 1 Related manuals

Ordering number	Document name
M 295 043	4300 Series Mobile Service Tester: Getting Started Manual
M 290 043	4300 Series Mobile Service Tester: User's Guide

Technical assistance

If you need assistance or have questions related to the use of this product, call or e-mail one of Willtek's technical assistance centers.

Table 2 Technical assistance centers

Region	Phone number	Fax number, email address
UK	+44 (0)20 8408 5720	+44 (0) 20 8397 6286 support.uk@willtek.com
Europe, Middle East, Asia, Africa	+49 (0)89 99641 386 +49 (0)89 99641 227	+49 (0)89 99641 440 support.eu@willtek.com
Americas	+1 317 595 2021 +1 866 WILLTEK	+1 317 595 2023 support.us@willtek.com

4300 Series Mobile Service Tester Overview

1

This chapter provides a general description of the 4300 Series Mobile Service Tester. Topics discussed in this chapter include the following:

About the 4300 Series Mobile Service Tester

The 4300 Series Mobile Service Tester is intuitive and user-friendly to operate. The 4300 offers single-button automated tests. This allows unskilled personnel to run a complete Go/No Go test of a mobile phone.

Whichever mode of operation you use, your test will be displayed on an easy to read LCD display with soft key menu-driven screens providing a complete summary analysis of analog or digital performance to improve repair and testing times. The display screen has selectable inverse video capability. The 4300 provides a wide range of testing variations, allowing for reprogramming channel parameters, base and mobile transmit power levels. All of these test features can be programmed from your personal computer, put on a floppy disk, and directly loaded into the 4300 via its PC-compatible disk drive; this allows for uploading, downloading, and distribution of programs and test data using standard PC software.

The 4300 provides increased flexibility with a cross point switch that controls the input and output of internal generators and filter combinations. Engineers can configure the bandpass filters required to accurately measure signal distortion and technicians can implement non-standard modulation tests using the two internal 10 Hz to 100 kHz generators that are programmable in 1 Hz steps.

What's new in this version

This manual refers to the following software configuration of the 4300:

Table 1 Software versions

4300 Series model	Version number
4301	1.3
4302, 4303 (CDMA)	5.1
4304, 4305 (TDMA)	7.5

Options

The models of the 4300 series are listed in section "Ordering information" on page i. The following options and accessories are recommended for use with the 4300 Series Mobile Service Tester:

ACELP vocoder software (TDMA only)

IS-136 custom software

Physical description

Elements such as the screen, keys, and connectors are described in detail in the 4300 user guide.

Specifications

For detailed specifications, please refer to the 4300 Series data sheets.

SCPI Fundamentals

2

This chapter describes the ideas behind SCPI programming. Topics discussed in this chapter are as follows:

Introduction

SCPI commands are based on a hierarchical structure known as a tree system. Associated commands are grouped together, analogous to the way leaves at the same level are connected to a common branch. Similar branches are, in turn, connected to thicker branches, until they meet at the root of the tree. Each branch of the command is called a *keyword* or *node*. To reference a particular leaf or *command*, the full path of keywords must be invoked. The path is formed by starting with the node closest to the root, and attaching the tributaries to it, each separated with a colon **(:)**, until the last branch of the command has been included.

Each keyword has both a long and a short form. In some cases, the two are the same. The long form is limited to a maximum of 12 characters and the short form is the first 1 to 4 characters of the long form. Either form can be used.

NOTE: SCPI language itself is case-insensitive, and any combination of upper and lowercase characters is acceptable. Also, it is not necessary to use all long forms or all short forms of each keyword in a command. Any combination of long and short forms is acceptable.

Example: For frequency, **`FREQuency`** is the long form and **`FREQ`** is the short form, either of these are acceptable, but **`FREQUEN`** is not.

In order to represent the long and short forms in the command listing, the short form is written in uppercase, and the rest of the long form is in lowercase.

Optional keywords, or *default nodes*, can be represented by more than one word. Default nodes are indicated by enclosing them in square brackets **([])**.

NOTE: Default node keywords may be omitted from the command path. If they are used, do not include the brackets with the keyword. Alternate keywords are listed as both versions of the command above its definition.

Parameter Types

Some commands have *parameters* or data associated with them. If parameters are defined for the command, each parameter will be enclosed in angle brackets **(<>)** if the parameter is required, and in square brackets **([])** if it is optional. A description of the parameters and the valid range for each will follow the command. Multiple parameters must be separated by a comma. Spaces or tabs (whitespace) are optional.

There are five basic SCPI command parameters:

Numeric

This parameter is represented by integer, floating point, or exponential notation (IEEE 488.2 NRf format or ANSI X3.42-1975). Some may optionally be followed by a unit suffix, if one is defined for the specific parameter.

Numeric values have upper and lower limits and a resolution associated with them. The keywords **`MINimum`** and **`MAXimum`** can be used in place of the numeric value in order to reference the minimum or maximum value for that parameter. The keyword **`DEFault`** can be used to set the parameter to its default values.

Responses are in floating point format with no exponential expression. Trailing zeros are omitted, unless it is the first digit to the right of the decimal point. For example, the value 102.675 MHz would be read back as: 102675000.0 if the command responds in units of Hz.

Integer

This parameter type is represented by decimal or hexadecimal integer, floating point, or exponential notation. Round all fractional portions to the nearest integer value. Unit suffixes are not allowed as integer types are unitless. Integers have upper and lower limits only since the resolution is always 1.

Preceed a hexadecimal value with "#h" or "#H". The keywords **`MINimum`** and **`MAXimum`** can be used in place of the numeric value in order to reference the minimum or maximum value for that parameter. The keyword **`DEfault`** can be used to set the parameter to its default values.

Responses are in integer format, with no decimal point.

Boolean

This parameter is represented by 0 or 1, or by the labels OFF or ON, with 0 representing OFF. Any numeric value that is nonzero is assumed to be 1.

Responses are either 0 or 1.

Enumerated

This parameter is IEEE character program data. It is defined as a string of ASCII alphanumeric characters representing a given state. The maximum length of the string is 12 characters.

Like command modes, enumerated parameters can be expressed by a long or short form

Example: EXTernal and INTernal are examples of enumerated parameters. The short form for these would be EXT and INT.

NOTE: The character data is not enclosed in quotation marks. Character program data responses are always reported as the short form of the expression.

String

This parameter is IEEE string program data. It is defined as ASCII characters enclosed in double quotes. A double quote character may be placed within the string by delimiting it with another double quote character

Example: The string **the " character** would be expressed as " the """ character."

If the double quotes are omitted and more than one word is defined in the string, an error will be flagged indicating that too many parameters have been sent.

If the string length sent is larger than the maximum defined for the parameter, the string will be accepted without flagging an error, although the string will be truncated to the maximum length, beginning with the first character of the string.

Example: "Test Information" would be truncated to "Test Info" if the maximum string length was 9.

The string response from a query is the ASCII text defined for the string, enclosed in double quotes.

Compound Commands

Compound commands are formed by connecting two or more commands on a single command line and separating them with a semicolon **(;)**.

Compound commands execute in the order they are received.

A *command path* is defined as the complete SCPI command list for the command.

A *parent command path* is defined as the current command path minus the last expressed node.

When the first command in a compound command begins with a colon **(:)**, it indicates that the entire command path is shown, whether the leading colon is present or not.

These commands are called *root commands* since they are referenced to the root node.

If the leading colon is omitted, the command is referenced to the parent command path of the prior command. This is called a *referenced command*.

Referenced commands are equivalent to sending the command preceded by the parent command path of the previous command (and a leading colon).

Example: `STATus:QUEStionable:CALibration:ENABle 1;PTRansition 1`

The leading colon was omitted from the first command. This is optional, since the first command has no previous command path on which to be referenced, and is a root command.

The second command in the line (following the semicolon) does not have a leading colon, and is referenced command to the root command.

Adding the parent command path to the beginning of the referenced command yields the following:

```
STATus:QUEStionable:CALibration:PTRansition 1
```

Commands in the list may alter the parent command path. A root command replaces the parent command path with its own command path minus the last node. A referenced command that requires more than one keyword (multilevel) extends the parent command path by adding its relative path minus the last node.

Example: `STAT:QUES:ENAB 1;BOARD:ENAB 1;SCP:ENAB 1`

the parent command paths after the first, second, and third commands are:

after 1st command - `STAT:QUES:`

after 2nd command - `STAT:QUES:BOARD:`

after 3rd command - `STAT:QUES:BOARD:SCP`

CAUTION ! The order in which commands are issued in a compound command line is very important. Ordering the commands incorrectly results in an error.

Commands having the shortest command paths should be placed at the start of the command line, and related commands placed in close proximity with each other.

CAUTION ! When a referenced command follows a command that ends in a default node, the default node must be included for the parent command path to be extracted properly.

SCPI does not allow nodes to be internally added to the parent command path if default nodes were not explicitly included. This is why the parent command path is defined as the current command path minus the last *expressed* node. An *implied* node (omitted default node) is not included unless it is followed by an expressed node.

Therefore the command,

```
PRINt:SELected:COLumn:TAB;BACKward
```

causes an error to occur on the second command. This is because the first command will execute `PRINt:SELected:COLumn:TAB:FORWARD` since Forward is a default node of TAB, but the parent command path is simply `PRIN:SEL:COL.` Since `BACKward` is not a sub-node of `COLumn`, SCPI defines this as a command error.

Avoid this problem by adding `:FORward` to the first command or by prefixing the second command with `TAB:`.

Common (Compound) commands do not affect the parent command path. They can be placed anywhere in the command line, and will not affect any relative commands that follow them.

Example: `STATus:QUEStionable:CALibration:ENABle 1;*CLS;PTRansition 1`

NOTE: When an error occurs in a compound command, only commands preceding the error will be executed; commands following the error will not be executed.

MINimum, MAXimum, and DEFault Values

All numeric and integer instrument settings have minimum and maximum limits. When the reset command (`*RST`) is sent, the unit is initialized to default values. These values can be set for each command, bypassing the label `MINimum`, `MAXimum`, or `DEFault` as the numeric or integer data.

Example: Set the SAT deviation upper and lower limits to maximum and minimum values, respectively by using the following command:

CALCulate:LIMit:DEViation:SAT:UPPer MAX;LOWer MIN

To query the SAT deviation upper and lower limits for numeric and integer parameter types use the following query command with the data selection (`MINimum`, `MAXimum`, or `DEFault`).

Example: `SOURce:AUDio:FREQuency? MAX` – maximum audio frequency value
`SYSTem:DATE? DEF,DEF,DEF` – default date values

NOTE: When a query command returns more than one value and a default, minimum or maximum value is requested, the number of `MIN/MAX/DEF` labels must match the number of numeric parameters, and must be separated by commas.

Query Commands

Most commands have a query form for extracting the current parameter settings of the unit. Some commands have no query form, and other commands exist only as a query. These exceptions will be noted in the definition part of the command descriptions.

Query commands are generated by adding a question mark **(?)** to the end of the command leaving no space between the command and the question mark. The parameters associated with a query follow the question mark.

Example: `:MEASure:SENSitivity? -120.0`

is the command for measuring the sensitivity of a mobile phone at -120 dBm.

When more than one query is placed in a single command line, the responses are separated by a semicolon.

When a query is performed of a parameter that has more than one value associated with it, the values are separated by a comma.

Example: `SYSTem:TIME?;*ESR?` produces the following response message:

16,8,52;4

to represent a time of 4:08:52 PM and an event status register value of 4.

A query used in a compound command along with commands that may change the value of the response, reflects the values at the time the query is received.

Example: `SOURce:AUDio:FREQ?;FREQ 2000;FREQ?;FREQ 3000;FREQ?` returns the following response:

1000;2000;3000

When the *valued coupled* parameters are queried in the same command line that they are being set, the query response reflects settings before the value coupled rules are executed.

NOTE: Coupled parameters are not updated until the terminator character is received for the command. Refer to “Value-Coupled Parameters” on page 11.

Value-Coupled Parameters

Instrument settings are considered value-coupled if the setting of one parameter affects or is dependent another.

Value-coupled parameters include:

- Autograph Start and Stop Frequencies
- Vehicle Speed and Delay Interval for Signal Quality with Fading
- Upper and Lower ESN Limits for Limit Table Auto Selection

Value-coupled parameters are executed as follows:

1. Verify that the parameter is within a valid absolute range.
2. Process the next command (if this is a compound command).
3. When all commands have been processed, check value-coupled parameter rules to determine actual settings.

If an error occurs at the first check (absolute limit check), none of the commands that follow it will be executed. If an error occurs at the second check, all non-coupled parameter commands will have been executed. Execution of value-coupled parameters is deferred until the entire command has been parsed, so that the value-coupled rules can be dependent on the combination of parameters changed. If an error occurs in the value-coupled rules, error -221, "Settings conflict" will be flagged.

When value-coupled parameters are queried in the same command line that they are being set, the query response reflects the settings before the value coupled rules are executed.

CAUTION ! Do not query the parameter in the same line that it is being modified, since it does not reflect the changes that the value-coupled rules may make.

Coupled-Value Rules

Autograph Start and Stop Frequencies

These parameters define the range of channels over which to perform an Autograph test. The basic principle of this rule is to assure that the start channel is the lower of the two frequencies.

`start = PROGram:AGRaph:CHANnel:STARt`

`stop = PROGram:AGRaph:CHANnel:STOP`

Rules:

If any of the following conditions are found, an error is generated and the parameter values are not changed:

`start = 1-799` and `stop = 991-1023`

`start = 1-799` and `stop = 1-799` and `start > stop`

`start = 991-1023` and `stop = 991-1023` and `start > stop`

`start = 1-1999` and `stop = 1-1999 PCS Band`

Vehicle Speed and Delay Interval for Signal Quality with Fading

These parameters are used for setting the fading when making signal quality (BER MAHO) measurements for TDMA. The following commands are renamed for clarity:

```
speed = SOURCe:SQUAlity:VSPeed
delay = SOURCe:SQUAlity:DINTerval
```

Rules:

- accept values if (speed > 0) or (delay = 0).
- else if only speed was changed, set delay to 0.
- else if only delay was changed, set speed to 0.1.
- else if both speed and delay were changed, indicate error and do not modify speed or delay.

Upper and Lower ESN Limits for Limit Table Auto Selection

These parameters are used for defining the lower and upper range of ESN values for a custom limit table. When AUTO select of the limit tables is enabled (see `CALC:LIMit:TYPE[:LOAD]:AUTO`) the ESN range limit and the manufacturer code defined for each of the custom limit tables is compared to the current phone registered in order to determine which table should be selected. The rules are used to assure that the LOWER limit is less than or equal to the UPPER limit.

```
lower = CALCulate:LIMit:MANufacturer:ESN:LOWer
```

```
upper = CALCulate:LIMit:MANufacturer:ESN:UPPer
```

Rules:

if `lower > upper`, indicate error and do not modify either parameter.

Sequential and Overlapped Commands

Remote commands can be of two types: sequential and overlapped.

A *sequential* command does not allow another command to begin until the current one has completed. Most of the commands for the 4300 are sequential commands.

An *overlapped* command allows additional commands to execute while the overlapped command is still in process. This is generally done on commands that may require a long time to execute, allowing other commands to be processed during the “idle” time of this command. If however, another command is received that requires the same hardware resources as the one that is still pending, no more commands are processed until the overlapped command completes. During the time that the overlapped command is in process, only commands that do not affect the behavior of or rely on the results of the overlapped command should be executed.

Example: `ALL:BREL;:MEAS:STAT:ST:DUR?` reports an incorrect duration value, since it executes immediately after the base release has begun. The base release requires more than 1 second to complete, but the ST measurement command attempts to read the result almost immediately after it has begun.

When it is desirable to delay processing of the next command until after the overlapped command has completed, the `*WAI` common command must be sent prior to the command that must wait for completion; the selected event to wait on must be setup by the `STATus:OPERation:COMPlete:ENABle` command.

Example:
```
STAT:OPER:COMP:ENAB 8
CALL:BREL;*WAI;:MEAS:STAT:ST:DUR?
```

The first command enables the operation complete flag for the base release command. The `*WAI` command will now wait until that command has completed execution before continuing with the next command. Since the base release command will have completed prior to the measurement command, the ST duration value should be correct.

All commands defined here are presumed to be sequential unless otherwise noted. When a command is defined as overlapped, the operation complete enable value will be defined for the command, as well as the resources that are required for that command. Another command that requires the resources that an overlapped command has the use of will suspend processing of remote commands until the overlapped command has completed.

All overlapped commands are sequential when sent from the sequencer instead of the remote port. This prevents the sequencer from executing any command until the last one has completed.

See Chapter 4 "TDMA SCPI Commands" and Chapter 2 "SCPI Fundamentals" for a listing of commands used for sequence operations, remote operations, and GPIB operations.

FIGURE 2-1. 4300 Audio Measurement Sources

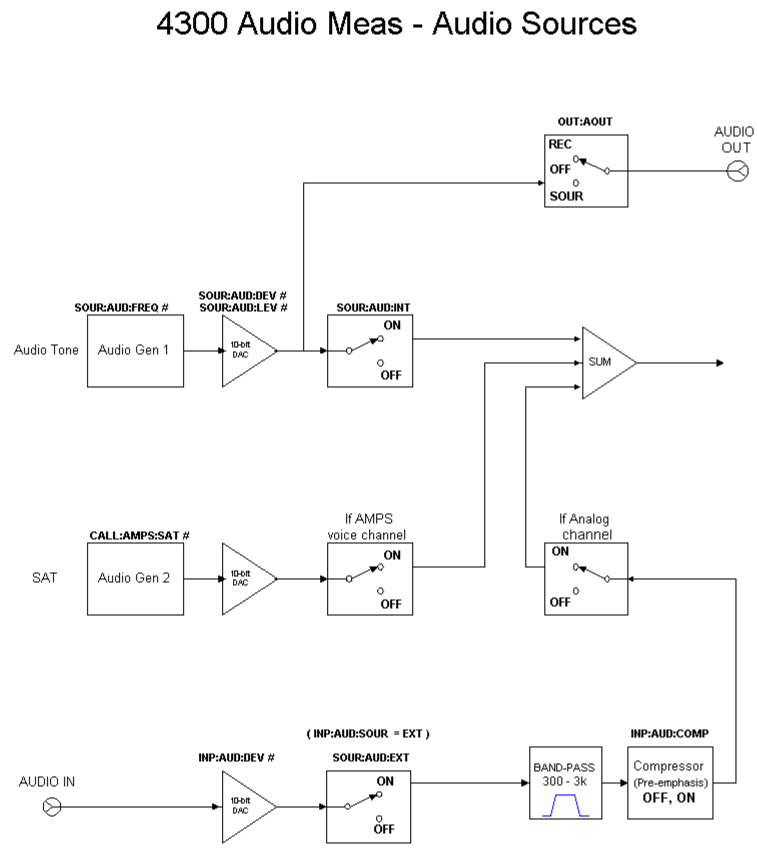
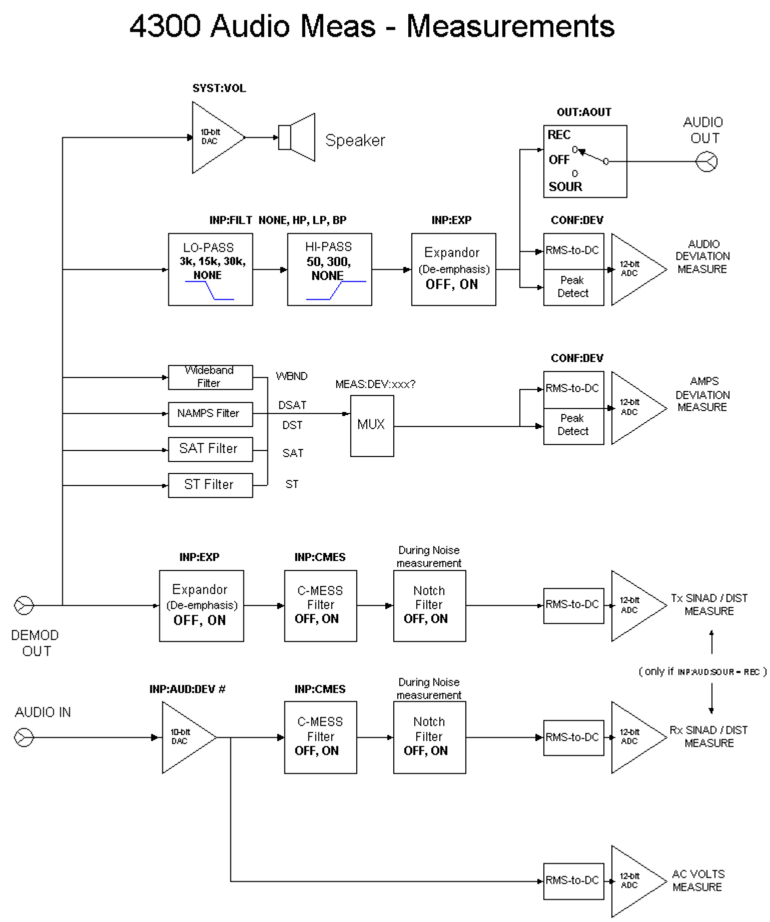


FIGURE 2-2. 4300 Audio Measurements



Program Sequence Operation

3

This chapter describes the functionality of the instrument. Topics discussed in this chapter are as follows:

General Operation

A program sequence is a text file that has been loaded into the 4300 either through the remote port or from the disk into one of five storage locations: `AUTO`, `QUICK`, or `CUSTOM1 - 3`.

It consists of a list of commands that are executed by the 4300 by pressing the corresponding key (AUTO, QUICK, or CUSTOM) on the front panel.

Program sequences use a BASIC-like syntax composed of a series of (SCPI) commands, program directives, comments and labels.

The sequence maximum length is 20K bits which includes a carriage return and line feed for each line and does not include text after "#" on each line.

The following sections define the components of a program.

Comments

Comments are created by setting the first non-whitespace character of the line to a '#'.

A comment may also be placed at the end of any program directive command except for the 'For' command.

Comments are not allowed on the same line as a SCPI command.

Comments are for documentation purposes only, and are ignored by the 4300.

Labels

Labels redirect the flow of the sequence in Branch and Gosub program directives.

They are case-sensitive; the label name SAMPLELABEL is handled differently from SampleLabel.

A label name is 1 - 40 characters in length, and is composed of a combination of upper and lower case letters, numbers 0 - 9, and the underscore character (_).

There is no limit to the number of labels that are defined in a sequence.

No checking for duplicate label names is performed. If a label name is duplicated, a branch statement references the one closest to the beginning of the sequence.

A label is defined by enclosing the label name in parenthesis (with no spaces) and placing a colon after the closing parenthesis.

The label may be preceded or followed by whitespace, but no commands or directives may be on the same line.

The following are valid labels:

(BeginMeasTests):	# VALID
(exit001):	# VALID (whitespace following end parenthesis)

(try_try_again): # VALID
(whitespace preceding beginning parenthesis)
#
(Not a label): # NOT VALID
(whitespace within parenthesis)

SCPI Commands

SCPI commands can be used as program sequencer commands and have the same context and operate in the same manner as remote operation, with three exceptions:

- all overlapped commands (a command that allows successive commands to begin prior to its completion) are treated as sequential commands.

Example: The CALL:ORIG command sent through the remote starts the mobile origination, but allows other commands (such as CALC:LIM:TYPE EIA) to execute before the origination has completed.

When the CALL:ORIG is sent as a command in a sequence, the next command does not start until mobile origination is completed.

- all measurement commands that require the ST to be `OFF` in order to measure accurately, pause before continuing if ST is detected.

A message on the display tells the user to press the SEND key on the mobile in order to turn ST off, so that the measurement can be made. Execution resumes when the ST is not detected.

- some substitutions may be made in the SCPI commands that are not available to remote operation.

The <num_param> entry can be substituted for a numeric argument, the <str_macro> may be substituted for an enumerated argument (providing the macro expands to a valid enumerated string) and both <num_param> and <str_macro> entries may be placed within double quotes for a string argument. This allows parameter values and loop values to be used in SCPI commands, rather than having to specify fixed arguments.

Several commands function differently when sent through the sequencer than when issued from the remote:

`PROG:AGRaph RUN`

This command (which starts the Autograph test) is not available through the sequencer.

`MEAS:DEV:PAUDio? [1 to 60]`

This command starts a continuous peak audio deviation measurement that terminates after the specified time duration (in seconds).

A special screen on the LCD indicates that a peak measurement test is in process and allows the user to terminate the test manually by pressing the CONTINUE key.

The time duration value is optional. If omitted, the test can only be terminated manually. This command is not available from the remote.

`DISK:LOAD:PROG <program name>`

This command loads the specified program from disk into a temporary location (not AUTO, QUICK, or CUSTOM).

Using the RUNPROG program directive following this command, allows program execution to be transferred to this temporary sequence.

When this sequence reaches the END directive, control is passed back to the original sequence at the command following the RUNPROG directive.

Only 1 level of program nesting is allowed. If this command is executed from within the nested program, an error is flagged.

Program Directives

Program directives control the program flow, defining/modifying/comparing variables that can be used in SCPI commands, as well as in other program directives, error handling, and a few other special functions.

They are interpreted by the program sequencer (not available to remote control).

Program directives consist of a command name (`DefineGlobal` or `Branch`) followed by an argument list. The argument list consists of zero or more program argument entries.

The command names are case-insensitive, but should be written as listed in this document for standardization purposes and readability.

Program Arguments

Program Arguments are required by the program directive and are composed of one or more base data types.

The types of arguments defined by the program sequencer include:

- A string of 1 to 40 alphanumeric characters (must begin with an letter).

NOTE: This character string is not enclosed in quotation marks.

- A numeric value consisting of either a <NUM_1> or a <num_param> entry.
- A character string argument consisting of either a <QSTR> or a <str_macro> entry.
- A list of one or more <string> and <numeric> entries separated by commas.

NOTE: The <string> is limited to only printable ASCII characters, but the <numeric> entry can be any 8-bit character having the range of 0 to 255 decimal.

Example: "This is a test", 13, 10 defines a string terminated by a carriage return (dec 13) and linefeed (dec 10).

Basic Data Types

<NUM_0>	Numeric integer value only.
	Consists only of the digits 0 - 9.
<NUM_1>	Numeric floating point value.
	Consists of optional + or - sign followed by zero or more numeric digits 0 - 9 followed by an optional decimal point and zero or more numeric digits.
<QSTR>	Quoted character string.
	Any printable ASCII characters (decimal value 32 to 126) enclosed in double quotes.
	To enclose a double quote character within the string, precede it by another double quote character (e.g. "This string contains the "" character.")

<num_param> A string of alphanumeric chars enclosed in angle brackets < >, where the string is the name of either a previously defined numeric global parameter (using the DefineGlobal command) or a loop parameter (using the FOR command).

The value for the numeric parameter is substituted for this string when the command is executed.

Global definitions may be used any time after their definition in the sequence (even in subroutine calls), but a loop parameter value is only defined during execution of the loop. Once a loop completes, its corresponding loop variable definition is no longer valid.

Example: for MacLevel = 0,7

```
call:plch:vmac <MacLevel>
meas:pow:tran?
prin:sel:str "MAC <MacLevel>:"
prin:sel:col:tab
prin:sel:item DATA
prin:sel:line:incr
NEXT
```

sets the MAC power values from 0 to 7, takes a power level measurement at each setting, and records both the MAC setting and power measurement in the print buffer.

<str_macro> One of the reserved string macros enclosed in angle brackets < >.

The string macro name is followed by a comma and either a <NUM_0> or <num_param> entry, and another comma and another <NUM_0> or <num_param> entry.

The first value is the character offset within the string at which to begin, and the second is the number of characters to use. If omitted, the character offset is 0 (start at first character) and the entire string length is used.

Example 1: <$LASTCMD>
specifies the entire last SCPI command string.

Example 2: <$LASTCMD,4>
skips the first 4 characters of the string.

Example 3: <$LASTCMD,4,3>
uses only 3 characters from the string starting with the 4th.

String Macros

$ENUM

The character string response that was returned from the last enumerated response from a SCPI query command can be accessed by using the enumerated alias <$ENUM>.

Use this alias wherever string or enumerated information is acceptable.

The string that is substituted for <$ENUM> is the short form response.

For instance, if the remote query had the following response

```
prog:name?
 CUST
```

then any occurrence of <$ENUM> following prog:name? is replaced by CUST.

Display this line on the LCD display using:

```
prog:name?
prog:str1"Running program: <$ENUM>".
```

Result: Running program: CUST.

$LASTCMD This macro is replaced with a string of characters indicating the last SCPI command header preceding the last SaveScpiCmd command. The command header does not include any parameters that were specified for the command.

Example 1:

```
meas:freq:err:sat?
SaveScpiCmd
prin:str 1,1,"<$LASTCMD>"
SaveScpiCmd
prin:str 2,1,"<$LASTCMD>"
```

prints the following two lines:

```
meas:freq:err:sat?
prin:str
```

The first SCPI command is duplicated in full (including the question mark) whereas the second command omits the parameters: 1,1,"meas:freq:err:sat?".

Both of these string alias definitions allow for selectively chopping of the specified string using optional offset and length parameters.

The offset specifies the starting character position, with the first character position being zero.

The length specifies the number of characters to supply.

If only one numeric value is given, it acts as the offset value.

Example 2:

```
prin:head1 "This is the first line of header text"
prin:head1?
prin:str 1,1,"<$QUERY,8>"
prin:str 2,1,"<$QUERY,8,9>"
prin:str 3,1,"<$QUERY,26,6>"
```

prints the following 3 lines:

```
the first line of header text
the first
header
```

$MEMFLOAT available in CMDA only.

$MEMSTR available in CMDA only.

$MEMSHORT available in CMDA only.

$MEMINTEGER available in CMDA only.

$MEMLONG available in CMDA only.

$NUMSTR The character string resulting from the last NumToStr or HexToStr command.

$PAGEBUF The current line from the page buffer (defined by PRINt:SEL:LINE).

This allows access to the text, not the printer control codes.

$QUERY The string response that was returned from the last SCPI string query command can be accessed by using the string alias <$QUERY>.

This alias is used wherever the string information is acceptable.

The string substitution for <$QUERY> is the full string response, excluding the quotation mark characters surrounding the string.

If the remote query had the following response:

prin:head1?
"This is the first line of header text"

then any occurrence of <$QUERY> following prin:head1? is replaced by: This is the first line of header text.

To place this line at the start of line 42 of the printer page buffer use:

prin:head1?
prin:str 42,1,"<$QUERY>"

$SERIALIN The serial input characters read by the SerialIn command.

$STRING The character string that was defined by the last SaveString command.

$STABLE The character string in the current program sequence file at the location defined by the SetTableSelect and SetTableLineIndex commands.

The following macros represent numeric integer values.

When expanded, they are converted into the shortest length decimal string possible (i.e. no leading zeroes or spaces, and no decimal point).

These macros cannot be used with the optional offset and length values (NUM_0 entries) that are used to access portions of the macro strings.

$ERRCODE The last SCPI error code that occurred.

$PASSFAIL The total number of tests executed. (SCPI measurement and call processing commands are considered tests.)

When one of these commands executes, the test name and its result is displayed on the LCD.

Logging information (measurement value and pass/fail status) is saved in memory for these commands and can be saved-to-disk or printout, and the total test counter value is incremented.

If the test failed, the total failures counter value is incremented. <$PASSFAIL> reports the total tests counter value; this is displayed on the LCD when the program sequence completes.

$FAIL The number of tests that have failed.

This reports the value of the total failures counter described in the $PASSFAIL description.

$PASS	The number of tests that have passed. This reports the value ($PASSFAIL - $FAIL) described in the $PASSFAIL description.
$RETCODE	The last SCPI error extended return code that occurred.
$SERIALLEN	The length of the string contained in the macro $SERIALIN.
$STRLEN	The length (number of characters) of the string contained in the macro $STRING.

Numeric Parameter Commands

Numeric parameter commands define parameters used by the program sequence in place of numeric values which allows a program to pull specific program parameters to the top of the file, making adjustments or modifications to these values easier.

They also modify parameters in a loop, pass values to subroutines, and to return values from subroutines. Global parameter names should be unique in the file. If a second definition of the same global parameter is made, the parameter value is changed rather than creating a new parameter. If a parameter has the same name as a loop parameter, the sequence uses the global parameter name when substituting a numeric value instead of the loop parameter.

Define Global <label> <numeric>

defines a numeric parameter by the name of <label> and assigns it the value <numeric>.

If the parameter is not defined, it is created. If it is defined, it modifies the value of the previous definition.

These parameters can be passed to subroutines (as arguments) and returned from subroutines. A maximum of 50 global parameters can be defined in a sequence.

The rules for parameter names is the same as for labels: 1 - 40 characters, upper and lower case letters plus numbers 0 - 9, the underscore character, no spaces are permitted within the name, and the upper/lower-case is preserved.

Example:

```
DefineGlobal   StartLoop 10
DefineGlobal  EndLoop   50
prin:sel:line 1
FOR LoopIndex = <StartLoop>, <EndLoop>
DefineGlobal  Param_1<LoopIndex>
DefineGlobal  Param_2  12
GOSUB Sub_DisplayIndex
NEXT
END
(Sub_DisplayIndex):
prin:sel:col <Param_2>
```

prin:sel:str "<Param_1>"
prin:sel:line:incr
RETURN
passes global parameters to a subroutine.

This code prints the values 10 to 50, one value per line, at column 12.

ModifyGlobal <label> operation <numeric>

operation	<numeric>
+	add numeric to the parameter value
-	subtract numeric from the parameter value
*	multiply the parameter value by numeric
/	divide the parameter value by numeric
%	perform modulo numeric arithmetic on the parameter
&	perform bitwise AND of numeric with the parameter value
|	perform bitwise OR of numeric with the parameter value
ˆ	perform bitwise EXCLUSIVE-OR of numeric with the parameter value

modifies a previously defined global parameter. If the parameter is not already defined, a program error will be reported.

For the modulo operator (%=), the parameter value and modulo base are first truncated to integer values (since this is an integer-only operation). The operation X modulo Y is executed as follows:

4. Y is successively subtracted from X (or added to if X < 0) until the absolute value is less than Y.
5. For positive values of X, this is equivalent to the remainder after X is divided by Y. This remainder value will always be greater than or equal to zero and less than Y.
6. For negative values of X, the result will be the same magnitude as if X was positive, but will always be negative or zero.

Modulo base values less than +2 have no real meaning, however performing a%=1 may be used for truncating a floating point value to an integer. It sets the global parameter to the largest integer value less than the (absolute) floating-point value.

The &=, |=, and ˆ= are all unsigned integer operations.

When these commands are used, both the global parameter and the numeric operator values are converted to 32-bit unsigned integer values prior to performing the operation. Floating point values are rounded to the closest integer value. If the decimal portion of the value is 0.5, it is rounded up to the more positive value.

Values that exceed the maximum positive 32-bit unsigned value are set to a 4,294,967,295 limit. Negative values are converted to their 2's complement form by subtracting from 4,294,967,296.

Converting a value to an unsigned 32-bit integer can be accomplished by using ModifyGlobal <param> |= 0.

Example:

```
DefineGlobal    TestValue    127.832#    = 127.832
ModifyGlobal    TestValue    += 17.34#    = 145.172
ModifyGlobal    TestValue    *= 3.14159#    = 456.0709
ModifyGlobal    TestValue  %= 1#    = 456
ModifyGlobal    TestValue    /= 12.0#    = 38
ModifyGlobal    TestValue    -= 61#    = - 23
ModifyGlobal    TestValue  %= 6#    = - 5
```

modifies a global parameter value.

Comments to the right of the commands indicate the value that the global value has after the command is executed.

AssignGlobalQuery <label>

obtains the value of the last numeric parameter queried and assigns it to a previously defined global parameter.

If the parameter is not defined, it is created.

AssignGlobalMeas <label>

obtains the value of the last measurement performed by a *Measure* subsystem command (or read by a *Read* subsystem command) and assigns it to a previously defined global parameter.

If the parameter is not defined, it is created.

If the last measurement was not successfully completed, the data is tagged as invalid, and this command returns an error (-40).

To prevent this error, precede the command with a `BranchIfInvalid` which handles the error case first.

Example:

```
FOR LoopIndex = 1, 1000
GOSUB NextChannel
meas:freq:err:rf?
AssignGlobalMeas   RF_freq_err     # read the meas value
CompareGlobal      RF_freq_err 3 # compare it to 3 (Hz)
BranchIfLE NoError                 # exit loop if >
BREAK
(NoError):
NEXT
......
# NextChannel
# performs a handoff to the next channel to test
```

```
#
NextChannel):
call:chan?
AssignGlobalQuery Channel   # read current channel setting
ModifyGlobal Channel += 10 # increment it by 10
CompareGlobal Channel 666   # compare to 666
BranchIfLE    Chan_OK# if > 666, reset it to 1
DefineGlobal Channel 1
(Chan_OK):
call:hoff:chan <Channel>; imm
RETURN
```

performs up to 1000 handoffs, incrementing the channel by 10 each time, and resetting to 1 if 666 is exceeded.

This sequence also performs an RF frequency error measurement for each channel, and aborts the loop if the error exceeds 3 Hz.

SetMemoryFloat available for CDMA only.

SetMemoryShort available for CDMA only.

SetMemoryInteger available for CDMA only.

SetMemoryLong available for CDMA only.

String Parameter Commands

String parameter commands define strings that are used by the program sequence in place of a discreet string definition.

This allows a common string value to be defined once and used multiple times. It also allows numeric values to be converted into strings.

SaveString <string_list> saves and joins string definitions; the resulting string can be accessed using the macro $STRING.

Example:

```
# test string definition command 'SaveString' and macro
$STRING.
#
DefineGlobal    PiValue     3.14159265359
NumToStr    <PiValue>, 6
SaveString "String ", "concatenation: ", "PI = ", <$NUMSTR>,
13, 10
SerialOut <$STRING>, 13, 10
```

SerialOut “The above string is “,<$STRLEN>,“chars in length.”, 13, 10
produces the following strings out the serial port (terminated by carriage return and linefeed):

String concatenation: PI = 3.141592
indicates that the string is 37 characters in length.

SaveScpiCmd saves the last SCPI command header executed internally by the sequencer, so that it can be accessed by the <$LASTCMD> string alias expansion, as well as the CompareScpiCmd command.

NOTE: The command header does not include any parameters associated with the command. This feature allows the Error Handler code to determine which SCPI command caused the error.

NumToStr <numeric1>, [numeric2]
<numeric1> the value to convert to a string
[numeric2] (optional) the number of decimal digits (up to 15) to use

converts the specified numeric value (or parameter) into a string.

Without a second argument, the value is truncated to an integer value.

With a second argument, the value indicates the number of decimal digits to use in the conversion. Access the resulting string using the macro $NUMSTR.

If the absolute value of <value> is greater than or equal to 1,000,000,000, the data is converted to exponential format (e.g. 1.234567e+15), otherwise it is converted to standard floating point notation (e.g. 1723.3333).

Example: DefineGlobal PiValue 3.14159265359
FOR loop = 0, 6
NumToStr <PiValue>, <loop>
prin:sel:str “<$NUMSTR>”
prin:sel:line:incr
NEXT
generates data in the page buffer as follows:
3
3.1
3.14
3.141
3.1415
3.14159
3.141592

HexToStr <numeric1>[numeric2]
<numeric1> value to convert to a string
[numeric2] (optional) number of hexadecimal digits (up to 8) to use

converts the specified numeric value (or parameter) into a hexadecimal string.

If the optional second argument is given, the value indicates the number of hexidecimal digits to use.

If this parameter is omitted, the default is 4 digits. Access the resulting string using the macro $NUMSTR.

Example: DefineGlobal HexData 2345
HexToStr <HexData>, 6
prin:sel:str "<$NUMSTR>"
generates data in the page buffer as follows:
000929

SetMemoryString <numeric integer>
<numeric integer>value from 1-20

allows the user to query a previously saved ASCII string in non-volatile memory.

Table Parameter Commands

Table parameter commands locate a table within the program sequence and extract string and numeric information to use as parameter values elsewhere in the program.

SetTableSelect <label>

specifies the name of the current table selection.

A table is defined as a specific label, followed by lines of text that make up the table.

Place the table at the end of the file following the END statement (before or after any subroutines).

Access the table information using the string designation <$TABLE,x1,x2> where x1 and x2 are the corresponding character offset and length. Select the line position in the table using SetTableLineIndex.

SetTableLineIndex <numeric>

specifies which line offset (0-999) accesses the table selected by the SetTableSelect command.

NOTE: There is no check performed on this value to determine if it exceeds the end of the table. The first line is specified by a value of 0. Negative numeric values are interpreted as 0.

Example: # Test setup conditions (IS-98A specified tests)

```
# set test table pointer
SetTableSelect TestSetupTable
# get offset to the selected test info
# (Test = value 1 to 12)
DefineGlobal TestIndex <Test>
SetTable LineIndex <TestIndex>
DefineGlobal Ior_Ioc <$TABLE, 8,5>
DefineGlobal Ioc <$TABLE,16,5>
DefineGlobal Pilot <$TABLE,24,5>
DefineGlobal Traffic <$TABLE,32,5>
DefineGlobal Rate <$TABLE,40,5>
DefineGlobal FerLimit <$TABLE,48,5>
DefineGlobal ConfFact <$TABLE,56,5>
DefineGlobal MaxFrames <$TABLE,64,5>
```

Test	Ior/ Ioc	Ioc	Pilot	Traffic	Rate	%FER	Conf	Max Test1
1	-1.0	-54.0	-7.0	-16.3	9600	3.0	95.0	5000
2	-1.0	-54.0	-7.0	-15.8	9600	1.0	95.0	5000
3	-1.0	-54.0	-7.0	-15.6	9600	0.5	95.0	5000
4	-1.0	-54.0	-7.0	-19.1	4800	1.0	95.0	5000
5	-1.0	-54.0	-7.0	-21.6	2400	1.0	95.0	5000
6	-1.0	-54.0	-7.0	-24.5	1200	1.0	95.0	5000
7	-1.0	-54.0	-7.0	-13.0	14400	0.5	95.0	5000
8	-1.0	-54.0	-7.0	-12.7	14400	1.0	95.0	5000
9	-1.0	-54.0	-7.0	-12.4	14400	0.5	95.0	5000
10	-1.0	-54.0	-7.0	-17.3	7200	1.0	95.0	5000
11	-1.0	-54.0	-7.0	-20.8	4300	1.0	95.0	5000
12	-1.0	-54.0	-7.0	-24.4	1800	1.0	95.0	5000

Compare Commands

compare a program parameter, loop value, SCPI query command response value, or measurement value to another value for use by a branch statement.

A conditional branch statement follows the compare statement.

These two statements allow the program flow to change based on run-time conditions, rather than fixed loops.

CompareLoop <label> <numeric>

compares the specified loop index parameter to a numeric value.

The results of the comparison are used by `BranchIfEQ, NE, LT, . . .` directives to modify the program flow.

Use this command inside the loop containing specified loop index name; it is valid within nested loops and subroutines that are called from within the loop.

Example: FOR ChannelSelect = 1, 1023
CompareLoop ChannelSelect 800
BranchIfLT ChannelValid
CompareLoop ChannelSelect 989
BranchIfLE ChannelInvalid
(ChannelValid):
call:hoff:chan <ChannelSelect>
call:hoff:imm
(ChannelInvalid):
NEXT
performs handoffs for channels 1 to 1023 (excluding 800-990) in
increments of 1.

CompareGlobal <label> <numeric>

compares a global parameter to a numeric value.

The results of the comparison are used by `BranchIf` directives to modify program flow.

CompareLastMeas <numeric>

compares the last measurement value to a fixed numeric value.

The results of the comparison can then be used by `BranchIf` directives to modify program flow.

Precede this command with a measurement command.

NOTE: The comparison is not affected by whether the limits are enabled/disabled.

A -40 "Disk BIOS" error is issued if the last measurement is invalid. To prevent this, precede the command with a `BranchIfInvalid` to handle the error case first.

Example: DefineGlobal MaxReadings100
(StartOfLoop):
meas:dev:aud?
CompareLastMeas 10000
BranchIfGE ExitLoop
ModifyGlobalMaxReadings -= 1
CompareGlobalMaxReadings 0
BranchIfGT StartOfLoop
(ExitLoop):
performs continuous audio deviation measurements until either a value of 10000 Hz is obtained, or 100 samples have been taken.

CompareNumericQuery <numeric>

compares the last numeric parameter queried to a fixed numeric value.

The results of the comparison can then be used by the `BranchIf` directives to modify program flow.

Example:
```
FOR Loop = 1, 100
    call:hoff:chan <Loop>; imm;
    call:chan?
    CompareNumericQuery <Loop>
    BranchIfEQ No_Problem
    prin:sel:str “HANDOFF ERROR ON CHANNEL <Loop>"
BREAK
    (No_Problem):
    NEXT
```
performs handoffs to channels 1 - 100 and checks to see if the channel is correct. If not, it prints an error and exits the loop.

CompareEnumQuery <label>

compares the last enumerated parameter queried to a specific value.

The result of the comparison is used by the `BranchIfEQ` and `BranchIfNE` directives to modify program flow.

<label> is the character string to compare the enumerated value to. Enumerated responses from the remote are always in short form (usually 4 characters or less) and are always in uppercase; the values required for the comparison are the full enumerated version.

For instance, a data rate query might respond with an enumerated value of QUAR that is the short form of QUARter. With this command, specify QUARter, with the first 4 characters uppercase and the others lowercase.

Example:
```
call:hoff:type NAMPs
    FOR Loop = 1, 100
(Restart_Loop):
    call:hoff:chan <Loop>; imm
    call:type?
            CompareEnumQuery NAMPS
    BranchIfEQ No_Problem
    call:page NAMPS
    prin:sel:str “HANDOFF ERROR ON CHANNEL <Loop>”
            Branch Restart_Loop
(No_Problem):
            NEXT
```
determines if the call has been dropped during an NAMPs handoff test, and if so, re-establishes the call by paging the mobile to a NAMPs channel and continues the test.

CompareStringQuery [numeric], <string>

compares the last string response from a SCPI query command to a specific <string> value.

The optional [numeric] value specifies a character offset to use on the response string. If omitted, the offset value is 0, and the comparison is performed starting with the first character of the query response string.

The result of the comparison is used by the Comparison Branch commands using the following logic:

1. Compare the character array response to string one character at a time (starting from the left) until a character mismatch occurs at character position x.
2. If no mismatch occurred, the = flag is set. If response[x] < string[x], the < flag is set
3. If response[x] > string[x], the > flag is set

Characters are evaluated as printable ASCII values (decimal values 32 to 127). The comparison is case sensitive, and the leading and trailing double quote characters are not included in the comparison.

The number of characters compared is determined by the length of <string>. Only the number of characters specified in this string are compared. The maximum string length that can be compared is 80. A field of length zero (two double quotes back to back) compares the specified offset location in the string to zero, which is placed at the end of the string. This determines if the end of the string has been reached.

Example: PRIN:HEAD1 "This is a simple test string."
PRIN:HEAD1?
DefineGlobal CmpIndex 5
is sent prior to the comparison commands.

NOTE: An embedded quote character must be preceded by another quote character to delimit it.

CompareStringQuery "This"
CompareStringQuery "This is a simple test string."
CompareStringQuery 10, "simple test"
CompareStringQuery <CmpIndex>, "is"
CompareStringQuery 30, ""
these comparisons are TRUE and set the branch status flag to branch if the **BranchIfEQ** directive follows the command.

The first two commands omit the offset value; the comparisons are performed from the start of the header string response. The first comparison only compares the first 4 characters of the string, while the second compares all 29 characters.

The third comparison illustrates the use of the offset value to compare a section of the string that begins with the 11th character (1st character = offset of 0) of the string. Eleven characters are compared.

The fourth comparison illustrates that a string response may be examined character or word at a time in a loop.

The last comparison is a test for the end of the string. No characters are placed between the two double quote characters, indicating a zero length string. The 30th character location in the string is tested for a zero (NULL). This indicates that the string is 29 characters long.

NOTE: Characters past the first NULL entry should not be tested, since they are not a part of the string.

CompareScpiCmd [numeric], <string>

compares the last SCPI command captured by the SaveScpiCmd command to a specific <string> value.

The optional [numeric] value specifies a character offset to use on the SCPI command string. If omitted, the offset value is 0, and the comparison is performed starting with the first character of the string. The results of the comparison can then be used by the Comparison Branch commands using the same logic as the CompareStringQuery command.

The string data is case sensitive, and the leading and trailing double quote characters are not included in the comparison. The number of characters compared is determined by the length of <string>. Only the number of characters specified in this string are compared. The maximum string length that can be compared is 80. The string comparison length is determined by the number of characters between the first and last double quote characters found.

This command is used by the error handler code to determine the command that caused the error.

See the sequencer error handler section for an example of this command.

CompareScpiError <numeric>

compares the last SCPI error code to the <numeric> value given.

If the value specified is < -499 or > -100, the SCPI error extended return code is used for the comparison instead.

NOTE: All errors are negative values.

SCPI error code -100 to -499
extended return code -1 to -99 (fatal) system errors
specific execution failure -500 to -999

NOTE: The return code value is a more specific error than the SCPI error code; many times the return code is 0 when the SCPI error code is not, however each return code value corresponds to a specific SCPI error code.

Branch Commands

Branch commands transfer program execution to a specified label in the sequence, based on a specific condition; this allows a program to alter its operation if a specific condition exists at the time the command is executed.

The types of branch statements (based on the type of condition used to determine whether to branch) include:

1. **Unconditional branches** always branch.
2. **System branches** determine what options (hardware and/or software) are installed in the 4300, and branch based on system capabilities. System branches do not require any setup conditions in order to function.
3. **Mobile type branches** determine the type of mobile that was last registered (through a previous registration or call origination command), and branch based on the capabilities of the mobile. Mobile type branches require that either a registration or an origination of the mobile being tested has successfully completed prior to the branch command.
4. **Call processing branches** determine the current call processing state and branch based on characteristics of that state.
5. **Boolean branches** determine the status of the last SCPI query that returned a boolean parameter and branch accordingly. Boolean branches require that the SCPI query, that returns a boolean response, preceded the branch command.
6. **Comparison branches** determine the results of a numeric (or enumerated) comparison and branch accordingly. The parameters that can be tested for branching consist of any SCPI query command that returns either a numeric or enumerated value, or a sequence global or loop parameter. Comparison branches require that one of the Compare directives has been executed prior to the branch command. Sequencer directives may be executed between the query command and the branch statement without affecting the branch condition.
7. **Measurement branches** determine the status of the last measurement completed and branch accordingly. When a measurement command is executed, the upper and lower limit values (and whether they are enabled) for the measurement are examined to determine if the measurement passed, failed, was not tested or did not complete. Measurement branches require that either a *Measurement* subsystem or a *Read* subsystem command is executed prior to the branch. Sequencer directives and non-measurement commands may be executed between the measurement command and the branch statement without affecting the branch condition.

Unconditional Branches
Branch <label>
GoTo <label>

redirect program execution to the specified label; the branch is always taken.

System Branches
BranchIfDcmOption <label>
BranchIfNoDcmOption <label>

determine if the DCM option board is installed in the 4300, which is required for TDMA.

The sequencer jumps to the label specified if the option is installed (for `BranchIfDcmOption`) or not installed (for `BranchIfNoDcmOption`). If the branch is not taken, the sequencer executes the next command in the sequence which allows the user to skip a section of code for TDMA testing, if the option is not installed.

BranchIfCcmOption<label>
BranchIfNoCcmOption<label>

These commands determine if the CCM option board is installed in the 4300, which is required for CDMA testing.

The sequencer jumps to the label specified if the option is installed (for `BranchIfCcmOption`) or not installed (for `BranchIfNoCcmOption`). If the branch is not taken, the sequencer executes the next command in the sequence which allows the user to skip a section of code for CDMA testing, if the option is not installed.

BranchIfFexOption<label>
BranchIfNoFexOption<label>

These commands determine if the FEX option board is installed in the 4300, which is required for PCS band testing.

The sequencer jump to the label specified if the option is installed (for `BranchIfFexOption`) or not installed (for `BranchIfNoFexOption`). If the branch is not taken, the sequencer executes the next command in the sequence which allows the user to skip a section of code for PCS band testing if the option is not installed.

BranchIfCdma2000Option<label>
BranchIfNoCdma2000Option<label>

These commands determine if the CMDA2000 option board is installed in the 4300, which is required for IS-2000 CDMA testing.

The sequencer jump to the label specified if the option is installed (for `BranchIfCdma2000Option`) or not installed (for `BranchIfNoCdma2000Option`). If the branch is not taken, the sequencer executes the next command in the sequence which allows the user to skip a section of code used for IS-2000 CDMA testing, if the option is not installed.

Mobile Type Branches
BranchIfAmps <label>
BranchIfNotAmps <label>
BranchIfNamps <label>
BranchIfNotNamps <label>
BranchIfTdma <label>
BranchIfNotTdma <label>
BranchIfCdma <label>
BranchIfNotCdma <label>

The sequencer jumps to the command following the specified label in the sequence if the mobile is the specified type.

If the mobile is not the specified type, the sequencer executes the next command in the sequence.

The mobile type is partially derived from the Station Class Mark sent by the mobile during a registration.

Additional information for CDMA phones is determined by status messages (requested during registration, page and origination).

BranchIfClass1<label>
BranchIfClass2<label>
...
BranchIfClass7<label>
BranchIfClass8<label>

The sequencer jumps to the command following the specified label in the sequence if the mobile is the specified power class.

If the mobile is not the specified power class, the sequencer executes the next command in the sequence.

The power class is derived from the Station Class Mark sent by the mobile during a registration.

NOTE: Roman numerals can be substituted for the class number (e.g. `BranchIfClassVIII` = `BranchIfClass8`).

BranchIfRCClass1<label>

The sequencer jumps to the command following the specified label location in the sequence if the radio configuration (RC) class of the current call (or service mode) is 1.

If the mobile is not on a CDMA2000 call (or in service mode), or if the current RC class is not 1, the branch is not taken.

NOTE: RC class 1 = forward RC 1 and 2 and reverse RC 1 and 2.

BranchIfRCClass2<label>

The sequencer jumps to the command following the specified label location in the sequence if the RC class of the current call (or service mode) is 2.

If the mobile is not on a CDMA2000 call (or in service mode), or if the current RC class is not 2, the branch is not taken.

NOTE: RC class 2 = forward RC 3, 4, 5 and reverse RC 3 and 4.

BranchIfChan666<label>
BranchIfChan832<label>

The sequencer jumps to the command following the specified label in the sequence if the mobile has the channel capability specified (666 if only channels 1-666 are allowed, 832 if channels 1-799 and 991-1023 are allowed).

If the mobile does not have the specified channel capability, the sequencer executes the next command in the sequence.

The bandwidth is derived from the Station Class Mark sent by the mobile during a registration.

BranchIfContinuous<label>
BranchIfDiscontinuous<label>

The sequencer jumps to the command following the specified label in the sequence if the mobile is the specified transmission type.

If the mobile is not the specified transmission type, the sequencer executes the next command in the sequence.

The power class is derived from the Station Class Mark sent by the mobile during a registration.

Call Processing Branches
BranchIfCtrlChan<label>
BranchIfVoiceChan<label>

branches based on whether or not the mobile is currently up on a voice (traffic) channel or on a control channel.

Boolean Branches
BranchIfOn<label>
BranchIfOff<label>

used immediately following a query of a boolean parameter.

The sequencer jumps to the label specified if the last boolean parameter queried was `ON` (for `BranchIfOn`) or `OFF` for `BranchIfOff`).

If the branch is not taken, the sequencer executes the next command in the sequence.

Example: meas:status:st:condition?
BranchIfOn ST_IsOn
print:str 1, 20, "ST OFF"
Branch NextTest
(ST_IsOn):
print:str 1, 20, "ST ON"
(NextTest):
outputs the string "ST OFF" if the signaling tone is not detected, and "ST ON" if it is.

Comparison Branches
BranchIfEQ
BranchIfNE
BranchIfGT
BranchIfGE
BranchIfLT
BranchIfLE

used immediately following a Compare directive.

The sequencer jumps to the label specified if the branch condition matches the comparison performed.

For the CompareEnumQuery directive, use only the **`BranchIfEQ`** or **`BranchIfNE`**, since "greater than" and "less than" have no meaning for enumerated values.

The branch conditions and the corresponding comparison conditions for which they branch include:

EQ<parameter>EQUAL TO <value>
NE<parameter>NOT EQUAL TO <value>
GT<parameter>GREATER THAN <value>
GE<parameter>GREATER THAN OR EQUAL TO<value>
LT<parameter>LESS THAN<value>
LE<parameter>LESS THAN OR EQUAL TO<value>

Example:
call:type?
CompareEnumQuery AMPS
BranchIfEQ UpOnACall
call:orig AMPS
(UpOnACall):
DefineGlobal Level 0
FOR Channel = 1, 100
call:hoff:chan <Channel>; vomit <Level>; imm
CompareLoop Channel 50
BranchIfLT MacIsSet
DefineGlobal Level 1
(MacIsSet):
NEXT
originates an AMPS call if not already on a call, and performs 50
handoffs at MAC 0 and 50 at MAC 1.

Measurement Branches
BranchIfPass <label>

branches only if the upper or lower limit for the measurement was enabled and the measurement value did not exceed any of the limits.

BranchIfFail<label>

branches only if the upper or lower limit for the measurement was enabled and the measurement value exceeds one of the limits.

BranchIfNotTested<label>

branches only if both upper and lower limits are disabled for the measurement or if the measurement did not complete successfully.

BranchIfInvalid<label>

branches only if the last measurement did not complete successfully.

```
Example:  alc:lim:type EIA
            DefineGlobal Pass   0
            DefineGlobal Fail   0
                        FOR MacLevel = 0, 7
            call:plch:vmac <MacLevel>
            meas:pow:tran?
            Gosub   TallyPassFail
            NEXT
            prin:str   10, 1, "Number of MAC levels passed: <Pass>"
            prin:str   11, 1, "Number of MAC levels failed:    <Fail>"
                       (TallyPassFail):
        BranchIfPass   IncrPass
        BranchIfFail    IncrFail
        Branch            Done
           (IncrPass):
        ModifyGlobal Pass + = 1
        Branch            Done
                        (IncrFail):
        ModifyGlobal Fail + = 1
           (Done):
        RETURN
```

performs power measurements at MAC levels 0 - 7 using the EIA specification for limit checking.

The number of measurements that pass and that fail are counted and printed.

Program Control Commands

Program control commands modify the sequential flow of a program in a fixed pre-determined fashion.

The normal program flow is to execute the next line in the program when the current one is done. Often, programs need to repeat similar actions. Loops and subroutines "package" the repetitive commands, which can then be executed as many times as necessary.

"Program nesting" is similar to subroutines. Other program sequences can be loaded in a temporary location and executed as subroutine from within another program.

FOR <label> = <numeric1>, <numeric2>, [numeric3]

numeric1	the initial value of the loop parameter
numeric2	the value of the loop parameter at which to terminate
numeric3	(optional) the value to increment the loop parameter by (+ or -)

begins a loop and defines a loop index parameter.

The name of the loop parameter uses the same definition as label definitions.

The loop value is a floating point quantity with a minimum/maximum absolute value of 1.0e+10 and a resolution of 0.000001. Loops may be nested within each other up to 40 deep.

The loop parameter may be used in place of numeric values in SCPI commands (similar to global parameters), providing it is not used outside the loop.

The increment value is optional. If omitted, it is set to +1 with a positive value indicating incrementation and a negative value indicating decrementation (zero is invalid).

The following loop termination test is performed at the start of each loop:

- if STEP is positive, the loop terminates when the loop value > STOP value.
- if STEP is negative, the loop terminates when the loop value < STOP value.

NEXT increments the loop parameter of a FOR loop by the step value, and returns to the line following the FOR directive.

For nested loops, it always returns to the closest FOR directive preceding it.

If the step value is positive and the current loop value is > the stop value, the loop is complete and program execution continues at the line following the NEXT command.

If the step value is negative, the loop completes when the current loop value < the stop value.

Example: FOR Channel = 10, 600, 10
call:hoff:chan <Channel>; imm
FOR MacLevel = 0, 7

call:plch:vmac <MacLevel>
meas:pow:tran?
NEXT
NEXT
This nested loop performs handoffs for every 10 channels from 10 to 600 and power measurements at all MAC levels 0 - 7 for each channel.

NOTE: The step argument inside loop is omitted, as it defaults to +1.

BREAK terminates a FOR loop prior to the loop index reaching its end value.

Program execution continues with the command following the NEXT directive that corresponds to this loop.

Example: FOR Channel = 1, 10
call:hoff:chan <Channel>; imm
BranchIfPass LoopOK
BREAK
(LoopOK):
NEXT
performs handoffs to channels 1 to 10, but aborts the loop if any of the handoffs fail.

GOSUB <label>

starts execution at the specified label, and returns to the command after this GOSUB command, when a RETURN directive has been executed.

NOTE: Subroutines may be nested up to 40 deep, and may contain FOR loops within them.

RETURN

returns control to the line following the GOSUB command that passed control to this subroutine.

END

terminates the current sequence.

If this was a nested sequence (see `RUNPROG`), control is passed back to the original sequence. Otherwise, the program sequence ends.

Use this directive in all sequences to avoid an error flag when the last line is executed. Place at the end of the main routine of the sequence preceding any subroutines to prevent the main routine from continuing execution into the subroutines.

RUNPROG [program select]
[program select] (optional)
AUTO, QUICK, CUSTOM1,
CUSTOM2, CUSTOM3

starts execution of another sequence, similar in operation to GOSUB.

All defined global parameters are maintained by the new sequence; the label definitions or loop parameters are not.

When the END command has been executed by the auxiliary sequence, control returns to the original sequence following this `RUNPROG` directive.

NOTE: This is referred to as program nesting; only one level of program nesting is allowed.

If `RUNPROG` is used by itself, a special sequence is run which must first be downloaded from disk (using `DISK:LOAD:PROG)` while running the sequence from which it is to be run, prior to the `RUNPROG` command.

The second form of the command runs the specified program sequence.

CAUTION ! Do not nest the current running program (i.e. If running `QUICK`, do not execute `RUNPROG QUICK`).

Serial Port Control Commands

SerialRemoteDisable

disables the serial input connector from interrupting the test sequence.

Any characters received after this command are read using `SerialIn`.

Cancel this command using `SerialRemoteEnable` when the program terminates.

SerialRemoteEnable

re-enables the serial input connector for remote input use following either the `SerialRemoteDisable` or the `SerialIn` command.

This function is performed automatically when the program terminates.

SerialOut <string_list>

outputs the specified string(s) and/or characters to the serial port.

Multiple strings can be combined with multiple characters on a single command line.

The characters are expressed in decimal, and allow control characters and other non-printable characters to be output.

NOTE: This command does not disable the remote input.

Example:

```
# Test use of string macros with SerialOut command
        PRINt:SELected:COL:TAB:DEFine 10, 20, 30
        SaveScpiCmd
        SerialOut "last SCPI cmd:         ", <$LASTCMD>, 13, 10
        SerialOut "(offset 6, len 20):    ", <$LASTCMD,6,20>, 13, 10
        *IDN?
        SerialOut "*IDN? response:        ", <$QUERY>, 13, 10
        SerialOut "(offset 3, len 10):    ", <$QUERY,3,10>, 13, 10
        INP:AUD?
        SerialOut "INP:AUD? response: ", <$ENUM>, 13, 10
        SerialOut 13, 10
```

outputs the following strings to the serial port:

```
last SCPI cmd:        PRINt:SELected:COL:TAB:DEFine 10, 20, 30
        (offset 6, len 20):    SELected:COL:TAB:DEF
```

```
*IDN? response:      WVTK,4300,12345678,4.3
(offset 3, len 10):  K,4300,12
INP:AUD? response:   DIR
```

SerialIn <numeric1>, <numeric2>, <numeric3>

<numeric1>	The max number of seconds to wait for a response. This value ranges from 0 to 60, where 0 or less = wait forever.
<numeric2>	The max number of chars to wait for before terminating. A value of 0 or less = do not terminate until the buffer fills (buffer size is 255).
<numeric3>	The decimal value (0 to 127) of the character to terminate on. A value of -1 = do not terminate due to data value.

reads from the serial input port until one or more of the specified conditions occurs.

No further commands are executed by the sequencer until the terminating condition is met.

Each of the terminating conditions (with the exception of the buffer full condition) can be disabled.

This command automatically disables serial input used for remote controlling the 4300. Re-enable serial remote when the sequence completes, using SerialRemoteEnable.

The response string can be accessed using the macro $SERIALIN.

Example:

```
# Test serial input command
SerialOut "Please enter 10 chars", 13, 10
SerialIn -1, 10, -1
SerialOut "The ", <$SERIALLEN>, "chars you typed were: "
SerialOut <$SERIALIN>, 13, 10

SerialOut "Now type a string ending with a carriage return",
13, 10
SerialIn -1, -1, 13
SerialOut "The ", <$SERIALLEN>, "chars you typed were: "
SerialOut <$SERIALIN>, 13, 10

SerialOut "Now type more chars within 10 secs", 13, 10
SerialIn 10, -1, -1
SerialOut "The ", <$SERIALLEN>, "chars you typed were: "
SerialOut <$SERIALIN>, 13, 10
```

communicates with a simple ASCII terminal to prompt for character input, and resends the string along with the number of characters received in each string.

The first SerialIn command places a limit on the total characters received; the command terminates when 10 characters are received.

The second SerialIn command uses a terminating character to terminate the SerialIn command. The 13 (decimal) represents an ASCII carriage return, and the program waits for serial input until this character is received.

The last SerialIn command specifies the timeout value, and terminates after 10 seconds, whether any serial input is received or not.

NOTE: Combinations of terminating conditions can be specified. All of the terminating conditions can be disabled, except for the number of characters.

NOTE: This command terminates if the buffer fills (255 characters received).

SerialPageLine

outputs the current line selection (using `PRINt:SEL:LINE)` and any printer control characters, from the page buffer to the serial port.

Example:

```
FOR Count = 1, 6
prin:sel:line <Count>
SerialPageLine
NEXT
```

outputs page buffer lines 1 - 6 to the serial port.

Other Commands

Delay <numeric>

is similar in form and function to `PROGam:DELay`.

The delay value <numeric> is specified in seconds, has a maximum value of 10, and a resolution of 0.01 s.

SetTestPass <string>
SetTestFail <string>

logs a test and whether it passed or failed.

The <string> is used as the name of the test placed on the LCD display and in the log header. The number of tests failed and total tests are incremented appropriately.

The total pass/fail, cumulative pass/fail, and the branch flags are set so that **`BranchIfPass`** and **`BranchIfFail`** can be used after this command.

ClearLastTest removes the last test from the log and decrements the total number of tests value.

If the last test failed, the test is removed from the test failures log and the number of test failures is decremented.

The pass/fail flags (total & cumulative) only change if only one test has been performed prior to the command and the flags reset to NOT TESTED.

ClearErrorCode sets the SCPI error code value to 0. (This is performed automatically when returning from the error handler.)

Sequencer Error Handling

When an error is encountered in executing a sequence, the normal action taken is to enter the PAUSE mode, where soft key selection such as CONTINUE and ABORT allow the user to take the appropriate actions.

The error condition and the sequencer command that caused the error are placed in the message window (in inverse video mode) near the bottom of the LCD display, to help identify the cause of the error.

Two types of sequencer errors can occur:

- sequencer directive errors
- SCPI command errors.

Sequencer directive errors indicate errors in the sequence itself; corrective action requires sequence modification in order to operate properly. Sequencer directive errors can also indicate a hardware failure. The sequence directive errors are listed in the next section.

SCPI command errors result from certain conditions of the mobile under test. Write an error handler by defining a special subroutine at the bottom of the sequence to perform any necessary actions.

Without an error handler, all SCPI errors are handled the same as sequencer directive errors.

With an error handler, all SCPI errors are routed to the error handler, where special functions are executed based on the error that occurred, the command that was executed, or any other condition that can be queried from the test equipment.

Special sequencer directives can only be used in this error handler routine; normal sequencer directives are also allowed, along with any SCPI command.

NOTE: If an error occurs with a SCPI command within the error handler routine, it is treated like a sequencer directive error and enters the PAUSE mode. For this reason, care must be used in constructing the error handler when using the SCPI commands, since errors from these cannot be recovered from without user intervention (pressing the CONTINUE or ABORT keys).

If an error handler is added to a program sequence, it must be the last function in the file. No subroutines, comments or definitions are allowed after the error handler. Whitespace (spaces, tabs, linefeeds) is the only item that may be placed after the error handler definition. This is because when an error occurs, the sequence is searched backwards from the end of the file, and the first word found must be the%ERROR_EXIT directive. If found, then the search is continued until either the%ERROR_HANDLER or the END directive is found. If%ERROR_HANDLER is found first, execution will begin from that point. If not found (or END was encountered first), the normal PAUSE on error condition is performed. At the end of the error handler, execution can either be resumed on the command that caused the error, or on the following command.

Upon return from the error handler function, the SCPI error code and the extended error code values are both cleared to 0.

The following commands control how a SCPI command error is handled. They can be placed anywhere in the program in order to change the error handling mode.

There are three possible actions that can be taken: stop the program, ignore the error, or perform some special action. The last case requires that the program has an error handler function defined within it, while the first two do not.

ScpiErrorHandler

causes SCPI errors to be handled by the error handler, if one is defined.

If no error handler is defined, errors force the sequence to enter the PAUSE mode and display the error that occurred.

This is the default setting when first beginning a sequence.

ScpiErrorPause

causes any errors from SCPI commands to force the sequence to enter the PAUSE mode and display the error, even if an error handler is defined.

The user is required to press a key on the 4300 in order to terminate or continue the sequence.

ScpiErrorContinue

causes SCPI errors to be ignored (and not reported) and continues execution with the next command.

Example:

```
# Check Authentication
#
prin:str 12,1, "Authentication SUPPORTED"
call:ssd:akey:checksum
ScpiErrorContinue
call:ssd?
```

CompareScpiError -653
BranchIfNE AuthCont
ClearLastTest
prin:str 12,1, "Authentication NOT SUPPORTED"
(AuthCont):
ScpiErrorPause
performs an SSD update command to verify authentication on a mobile.

It disables SCPI errors during the test and re-enables them when complete. Immediately following the SSD update command, a check of the SCPI error code is made to determine if the mobile did not respond to the SSD update command, indicating authentication is not supported by this mobile. If this is the cause of the error, the test results are eliminated from the log, and the page buffer results will indicate that the test was not performed (Authentication NOT SUPPORTED). Any other error indicates that Authentication is supported, but failed the SSD update test.

%ERROR_HANDLER

This must be placed at the beginning of the error handler code.

This is the location from which execution will begin when a SCPI error occurs.

NOTE: The '%' character must be the first character of this command.

%ERROR_EXIT

This must be placed at the end of the error handler code.

It must also be the last non-whitespace in the sequence.

If the error handler executes this command, it is treated in the same manner as the ErrorPause directive.

ErrorReturn

causes the sequence to continue execution with the command following the SCPI command that caused the error.

This picks up where the sequencer left off prior to the error without re-executing the command that failed.

ErrorRepeat

causes the sequence to continue execution with the SCPI command that caused the error. This picks up where the sequencer left off prior to the error and re-executes the command that failed.

ErrorPause

An error handler-only command.

One of the return commands from the error handler that returns to the offending command, indicates the error and pauses, waiting for the user to press the CONTINUE key to begin executing on the next line.

ERROR <numeric>, [numeric]

allows the error handler to execute code based on the error that occurred.

The first form allows the code following this directive to be executed only if the error code that occurred was equal to <numeric>.

If a different error occurred, all of the lines will be skipped until another ERROR (or the%ERROR_EXIT) directive is encountered. The second form defines a range of error code values. If the error that occurred is equal to either of these values or any value between this range, the code following this directive will be executed, otherwise it will be skipped.

See chapter X for a list of the SCPI error codes.

ERROR

indicates that the following code will be executed, regardless of the error code value.

This terminates a series of ERROR <value> directives, so that all other errors not specified branch to this point.

NOTE: If the specific error codes caught by preceding ERROR <value> directives do not exit the error handler or branch around this statement, they execute the code that follows this directive.

This does not exclude previously caught error codes.

Example:

```
%ERROR_HANDLER
SaveScpiCmd
ERROR -258
prog:str1 "Write protect tab set on disk."
prog:str2 "Please correct & re-insert."
prog:pause
prog:str1 ""
prog:str2 ""
ErrorRepeat
ERROR -603
CompareScpiCmd "call:brel"
BranchIfNE   ErrLabel001
ErrorReturn
ERROR
(ErrLabel001):
%ERROR_EXIT
```

checks for a disk write error (-258) due to the write-protect tab being set on the floppy disk and prompts the user to remove the disk and correct the tab setting before rerun the disk write command.

This code checks for an error from the base release command. If it is due to the mobile no longer being up on a call (-603), the error is ignored. All other SCPI errors place the 4300 in the PAUSE mode; the error is displayed in the sequencer message window on the LCD display.

RS-232 Remote Operation Mode

General

Using the RS-232 Remote Operation Mode, the 4300 can be controlled via the serial port to execute all cellular tests and measurements that are performed from the front panel.

This allows for automating the cellular phone testing process by an external controlling device (personal computer) connected to the 4300 via the RS-232 serial ports with the supplied RS-232 serial cable.

The 4300 supports a simple 3-wire RS-232 interface, using only RxD, TxD, and ground.

NOTE: Hardware flow control methods (RTS/CTS and DTR/DSR) are not supported.

The RS-232 connector is a standard DB-9 (9-pin) female connector with TxD on pin 2, RxD on pin 3, and ground on pin 5.

The 4300 is configured as Data Terminal Equipment (DTE), transmitting on TxD and receiving on RxD. The external controlling device transmits on RxD (pin 3) and receives on TxD (pin 2), to communicate with the 4300.

If the external controlling device is not configured this way, reverse pins 2 and 3 with a null modem connector or swap these two pins in the cable connecting the 4300 to the controller.

Setup

The 4300 supports the following communication formats:

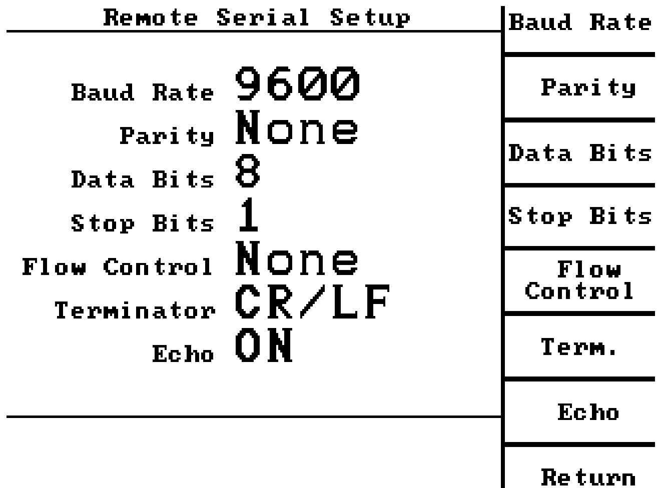
Baud Rate: 19200, 9600, 4800, 2400, 1200, 300

Parity: NONE, ODD, EVEN, MARK, SPACE

Stop Bits: 1, 2

Bits/Char: 7, 8

Set up the communications parameters between the 4300 and the controlling device via Remote Serial Setup Screen as follows: press SETUP and the following soft keys:
More
More
Remote Setup
Serial Setup



Configure the parameters in table X to match those of the controlling terminal.

Change these parameters by pressing the associated soft key until the desired setting is displayed.

NOTE: The parameter changes take affect immediately and are saved when the 4300 is turned off.

Baud Rate set to any of the following values: 19200, 9600, 4800, 2400, 1200, 300.

The factory default value is 9600 baud.

Parity selections are NONE, ODD, EVEN, SPACE, and MARK.

The appropriate parity bit is added to each character sent if NONE is not selected. The extra parity bit is also required on each character received; characters received with parity errors are still accepted.

The factory default value is NONE.

Data Bits selects the number of bits each character represents on input and output.

This does not include the parity bit, if enabled, or the stop bits.

The allowable values are 7 and 8, with factory default being 8.

Stop Bits selects the number of stop bits to append to each character.

The allowable values are 1 and 2, with factory default being 1.

Flow Control enables or disables the XON/XOFF software flow control. (RTS/CTS hardware flow control is not supported.)

Flow control prevents the loss of data during communication by indicating when the test equipment is ready to receive another character.

When the 4300 is receiving data, an XOFF character (control-S or decimal 19) is issued when there are only 100 characters available in the input buffer. (The sender must be able to detect that an XOFF character has been sent and pause its transmission, prior to overflowing the receiver's input buffer.)

When at least 200 characters are again available in the input buffer, an XON character (control-Q or decimal 17) is sent to tell the user to continue transmission. Since the input buffer is 8192 characters in length, this is usually only necessary for commands that exceed this length.

If XON/XOFF enabling echo, and verifying the characters received match those sent are disabled is another method of flow control.

The factory default value is NONE.

Terminator specifies the output terminator character(s) to send at the end of each response.

The input terminator definition is not modified by this parameter. Input is terminated with either a carriage return (control-M or decimal 13) or a linefeed (control-J or decimal 10) character.

The output terminator selections are CR (carriage return only), LF (linefeed only), or CR/LF (carriage return followed by a linefeed).

The factory default value is CR/LF.

Echo enables or disables echoing of the characters received by the 4300 back to the user.

When a terminator is received, the output terminator character selection is echoed, rather than the character received.

If both a carriage return and a linefeed character are received, they are treated as a single terminator character. If the CR/LF output terminator is selected and a single linefeed character is received, both a carriage return and a linefeed will be output.

This can be used for flow control on a character by character basis but is slower than XON/XOFF.

The factory default value is ON.

Special Control Characters

Remote command input consists only of ASCII text characters.

Only the printable characters from hexidecimal 20 (ASCII space character) to hexidecimal 7F are accepted as command input.

Non-ASCII values (hexidecimal 80 to hexidecimal FF) are considered an error.

Certain control characters (hexidecimal 00 to hexidecimal 1F) are used as special commands that simulate the GPIB special signals.

Any control character received, with the following exceptions, is converted to an ASCII space character:

Control – C(hex 03)INTERFACE CLEAR

resets the remote port and make it ready to receive new input, aborting any pending remote operations.

The following actions are executed the beginning of a remote session:

resets the remote input buffer
resets the remote output buffer
resets the error queue, eliminating any pending errors
resets the STATUS event subsystem
resets the remote port hardware
resets all software operations to prepare for new remote command

NOTE: This command requires approximately 3 seconds to complete. The remote controller waits to send a command after an (IFC) to prevent possible loss of input characters.

Control – X(hex 18)DEVICE CLEAR

resets the remote port input and output buffers to make one or more devices ready to receive new input, aborting any pending remote operations.

It performs the same functions as the INTERFACE CLEAR.

Control – L(hex 0C)GO TO LOCAL

allows the remote port to put the test equipment in local mode.

Front panel operations are prohibited while the remote port has control of the test equipment. Only the EXIT REMOTE soft key is functional; it can be locked out by the LOCAL LOCKOUT command.

When the GO TO LOCAL signal is received, front panel operation is restored, even if LOCAL LOCKOUT is enabled.

Remote mode is entered when any remote command is received.

Control – R(hex 12)GO TO REMOTE

allows the remote port to put the test equipment into remote mode.

Remote mode is entered when any remote command is received; this allows the unit to be in remote without having to execute additional commands.

Control – X(hex 1A)LOCAL LOCKOUT

allows the remote port to lock out all front panel functionality until a GO TO LOCAL command is received or the power is recycled on the test equipment.

Use this if the controller is running a remote program that has a long run time, to ensure that an accidental pressing of front panel keys does not affect operation.

Control – S(hex 13)XOFF
Control – Q(hex 11)XON

used for flow control of the 4300 remote responses.

For long response messages, the controller may not have a large enough buffer to hold the data, and may have to read it in sections.

Send XOFF to pause the 4300 from sending additional characters until the controller is ready or until the XON command is received.

The controller issues the XOFF prior to filling up the buffer (since the 4300 sends characters until it has received and decoded the XOFF command.

At 19200 baud, as many as 200 characters are sent while the controller is issuing XOFF. To prevent sending XOFF/XON many times, do not send XON until there is sufficient room for additional data.

If XOFF is issued to the 4300, ensure that XON is sent after all the responses have been collected, or the 4300 will not send any more responses.

GPIB OPERATION

General

The 4300 can be controlled externally via the IEEE STD 488 Port using the GPIB Operation Mode which allows for automated cellular phone testing using an external controlling device connected to the 4300 via the 488 Port.

GPIB System Constraints

The 4300 is IEEE 488.2-compliant and implements the following interface functions:

SH1	Complete source handshake capability
AH1	Complete acceptor handshake capability
T6	Basic talker (no talk only mode)
TE6	Basic extended talker
L4	Basic listener (no listen only mode)
LE4	Basic extended listener
SR1	Complete service request capability
RL1	Complete remote/local capability
PP1	Remote parallel poll capability
DC1	Complete device clear capability
DT0	No device trigger capability
C0	No controller capability
E2	Three-station drivers used for driving bus lines

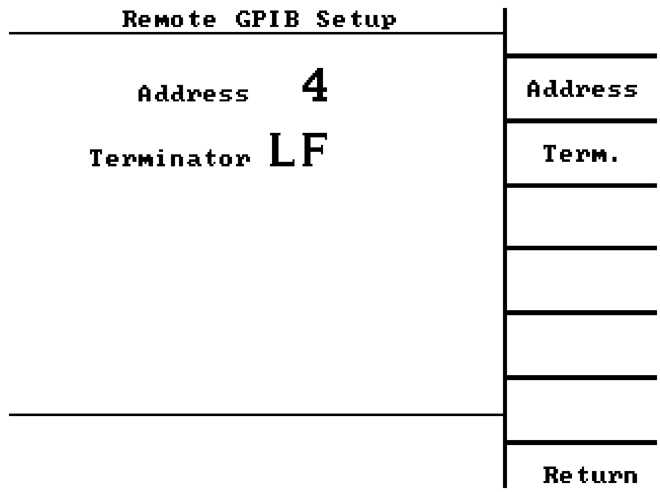
CAUTION: To prevent data loss, read the requested responsedata prior to sending another command.

Any command received following a command terminator (linefeed or EO1) of a query command clears the output buffer and signals a QUERY INTERRUPTED error.

Setup

To select GPIB primary address and output terminator characters at the Remote GPIB setup screen, press SETUP and the following soft keys:

More
More
Remote Setup
GPIB Setup



Configure these parameters to match the controlling terminal:

Address selects the primary GPIB address, and has a valid range of 0 to 31.

Change it by pressing the Address soft key followed by either the numeric entry or the up/down arrow keys.

Secondary addressing is disabled on power up of the test equipment and when programmed through the front panel.

Enable it using remote command SYST:COMM:GPIB:ADDR.

The factory default address is 4.

Terminator specifies the output terminator character(s) to send at the end of each response.

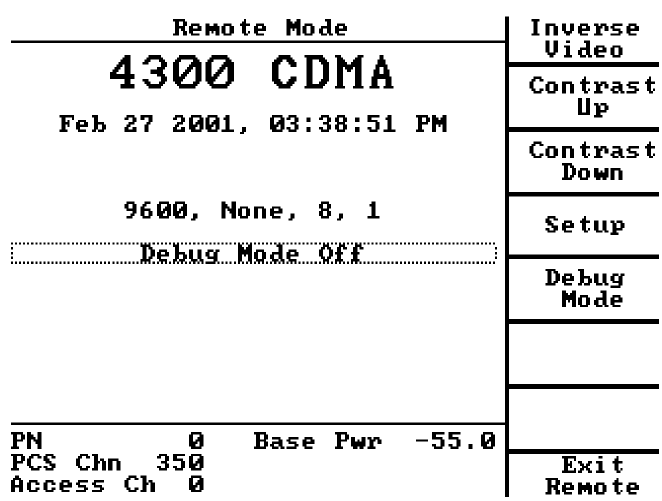
The input terminator definition is not modified by this parameter. Input is terminated with either a carriage return (control-M or decimal 13) or a linefeed (control-J or decimal 10) character, or when an EOI (End or Identify) GPIB bus signal is received.

The output terminator selections are CR (carriage return only), LF (linefeed only), or CR/LF (carriage return followed by a linefeed). An EOI is issued on the last character sent.

The factory default value is CR/LF.

Front Panel Screen

When the 4300 is operating in the remote mode, the following screen is displayed:



Inverse Video

Press Inverse Video to toggle between inverse video on and inverse video off.

Contrast Up/Contrast Down

Use Contrast Up and Contrast Down to adjust the contrast level of the display.

Setup

See 4300 Operations Manual.

Debug Mode

allows the user to view various errors, characters, and commands executed in the remote mode.

Press the Debug Mode soft key until the desired function is displayed.

Function	Function Description
Debug Mode Off:	No information is reported in the remote message window in this mode.
Error Queue:	When the ERROR status indicator is displayed above the remote communication parameters, it indicates that a remote command error has occurred, and the error has been logged to the remote error queue. If Error Queue is selected, the remote command error messages are placed in the remote message window beginning with the oldest message at the top. A maximum of 3 messages can be displayed in the window, although the error queue can hold a maximum of 10 messages. As the errors are read from the remote port using SYST:ERR?, they are read out from the queue from the oldest message to the most recent. As they are read out, they are deleted from the queue. A device clear clears out the error queue.
Last Error In Queue:	This is similar to the debug mode, except that it only displays the most recent error in the error queue.
Characters Received:	displays the last 210 characters received from the remote port. This does not indicate how much of the input has been executed, only what has been received. Control characters are displayed using the IBM mode standard character set. A device clear will clear out the input buffer, as well as this message window.
Characters Sent:	displays the last 210 characters output to the remote port. Control characters are displayed using the IBM mode standard character set. A device clear clears out the output buffer, as well as this message window.

Commands Executed: displays the last commands executed by the 4300.

It displays up to 7 commands if they are less than 31 characters in length. Longer commands use up more than 1 line in the message window.

The characters are placed in this window as they are parsed. Terminator characters are not displayed in this window. A completed command indicates that the command has begun execution.

A device clear clears out this message window.

Responses: displays the last response messages that were generated by the 4300 for output to the remote port.

It displays up to 7 responses if they are less than 31 characters in length. Longer responses use up more than 1 line in the message window.

The characters are placed in this window as they are generated. This window does not indicate that the characters have been output to the remote port. Terminator characters are not displayed in this window.

A completed response indicates that the message available flag (MAV = bit 4) in the serial poll register has been set and that the message is ready for reading by the remote device.

A device clear clears out this message window.

Service Requests and Test Equipment Status

The service request (SRQ) is a special hardware line on the IEEE 488 bus that signals the GPIB controller that a certain event or condition occurred, without requiring the controller to continuously perform queries to the 4300.

When the system controller receives an SRQ, its response is to determine which test equipment is requesting service, by performing a serial poll of each test equipment on the bus.

When a serial poll is performed on the 4300, the response is an 8-bit status value, with bit 6 representing SRQ status.

If the 4300 issues a service request, this bit is set to a 1.

NOTE: The status value indicates the type of event that caused the 4300 to issue the SRQ, and is the same as the value returned from the common command *STB?.

When the serial poll is performed on the test equipment, it no longer asserts its SRQ. If an SRQ condition still exists, it may be due to another test equipment on the bus also issuing an SRQ, or another SRQ-enabled event.

The conditions that generate an SRQ are programmable and explained in detail in the IEEE Common Command definitions.

NOTE: It is not necessary to implement service request interrupt handling in the system controller to perform a serial poll.

Serial polling is an efficient method of synchronizing the controller with the test equipment.

Input commands are buffered in the 4300 to speed communication with the bus and prevents the controller from knowing how many of the commands that have been sent to the 4300 have been executed.

Performing serial polls enables the controller to determine whether a response is ready to be read, an error condition has occurred, or an overlapped command has completed.

IEEE Special Signals

IEEE Special Signals are sent as GPIB bus signals (interface clear), as GPIB Universal Command Group (UCG), or Addressed Command Group (ACG) commands, requiring the bus ATN line be asserted with the command.

NOTE: These functions are defined by IEEE 488.1.

INTERFACE CLEAR

resets the remote port and prepares it to receive new input, aborting any pending remote operations and is performed once at the beginning of a remote session.

One of the pins on the GPIB bus (IFC) is sent to all devices that are connected to the bus.

The following actions are executed:

- clears the remote input buffer
- clears the remote output buffer
- clears the remote error queue
- resets the STATUS reporting system
- resets the remote port hardware
- resets all software operations to be ready for new remote commands.

NOTE: This command requires ~ 3 seconds to complete. The remote controller waits before sending a command after an IFC.

DEVICE CLEAR

resets the GPIB remote operation of one or more test equipment connected to the bus.

It can be sent as a Universal Command (DCL) that affects all test equipment on the bus, or as an Addressed Command (SDC) that only affects specified GPIB addresses.

This command is similar to INTERFACE CLEAR.

The Data Accepted (DAC) handshake signal on the bus waits until the command has completed, to prevent possible loss of data if commands are sent to the 4300 prior to this command.

GO TO LOCAL

This Addressed GPIB command allows the remote port to put the test equipment in local mode.

Front panel operations are prohibited while the remote port has control of the test equipment. The only front panel key that is functional is the EXIT REMOTE soft key, which can be locked out by the LOCAL LOCKOUT command.

When the GO TO LOCAL signal is received, all front panel operation is restored, even if LOCAL LOCKOUT is enabled. Remote mode is enabled when any remote command is received.

LOCAL LOCKOUT

This Universal GPIB command allows the remote port to lock out all front panel functionality until either a GO TO LOCAL command is received, or the power is recycled on the test equipment.

Use this if the controller is running a remote program that has a long run time and the user wishes to guarantee that no one will accidentally press a front panel key that could hinder the program.

TDMA SCPI Commands

4

This chapter describes the functionality of the instrument. Topics discussed in this chapter are as follows:

CALCulate Subsystem

The *Calculate* subsystem commands perform postprocessing on the measurements to convert the data into a form more useful by the application.

CALCulate:LIMit:FAIL:CUMulative

resets the cumulative pass/fail flag that is reported by `CALC:LIM:FAIL:CUM?`.

This allows the flag to represent a group of sequential measurement results.

NOTE: This is a sequencer-only command.

CALCulate:LIMit:FAIL:CUMulative?

determines if all of the measurements performed since either the beginning of the test sequence or the last `CALC:LIM:FAIL:CUM` command passed or if any of them failed the specified limits.

NOTE: This is a sequencer-only command.

0 - indicates all measurements passed.
1 - indicates at least one measurement failed or the limits were not enabled.

Response: 0 or 1

Units: NONE

Example: MEAS:DEV:SAT?
5000
MEAS:DEV:ST?
10000
MEAS:DEV:RES?
5600
CALC:LIM:FAIL:CUM?
0

CALCulate:LIMit:FAIL[:LAST]?

determines if the last measurement performed passed or failed the specified limits.

0 - indicates the measurement passed
1 - indicates the measurement failed or the limits were not enabled

Response: 0 or 1
Units: NONE
Example: MEAS:DEV:SAT?
5000
CALC:LIM:FAIL?

0
indicates that all measurements passed the limit checks.

CALCulate:LIMit:MANufacturer:CODE <mfr code>
<mfr code> numeric integer value from 0 - 255

defines the manufacturer code for the custom limit table.

This value determines the custom table selection when `AUTO` select is enabled.

See `CALC:LIM:TYPE[:LOAD]:AUTO`.

Example: CALC:LIM:MAN:CODE 87
defines the manufacturer code as 87 for the current custom limit table selection.

CALCulate:LIMit:MANufacturer:ESN:LOWer <esn value>
<esn value> numeric integer value from 0 - 16777215 (0xFFFFFF)

defines the lower ESN value for the custom limit table.

This value determines the custom table selection when the `AUTO` select in enabled.

See `CALC:LIM:TYPE[:LOAD]:AUTO`.

This parameter is valued-coupled with `CALC:LIM:MAN:ESN:UPPer`. It cannot be set higher than the value specified by UPPer.

Example: CALC:LIM:MAN:ESN:LOW #h09999
defines the lower ESN value to be 0x09999 for the current custom limit table selection.

CALCulate:LIMit:MANufacturer:ESN:UPPer <esn value>
<esn value> numeric integer value from 0 - 16777215 (0xFFFFFF)

defines the upper Electronic Serial Number (ESN) value for the custom limit table.

This value determines the custom table selection when `AUTO` select is enabled.

See `CALC:LIM:TYPE[:LOAD]:AUTO`.

This parameter is value-coupled with `CALC:LIM:MAN:ESN:LOWer`. It cannot be set lower than the value specified by `LOWer`.

Example: CALC:LIM:MAN:ESN:UPP #h39999
defines the upper ESN value to be 0x39999 for the current custom limit table selection.

CALCulate:LIMit:MANufacturer:NAME <mfr name>
<mfr name> a string of 1 - 21 characters enclosed in double quotes

defines a name for the custom limit table.

Example: CALC:LIM:MAN:NAME “my limit table”
defines the name “my limit table” for the current custom limit table selection.

CALCulate:LIMit:TYPE[:LOAD]:AUTO <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables auto selection of the custom limit table, based on the manufacturer code and ESN range for the 5 custom tables defined.

No change occurs in the limit table selection when disabled.

The limit table selection is immediately modified when enabled whenever a registration or origination of a phone is performed.

The registration determines the manufacturer and ESN for the phone, which is used to determine if a different limit table selection should be made.

If more than one custom limit table matches the registration information, the first one found (searching from `MFC1` to `MFC5`) is selected. If none of the custom limit tables matches the registration information, `MFC1` is set as the default.

Example: CALC:LIM:TYPE:AUTO ON
enables auto selection of the limit table.

CALCulate:LIMit:TYPE[:LOAD][:SELect] <limit table select>
<limit table select> NONE, EIA, MFC1, MFC2, MFC3, MFC4, MFC5

selects the limit table to use when performing measurements:

- If `NONE` is selected, all limits are disabled and limit parameter changes are not allowed.
- If `EIA` is selected, the limits are set to values specified by the EIA standard and changes are not allowed.
- Limit parameter values can only be modified if one of the Manufacturer/Custom tables is selected (`MFC1-MFC5`).

These tables are stored in non-volatile static memory, allowing them to be modified and retain their values when the power to the 4300 is turned off.

NOTE: The factory settings for the custom tables is the same as the EIA table.

Example 1: CALC:LIM:TYPE EIA
selects the EIA limits and disables the limit parameter commands.

Example 2: CALC:LIM:TYPE MFC3
selects custom table 3 limits and enables the limit parameter commands.

CALCulate:LIMit:TYPE:STORe <limit table>
<limit select> MFC1, MFC2, MFC3, MFC4, MFC5

saves the current limit table setup to the specified user definable limit table.

Limit table value changes can only be made if one of the user definable selections is the current limit table selection.

CALCulate:LIMit: {*parameter*}**:LOWer[:DATA] <value>**
<value> numeric value specified by {*lower*} in Table 3-2.

sets the lower limit value for the selected parameter.

NOTE: This is a relative value that is added to the `NOMinal` value to determine the absolute minimum limit that the measurement is compared to.

The current custom limit table selection automatically updates when this parameter value changes.

CALCulate:LIMit: {*parameter*}**:LOWer:STATe <boolean>**
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the minimum limit when limit checking the selected parameter.

The current custom limit table selection automatically updates when this parameter value changes.

CALCulate:LIMit: {*parameter*}**:NOMinal <value>**
<value> numeric value specified by {*nominal*} in Table 3-2.

sets the nominal limit value for the selected parameter.

NOTE: This affects both the upper and the lower absolute limits that the measurement is compared to. The absolute minimum and maximum limits are derived by adding the `NOMinal` value to the `LOWer` and `UPPer` values respectively. The current custom limit table selection automatically updates when this parameter value changes.

CALCulate:LIMit: {*parameter*}**:STATe <boolean>**
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables limit checking for the selected parameter.

The current custom limit table selection automatically updates when this parameter value changes.

CALCulate:LIMit: {*parameter*}**:UPPer[:DATA] <value>**
<value> numeric value specified by {*upper range*} in Table 3-2.

sets the upper limit value for the selected parameter.

NOTE: This is a relative value added to the `NOMinal` value to determine the absolute maximum limit that the measurement is compared to.

The current custom limit table selection automatically updates when this parameter value changes.

CALCulate:LIMit: {*parameter*}:UPPer:STATe <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the maximum limit when limit checking the selected parameter.

The current custom limit table selection automatically updates when this parameter value changes.

Tables 3-1 and 3-2 contain the limit parameter selections that are substituted into the limit table parameter commands.

{parameter}	{description}
ATIMe1	Acquisition time for 2nd channel power >= -90 dBm
ATIMe2	Acquisition time for 2nd channel power < -90 & >= -103 dBm
ATIMe3	Acquisition time for 2nd channel power < -103 dBm
BER:ANALog	Analog BER measurements
BER:LOOPback	Loopback BER measurements
BER:MAHO	MAHO BER (or channel quality) measurements
CSTime:FALL	Carrier switching fall time (TDMA modulation envelope)
CSTime:RISE	Carrier switching rise time (TDMA modulation envelope)
DEViation:AUDio[:AMPS]	audio and peak audio deviation measurements in AMPS mode
DEViation:AUDio:NAMPs	audio and peak audio deviation measurements in NAMPS mode
DEViation:NAMPs	DSAT and DST deviation measurements
DEViation:RESidual	Tx residual deviation measurements
DEViation:SAT	SAT deviation measurements
DEViation:ST	ST deviation measurements
DEViation:WBND	wideband data deviation measurements
DISTortion:RECeive	Rx distortion measurements
DISTortion:TRANsmit	Tx distortion measurements
DURation:FLASh	ST duration measurements for a hook flash
DURation:HANDoff	ST duration measurements for handoffs
DURation:RELease	ST duration measurements for a release
FREQuency:RF[:AMPS]	Tx frequency measurements in AMPS mode
FREQuency:RF:NAMPs	Tx frequency measurements in NAMPS mode

{parameter}	{description}
FREQuency:SAT	SAT frequency (and frequency error) measurements
FREQuency:ST	ST frequency (and frequency error) measurements
FREQuency:RF:TDMA	Tx frequency (and frequency error) measurements in TDMA mode
POWer:MAC*n* = 0 to 10	Mobile power measurements for MAC level *n*
POWer:TRANsmitter:DROOp	TDMA transmitter droop power
POWer:TRANsmitter:PEAK	TDMA transmitter peak power
RSSI	RSSI measurements
SENSitivity[:AMPS]	Sensitivity measurements for AMPS mode
SENSitivity[:AMPS]	Sensitivity measurements for NAMPS mode
SENSitivity:BER	Sensitivity BER limits for TDMA mode
SENSitivity:TDMA	Sensitivity measurements for TDMA mode
SINad:RECeive	Rx SINAD measurements
SINad:TRANsmit	Tx SINAD measurements
TALignment	Time alignment measurements
VERRor:MAGN:PEAK	TDMA peak magnitude error measurements
VERRor:MAGN:RMS	TDMA RMS magnitude error measurements
VERRor:NORM:PEAK	TDMA normalized peak vector error measurements
VERRor:NORM:RMS	TDMA normalized RMS vector error measurements
VERRor:OOFFset	TDMA origin offset vector error measurements
VERRor:PEAK	TDMA peak vector error measurements
VERRor:PHASe:PEAK	TDMA peak phase error measurements
VERRor:PHASe:RMS	TDMA RMS phase error measurements
VERRor:RMS	TDMA RMS vector error measurements

	Factory Settings			
{parameter}	{nominal}	{upper}	{lower}	{res}{units}
ATIMe1	0	130	0	1msec
ATIMe2	0	250	0	1msec
ATIMe3	0	250	0	1msec
BER:ANALog	0	0	0	0.01%
BER:LOOPback	0	3.0	0	0.01%
BER:MAHO	0	1	-1	1 (none)
CSTime:FALL	-60	0	0	0.1 dBW
CSTime:RISE	0	4.0	-20	0.1dB
DEViation: AUDio [:AMPS]	0	12000	0	1Hz
DEViation: AUDio: NAMPs	0	5000	0	1Hz
DEViation:NAMPs	700	70	-70	1Hz
DEViation:RESidual	0	200	0	1Hz
DEViation:SAT	2000	200	-200	1Hz
DEViation:ST	8000	800	-800	1Hz
DEViation:WBND	0	12000	0	1Hz
DISTortion:RECeive	25	0	0	0.1 dB
DISTortion:TRANsmit	0	0	0	0.1dB
DURation:FLASh	400	40	-40	1msec
DURation:HANDoff	50	5	-5	1msec
DURation:RELease	1800	180	-180	1msec
FREQuency:RF[:AMPS]	0	2060	-2060	1Hz
FREQuency:RF:NAMPs	0	830	-830	1Hz
FREQuency:RF:TDMA				1Hz
FREQuency:SAT	0	1	-1	1Hz
FREQuency:ST	0	1	-1	1Hz
POWer:MAC*n=0–*	4.5	2.0	-4.0	0.1 dBW
POWer:MAC*n=1*	0.5	2.0	-4.0	0.1 dBW
POWer:MAC*n=2*	-3.5	2.0	-4.0	0.1 dBW
POWer:MAC*n=3*	-7.5	2.0	-4.0	0.1 dBW
POWer:MAC*n=4*	-11.5	2.0	-4.0	0.1 dBW
POWer:MAC*n=5*	-15.5	2.0	-4.0	0.1 dBW
POWer:MAC*n=6*	-19.5	2.0	-4.0	0.1 dBW
POWer:MAC*n=7*	-23.5	2.0	-4.0	0.1 dBW
POWer:MAC*n=8*	-27.5	3.0	-6.0	0.1 dBW
POWer:MAC*n=9*	-31.5	6.0	-6.0	0.1 dBW
POWer:MAC*n=10*	-35.5	9.0	-9.0	0.1 dBW
POWer:TRANsmitter:DROOp	0	.01	-.01	0.001 dB
POWer:TRANsmitter: PEAK	0	0	0	0.1 dBW
RSSI	0	3	-3	1 (none)
SENSitivity[:AMPS]	-116	0	0	0.1 dBW
SENSitivity:BER	0	5	0	1 (none)
SENSitivity:NAMPs	-118	0	0	0.1 dBW

	Factory Settings			
{parameter}	**{nominal}**	**{upper}**	**{lower}**	**{res}{units}**
SENSitivity:TDMA	-110	0	0	0.1 dBW
SINad:RECeive	12	0	0	0.1dB
SINad:TRANsmit	0	0	0	0.1dB
TALignment	0	0.25	–0.25	0.125 sym
VERRor:MAGN:PEAK	0	0	0	0.1%
VERRor:MAGN:RMS	0	0	0	0.1%
VERRor:NORM:PEAK	0	35	0	0.1%
VERRor:NORM:RMS	0	25	0	0.1%
VERRor:OOFFset	-20	0	0	0.1 dBc
VERRor:PEAK	0	25	0	0.1%
VERRor:PHASe:PEAK	0	0	0	0.01 deg
VERRor:PHASe:RMS	0	0	0	0.01 deg
VERRor:RMS	0	12.5	0	0.1%

CALibration Subsystem

The *Calibration* subsystem commands allow the test equipment to compensate for various factors.

CALibration[:ALL]:AUTO <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the auto-calibration for the Digital Control Module (DCM).

The calibration is performed every 10 minutes for the first 30 minutes of operation, and every 30 minutes thereafter. The calibration takes 15 minutes and pre-empts all tasks.

This prevents disk accesses, sequencer operation, and remote operation during the calibration.

NOTE: This command is valid only if the DCM option is installed.

CALibration[:ALL][:IMMediate]

performs the required calibration for the DCM.

NOTE: This command is valid only if the DCM option is installed.

CALibration:CABLe:COMPensation <boolean>
<boolean> OFF or ON (0 = OFF and 1 = ON)

enables or disables the cable loss factor.

CALibration:CABLe:DATA*[n]* <cable loss>
[n] numeric integer value 1 - 3
<cable loss> numeric value from 0.0 - 5.0 (resolution of 0.1) dB

defines the cable loss value for up to three cables.

NOTE: The cable loss value is selected using `CAL:CABL:SEL` only when `CAL:CABL:COMP` is set to `ON`.

CALibration:CABLe:SELection <cable select>
<cable select> numeric integer value 1 - 3

selects which cable loss factor to use.

NOTE: The cable loss value is selected using `CAL:CABL:SEL` only when `CAL:CABL:COMP` is set to `ON`.

Call Processing Subsystem

The *Call Processing* subsystem serves several purposes.

One purpose of the call processing subsystem is to allow the user to control the parameters used during call processing. Some of the parameters that can be controlled are the type of processing (analog or digital), the channel and band that the unit used to communicate with the mobile, and the system parameters that are transmitted to the mobile through overhead
messages.

The call processing subsystem is also used to initiate any call processing activity: ordered registrations, pages and originations, handoffs, flashes and alerts, transmit power changes, and any other order that might be sent to the mobile.

The call processing subsystem can determine the status of call processing or the mobile that is being tested including: current call processing type channel, completion status of call processing commands, and mobile identity.

CALL:ABURst?

returns the type of access burst that the mobile is directed to use on the reverse digital control channel.

Response:	<character data>	
	NORMal	normal burst
	ABBReviated	abbreviated burst
Units:	none	

CALL:ALERt:CNI1:NPLan <number plan>
CALL:ALERt:CNI2:NPLan <number plan>
<number plan> numeric integer value 0 – 15

sets the Numbering Plan Identification (as defined in ANSI T1.607) to be used for the caller ID for Alert With Info messages sent to the mobile.

TDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

where:	
0	Unknown
1	ISDN/Telephony numbering plan (CCITT E.164 and E.163)
2	Reserved
3	Data Numbering Plan (CCITT X.121)

4	Telex numbering plan (CCITT F.69)
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Private numbering plan
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved for extension

See `CALL:HFLash:TRANsmit:CNI1:NPLan`.

CALL:ALERt:CNI1:NTYPe <number type>
CALL:ALERt:CNI2:NTYPe <number type>
<number type> numeric integer value 0 - 7

sets the Type of Number as defined in ANSI T1.607 to be used for the caller ID for Alert With Info messages sent to the mobile.

TDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

where:	
0	Unknown
1	International number
2	National number
3	Network-specific number
4	Subscriber number
5	Reserved
6	Abbreviated number
7	Reserved for extension

See `CALL:HFLash:TRANsmit:CNI1:NTYPe`.

CALL:ALERt:CNI1:NUMBer <caller id info>
CALL:ALERt:CNI2:NUMBer <caller id info>
<caller id info> ASCII string (15 character maximum) enclosed in double quotes

defines the text (phone number, name, etc.) to be used for the caller ID for Alert With Info messages sent to the mobile.

TDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

CALL:ALERt:CNI1:PI <presentation indicator>
CALL:ALERt:CNI2:PI <presentation indicator>
<presentation indicator> numeric integer value 0 - 3

sets the Presentation Indicator selection (as defined in ANSI T1.607) to be used for the caller ID for Alert With Info messages sent to the mobile.

TDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for CNI1.

where:	
0	Presentation allowed
1	Presentation restricted
2	Number not available
3	Reserved

See `CALL:HFLash:TRANsmit:CNI1:PI`.

CALL:ALERt:CNI1:SI <screening indicator>
CALL:ALERt:CNI2:SI <screening indicator>
<screening indicator> numeric integer value 0 - 3

sets the Screening Indicator selection (as defined in ANSI T1.607) to be used for the caller ID for Alert With Info messages sent to the mobile.

TDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

where:	
0	User-provided, not screened
1	User-provided, verified and passed

2	User-provided, verified and failed
3	Network-provided

See `CALL:HFLash:TRANsmit:CNI1:SI`.

CALL:ALERt[:IMMediate]

sends the Alert With Info message to the mobile phone, with the information specified by `CALL:ALERt:SELect`.

If the first parameter is `NONE`, an Alert message is sent to the mobile phone (instead of the Alert With Info).

NOTE: The phone must be up on a call when this command is executed, or an error will be indicated.

NOTE: This command cannot be queried.

CALL:ALERt:INFOrmation:AMPS <boolean>
<boolean> OFF, ON (0 for OFF and 1 for ON)

enables or disables the Alert with Info and Flash with Info messages sent to the mobile (if the mobile protocol indicates it is EIA 553 or IS-54A).

See `CALL:MDATa:STD?`.

NOTE: These messages are not supported in EIA 553 protocol.

CALL:ALERt:SELect <info select1>, [info select2], [info select3]
<info select> NONE, SIGNal, CNI1, CNI2

selects the information to be sent to the mobile phone when the `CALL:ALERt[:IMMediate]` command is executed.

On an AMPS channel, only one parameter is allowed (only `CNI1` or `NONE)`.

On a TDMA traffic channel, up to three selections may be made and can be any of the choices for <info select>.

The selections represent the following information:

NONE no information sent
SIGNal alert signal (for distinctive ring)
CNI1 calling number indicator (caller ID info)
CNI2 2nd calling number indicator (2nd caller ID info)

CALL:ALERt:SIGNal:CADence <cadence select>
<cadence select> numeric integer value 0 - 63

sets the cadence of the ring used for the Alert With Info messages sent to the mobile.

where:	
0	No tone
1	long
2	short-short
3	short-short-long
4	short-short-2
5	short-long-short
6	short-short-short-short
7	PBX long
8	PBX short-short
9	PBX short-short-long
10	PBX short-long-short
11	PBX short-short-short-short
12-63	Reserved

See `CALL:HFLash:TRANsmit:SIGNal:CADence`.

CALL:ALERt:SIGNal:PITCh <pitch select>
<pitch select> MEDium, HIGH, LOW, REServed

sets the pitch of the ringing tone used for the Alert With Info messages sent to the mobile.

CALL:BAND?

returns the BAND selection for the voice or control channel currently in use.

Response: <character data> CELL or PCS

Example: CALL:BAND?
PCS
CALL:HOFF:BAND CELLULAR; IMM
CALL:BAND?
CELL

CALL:BRELease

performs a base release to terminate the current call, and returns to the current setup for the forward control channel specified by the `CALL:CONTROL` parameters.

This is an overlapped command with an enable code of 0008 hexidecimal and uses the *Call Processing* resources.

See `STAT:OPER:COMP`.

When on an amps call, the test equipment performs an signaling tone (ST) duration measurement upon completion. By following this command with the ST duration measurement query, the ST duration of the base release is reported.

NOTE: The phone must be up on a call when this command is executed, or an execution error will be indicated.

NOTE: This command cannot be queried.

CALL:BURSt?

returns the burst type currently being used for TDMA traffic channel.

Response: <numeric integer>

where: 0 = normal burst for cell-to-cell handoffs
1 = normal burst within a cell
2 = shortened burst for cell-to-cell handoffs
3 = reserved

CALL:CHANnel?

returns the channel number through which the tester and the mobile are communicating.

Response: <numeric integer>
Units: none
Example: CALL:CHAN?
330
indicates that the current channel is 330.

CALL:CONTrol:BAND <band select>
<band select> CELLular or PCS

sets the BAND selection (Cellular or PCS) for the control channel.

NOTE: The PCS band selection is allowed only if the PCS option is installed.

CALL:CONTrol:CHANnel <channel select>
<channel select> numeric value 1 - 1999

sets the channel number to be used for systems A or B.

The selection is activated by the `CALL:FCC ON` command.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:CONTrol:DCC <code value>
<code value> numeric value 0 - 3

selects the Digital Color Code (DCC) to use for system A (or B).

CALL:CONTrol:DCCHannel:ABURst <access burst type>
<access burst type> NORMal or ABBReviated

selects the type of access burst to use for the reverse digital control channel.

CALL:CONTrol:DCCHannel:CUSTom:CLEar

clears the custom digital control channel hyperframe data.

For each of the 64 superframe phases of the custom Digital Color Code (DCC), the 109 bits of data are set to 0 and the sync, Shard Channel Feedback (SCF), and Coded Super Frame Phase (CSFP) are determined by the 4300.

CALL:CONTrol:DCCHannel:CUSTom[:DATA] <sframe index>,
<sframe phase>, <channel type>, <data>[Sync, SCF, CSFP]

<sframe index>	numeric value 0 - 1
	where: 0 = primary superframe 1 = secondary superframe
<sframe phase>	numeric value 0 - 31
<channel type>	FBCCh, EBCCh, SBCCh, SPACh, or REServed
<data>	string of 28 hexadecimal digits (0-F) enclosed in double quotes
<sync>	numeric value 0 to 268435455 (0 to FFFFFFF hexadecimal) or -1 or DEF to use the default sync
<SCF>	numeric value 0 to 4194303 (0 to 3FFFFF hexadecimal) or -1 or DEF to use the default sync
<CSFP>	numeric value 0 to 4095 (0 to FFF hexadecimal) or -1 or DEF to use the default sync

sets the data to be transmitted in the selected superframe phase of a digital control channel.

NOTE: Requires CUSTOM-136 option.

The data represents 109 bits of layer 3 data and the associated layer 2 header field.

Every 2 characters represent 2 hexadecimal digits, or 1 byte (8 bits of data). The least significant 109 bits are used.

The sync represents 28 bits to be transmitted at the beginning of the slot.

The SCF represents 22 bits of shared channel feedback data.

The CSFP represents 12 bits of coded superframe phase data.

The sync, SCF, and CSFP are optional. If not included, the 4300 determines the values for this data.

To create an entire hyperframe, this command must be sent 64 times, once for each superframe phase for both the primary and secondary superframe.

Once the data has been set and `CALL:CONTrol:TYPE` is set to `CUSTom`, then `CALL:FCC ON` causes the new hyperframe to be built and transmitted after completion of the current hyperframe.

Example: CALL:CONT:DCCH:CUST 0, 2, FBCCH,..
"0122334455667788899AABBCCDDEE",DEF,#h3FF
sets the fast broadcast channel data in phase 2 of the primary superframe and the SCF value to hexadecimal 3FF.

NOTE: The 4300 determines the proper default values for sync and CSFP.

CALL:CONTrol:DCCHannel:CUSTom[:DATA]? <index>, <phase>

<index> numeric value 0 to1
where: 0 = primary superframe
1 = secondary superframe

<phase> numeric value 0 - 31

returns the 109 bits of data in the selected superframe phase of the custom DCC.

NOTE: Requires CUSTOM-136 option.

Response: <28 char string> 14 bytes (least-significant 109 bits) of data

Example: "CALL:CONTrol:DCCHannel:CUSTom? 0.0...
"00112233445566778899AABBCCDD"

CALL:CONTrol:DCCHannel:CUSTom:FREE?

returns the number of free entries of superframe phase data or transmit data available.

Each `CALL:CONTrol:DCCHannel:CUSTom:DATA` or `CALL:CONTrol:DCCHannel:CUSTom:TRANsmit` command uses one entry.

This query returns the number of additional commands that may be sent before the buffer overflows.

Requires CUSTOM-136 option.

Response: <numeric integer>

Units: none

Example: CALL:CONTrol:DCCH:CUST:FREE?
1
indicates that only 1 more CALL:CONTrol:DCCHannel:CUSTom:DATA may be sent prior to the next CALL:FCC ON command.

CALL:CONTrol:DCCHannel:CUSTom:TRANsmit <repeat count>
<repeat count> numeric value 1 - 32767

defines the number of times the previously defined hyperframe should be transmitted on the digital control channel.

`CALL:FCC ON` begins transmission on the digital control channel.

This command is used to specify a sequence of hyperframes to be transmitted on the DCC.

Requires CUSTOM-136 option.

Example 1: (1 continuous hyperframe)

CALL:CONTrol:TYPE CUSTom
CALL:CONTrol:DCCH:CUSTom:CLEar
CALL:CONTrol:DCCHannel:SLOT 0,0, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom 0,1, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom 0,2, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom
0,3,FBCCH,"00112233445566778899AABBCCDDEE"
CALL:FCC ON
causes the custom digital control channel hyperframe to be transmitted continuously.

Example 2: (sequence of 3 hyperframes, the last transmitted
continuously)

CALL:CONTrol:TYPE CUSTom
CALL:CONTrol:DCCH:CUSTom:CLEar
CALL:CONTrol:DCCH:CUSTom 0,0, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom 0,1, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom 0,2, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom 0,3, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom:TRANsmit 3
CALL:CONTrol:DCCH:CUSTom 0,1, FBCCH,...
"EEDDCCBBAA99887766554433221100"
CALL:CONTrol:DCCH:CUSTom:TRANsmit 3
CALL:CONTrol:DCCH:CUSTom 0,1, FBCCH,...
"000000000000000000000000000000"
CALL:FCC ON

causes a custom DCC to be transmitted for 3 hyperframes with the first 4 superframe phases identical, followed by 3 hyperframes with the superframe phase 1 reversed, followed by continuous hyperframes with the superframe phase 1 set to all 0's.

Example 3: (sequence of 3 hyperframes, then stop)

CALL:CONTrol:TYPE CUSTom
CALL:CONTrol:DCCH:CUSTom:CLEar
CALL:CONTrol:DCCH:CUSTom 0,0, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom 0,1, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom 0,2, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom 0,3, FBCCH,...
"00112233445566778899AABBCCDDEE"
CALL:CONTrol:DCCH:CUSTom:TRANsmit 3
CALL:CONTrol:DCCH:CUSTom 0,1, FBCCH,...
"EEDDCCBBAA99887766554433221100"
CALL:CONTrol:DCCH:CUSTom:TRANsmit 3
CALL:CONTrol:DCCH:CUSTom 0,1, FBCCH,...
"000000000000000000000000000000"
CALL:CONTrol:DCCH:CUSTom:TRANsmit 5
CALL:FCC ON

causes a custom digital control channel to be transmitted for 3 hyperframes with the first 4 superframe phases identical, followed by 3 hyperframes with the superframe phase 1 reversed, followed by 5 hyperframes with the superframe phase 1 set to all 0's.

After the fifth all 0's hyperframe, hyperframe transmission ends.

CALL:CONTrol:DCCHannel:DVCC <DVCC vector select>
<DVCC vector select> numeric value 0 - 255

selects the Digital Verification Color Code (DVCC) to use for the DCC.

CALL:CONTrol:DCCHannel:MWAIting:NUMBer <value>
<value> numeric integer 0 - 63

sets the number of messages waiting value to be used when issuing the next Message Waiting.

A value of 0 indicates no messages and a value of 63 indicates an unknown number of messages greater than 0. All other values indicate the number of messages waiting.

CALL:CONTrol:DCCHannel:MWAIting[:IMMediate]

issues the Message Waiting message on the DCC using a SPACH message and the current Message Waiting parameters defined.

This command only issues the message if on a TDMA control channel.

CALL:CONTrol:DCCHannel:MWAIting:TYPE <value>
<value> numeric integer 0 - 15

sets the type of message to be used when the next Message Waiting message is issued.

This 4-bit parameter is defined as follows:

0 Voice Messages
1 SMS Messages
2 G3-Fax Messages
3 Reserved
4 – 7 User Specific Messages
8 – 15 Reserved

CALL:CONTrol:DCCHannel:POINter:OFFSet <value>
<value> numeric value -20 to 0 in 2 dB steps

sets the analog pointer signal power level relative to the base power level.

CALL:CONTrol:DCCHannel:POINter:STATe <boolean>
<boolean> OFF, ON (0 = OFF, 1 = ON)

enables or disables transmission of the analog pointer channel.

The analog channel selection is defined as 33 channels below the TDMA control channel selection (`CALL:CONTrol:CHANnel`).
For TDMA channels 1 – 33, the analog channel is 989 channels above it. For TDMA channels 990 - 1023, the analog pointer is not available.

The DCCH channel number is broadcast in the overhead message on this analog control channel to redirect the mobile to it. This makes acquisition on a digital control channel faster if the mobile prefers analog service.

CALL:CONTrol:DCCHannel:RATE <rate select>
<rate select> FULL

selects the rate to use for the DCC.

The 4300 supports full rate phones.

CALL:CONTrol:DCCHannel:SLOT <slot select>
<slot select> numeric value 1 - 6

selects the time slot to use for the digital control channel.

If full rate traffic channel is selected, then a slot select value of either 1 or 4 shall be interpreted as meaning slots 1 and 4, 2 or 5 shall mean 2 and 5, and 3 or 6 shall mean 3 and 6.

The 4300 supports full rate phones.

CALL:CONTrol:DCCHannel:SMS:DAR <boolean>
<boolean> OFF, ON (0=OFF and 1=ON)

sets the Delivery Acknowledge Required (DAR) parameter in the Short Message Services (SMS) message as follows:

0 = Prohibited
1 = Required

CALL:CONTrol:DCCHannel:SMS:DT <value>
<value> numeric integer 0 - 3

sets the Display Time parameter in the SMS message as follows:

TABLE 1.

Display Time	Function
0	Temporary
1	Default
2	Invoke
3	Reserved

CALL:CONTrol:DCCHannel:SMS[:IMMediate]

issues the SMS Deliver message on the digital control channel to be sent to the mobile using the current SMS parameters defined.

This command only issues the message if on a TDMA control channel.

CALL:CONTrol:DCCHannel:SMS:MAR <boolean>
<boolean> OFF, ON (0=OFF and 1=ON)

sets the Manual Acknowledge Required (MAR) parameter in the SMS message as follows:

0 = Prohibited
1 = Requested

CALL:CONTrol:DCCHannel:SMS:MUPDate <boolean>
<boolean> OFF, ON (0=OFF and 1=ON)

sets the Message Updating (MUPD) parameter in the SMS message as follows:

0 = overwrite
1 = new

CALL:CONTrol:DCCHannel:SMS:PI <Privacy Level>
<Privacy Level> numeric integer 0 - 3

sets the Privacy Indicator (PI) parameter in the SMS message as follows:

0 = Not Restricted
1 = Restricted
2 = Confidential
3 = Secret

CALL:CONTrol:DCCHannel:SMS:UDATa <user message>
<user message> ASCII string up to 32 characters in length enclosed in double quotes

sets the text data to send in the SMS message.

CALL:CONTrol:DCCHannel:SMS:UI <value>
<value> numeric integer 0 - 3

sets the Urgency Indicator (UI) parameter in the SMS message as follows:

0 = Bulk
1 = Normal
2 = Urgent
3 = Very Urgent

CALL:CONTrol:DCCHannel:SMS:VALDty <value>
<value> numeric integer 0 - 7

sets the Validity (VALD) parameter in the SMS message.

CALL:CONTrol:DCCHannel:SOCBsmc:BSMC <value>
<value> numeric integer 0 - 255 (00 - FF hexidecimal)

sets the Base Station Manufacturer's Code (BSMC) that is sent out on the DCCH.

This value is only sent out when on a TDMA control channel and the message status bit is on.

A list of BSMC values associated with particular service providers is available in the IS-136.1-A specification under Annex B.

CALL:CONTrol:DCCHannel:SOCBsmc:SOC <value>
<value> numeric integer 0 - 4095 (000 - FFF hexidecimal)

sets the System Operator Code (SOC) that is sent out on the DCCH.

This value is only sent out when on a TDMA control channel and the message status bit is on.

A list of SOC values associated with particular service providers is available in the IS-136.1-A specification under Annex B and is also at www.tiaonline.org.

CALL:CONTrol:DCCHannel:SOCBsmc:STATus <boolean>
<boolean> OFF, ON (0=OFF and 1=ON)

enables or disables sending the SOC/BSMC message on the DCCH.

CALL:CONTrol:DCCHannel:TALignment <TA command select>
<TA command select> numeric value 0 - 31

selects the Time Alignment (TAL) parameter to use for the digital control channel.

CALL:CONTrol:MAC <mobile power level>
<mobile power level> numeric value 0 - 7
(0 - 10 for digital control channel)

selects a digital Mobile Attenuation Code (MAC) level for the analog control channel.

The selection is activated by the `CALL: FCC ON` command.

If the mobile power level is greater than 7 when `CALL:CONTrol:TYPE` is changed to `ANALog`, then the mobile power level is set to 7.

CALL:CONTrol:SID <SID value>
<SID value> numeric value 0 - 32767

selects the System Identification (SID) code to use for system A (or B).

CALL:CONTrol:SYSTem <system select>
<system select> A or B

selects whether system A or B is used.

CALL:CONTrol:TYPE <channel type>
<channel type> ANALog, DIGital, CUSTom
ANALog AMPS and IS-54B operation
DIGital IS-136 operation
NOTE: Requires IS-136 option.
CUSTom IS-136 Layer 1 operation
NOTE: Requires CUSTOM-136 option.

selects the type of control channel to use for system A (or B).

When `CUSTom` is selected, the 4300 outputs the hyperframe as defined by `CALL:CONTrol:DCCHannel:CUSTom:DATA`.

If the mobile power level is greater than 7 when `CALL:CONTrol:TYPE` is changed to `ANALog`, then the mobile power level is set to 7.

CALL:ERRor?

returns the status of the last call processing execution command.

This query is used to detect when a mobile phone fails to complete a call processing task (handoff, registration, origination, page).

Response: <numeric integer>
0 = passed
1 = failed
2 = aborted
3 = overlapped command is still in process

Units: none

Example: CALL:PAGE AMPS;*WAI;
CALL:ERR?
0
indicates that the page completed successfully.

CALL:FCC <boolean>
<boolean> OFF, ON (0 = OFF, 1 = ON)

sets the 4300 and mobile phone to the control channel parameters specified by the control channel parameter setup commands.

If the parameter is set to `OFF`, the control channel of the 4300 is turned off.

Set up the control channel parameters prior to executing this command.

CALL:HFLash[:RECeive]:DATA?

returns the information sent by the phone when it performs a hook flash with information.

Response: <15 char string> the information sent by the mobile phone.

Example: CALL:PAGE TDMA; *WAI
CALL:HFL:DATA?
"1234567"

Requirements: Mobile must have performed a hook flash with information to obtain valid data.

Resources required: none

CALL:HFLash[:RECeive][:IMMediate]

instructs the unit that a hook flash ST duration measurement is about to be made.

This is an overlapped command with an enable code of 0x0010 hexidecimal (16 decimal) requiring no exclusive resources and does not conflict with any other overlapped command.

See `STAT:OPER:COMP`.

Press the proper key on the handset to perform the hook flash, which sets the operation complete flag for an `HOOKflash` (used by the `*WAI` command).

Follow this command with an ST duration measurement query in order to return the measurement value.

The mobile must be up on a call when this command is executed, or an execution error is indicated.

This command cannot be queried.

CALL:HFLash:TRANsmit:CNI1:NPLan <number plan>
CALL:HFLash:TRANsmit:CNI2:NPLan <number plan>
<number plan> numeric integer value 0 - 15

sets the Numbering Plan (`NPL`) identification as defined in ANSI T1.607 to be used for the caller ID for Flash With Info messages sent to the mobile.

TDMA allows two caller IDs, and the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

The definitions for <number plan> include:

TABLE 2.

0	Unknown
1	ISDN/Telephony numbering plan (CCITT E.164 and E.163)
2	Reserved
3	Data Numbering Plan (CCITT X.121)
4	Telex numbering plan (CCITT F.69)
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Private numbering plan
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved for extension

CALL:HFLash:TRANsmit:CNI1:NTYPe <number type>
CALL:HFLash:TRANsmit:CNI2:NTYPe <number type>
<number type> numeric integer value 0 - 7

sets the type of number as defined in ANSI T1.607 to be used for the caller ID for Flash With Info messages sent to the mobile.

TDMA allows two caller IDs, the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

The definitions for <number type> include:

TABLE 3.

0	Unknown
1	International number
2	National number
3	Network-specific number
4	Subscriber number
5	Reserved
6	Abbreviated number
7	Reserved for extension

CALL:HFLash:TRANsmit:CNI1:NUMBer <caller id info>
CALL:HFLash:TRANsmit:CNI2:NUMBer <caller id info>
<caller id info>ASCII string (15 character maximum) enclosed in double quotes

defines the text used for the caller ID for Flash With Info messages sent to the mobile.

TDMA allows two caller IDs, the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

CALL:HFLash:TRANsmit:CNI1:PI <presentation indicator>
CALL:HFLash:TRANsmit:CNI2:PI <presentation indicator>
<presentation indicator> numeric integer value 0 - 3

sets the Presentation Indicator selection (as defined in ANSI T1.607) to be used for the caller ID for Flash With Info messages sent to the mobile.

TDMA allows two caller IDs, the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

The definitions for the <presentation indicator> include:

TABLE 4.

0	Presentation allowed
1	Presentation restricted
2	Number not available
3	Reserved

CALL:HFLash:TRANsmit:CNI1:SI <screening indicator>
CALL:HFLash:TRANsmit:CNI2:SI <screening indicator>
<screening indicator> numeric integer value 0 - 3

sets the Screening Indicator selection (as defined in ANSI T1.607) to be used for the caller ID for Flash With Info messages sent to the mobile.

TDMA allows two caller IDs, and the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

The definitions for the <screening indicator> include:

TABLE 5.

0	User-provided, not screened
1	User-provided, verified and passed
2	User-provided, verified and failed
3	Network-provided

CALL:HFLash:TRANsmit[:IMMediate]

sends the Flash With Info message to the mobile phone, with the information specified by `CALL:HFLash:TRANsmit:SELect`.

If the first parameter specified is `NONE`, a Hook Flash message is sent to the mobile phone instead of the Flash With Info.

If the first parameter specified is MWI and the mobile is on an AMPS call (dual-mode phone only), the Message Waiting message is sent to the mobile phone instead of the Flash With Info.

The phone must be up on a call when this command is executed, or an error is indicated.

This command cannot be queried.

CALL:HFLash:TRANsmit:MWAIting [:Number] <number of msgs>
<number of msgs> numeric integer value 0 - 63

sets the number of messages waiting for the message waiting feature.

A value of 0 indicates `NONE`.
A value of 63 indicates some undetermined number > 0.

CALL:HFLash:TRANsmit:SELect <sel>[, sel2] [, sel3] [, sel4]
<sel1-4> NONE, SIGNal, MWI, CNI1, CNI2

selects the information to be sent to the mobile phone when `CALL:HFLash:TRANsmit[:IMMediate]` is executed.

On an AMPS channel, only the first parameter is allowed, and must be either `MWI`, `CNI1`, or `NONE`.

On a TDMA traffic channel, up to four selections may be made and can be any of the following choices for <info sel>.

NONE no information sent
SIGNal alert signal (for distinctive ring)
MWI message waiting indicator
CNI1 calling number indicator (caller ID info)
CNI2 second calling number indicator (second caller ID info)

CALL:HFLash:TRANsmit:SIGNal:CADence <cadence select>
<cadence select> numeric integer value 0 - 63

sets the cadence of the ring for Flash With Info messages sent to the mobile.

Each of the patterns described below are repeating:

TABLE 6.

Value	Description
0	No tone
1	long
2	short-short
3	short-short-long
4	short-short-2
5	short-long-short
6	short-short-short-short
7	PBX long

TABLE 6.

8	PBX short-short
9	PBX short-short-long
10	PBX short-long-short
11	PBX short-short-short-short
12-63	Reserved

CALL:HFLash:TRANsmit:SIGNal:PITCh <pitch select>
<pitch select> MEDium, HIGH, LOW, REServed

sets the pitch of the ringing tone for Flash With Info messages sent to the mobile.

CALL:HOFF:BAND <band select>
<band select> CELLular or PCS

sets the `BAND` selection (Cellular or PCS) to be used for the next handoff.

If a band change (PCS to cellular or cellular to PCS) is dictated by the handoff, a dedicated digital traffic channel handoff message is sent to the mobile phone instead of the normal Handoff message.

The PCS band selection is allowed only if the PCS option is installed.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:BURSt <burst select>
<burst select> numeric value 0 - 3

selects the type of burst to be used during the handoff to a TDMA traffic channel.

where: 0 = normal burst for cell-to-cell handoffs
1 = normal burst within cell
2 = shortened burst for cell-to-cell handoffs
3 = reserved

This applies only to the TDMA channel type.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the handoff voice channel number.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF[:IMMediate]

performs a handoff to the analog voice or digital traffic channel specified by the handoff parameter setup commands.

Set up the handoff parameters prior to executing this command.

This command indicates an error if the `CALL:HOFF:TYPE` selection is TDMA and the DCM option is not installed.

CALL:HOFF:MAC <mobile power level>
<mobile power level> numeric value 0 - 10 (AMPS = 0 - 7)

selects the handoff mobile access code (MAC) level.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:OFFSet <channel offset select>
<channel offset select> MID, LOW, or HIGH

selects the handoff channel offset.

This applies only to the NAMPS channel type.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:PROTocol <select>
<select> IS54 or IS136

selects which protocol to use after a handoff to an AMPS channel.

The IS54 selection handles a release as designated by IS-54 standard, where SAT and ST are changed from (1,0) to (1,1) 1.8 seconds prior to release.

The IS136 selection handles a release as defined by IS-136 standard which uses Release Order messaging from base DCCH or mobile with a Release Complete message returned for confirmation.

This protocol does not use the SAT and ST signaling for releases.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:RATE <rate select>
<rate select> FULL

selects the rate of the handoff traffic channel.

This applies only to the TDMA channel type.

The 4300 supports full rate phones.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:SAT<SAT select>
<SAT select> numeric value (AMPS = 0 - 2, NAMPS = 0 - 6, TDMA = 0 - 255)

selects the SAT, DSAT, or DVCC vector code to handoff to.

where <SAT select> represents:

SAT frequency vector for AMPS

where: 0 = 5970 Hz
1 = 6000 Hz
2 = 6030 Hz

DSAT vector for NAMPS

where: 0 = 2556CB
1 = 255B2B
2 = 256A9B
3= 25AD4D
4 = 26AB2B
5 = 26B2AD
6 = 2969AB

DVCC vector for TDMA

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:SLOT <slot select>
<slot select> numeric value 1 - 6

selects the handoff time slot.

If full rate traffic channel is selected, then a slot select value of either 1 or 4 is interpreted as meaning slots 1 and 4, 2 or 5 means 2 and 5, and 3 or 6 shall mean 3 and 6.

This applies only to the TDMA channel type.

The 4300 supports full rate phones.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:TALignment <TA command select>
<TA command select> numeric value 0 - 31

selects the time alignment parameter for the traffic channel to handoff to.

This applies only to the TDMA channel type.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:TYPE <voice type>
<voice type> AMPS, NAMPS, or TDMA

selects the voice channel type to handoff to.

The TDMA selection is valid only if the DCM option is installed.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:INITial:AMPS:AVAIlable <boolean>
<boolean> ON, OFF (1 = ON, 0 = OFF)

determines if a voice channel is available when originating a call from an AMPS control channel.

If no voice channel is available, the mobile generates a system (fast) busy signal.

CALL:INITial:AMPS:BAND <band select>
<band select> CELLular or PCS

sets the `BAND` (800 MHz cellular or 1900 MHz PCS) for the next page or origination to an AMPS voice channel.

The PCS band selection is allowed only if the PCS option is installed.

CALL:INITial:AMPS:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the initial voice channel number to be used for the AMPS system.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:INITial:AMPS:MAC <mobile power level>
<mobile power level> numeric value 0 - 7

selects the MAC level to be used for the AMPS voice channel.

CALL:INITial:AMPS:PROTocol <select>
<select> IS54 or IS136

selects the initial protocol to use when paging or originating to an AMPS channel.

The IS54 selection handles a release as designated by IS-54 standard, where SAT and ST are changed from (1,0) to (1,1) 1.8 seconds prior to release.

The IS136 selection handles a release as defined by IS-136 standard which uses Release Order messaging from base DCCH or mobile with a Release Complete message returned for confirmation.

This protocol does not use the SAT and ST signaling for releases.

CALL:INITial:AMPS:SAT <SAT frequency select>
<SAT frequency select> numeric value 0 - 2

selects the SAT frequency code to be used for the AMPs system.

where: 0=5970 Hz
1=6000 Hz
2=6030 Hz

CALL:INITial:NAMPs:BAND <band select>
<band select> CELLular or PCS

sets the `BAND` selection (800 MHz cellular or 1900 MHz PCS) for the next page or origination to an NAMPS voice channel.

The PCS band selection is only allowed if the PCS option is installed.

CALL:INITial:NAMPs:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the initial voice channel number to be used for the NAMPS system.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular) the command is not executed and an error is generated.

CALL:INITial:NAMPs:DSAT <DSAT vector select>
<DSAT vector select> numeric value 0 - 6

selects the DSAT vector code to be used for the NAMPS system.

where: 0 = 2556CB
1 = 255B2B
2 = 256A9B
3 = 25AD4D
4 = 26AB2B
5 = 26B2AD
6 = 2969AB

CALL:INITial:NAMPs:MAC <mobile power level>
<mobile power level> numeric value 0 - 7

selects the MAC level to be used for the NAMPS voice channel.

CALL:INITial:NAMPs:OFFSet <channel offset select>
<channel offset select> MID, LOW, or HIGH

selects the channel offset to be used for the NAMPS system.

CALL:INITial:SYSA:BAND <band select>
CALL:INITial:SYSB:BAND <band select>
<band select> CELLular or PCS

sets the `BAND` (800 MHz cellular or 1900 MHz PCS) for the control channel.

It is used for the next `CALL:FCC ON` command if `CALL:INITial:SYStem` is set to A (or B).

The PCS band selection is only allowed if the PCS option is installed.

CALL:INITial:SYSA:CHANnel <channel select>
CALL:INITial:SYSB:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the channel number to be used for system A (or B).

If the channel is not defined (800-989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:INITial:SYSA:DCC <code value>
CALL:INITial:SYSB:DCC <code value>
<code value> numeric value 0 - 3

selects the Digital Color Code (DCC) for systems A (or B).

CALL:INITitial:SYSA:DCCHannel:ABURSt <access burst type>
CALL:INITitial:SYSB:DCCHannel:ABURSt <access burst type>
<access burst type> NORMal or ABBReviated

selects the type of access burst to use for the reverse digital control channel for system A (or B).

CALL:INITial:SYSA:DCCHannel:DVCC <DVCC vector select>
CALL:INITial:SYSB:DCCHannel:DVCC <DVCC vector select
<DVCC vector select> numeric value 0 - 255

selects the Digital Verification Color Code (DVCC) for the digital control channel for system A (or B).

CALL:INITial:SYSA:DCCHannel:POINter:OFFSet <value>
CALL:INITial:SYSB:DCCHannel:POINter:OFFSet <value>
<value> numeric value -20 - 0 in 2 dB steps

sets the analog pointer signal power level relative to the base power level for systems A and B respectively.

CALL:INITial:SYSA:DCCHannel:POINter:STATe <boolean>
CALL:INITial:SYSB:DCCHannel:POINter:STATe <boolean>
<boolean> OFF, ON (0 = OFF, 1 = ON)

enables or disables the analog pointer for systems A and B.

The analog channel selection is defined as 33 channels below the TDMA control channel selection (`CALL:INITial:SYSA:CHANnel`).

For TDMA channels 1 – 33, the analog channel is 989 channels above it.

For TDMA channels 990 - 1023, the analog pointer is not available.

CALL:INITial:SYSA:DCCHannel:RATE <rate select
CALL:INITial:SYSB:DCCHannel:RATE <rate select>
<rate select> FULL

selects the rate to use for the digital control channel for systems A and B.

The 4300 only supports full rate phones.

CALL:INITial:SYSA:DCCHannel:SOCBsmc:BSMC <value>
CALL:INITial:SYSB:DCCHannel:SOCBsmc:BSMC <value>
<value> numeric integer 0 - 255 (00 - FF hexidecimal)

sets the initial value for the base station manufacturer's code of systems A and B.

CALL:INITial:SYSA:DCCHannel:SOCBsmc:SOC <value>
CALL:INITial:SYSB:DCCHannel:SOCBsmc:SOC <value>
<value> numeric integer 0 - 4095 (000 - FFF hexidecimal)

sets the initial System Operator Code value for systems A and B.

CALL:INITial:SYSA:DCCHannel:SOCBsmc:STATus <boolean>
CALL:INITial:SYSB:DCCHannel:SOCBsmc:STATus <boolean>
<boolean> OFF, ON (0 = OFF, 1 = ON)

sets the initial SOC/BSMC message status for systems A and B.

CALL:INITitial:SYSA:DCCHannel:TALignment <TA value>
CALL:INITitial:SYSB:DCCHannel:TALignment <TA value>
<TAvalue> numeric value 0 - 31

selects the Time Alignment (`TAL`) parameter to use for the digital control channel for systems A and B.

CALL:INITial:SYSA:MAC <mobile power level>
CALL:INITial:SYSB:MAC <mobile power level>
<mobile power level> numeric value 0 - 10 (0 - 7 for analog control channel)

selects the Mobile Attenuation Code (`MAC`) level for control channel A (or B).

If mobile power level is greater than 7 when `CALL:INIT:SYSA:TYPE ANALog` is received, the mobile power parameter is changed to 7.

CALL:INITial:SYSA:SID <SID value>
CALL:INITial:SYSB:SID <ID value>
<SID value> numeric value 0 - 32767

selects the System Identification (`SID`) code to use for systems A and B.

CALL:INITial:SYSA:TYPE <channel type>
CALL:INITial:SYSB:TYPE <channel type>
<channel type> ANALog, DIGital
ANALog AMPS and IS-54B operation
DIGital IS-136 operation (Requires IS-136 option.)

selects the type of control channel to use for systems A and B.

If the mobile power level (`CALL:INITial:SYSA:MAC`) is greater than 7 when the type is changed to `ANALog`, then the mobile power level is set to 7.

CALL:INITial:SYSTem <system select>
<system select> A or B

selects whether system A or **B** is used.

CALL:INITial:TDMA:BAND <band select>
<band select> CELLular or PCS

sets the `BAND` (800 MHz cellular or 1900 MHz PCS) for the next page or origination to a TDMA voice channel.

The PCS band selection is allowed only if the PCS option is installed.

CALL:INITial:TDMA:BURSt <burst select>
<burst select> numeric value 0 - 3

where: 0 = normal burst for cell-to-cell handoffs
1 = normal burst within cell
2 = shortened burst for cell-to-cell handoffs
3 = reserved

selects the type of burst to be used when a TDMA traffic channel is initially established.

CALL:INITial:TDMA:CHANnel <channel select>
<channel select> numeric value 1 - 1999 (excluding 800 - 989, 1024 - 1999 for cellular)

selects the initial traffic channel number for the TDMA system.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:INITial:TDMA:DVCC <DVCC vector select>
<DVCC vector select> numeric value 0 - 255

selects the Digital Verification Color Code (DVCC) for the TDMA system.

CALL:INITial:TDMA:MAC <mobile power level>
<mobile power level> numeric value 0 - 10

selects the digital Mobile Attenuation Code (MAC) level for the TDMA voice channel.

CALL:INITial:TDMA:RATE <rate select>
<rate select> FULL

selects the rate of the traffic channel for the TDMA system.

The 4300 supports full rate phones.

CALL:INITial:TDMA:SLOT <slot select>
<slot select> numeric value 1 - 6

selects the initial time slot to be used for the TDMA system.

If full rate traffic channel is selected, then a slot select value of either 1 or 4 means slots 1 and 4; 2 or 5 means 2 and 5; 3 or 6 means 3 and 6.

The 4300 supports full rate phones.

CALL:INITial:TDMA:TALignment <TA command select>
<TA command select> numeric value 0 - 31

selects the Time Alignment (TAL) parameter for the initial traffic channel to be used for the TDMA system.

CALL:INITial:VCHannel <voice type>
<voice type> AMPS, NAMPS, or TDMA

selects the voice channel type to be used.

The TDMA selection is only valid if the DCM option is installed.

CALL:MAC?

returns the Mobile Attenuation Code (MAC) level at which the mobile is transmitting.

Response: <numeric integer>

TABLE 7.

Channel Type	Response Code	Range of Values
analog control (A or B)	CMAC	0-7

TABLE 7.

Channel Type	Response Code	Range of Values
digital control (DCCA or DCCB)	DMAC	0–10
voice (AMPS or NAMPS)	VMAC	0–7
traffic (TDMA)	DMAC	0–10

Units: none

Example: CALL:MAC?
1
indicates that the current MAC level = 1.

CALL:MDATa:CODEc?

returns the voice codec supported by the mobile.

It returns an enumerated response indicating this type. The mobile type is retrieved from the phone during registration. If a registration has not yet been performed, the codec type is `UNKnown`.

Response: <character data> VSEL, ACELp, UNKnown.

CALL:MDATa:DCCHannel:EMERgency?

returns the phone's emergency call flag.

An origination on a digital control channel must be performed on the phone prior to this query to obtain valid information.

Requires IS-136 option.

Response: <numeric integer 0 or 1>

CALL:MDATa:DCCHannel:PVERsion?

returns the phone's protocol version.

Perform a registration or origination on the mobile's digital control channel prior to this query, to obtain valid information.

Requires IS-136 option.

Response: numeric integer>

Units: none

Example: CALL:MDAT:DCCH:PVER?
2
indicates that the mobile supports IS-136 protocol.

CALL:MDATa:DCCHannel:RACHannel:CLEAr

clears out the RACH buffer received from the mobile on the reverse digital control channel, allowing one of these for further collection without losing data to buffer overflow.

This is because
`CALL:MDATa:DCCHannel:RACHannel[:DATA]?` returns up to 25 bursts and the 4300 collects up to 250.

CALL:MDATa:DCCHannel:RACHannel[:DATA]? [<number of bursts>]
<number of bursts> numeric value 1 - 25

returns the information received from the mobile on the reverse digital control channel.

The optional parameter indicates the maximum number of bursts of data to return. If the optional parameter is omitted, then one burst of data will be returned. The 4300 is capable of storing up to a maximum of 250 bursts of data received from the mobile on the reverse digital control channel.

Requires CUSTOM-136 option

Response: <hyperframe #>, <superframe #>, <superframe phase>, <data>
where:
<hyperframe #>= -1 if an overflow error occurred
= -2 if an underflow error occurred
= non-negative number
(the number of hyperframes that have been transmitted since the last CALL:FCC ON command was sent.)
<superframe #>= 0 or 1 within the hyperframe
<superframe phase> = 0 - 31 for the phase number within the hyperframe (this is the number of bursts of data missing if an overflow error occurred.)
<data> = 32 ASCII characters of data (0-9, A-F)
represents the 16 bytes of data received from the mobile. The 122 bits of data from the mobile is right justified in the 16 bytes of data.

Example 1:

CALL:MDAT:DCCH:RACH? 2
3, 0, 23, "00112233445566778899AABBCCDDEEFF",
3, 1, 15,"00112233445566778899AABBCCDDEEFF",...

The first 16 bytes of data was received in the primary super frame, phase 23, within the 3rd hyperframe since the last CALL:MDATa:DCCHannel:RACHannel? query.

The second 16 bytes of data was received in the secondary super frame, phase 15, within the 3rd hyperframe since the lasCALL:MDATa:DCCHannel:RACHannel? query.

Example 2:

CALL:MDAT:DCCH:RACH? 3
3, 0, 23, "00112233445566778899AABBCCDDEEFF", -1, 0, 3, "00000000000000000000000000000000", 3, 1, 15, "00112233445566778899AABBCCDDEEFF",...

This is similar to Example 1 except that 3 bursts (slots) of data were received but not saved due to an overflow error.

CALL:MDATa:DCCHannel:RTYPe?

returns the mobile's registration type.

A registration on a digital control channel must be performed on the phone prior to this query to obtain valid information.

Requires IS-136 option.

Response: <numeric integer>

TABLE 8.

where:	
0	Power Up
1	Power Down
2	Location Area
3	Forced
4	Periodic
5	Deregistration
6	New System
7	ACC to DCCH
8	TMSI Timeout
9	User Group

Units: none

Example: CALL:REG; *WAI
CALL:MDAT:DCCH:RTYP?
3
indicates that a forced registration occurred.

CALL:MDATa:DCCHannel:SCODe?

returns the phone's service code.

A registration or origination on a digital control channel must be performed on the phone prior to this query to obtain valid information.

Requires IS-136 option.

Response: <numeric integer>

TABLE 9.

where:	
0	Analog Speech only
1	Digital Speech only
2	Analog or Digital - Analog
3	Analog or Digital - Digital
4	Async Data
5	G3 Fax
6	Service Rejected

Units: none

Example: CALL:REG; *WAI
CALL:MDAT:DCCH:SCOD?
3
indicates that the phone is capable of analog or digital speech - digital preferred.

CALL:MDATa:ESN:FORMat <format select>
<format select> STANdard1, STANdard2

selects whether to include the 6-bit reserved field of the Electronic Serial Number (`ESN`) as part of the serial number or as a separate entity.

This only affects the `CALL:MDATa:ESN[:IMMediate]` and `CALL:MDATa:ESN:SNUMber` responses.

`STANdard1` considers them to be separate values.

`STANdard2` combines them into a single serial number value.

CALL:MDATa:ESN:[IMMediate]?

returns the `ESN` of the mobile as a string of ASCII decimal numeric characters separated into groups by ASCII spaces.

The number of groups and the value of the numbers contained in those groups are based on whether `STANdard1` or `STANdard2` is chosen for the `CALL:MDATa:ESN:FORMat` parameter and whether or not the mobile supports expanded ESN.

TABLE 10.

Standard	# group s	String Length	Response Format
STANdard1	3	13	"MMM RR SSSSSS"
STANdard2	2	12	"MMM NNNNNNNN"

MMM 8-bit manufacturer's code field (bits 31-24)

RR 6-bit reserved field (bits 23-18)

SSSSSS 18-bit serial number field (bits 17-0)

NNNNNNNN 24-bit reserved plus serial number field (bits 23-0)

Response: <13 char string>

Example: CALL:MDAT:ESN:FORM STAN1
CALL:MDAT:ESN?
"162 02 188303"
CALL:MDAT:ESN:FORM STAN2
CALL:MDAT:ESN?
"162 00712591"

CALL:MDATa:ESN:MCODe?

returns the phone manufacturer's code contained within the ESN.

A registration or origination must be performed on the phone prior to this query to obtain valid information.

Response: <numeric integer>

Example: CALL:REG;*WAI?
CALL:MDAT:ESN:MCOD?
130
indicates the manufacturer's code for Motorola.

CALL:MDATa:ESN:REServed?

returns the 6-bit reserved field portion (bits 18-23) of the ESN of the mobile phone as a decimal numeric value.

Response: <numeric integer>

CALL:MDATa:ESN:SNUMber?

returns the serial number portion of the ESN of the mobile phone as a decimal numeric value.

If `CALL:MDATa:ESN:FORMat` is set to `STANdard1`, the serial number consists of the 18 least-significant bits (bits 0-17) of the 32-bit ESN value.

If `CALL:MDATa:ESN:FORMat` is set to `STANdard2`, the serial number consists of the 24 least significant bits (bits 0-23).

The extra six bits added in `STANdard2` are defined by IS-54B as a serial number extension that is reserved for allocation by the FCC.

Response: <numeric integer>

CALL:MDATa:MANufacturer?

returns the phone's station class mark.

Perform a registration or origination on the phone prior to this query to obtain valid information.

Response: <35 char string> the manufacturer's name corresponding to the manufacturer code value (MCODe) of the mobile's ESN.

Example: CALL:REG;*WAI?
CALL:MDAT:ESN:MAN?
Motorola
indicates the manufacturer's name corresponding to the manufacturer's code, in this case, Motorola.

CALL:MDATa:MIN?

returns the phone's Mobile Identification Number (`MIN`).

Perform a registration or origination on the phone prior to this query, to obtain valid information.

Response: <13 char string>
10 digit telephone number in the US standard format

Example: CALL:REG;*WAI?
CALL:MDAT:MIN?
(618) 623-9404
indicates that this mobile's number is as shown.

CALL:MDATa:MPCI?

returns the 2-bit Mobile Protocol Capability Indicator (MPCI) value that is received from the phone during a registration, page or origination from an analog control channel.

This value is not updated on a digital control channel.

The value represents the mobile's supported protocol.

See CALL:MDATa:STD for command description.

Response: <numeric integer> 0-3

CALL:MDATa:SCM?

returns the phone's Station Class Mark (SCM).

Perform a registration or origination on the phone prior to this query, to obtain valid information.

Response: <numeric integer>

Example: CALL:REG;*WAI?
CALL:MDAT:SCM?
3
indicates that the phone's station class mark is 3.

CALL:MDATa:STD?

the response string indicates the protocols, received during a registration, that are supported by the mobile.

Perform a registration or origination on the mobile prior to this query, to obtain valid information.

If registration occurs on a digital control channel, the following string values are returned, based on the 4-bit Protocol Version field:

TABLE 11.

Protocol Version	Response String
0	"EIA 553,IS-54-A"
1	"EIA 627,IS-54-B"
2	"IS-136 Rev 0"
3	"Reserved"
4	"IS-136 Rev A"
5 – 15	"Reserved"

If registration occurs on an analog control channel, the following string values are returned based on the 2-bit MPCI field:

TABLE 12.

MPCI	Response String
0	"EIA 553,IS-54-A"
1	"EIA 627,IS-54-B"
2	"IS-95"
3	"IS-136"

The protocol information is updated during a page and origination.

Response: <numeric integer>

CALL:MDATa:TYPE?

returns the mobile phone type indicating what type(s) of voice/traffic channels the mobile is capable of using.

A registration or origination must be performed on the phone prior to this query to obtain valid information.

Response: <character data> AMPS, NAMP, or TDMA

Example: CALL:REG;*WAI?
CALL:MDAT:TYPE?
TDMA
indicates that the mobile is capable of dual-mode operation on AMPS voice or TDMA traffic channels.

CALL:MEM?

returns an indication of whether Message Encryption Mode (`MEM`) is on or off.

An on is represented by a value of 1, and an off is represented by the value 0.

Does not exist for AMPS.

Response: <numeric integer 0 or 1>

Units: none

Example: CALL:MEM?
0
indicates that the message encryption mode is off.

The 4300 does not enable message encryption mode.

CALL:MRELease

instructs the unit that a mobile release ST duration measurement is about to be made.

This command cannot be queried.

Pressing the proper key on the handset performs the mobile release, which sets the operation complete flag for a `RELease` command (used by the `*WAI` command).

Follow this command by an ST duration measurement query in order to return the measurement.

The phone must be up on a call when this command is executed, or an execution error will be indicated.

When the call is terminated, the 4300 returns to the forward control channel conditions specified by `CALL:CONTrol`.

This overlapped command requires an enable code of 0008 hexidecimal, uses no exclusive resources, and does not conflict with any other overlapped command.

See `STAT:OPER:COMP`.

CALL:OFFSet?

returns the NAMPS channel offset through which the instrument and the mobile are communicating.

This query has meaning only if the current channel type is NAMPS.

Response: <character data> MID, LOW, or HIGH

Example: CALL:PAGE NAMPs;*WAI?
CALL:OFFS?
LOW
indicates that the current NAMPS channel offset is LOW.

CALL:ORIGination:DIALed?

returns the called party number sent by the mobile phone during an origination process.

This number corresponds to the digits dialed on the mobile's keypad when a call is made.

Response: <16 char string> the digits dialed as sent by the mobile phone.

Requirements: Unit must have successfully completed an origination for the information to be valid.

Example: CALL:ORIG;*WAI
CALL:ORIG:DIAL?
623-9404
indicates that the mobile reported the above number as the digits dialed to place the call.

CALL:ORIGination[:IMMediate] <voice type>
<voice type> AMPS, NAMPS, or TDMA (optional)

sets the unit up from the initial system data for the selected type of call to be originated from the phone.

This command cannot be queried.

This overlapped command requires an enable code of 0002 hexadecimal and uses the *Call Processing* resources.

If the optional voice type is not specified in the command, the type specified by the `CALL:INITial:VCHannel` is used.

The TDMA selection is valid only if the DCM option is installed.

See `STAT:OPER:COMP`.

CALL:PAGE[:IMMediate] <voice type>
<voice type> AMPS, NAMPS, or TDMA

pages the mobile to the selected voice channel using the parameters set in the initial system.

When the mobile receives the page, press the proper key on the handset to answer the call.

Either an `Alert` or an `Alert With Info` message is automatically sent to the mobile when it is on a voice or traffic channel.

Use the parameters in `CALL:ALERt:XXX`.

This overlapped command requires an enable code of 0004 hexidecimal and uses the *Call Processing* resources.

See `STAT:OPER:COMP`.

If the voice type is omitted, the type is specified by `CALL:INITial:VCHannel`.

This command cannot be queried.

The TDMA selection is valid only if the DCM option is installed.

CALL:PAGE:MIN1 <lower MIN value>
<lower MIN value> numeric integer value 0 - 9999999

sets the lower 7 digits of the Mobile Identification Number (`MIN`) of the phone to be paged using `CALL:PAGE[:IMMediate]`.

These digits are designated as `MIN1`.

CALL:PAGE:MIN2 <upper MIN value>
<upper MIN value> numeric integer value 0 - 999

sets the upper 3 digits of the Mobile Identification Number (`MIN`) of the phone to be paged using `CALL:PAGE[:IMMediate]`.

These digits are designated as `MIN2`.

CALL:PLAYer:DIC <boolean>
<boolean> OFF, ON (0= OFF, 1=ON)

enables or disables the Delay Interval Compensation (`DIC`) function in the mobile phone.

This command is valid only if the DCM option is installed.

CALL:PLAYer:DTX <boolean>
<boolean> OFF, ON (0= OFF, 1=ON)

enables or disables the Discontinuous Transmission (`DTX`) function in the mobile phone.

This command is valid only if the DCM option is installed.

CALL:PLAYer:TYPE <channel select>
<channel select> FACCh or SACCh

selects the message channel to be used during mobile testing on a TDMA traffic channel.

The FACCH and SACCH signalling channels are used for the transmission of control and supervision messages between the base station and the mobile station.

This command is valid only if the DCM option is installed.

CALL:PLCH:MAC <mobile power level>
<mobile power level> numeric value 0 - 10 (AMPS = 0 - 7)

sets the current mobile power to a specified value if a voice channel (AMPS, NAMPS, TDMA) is assigned.

CALL:PM?

returns an indication of whether privacy mode (PM) is on or off.

An on is represented by a value of 1, and an off is represented by the value 0.

Response: <numeric integer 0 or 1>

Units: none

Example: CALL:PAGE TDMA; *WAI
CALL:PM?
0
indicates that the privacy mode is off.

The 4300 does not enable privacy mode.

CALL:RATE?

returns the rate of the TDMA traffic channel through which the instrument and the mobile are communicating.

This query has meaning only if the current channel type is TDMA/DCCA/DCCB.

Response: <character data> FULL or HALF

Example: CALL:RATE?
FULL
indicates that the current TDMA channel is using full rate: 2 slots per frame.

The 4300 supports full rate phones.

CALL:REGistration

performs a registration and saves the results internally.

After this command completes, with no errors, the data is accessible using CALL:MDAT:XXX.

See *Call Processing* section.

Save the data in the print buffer using PRIN:ITEM and PRIN:SEL:ITEM.

This command cannot be queried.

CALL:RELease:POINter <boolean>
<boolean> OFF, ON (0= OFF, 1 = ON)

enables or disables placing the DCCH information (channel number) in the TDMA release message to the mobile.

When enabled, the mobile is redirected to the digital control channel upon performing a release to make reacquisition faster if the mobile prefers analog service.

CALL:SAT?

returns the SAT, DSAT, DCC, or DVCC code being used to maintain communication with the mobile.

Response: <numeric integer>

TABLE 13.

Channel Type	Response Code	Range of Values
Control (A or B)	DCC	0-3
AMPS	SAT	0-2 where: 0 = 5970 Hz 1 = 6000 Hz 2 = 6030 Hz
NAMPS	DSAT	0-6
TDMA	DVCC	0-255

Units: none

Example: CALL:SAT?
2
indicates that the SAT frequency is 6030 Hz.

CALL:SID?

returns the System Identification (SID) used on a control channel.

This query has meaning only if the current channel type is control channel (system A or B).

Response: <numeric integer>

Units: none

Example: CALL:FCC ON
CALL:SID?
2000
indicates that the current system identification code is 2000.

CALL:SLOT?

returns the TDMA channel (or digital control channel) time slot through which the instrument and the mobile are communicating.

This query has meaning only if the current channel type is TDMA/DCCA/DCCB.

Response: <numeric integer>

Units: none

Example: CALL:SLOT?
1
indicates that the current TDMA time slot is 1 if half rate and 1+4 if full rate.

See `CALL:RATE?`.

CALL:SRVMode:BAND <band select>
<band select> CELLular or PCS

sets the `BAND` (800 MHz cellular or 1900 MHz PCS) for the Service Mode.

The PCS band selection is allowed only if the PCS option is installed.

CALL:SRVMode:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the voice channel number for service mode.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular) the command is not executed and an error is generated.

CALL:SRVMode[IMMediate]

places the 4300 in service mode, using the parameters specified by the various Service Mode Parameter Setup commands.

This allows certain measurements to be performed without establishing a traffic or voice channel.

The RF hardware of the 4300 is set as if it is up on a call and some of the measurement commands are permitted. It is assumed that the mobile is set to the same RF settings (i.e. band, channel, channel type, etc.) so that the measurements can be performed.

This command indicates an error if the `CALL:SRVMode:TYPE` selection is TDMA and the DCM option is not installed.

CALL:SRVMode:MAC <mobile power level>
<mobile power level> numeric value 0 - 10 (AMPS = 0 - 7)

selects the Mobile Attenuation Code (`MAC`) level for service mode.

CALL:SRVMode:OFFSet <channel offset select>
<channel offset select> MID, LOW, or HIGH

selects the channel offset for the NAMPS service mode.

CALL:SRVMode:PROTocol <select>
<select> IS54 or IS136

selects the protocol used in the AMPS service mode.

The IS54 selection handles a release as designated by IS-54 standard, where SAT and ST are changed from (1,0) to (1,1) for 1.8 seconds prior to release.

The IS136 selection handles a release as defined by IS-136 standard which uses Release Order messaging from base DCCH or mobile with a Release Complete message returned for confirmation.

IS136 protocol does not use the SAT and ST signaling for releases.

CALL:SRVMode:RATE <rate select>
<rate select> FULL

selects the rate of the traffic channel for service mode.

This applies only to the TDMA channel type.

The 4300 supports full rate phones.

CALL:SRVMode:SAT <vector select>
<vector select> numeric value
(AMPS = 0 - 2, NAMPS = 0 - 6, TDMA = 0 - 255)

selects the SAT, DSAT, or DVCC vector code for service mode.

where <vector select> represents:

SAT frequency vector for AMPS

where:
0 = 5970 Hz
1 = 6000 Hz
2 = 6030 Hz

DSAT vector for NAMPS

where:
0 = 2556CB
1 = 255B2B
2 = 256A9B
3 = 25AD4D
4 = 26AB2B
5 = 26B2AD
6 = 2969AB

DVCC vector for TDMA

CALL:SRVMode:SLOT <slot select>
<slot select> numeric value 1 - 6

selects the time slot for service mode.

If full rate traffic channel is selected, then a slot select value of either 1 or 4 shall be interpreted as meaning slots 1 and 4, 2 or 5 shall mean 2 and 5, and 3 or 6 shall mean 3 and 6.

This applies only to the TDMA channel type.

The 4300 supports full rate phones.

CALL:SRVMode:SPECtrum:PEAKsearch:BWIDth <bandwidth>
<bandwidth> numeric value 0.19 - 40.09, resolution 0.19

sets the frequency bandwidth (in kHz) over which to perform a peak search. See `MEASure:SRVMode:SPECtrum:PEAK`.

This parameter is value-coupled with the frequency offset parameter for `CALL:SRVMode:SPECtrum:PEAKsearch:OFFSet` as follows:

OFFSet - 0.5 * (BWIDth – 0.19) < 40.09kHz

If the validation inequality is false, then the previous value is saved and a "Settings Conflict" error is generated.

CALL:SRVMode:SPECtrum:PEAKsearch:OFFSet <offset>
<offset> numeric value –19.95 to +19.95, resolution 0.19

sets the center of the frequency band over which to perform a peak search.

The offset value is expressed in kHz and is relative to the current channel selection in service mode (`CALL:SRVMode:CHANnel`).

This parameter is value-coupled with the bandwidth parameter `CALL:SRVMode:SPECtrum:PEAKsearch:BWIDth` as follows:

OFFSet - 0.5 * (BWIDth – 0.19) < 40.09kHz

If the validation inequality is false, then the previous value is saved and a "Settings Conflict" error is generated.

CALL:SRVMode:SPECtrum:STATe <boolean>
<boolean> OFF, ON (0 = OFF, 1 = ON)

The 4300 must be in TDMA service mode prior to issuing this command or an error will occur.

Setting this command to ON loads the spectrum mode software, which is required prior to making spectrum measurements.

Setting this command to OFF stops the spectrum measurements, unloads the spectrum mode software and reloads the normal measurement software to allow normal measurements to be made.

Measurements other than the spectrum mode measurements cannot be made while in the spectrum mode.

Switching this state may take up to 30 seconds.

CALL:SRVMode:SPECtrum:TRANsmit <data>
<data> SILent, ONES, ZEROs, or RANDom

selects the data to be transmitted by the 4300 when in the spectrum mode as follows:

TABLE 14.

SILent	DTC (Digital Traffic Channel) with zeros for voice data
ONES	transmitting all ones with no DTC
ZEROs	transmitting all zeros with no DTC
RANDom	transmitting a pseudo-random code called PN9

CALL:SRVMode:TALignment <TA command select>
<TA command select> numeric value 0 - 31

selects the Time Alignment (TAL) parameter for the traffic channel for service mode.

This applies only to the TDMA channel type.

CALL:SRVMode:TYPE <voice type>
<voice type> AMPS, NAMPS, or TDMA

selects the voice channel type for service mode.

CALL:SSD:AKEY:CHECksum

modifies the last 6-digits in the AKEY string (the checksum digits) so that the number is valid.

This does not guarantee that the AKEY number is valid, or that it matches the phone. It simply adjusts it for a valid checksum.

This is useful only if the AKEY of the mobile is known except for the checksum digits, or when the AKEY value has been set to 0, and the mobile has not yet been assigned an A-key.

CALL:SSD:AKEY[:DATA] <auth key string>
<auth key string> string of 26 decimal digits (0-9) enclosed in double quotes

selects the AKEY value used in the Shared Secret Data (SSD) Update test.

For proper operation, it must match the mobile phone AKEY value.

If an AKEY has not been assigned to the mobile phone, enter the default value.

To select a default value enter "00000000000000000000000000" for the AKEY value, and send the CALL:SSD:AKEY:CHECksum command to calculate the checksum digits (the last 6 digits of the string).

CALL:SSD:AUTHbs?

returns the AUTHBS value calculated by the 4300 (from the RANDBS value sent by the mobile) for the last SSD Update test performed.

Response: <18-bit numeric integer> (value 0 to 262143)

CALL:SSD[:IMMediate]?

performs an SSD Update test which requires that a valid AKEY value has been entered into the 4300 (using CALL:SSD:AKEY[:DATA]) and that it matches the AKEY value in the mobile phone being updated.

This update procedure consists of first verifying the AKEY checksum digits. If the AKEY checksum is valid, the 4300 sends the SSD Update Order (including the RANDSSD value) to the mobile phone, and then waits for the Base Station Challenge Order message sent by the mobile. The 4300 then uses the RANDBS value returned by the mobile message to calculate the AUTHBS value, and passes it back to the mobile phone with the Base Station Challenge Order Confirmation message. The mobile will determine if the AUTHBS value is valid, and will update its SSD value if so. It returns an SSD Update Order Confirmation message containing the pass/fail status. This pass/fail status is returned as a response message.

The phone must be up on an AMPS or TDMA call for this command to work. If an error occurs (the mobile is not responding to the SSD Update Order) an error is placed in the error queue, and no response is returned.

Response: 1 if PASS, 0 if FAIL

CALL:SSD:RANDbs?

returns the RANDBS value received from the mobile during the last SSD Update test.

Since this value is a 32-bit value and some remote controllers may have difficulty with an unsigned 32-bit value, it is returned as an ASCII string with 10 decimal digits enclosed in double quotes.

Response: <10-digit decimal numeric string>

Example: CALL:SSD:RAND?
"1298283512"

CALL:SSD:RANDssd:AUTO <auto select>
<auto select> ONCE, OFF, ON

enables or disables auto-selection of the RANDSSD value used in the SSD Update test.

If ONCE is selected, a new RANDSSD value is chosen immediately. This value remains in affect until this command is executed again or until `CALL:SSD:RANDssd[:DATA]` changes the value.

The OFF selection does not change the RANDSSD value; it disables the auto selection each time an SSD Update is performed.

If ON is selected, no change is made to the current RANDSSD value, but a new value will be selected whenever `CALL:SSD[:IMMediate]?` begins a new SSD Update test.

CALL:SSD:RANDssd[:DATA] <random number string>
<random number string>string of 17 decimal digits (0-9) enclosed in double quotes

selects the RANDSSD value to use in the SSD Update test.

CAUTION: `CALL:SSD:RANDssd:AUTO` must be OFF to prevent the auto selection from overwriting this value.

CALL:SSD:SSDA <shared data string>
<shared data string> string of 20 decimal digits enclosed in double quotes or string of 16 hexadecimal digits preceded by 'H' or 'h' with entire string enclosed in double quotes.

allows direct access to modify the SSDA value.

Example: CALL:SSD:SSDA
"12345678901234567890"
CALL:SSD:SSDA
"HAB54A98CEB1F0AD2"
sets the SSDA to the same value, since 12345678901234567890 decimal is equal to AB54A98CEB1F0AD2 hexadecimal.

CALL:SSD:SSDA? [format]
[format] (optional) DECimal or HEXadecimal

reports the current SSDA value.

The SSDA value is set automatically when an SSD update is performed with the `CALL:SSD[:IMMediate]?` command, and when set directly with the `CALL:SSD:SSDA` command. The optional format parameter specifies whether the response should be formatted as a 20-digit decimal value or a 16-digit hexadecimal value.

The default is decimal format.

Response: <20-digit decimal or 16-digit hexadecimal string>

Example: CALL:SSD:SSDA "12345678901234567890"
CALL:SSD:SSDA?
"12345678901234567890"
CALL:SSD:SSDA? HEX
"AB54A98CEB1F0AD2"

CALL:SSD:SSDB? [format]
[format] (optional) DECimal or HEXadecimal

reports the current SSDB value.

The SSDB value is set automatically when an SSD update is performed with the `CALL:SSD[:IMMediate]?` command. The optional format parameter specifies whether the response should be formatted as a 20-digit decimal value or a 16-digit hexadecimal value.

The default is decimal format.

Response: <20-digit decimal or 16-digit hexadecimal string>

Example: CALL:SSD:SSDB?
"11111111111111111111"
CALL:SSD:SSDB? HEX
"9A3298AFB5AC71C7"

CALL:TALChange <TA command select>
<TA command select> numeric value 0 - 31

sends a Time Alignment (`TAL`) command to the phone with the specified time alignment parameter.

This command is valid only if the DCM option is installed.

Requirements: Unit must be up on a TDMA call in manual mode for the command to be executed.

CALL:TALignment?

returns the Time Alignment (TAL) parameter which specifies timing for the mobile's burst transmitted on the reverse TDMA traffic channel.

This query has meaning only if the current channel type is TDMA/DCCA/DCCB.

Response: <numeric integer>

Units: none

Example: CALL:TAL?
3
indicates that the current time alignment is 3:1.5 symbol adjustment from the standard offset.

CALL:TYPE?

returns the type of channel through which the instrument and the mobile are communicating.

The type of channel can be a control channel (system A or B), a digital control channel (system A or B), an AMPS or NAMPS voice channel, or a TDMA traffic channel.

Response: <character data> A, B, DCCA, DCCB, AMPS, NAMPS, TDMA, CUST

TABLE 15.

A	analog control channel on system A
B	analog control channel on system B
DCCA	digital control channel on system A
DCCB	digital control channel on system B
AMPS	AMPS voice channel (Advanced Mobile Phone System)
NAMPS	NAMPS voice channel (narrow-AMPS)
TDMA	TDMA traffic channel (Time-Domain, Multiple-Access)
CUST	CUSTOM IS-136 control channel

Example: CALL:TYPE?
A
indicates that the current channel type is a control channel on system A.

CALL:UNIQue:AUTHu:BASE?

returns the AUTHU value calculated by the 4300 for the last Unique Challenge test performed.

Response: <18-bit numeric integer> (value 0 - 262143)

CALL:UNIQue:AUTHu[:MOBile]?

returns the AUTHU value sent back from the mobile phone for the last Unique Challenge test performed.

Response: <18-bit numeric integer> (value 0 - 262143)

CALL:UNIQue[:IMMediate]?

performs the Unique Challenge test by sending the Unique Challenge Order (including the RANDU value) to the mobile phone.

The tester waits for the Unique Challenge Order Confirmation message response from the mobile, and reads the AUTHU value sent with the message.

The AUTHU value is compared to the value calculated by the 4300 using the Cellular Authentication and Voice Encryption (CAVE) algorithm.

The pass/fail status is returned as a response message.

The mobile must be up on an AMPS or TDMA call for this command to work.

If an error occurs (the mobile does not respond to the Unique Challenge Order), an error is placed in the error queue, and no response is returned.

Response: 1 = PASS, 0 = FAIL

CALL:UNIQue:RANDu:AUTO <auto select>
<auto select> ONCE, OFF, ON

decides whether or not to allow the 4300 to automatically pick a random number when the Unique Challenge test is performed.

If ONCE is selected, a new RANDU value is chosen immediately. This value will remain in affect until this command is again executed, or the `CALL:UNIQue:RANDu[:DATA]` command changes the value.

The OFF selection does not change the RANDU value, and disables the auto selection each time a Unique Challenge is performed.

If ON is selected, no change is made to the current RANDU value; a new value is selected whenever `CALL:UNIQue[:IMMediate]?` begins a new Unique Challenge test.

CALL:UNIQue:RANDu[:DATA] <random number>
<random number> numeric integer value from 0 - 16777215

selects the `RANDU` value to use in the Unique Challenge test.

CAUTION: `CALL:UNIQue:RANDu:AUTO` must be `OFF` to prevent the auto selection from overwriting this value.

CALL:VOICe:CODec <select>
<select> VSELp or ACELp

selects the voice codec to be used for IS-136.

This parameter cannot be modified if on a TDMA call or in service mode.

The codec selection must match the one used by the mobile and must be installed in the 4300, otherwise, the voice type is set to `LOOPback` mode and cannot be modified.

CALL:VOICe:DELay <voice delay>
<voice delay>numeric value 0 - 5, resolution 0.001 seconds

sets the delay interval when the `LOOPback` voice mode is selected.

See `CALL:VOICe:TYPE`.

CALL:VOICe:TYPE <voice mode>
<voice mode>NORMal, LOOPback, RECeiver, or SILent

selects the voice mode used while on a TDMA traffic channel.

TABLE 16.

where:	
NORMal	The user speaks into the handset microphone and the transmitted audio is heard on the 4300's speaker.
LOOPback	The user speaks into the handset microphone and hear his speech retransmitted to the phone with a delay. The delay is selectable. See CALL:VOICE:DELAY.
RECeiver	The 4300 transmits a pre-stored speech sequence to the phone, thereby testing the mobile's receive audio path.
SILent	No audio processing is performed by the unit.

This command is valid only if the DCM option is installed.

CONFigure Subsystem

The *Configure* subsystem commands perform any setup functions required to make measurements.

CONFigure:DEViation <detect select>
<detect select>PEAK, POSitive, NEGative, or RMS

selects the method of measurement to use when taking a deviation measurement.

Example: CONF:DEV POS
MEAS:DEV:WBAN?
1111
configures the unit to use the positive peak wideband method, and measures the wideband deviation.

DIAGnostic Subsystem

The *Diagnostic* subsystem consists of commands that verify the proper operation of the test equipment.

DIAGnostic[:ALL]?

performs an overall diagnostic test of the unit.

This command is the same as the `*TST?` common command defined by IEEE-488.2. The response reported back is the bit values of the board tests that failed. A zero indicates that all tests passed.

The bit values for the boards are the same as for the `STATus:QUES:BOARd` register, which is set when this command is executed.

These bit values are:

TABLE 17.

bit 4	(dec 16)	transceiver board
bit 3	(dec 8)	audio measurement board
bit 2	(dec 4)	DCM board
bit 1	(dec 2)	AMPS board
bit 0	(dec 1)	SCP board

Response: <numeric integer>

Example: DIAG?
5
indicates that the DCM and SCP boards failed self-test.

DIAGnostic:BOARd? <board select>
<board select>SCP, AMPS, DCM, AMEas, or XCVR

performs a self-test on a specified board.

The bit values for the boards are the same as for the `STATus:QUES:BOARd` register, which is set when this command is executed.

TABLE 18.

DCM		
bit 6	(dec 64)	DCM is dead
bit 5	(dec 32)	loopback diagnostics failure
bit 4	(dec 16)	dual port ram failure on DCM side

TABLE 18.

bit 3	(dec 8)	Dual Port ram failure on SCP side
bit 2	(dec 4)	version ID failure
bit 1	(dec 2)	restart failure
bit 0	(dec 1)	generic failure
SCP		
bit 4	(dec 16)	remote port failures
bit 3	(dec 8)	disk drive failures
bit 2	(dec 4)	printer failures
bit 1	(dec 2)	static RAM memory failure
bit 0	(dec 1)	generic failure
All other cards		
bit 0		set for a failure, and cleared otherwise

Response: <numeric integer>
Example:DIAG:BOAR? DCM
32
indicates that the DCM failed the loopback diagnostics tests.

DIAGnostic:OPTion[:DATA]?

returns a list of the options installed in the system.

See `DIAGnostic:OPTion:STATus?` for a list of responses.

Response: <ASCII string data>

Example: DIAG:OPT?
"DCM-TDMA"
indicates that the DCM card is installed for TDMA operation.

DIAGnostic:OPTion:STATus?

returns the bit value of the sum of all installed system options.

Response: <numeric integer (decimal value)>

Bit	Value	Short Name	Definition
0	1	"DCM-TDMA"	TDMA support board is installed

2	4	"CUSTOM-136"	Custom-136 control channel option - allows user to specify control channel information - is enabled for TDMA unit
3	8	"FEX"	PCS option is installed
4	16	"GPIB"	GPIB communication is supported - always available in 4300
5	64	"VSELP"	VSELP audio processing option is installed for TDMA unit
7	128	"ACELP"	ACELP audio processing option is installed for TDMA unit

Example: DIAG:OPT:STAT?
3
indicates that the DCM card is installed for TDMA operation and IS-136 digital control channel operation is enabled.

DISK Subsystem

The *Disk* subsystem directly access the floppy disk to perform specific disk-related activities including: formatting the disk, reading a file, writing to a file, deleting a file, modifying a filename, and listing a directory of the disk.

DISK:CATalog
DISK:DIRectory

produces a directory listing output of the disk currently inserted in the disk drive.

It may require several seconds to build up the response string.

The response is ASCII text enclosed in double quotes, and contains the filenames of each file found on the disk, along with the file size (in bytes), the date stamp (of each file), and the time stamp (of each file).

File entries are separated by newline characters, and the date for each file is separated by a comma.

DISK:CATalog? <extension filter>
DISK:DIRectory? <extension filter>
<extension filter>ALL, AGR, BIN, CAL, HDR, LIM, LOG, PAG, PRG

produces a directory listing output of the disk currently inserted in the disk drive.

The extension filter value placed after the command indicates which type of files to list on the directory.

If `ALL` is used, the command produces a directory of all files found on the disk.

This is the same response that the previous command produces.

For any of the other selections, only the files having the specified extension will be included in the directory listing.

TABLE 19.

Extension	File type
AGR	the output of an Autograph test when log-to-disk is enabled
BIN	the executable program for the 4300 (4300-scp.bin)
CAL	a calibration table file
HDR	a header file
LIM	a limit table that has been saved to disk
LOG	the output of a sequence when log-to-disk is enabled.
PAG	a page buffer that has been saved to disk

TABLE 19.

Extension	File type
PRG	a program sequence file

DISK:DELete <filename>
<filename>string containing the filename to delete

deletes the specified file from the disk directory.

If the file does not exist, error - 256 "File not found" is issued.

DISK:ERRor?

returns the status of the last completed disk command.

A value of 0 indicates the command completed with no errors.

A negative code indicates an error code and does not indicate that the last disk command has completed.

This condition is true when the Disk Command bit in the PendingStatus register is not set.

See chapter 5 for a list of error codes.

DISK:FORMat
DISK:INITiate

formats the floppy disk in the disk drive.

CAUTION: Any data on the disk is erased during a disk format.

DISK:LOAD:HEADer <filename>
<filename>string containing name of file to load

loads the specified header file from the disk into internal memory.

The header file consists of up to 10 lines of ASCII text that are included at the top of log files, and can be inserted into the page buffer using the `PRINt:HEADer1-10` commands.

All header filenames have an.hdr extension.

DISK:LOAD:LIMit <limit table select>, <filename>
DISK:LOAD:SPEC <limit table select>, <file name>
<limit table select>MFC1, MFC2, MFC3, MFC4, MFC5
<filename> string containing name of file to load

loads the specified limit table file from the disk into internal memory.

The limit table is loaded into the custom limit table specified by the first parameter.

All limit table filenames have a.lim extension.

DISK:LOAD:PAGE <filename>
<filename>string containing name of file enclosed in double quotes

loads the specified page buffer file from the disk into internal memory.

This file consists of up to 60 lines of ASCII text (up to 160 characters per line) that can include several standard control characters for producing special fonts (boldface, underscore, expanded and compressed).

The page buffer can be modified with the *Print* subsystem commands and out put to the printer using `PRINt:DUMP`.

Page buffer filenames have a .pag extension.

DISK:LOAD:PROGram <filename>
<filename>string containing name of program file to load

loads the specified program sequence file from the disk into internal memory.

If the command is issued from the remote port, the program is placed in the non-volatile location specified by `PROGram[:SELected]:NAME` (`AUTO`, `QUICK`, or `CUSTOM`).

If `CUSTOM` is selected, the custom sequence location is specified by `PROGram:CUStom:NUMBer`.

These parameters must be set up prior to issuing `DISK:LOAD:PROGram`.

If the command is issued from the sequencer (if a program is currently running and this command is executed from within the program), the specified program file will be loaded into a temporary location that can only be accessed by the sequencer using the `RUNPROG` command to nest programs. This allows one program to pass control to another, and return to the original program following the `RUNPROG` command when the nested program completes.

Only one level of nesting is allowed; "-284, Program currently running" error is flagged if this command is executed while running a nested program.

CAUTION: The temporary location of the nested program is not saved when the power to the unit is removed.

DISK:REName <current filename>, <new filename>
DISK:MOVE <current filename>, <new filename>
<current filename>string containing the current filename enclosed in double quotes
<new filename>string containing the new filename enclosed in double quotes
renames a file.

The filenames must include an extension that identifies the file type and the extensions for both filenames must match.

See `DISK:DIR` for a list of the file extensions.

DISK:STORe:AGRaph

stores the last autograph log buffer to disk after the autograph test has completed.

This is useful if the log-to-disk feature is not enabled when the autograph was run.

The filename is generated internally by taking the first 7 digits of the last recorded ESN from a mobile phone (using the format defined by CALL:MDATa:ESN:FORMat) plus a single character A-Z, chosen so as to not overwrite a previous log file with the same ESN already existing on the disk.

The filename has an .agr extension.

If no unique filename can be created in this manner (26 autograph files with the same ESN code), no file is generated and an error code is returned.

DISK:STORe:LIMit <limit table select>, <filename>
DISK:STORe:SPEC <limit table select>, <file name>
<limit table select>EIA, MFC1, MFC2, MFC3, MFC4, MFC5
<filename> string containing name of file to save to stores the specified limit table data into the specified file.

All limit table filenames have a .lim extension.

CAUTION: If the filename already exists when this command is executed, the old file is overwritten.

DISK:STORe:LOG

stores the last program sequence log buffer (up to 500 lines) to disk after the sequence has completed.

This is useful if the log-to-disk feature is not enabled when the sequence is run, and only the last 500 test results were required to be reported.

The filename is generated internally by taking the first 7 digits of the last recorded ESN from a mobile phone (using the format defined by CALL:MDATa:ESN:FORMat) plus a single character A-Z, chosen so as to not overwrite a previous log file with the same ESN already existing on the disk.

The filename has the .log extension.

If no unique filename can be created in this manner (26 log files with the same ESN code), no file is generated and an error code is returned.

DISK:STORe:PAGE <filename>
<filename>string containing name of saved file

stores the current page buffer text into the specified file.

This file consists of 60 lines of ASCII text (up to 160 characters per line) that can include several standard control characters for producing special fonts (boldface, underscore, expanded and compressed).

CAUTION: If the filename already exists when this command is executed, it is appended to the end of the file.

DISK:STORe:PROGram <program select>, <filename>
<program select>AUTO, QUICk, CUSTom
<filename>string containing name of file to save to

stores the specified program sequence into the specified file.

If `CUSTOM` is selected, the custom sequence location (1-3) stored will be specified by `PROG:CUST:NUMB`.

The program sequence has a .prg extension.

CAUTION: If the filename already exists when this command is executed, the old file is overwritten.

DISPlay Subsystem

DISPlay:CONTrast <contrast>
<contrast> numeric value 0 - 31

adjusts the contrast level of the LCD display.

DISPlay:INVerse <boolean>
<boolean>OFF, ON (0= OFF, 1= ON)

enables or disables the inverse video feature of the LCD display.

DISPlay:RATio[:IMMediate] <source>
<source>RECeive or TRANsmit

selects whether the Rx or Tx SINAD/distortion measurement is displayed on the AMPS and NAMPS RF Tests Screen on the LCD in both Manual and Service Mode front panel operation.

DISPlay:RATio:RECeiver <type>
<type>SNDRatio or DISTortion

selects whether the Rx SINAD or Rx distortion measurement is displayed on the AMPS and NAMPS RF Tests screen on the LCD in both Manual and Service Mode front panel operation if `DISPlay:RATio[:IMMediate]` is set to `RECeive`.

DISPlay:RATio:TRANsmitter <type>
<type>SNDRatio or DISTortion

selects whether the Tx SINAD or Tx distortion measurement is displayed on the AMPS and NAMPS RF Tests Screen on the LCD in both Manual and Service Mode front panel operation if `DISPlay:RATio[:IMMediate]` is set to `TRANsmit`.

DISPlay:UNITs:FREQuency:RF <unit select>
<unit select>HZ or KHZ

selects which RF frequency units to display on the LCD during front panel operation.

DISPlay:UNITs:POWer:TRANsmitter <unit select>
<unit select>WATT, DBW, or DBM

selects which transmitter power units to display on the LCD during front panel operation.

When `WATT` is selected, the unit automatically selects either Watts (W) or milliWatts (mW) for transmitter power measurements.

IEEE Common Commands

*CLS

resets the status reporting structure.

See *Status* subsystem commands.

When `*CLS` is executed:

- the Status Byte Register (`*STB?`) is cleared,
- the Event Status Register (`*ESR?`) is cleared,
- all `STAT:OPER and STAT:QUES EVENT` registers are cleared,
- all `STAT:OPER:COMP` registers are cleared, and
- the error queue is cleared.

*ESE <enable bits>
<enable bits> numeric integer value from 0 - 255

sets the mask for the Event Status Register to produce an ESR summary bit that is set in the Status Byte Register.

See `*ESR?` and `*STB?`.

This register is bitwise ANDed to the ESR register to produce the summary bit.

This command can be queried (`*ESE?`) and responds with the current mask value set (decimal format).

*ESR?

reports back the 8-bit Event Status Register defined by IEEE 488.2.

Four of the bits are used to indicate the occurrence of a specific type of error, while the remaining bits indicate other miscellaneous events.

The **Power On** bit is only set at power up, and is used to determine if the power has been recycled.

The **User Request** bit is set when the Exit Remote front panel key is used to exit remote mode.

The **Command Error** bit is set when a command error occurs (SCPI error code = -100 to -199).

The **Execution Error** bit is set when an execution error occurs for a command (SCPI error code = -200 to -299).

The **Device Dependent Error** bit is set when a device dependent error occurs for a command (SCPI error = -300 to -399).

The **Query Error** bit is set when a query error occurs for a command (SCPI error code = -400 to -499).

The **Request Control** bit is not used.

The **Operation Complete** bit is set when any overlapped command that has its corresponding `STAT:OPER:COMP:ENAB` bit set has completed.

TABLE 20.

bit 7	(dec 128)	Power On
bit 6	(dec 64)	User Request
bit 5	(dec 32)	Command Error
bit 4	(dec 16)	Execution Error
bit 3	(dec 8)	Device Dependent Error
bit 2	(dec 4)	Query Error
bit 1	(dec 2)	Request Control
bit 0	(dec 1)	Operation Complete

Response: <numeric integer (decimal value)>

Example: *ESR?

17

indicates that an execution error has occurred and an overlapped command has reported completion.

*IDN?

returns an ASCII string response identifying the test equipment.

There are four fields to the response string, separated by a comma.

The first field identifies the test equipment manufacturer.

The second field indicates the model number of the test equipment.

The third field indicates the serial number of the test equipment.

The fourth field indicates the software revision installed in the test equipment's control module.

Response: <ASCII string>

Example: *IDN?
"WVTK, 4300,12345678,7.1"
indicates the test equipment is a Wavetek 4300 with control module
software revision 7.1 installed and that the serial number of the test
equipment is 12345678.

*OPC

enables the Operation Complete bit in the Event Status Register when all pending commands have completed.

See `*ESR?`.

It is also used for sending a service request signal to the GPIB controller when overlapped commands have completed.

The service request signal is not available for serial remote mode.

Prior to sending this command, the `STATus:OPERation:COMPlete:ENABle` register must be setup with the enable bits set for the overlapped commands desired, to signal their completion.

For the service request to be sent upon completion, the Operation Complete (bit 0) of the Event Status Enable register (`*ESE`) must be set and the Event Status Register Summary (bit 5) and My Summary Status (bit 6) must both be set in the Status Register Enable (`*SRE`).

*OPC?

is similar to the `*OPC` and `*WAI` commands, but does not require the service request signal, which allows it to be used in GPIB and serial remote modes.

The `STATus:OPERation:COMPlete:ENABle` register must be setup with the enable bits set for the overlapped commands desired to signal their completion.

Neither the `*ESE` or the `*SRE` registers need to be setup.

This command does not complete until the selected overlapped commands have signaled completion. The response message, an ASCII '1', is returned when this command completes.

The Message Available bit is set in the Status Byte Register when the message is available for reading.

Response: <numeric integer> (always a 1)

Example: STAT:OPER:COMP:ENAB 4
CALL:PAGE
*OPC?
1
occurs after the CALL:PAGE command has completed.

*RCL <setting index>
<setting index>numeric integer 0 - 9

sets the 4300 parameters to those defined in the specified stored setting location.

The list of parameters that are modified is the same as the list for `*RST`.

If the stored setting location is empty or has invalid data in it, the current parameters do not change and the appropriate error is placed in the error queue.

*RST

initializes many 4300 parameters to their default values, in order to prevent inconsistent operation (due to uninitialized parameters) between remote sessions.

Example: Use this command if a remote program performed various TDMA measurements (including BER) but the BER measurement interval was not setup beforehand.

Set any parameter that affects the operation of a program once at the start of each remote session, as the program may not behave consistently if parameters are modified between uses.

Sending `*RST` is the most effective method of initializing parameters of current and future commands.

The parameters initialized with this command are the same as those that are saved and recalled using `*SAV` and `*RCL`.

The list of remote command parameters set by this command, along with the default value they are set to, include:*

TABLE 21.

SCPI Command	Default Value
CALCulate:LIMit:TYPE[:LOAD]:AUTO	OFF
CALCulate:LIMit:TYPE[:LOAD][:SELect]	EIA
CALL:ALERt:CNI1:NPLan	0
CALL:ALERt:CNI2:NPLan	0
CALL:ALERt:CNI1:NTYPe	0
CALL:ALERt:CNI2:NTYPe	0
CALL:ALERt:CNI1:NUMBer	"8008511198"
CALL:ALERt:CNI2:NUMBer	"8008511198"
CALL:ALERt:CNI1:PI	0
CALL:ALERt:CNI2:PI	0
CALL:ALERt:SELect	SIGNAL, CNI1
CALL:ALERt:CNI1:SI	0
CALL:ALERt:CNI2:SI	0
CALL:ALERt:SIGNal:CADence	1
CALL:ALERt:SIGNal:PITCh	MEDIUM
CALL:CONTrol:CHANnel	330
CALL:CONTrol:CMAC	2
CALL:CONTrol:DCC	0
CALL:CONTrol:SID	19
CALL:CONTrol:SYSTem	A
CALL:HFLash:TRANsmit:CNI1:NPLan	0
CALL:HFLash:TRANsmit:CNI2:NPLan	0
CALL:HFLash:TRANsmit:CNI1:NTYPe	0
CALL:HFLash:TRANsmit:CNI2:NTYPe	0
CALL:HFLash:TRANsmit:CNI1:NUMBer	"8008511198"

TABLE 21.

SCPI Command	Default Value
CALL:HFLash:TRANsmit:CNI2:NUMBer	"8008511198"
CALL:HFLash:TRANsmit:CNI1:P1	0
CALL:HFLash:TRANsmit:CNI2:PI	0
CALL:HFLash:TRANsmit:CNI1:SI	0
CALL:HFLash:TRANsmit:CNI2:SI	0
CALL:HFLash:TRANsmit:SELect	MWI,SIGNAL,CNI1,CNI2
CALL:HFLash:TRANsmit:SIGNal:PITCh	MEDIUM
CALL:HFLash:TRANsmit:SIGNal:CADence	1
CALL:HFLash:TRANsmit:MWAIting	1
CALL:HOFF:BURSt	0
CALL:HOFF:CHANnel	100
CALL:HOFF:OFFSet	MID
CALL:HOFF:RATE	FULL
CALL:HOFF:SAT	0
CALL:HOFF:SLOT	1
CALL:HOFF:TALignment	0
CALL:HOFF:TYPE	AMPS
CALL:HOFF:VMAC	2
CALL:INITial:AMPS:IVC	330
CALL:INITial:AMPS:SAT	0
CALL:INITial:AMPS:VMAC	2
CALL:INITial:NAMPs:DSAT	2
CALL:INITial:NAMPs:IVC	330
CALL:INITial:NAMPs:OFFSet	MID
CALL:INITial:NAMPs:VMAC	2
CALL:INITial:SYSA:CHANnel	330
CALL:INITial:SYSB:CHANnel	334
CALL:INITial:SYSA:CMAC	2
CALL:INITial:SYSB:CMAC	2
CALL:INITial:SYSA:DCC	0

TABLE 21.

SCPI Command	Default Value
CALL:INITial:SYSB:DCC	0
CALL:INITial:SYSA:SID	19
CALL:INITial:SYSB:SID	80
CALL:INITial:SYSTem	A
CALL:INITial:TDMA:BURSt	0
CALL:INITial:TDMA:DMAC	2
CALL:INITial:TDMA:DVCC	15
CALL:INITial:TDMA:ITC	330
CALL:INITial:TDMA:RATE	FULL
CALL:INITial:TDMA:SLOT	1
CALL:INITial:TDMA:TALignment	0
CALL:INITial:VCHannel	AMPS
CALL:PLAYer:TYPE	FACCH
CALL:PLAYer:DTX	OFF
CALL:PLAYer:DIC	OFF
CALL:SRVMode:TYPE	AMPS
CALL:SRVMode:CHANnel	100
CALL:SRVMode:OFFSet	MID
CALL:SRVMode:SLOT	1
CALL:SRVMode:SAT	0
CALL:SRVMode:VMAC	2
CALL:SRVMode:RATE	FULL
CALL:SRVMode:TALignment	0
CALL:SSD:RANDssd:DATA	"00000000000000000"
CALL:SSD:RANDssd:AUTO	OFF
CALL:UNIQue:RANDu:DATA	0
CALL:UNIQue:RANDu:AUTO	OFF
CALL:VOICe:TYPE	NORMAL
CALL:VOICe:DELay	1.0 sec
CONFigure:DEViation	PEAK

TABLE 21.

SCPI Command	Default Value
INPut:AUDio[:SOURce]	RECeive
INPut:AUDio:CMESsage	ON
INPut:AUDio:COMPressor	OFF
INPut:AUDio:DEViation	1000 Hz
INPut:EXPandor	OFF
INPut:FILTer	Bandpass, 300 - 3000
INPut:RF	DIRect
MEASure:POWer:TRANsmitter:BURSt	NORMAL
MEASure:BER:MINTerval	1.0%
SOURce:AUDio[:SOURce]:INTernal	ON
SOURce:AUDio[:SOURce]:EXTernal	OFF
SOURce:AUDio:FREQuency	1000 Hz
SOURce:AUDio:DEViation	8000 Hz
SOURce:POWer:LEVel	-70.0 dBm
SOURce:POWer:STATe	ON

*System parameters (those that affect the 4300 and the user/controller interface) are excluded from this list.

*SAV <setting index>
<setting index>numeric integer 0 - 9

saves the current parameter values at the specified stored setting location.

A default label is automatically assigned to the location: "Setting #N" where N is the <setting index> value.

If the specified setting location contains a stored setting prior to executing this command, the original setting information is overwritten.

The stored setting data is contained in non-volatile memory which retains these settings when power is removed from the instrument.

The list of parameters that are stored is the same as for `*RST`.

*SRE <enable bits>
<enable bits>numeric integer value from 0 - 255

sets the mask for the Status Byte Register (see `*STB?`) to produce a My Summary Status bit which is placed in bit 6 of the Status Byte Register.

If bit 6 of the SRE register is set and the My Summary Status bit in the STB register is also set, a service request is issued (GPIB mode only).

This command can be queried (`*SRE?`), and responds with the current mask value set (decimal format).

*STB?

reports back the 8-bit Status Byte register defined by IEEE 488.2.

This command consists of summary bits from 3 other register structures (Event Status, Operational Status, and Questionable Status).

The My Summary Status bit represents a summary of the rest of the bits in this register, masked by the Status Register Enable.

See `*SRE` command.

For GPIB mode, this register can be read by performing a serial poll of the instrument where bit 6 (My Summary Status) represents the service request line status.

The definitions of the Status Byte bits are:

TABLE 22.

bit 7:	(dec 128)	OPERational status summary
bit 6:	(dec 64)	My Summary Status
bit 5:	(dec 32)	Event Status Register summary
bit 4:	(dec 16)	Message Available
bit 3:	(dec 8)	QUEStionable status summary
bit 2:	(dec 4)	Error Queue Status
bit 1:	(dec 2)	Remote Command Completed
bit 0:	(dec 1)	-not used-

The **Message Available** bit is set when a query response has been generated and is ready for reading. This bit is used to determine when to read a query response after the query command has been issued. Some measurement responses may take more than 1 second to generate the response.

The **Error Queue Status** bit is set when the error queue contains at least one error. Up to 10 errors can be accumulated in the error queue; this bit uses `SYSTem:ERRor?` to determine when all of the errors have been read.

The **Remote Command Completed** bit is cleared when a remote command is being parsed and set after the terminator is received and the command has been executed or if the command generated an error.

This command is self-destructive. After the command returns the current value of the register, the register is cleared to 0.

Response: <numeric integer (decimal value)>

Example: *STB?
80
indicates that a response message is available for reading, and the service request was issued (if GPIB remote). The service request is negated after this command is executed.

If a serial poll had been performed on the test equipment, the response would have been a value of 80 decimal, and the service request would have been negated after the serial poll had been conducted.

*TST?

performs a complete self-test and reports the result as pass/fail.

See `DIAG?`.

Response: <numeric 0 or 1>

Example: *TST?
0
indicates that the self-test diagnostics passed.

*WAI

This command is similar to `*OPC` and `*OPC?`, but does not require the service request signal and does not produce a response message that must be read.

The `STATus:OPERation:COMPlete:ENABle` register must be setup with the enable bits set for the overlapped commands to signal their completion.

Neither the `*ESE` or the `*SRE` registers need to be setup.

This command will not complete until the selected overlapped commands have signaled completion.

Example: STAT:OPER:COMP:ENAB 1
CALL:REG; *WAI
CALL:MDAT:ESN?

By placing the *WAI command after the CALL:REG command, the command has been converted from an overlapped command into a sequential command. The next command, reading the ESN data acquired by the registration, will not execute until the registration has completed.

INPut Subsystem

The *Input* subsystem commands condition the incoming signal prior to measurement.

INPut:AUDio:CMESsage <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

inserts or removes the `CMESSAGE` filter in the Audio In path prior to the Rx SINAD/distortion measurements.

Example: INP:AUD:CMES ON
inserts the CMESSAGE filter for SINAD measurements.

INPut:AUDio:COMPressor <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

inserts or removes the `COMPressor/PRE-emphasis` from the path between the external Audio In jack and the Tx modulation.

This has no effect unless the `INPut:AUDio` selection is `EXTERNAL`.

INPut:AUDio:DEViation <dev value>
<dev value>numeric value from 0.0 - 20000.0 Hz (resolution of 25.0)

sets the audio input deviation in units of Hz, KHz, or MHz.

The units are assumed to be Hz if not otherwise specified.

Example: INP:AUD:DEV 1500
sets the audio input deviation to 1500 Hz.

INPut:AUDio[:SOURce] <input select>
<output select>RECeive or EXTernal

selects the source of the input modulation.

`RECEIVE` is used for SINAD and other measurements.

`EXTERNAL` selects an external modulation source that is connected to the Audio In jack on the front panel.

Example: INP:AUD REC
allows mobile audio measurements to be made.

INPut:CMESsage <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

inserts or removes the C-message filter in the RF path prior to the Tx SINAD/distortion measurements.

INPut:EXPandor <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

inserts or removes the `EXPandor` from the receive audio measurement path for performing deviation measurements and may be cascaded with input filters.

See `INPut:FILTer`.

Example: INP:AUD:EXP ON
inserts the EXPandor.

INPut:FILTer NONE
INPut:FILTer HP, <hi-pass freq>
INPut:FILTer LP, <lo-pass freq>
INPut:FILTer BP, <hi-pass freq>, <lo-pass freq>
<hi-pass freq>numeric value: 50, 300
<lo-pass freq>numeric value: 3000, 15000, 30000

selects the input filters used between the receiver demodulator and the deviation meter.

The four filter type selections are:

none, high pass only, low pass only, and bandpass.

There are 2 high-pass filter selections and 3 low-pass selections.

The bandpass filter is a combination of any high and low pass filter selections.

The units for the filter frequencies are assumed to be Hz.

In remote mode, the query form of this command returns all 3 parameters; in sequence mode, only the filter type is returned.

<filter type>, <hi-pass freq>, <lo-pass freq>
where: <filter type> = NONE, HP, LP, BP

Example 1: INP:FILT BP, 300, 30000
selects bandpass filtering with a 300 Hz high-pass cutoff frequency and a 30000 Hz low-pass cutoff frequency.

Example 2: INP:FILT?
LP, 15000
indicates that the 15 kHz low-pass filter is currently selected.

INPut:RF <input select>
<input select>DIRect or OFFair

selects whether to insert the 20 Watt 20 dB pad in the RF input line.

`DIRect` inserts the pad, allowing a direct connection to be made from the mobile transmitter to the RF input. The base transmitter power range for this selection is -50 to -125 dBm. This selection is required for making power measurements.

`OFFair` removes this pad to allow greater sensitivity. This selection is used if the RF input is connected to an antenna to receive the mobile transmitter RF signal. Base transmitter power for this selection is -30 to -105 dBm.

Power measurements cannot be made in this mode.

MEASurement Subsystem

The *Measurement* subsystem commands initiate and report measurements.

These commands perform limit checking as specified by the current limit table parameters. See `CALCulate:LIMit`.

Information about the measurement (value, units, limits) may be output to the printer buffer using `PRINt:ITEM`. Many of these commands require the unit to be in a particular state in order to execute, as noted in the Requirements section of the command description.

Certain measurements have a corresponding `READ` subsystem command which reports the previous measurement value taken. This allows a measurement to be made once, and the data can be read or printed in different units, without taking another measurement.

MEASure:ATIMe?

performs a handoff, and measures and returns the acquisition time of the mobile.

The acquisition time is measured from the time the mobile sends the handoff acknowledgment to the time the carrier appears on the new channel.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a TDMA call

Response: <numeric floating point>

Resolution: 0.1

Units: milliseconds

Limit checking: uses the values in CALC:LIM:ATIMe command.

Example: CALL:PAGE TDMA; *WAI
MEAS:ATIM?
230
indicates that the acquisition time is 230 ms.

MEASure:BER?

performs one of three BER measurements, depending on the mode of operation.

If the unit is on an AMPS call, then an analog BER measurement is performed using the number of samples specified by `MEASure:BER:ANALog:SAMPles`.

If on a TDMA call, then a MAHO BER measurement is performed.

If in TDMA service mode, then a loopback BER measurement is performed using the time interval specified by `MEASure:BER:MINTerval`.

The measured value is returned along with an indication of which measurement was performed.

This command indicates an error if not on a call or service mode.

Requirements: Unit must be up on an AMPS or TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <measurement indication>, <numeric floating point>

where:

<measurement indication> ANAL analog
MAHO mobile assisted handoff
LOOP loopback

<numeric floating point> mobile reported BER code if MAHO (0-7), percentage BER if analog or loopback

Resolution: 1 if MAHO; 0.01 if analog or loopback

Units: none if MAHO BER, percentage if analog or loopback BER

Limit checking: uses the values in CALC:LIM:BER:ANALog, CALC:LIM:BER:MAHO, or CALC:LIM:BER:LOOPback command.

Example 1: CALL:PAGE AMPS; *WAI
MEAS:BER?
ANAL,2.5
indicates that analog BER is 2.5% (2.5% of the audit messages were not acknowledged) (unit on AMPS call).

Example 2: CALL:PAGE TDMA; *WAI
MEAS:BER?
MAHO,4.0
indicates that the MAHO BER code is 4 (BER is between 1.0 and 2.0%) (unit on TDMA call).

Example 3: CALL:SRVM:TYPE TDMA; *WAI
MEAS:BER?
LOOP,1.25
indicates that the Loopback BER is 1.25% (unit in TDMA service mode).

MEASure:BER:ANALog:SAMPles <number of samples>
<number of samples>numeric integer value 1 - 1000

selects the number of audit messages to use when an analog BER measurement is performed using `MEASure:BER?` while the unit is in AMPS call mode.

A larger number of samples provides a more accurate analog BER measurement.

MEASure:BER:MINTerval <time interval>
<time interval> numeric value 0.1 - 50.0, resolution 0.1 seconds

selects the measurement interval to use when a loopback BER measurement is performed using `MEASure:BER?` while the unit is in service mode.

A longer measurement interval provides a more accurate loopback BER measurement.

MEASure:CURRent[:DC]?

performs a DC input current measurement for the mobile.

Requirements: none

Response: <numeric floating point>

Resolution: 0.01 Amp

Units: Amps

Limit checking: none

Example: MEAS:CURR?
1.300000
indicates that the DC input current to the phone is 1.3 amps.

MEASure:DEViation:AUDio?

performs an audio deviation measurement.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPS only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values from CALC:LIM:DEViation:AUDio.

MEASure:DEViation:DSAT?

performs a DSAT deviation measurement.

Requirements: Unit must be up on a call or in service mode (NAMPS only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values from CALC:LIM:DEViation:NAMPs.

MEASure:DEViation:DST?

performs a DST deviation measurement.

Requirements: Unit must be up on a call or in service mode (NAMPS only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values from CALC:LIM:DEViation:NAMPs.

MEASure:DEViation:PAUDio <duration>
<duration>numeric integer value 0 - 60 (time in seconds)

initiates a peak audio deviation measurement for the specified amount of time but does not return a measurement value.

Use `READ:DEViation:PAUDio?` following completion of the measurement to obtain measurement value.

This overlapped command allows other commands to be executed while the measurement is in progress; delay the `READ` command until the measurement completes.

Bit 7 (decimal value 128) of the PendingStatus register is set upon completion of the peak measurement.

See the *Status* subsystem commands and "Overlapped and Sequential Commands" at the beginning of Appendix E for an explanation of how the **PendingStatus** flag provides notification to the controller of when the measurement completes and is ready to be read.

This command requires the exclusive use of one of the measurement tasks in the 4300; any other measurement command that uses the same measurement task must wait until the task is available.

The following commands, when placed after `MEAS:DEV:PAUD`, execute only after the peak audio measurement is complete:

MEASure:POWer:TRANsmit[:IMMediate]?
MEASure:POWer:DC?
MEASure:CURRent[:DC]?
MEASure:VOLTage[:DC]?
MEASure:VOLTage:AC?
MEASure:SNDRatio:RECeiver?
MEASure:SNDRatio:TRANsmitter?
MEAS:DEViation:*XXXX*(all of the deviation measurements)

Requirements: Unit must be up on an AMPS or NAMPS call, or in service mode.

MEASure:DEViation:PAUDio? <duration>
<duration>numeric integer value 0 - 60 (time in seconds)

functions the same as the non-query form of the command except that it is non-overlapped (next command is not processed until this command completes) and it returns the peak deviation measurement value in Hz.

Requirements: The phone must be up on an AMPS or NAMPS call or in service mode.

Response: <numeric integer>

Units: Hz

This command is not valid through the remote port. It can only be used from within a program.

MEASure:DEViation:RESidual?

performs a residual deviation measurement.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer>

Units: Hz

Limit checking: uses the values from CALC:LIM:DEViation:RESidual.

MEASure:DEViation:SAT?

performs a SAT deviation measurement.

Requirements: Unit must be up on a call (AMPS only) or in service mode.

Response: <numeric integer>

Units: Hz

Limit checking: uses the values from CALC:LIM:DEViation:SAT.

Example: CALL:PAGE AMPS; *WAI
MEAS:DEV:SAT?
1880
represents 1880 Hz measured SAT deviation.

MEASure:DEViation:ST?

turns the signaling tone ON, performs an ST deviation measurement.

The ST remains on after this query has been executed. Press the proper key on the phone handset to turn the ST off.

The phone may ring during this measurement. Do not answer the phone. Answering the phone could cause measurement errors.

Requirements: Unit must be up on a call (AMPS only) or in service mode.

Response: <numeric integer>

Units: Hz

Limit checking: uses the values from CALC:LIM:DEViation:ST.

MEASure:DEViation:WBANd?

performs a wideband deviation measurement.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer>

Units: Hz

Limit checking: uses the values from CALC:LIM:DEViation:WBANd.

MEASure:DISTortion:RECeiver?

performs a receiver distortion measurement.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values from CALC:LIM:DISTortion:RECeive

Example: CALL:PAGE AMPS; *WAI
MEAS:DIST:REC?
3.2
indicates that the Rx distortion is 3.2%.

MEASure:DISTortion:TRANsmitter?

performs a transmitter distortion measurement.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses values from CALC:LIM:DISTortion:TRANsmit.

Example: CALL:PAGE AMPS; *WAI
MEAS:DIST:TRAN?
12.5
indicates that the Tx distortion is 12.5%.

MEASure:DTMF [:FULL]

initializes the DTMF key string prior to sending `MEAS:DTMF?` from the remote.

The string is initialized to 12 ASCII spaces, and the last character read is initialized to 0. The PASS/FAIL status is set to PASS when all 12 keys have been pressed.

The operation pending flag (see *Status* subsystem) for DTMF is set when this command is received, and is cleared when all 12 DTMF keys have been received.

This overlapped command uses an enable code of 0020 hexidecimal (32 decimal) and requires no exclusive resources (does not conflict with any other overlapped command).

Requirements: Unit must be up on a call (AMPS, NAMPS, TDMA) in manual mode.

MEASure:DTMF [:FULL]?

reports the value of the last DTMF key pressed (the ASCII code represented in decimal format), along with a string indicating which of the 12 DTMF keys have been pressed since the last `MEAS:DTMF` command (not query) was received.

The string is 12 characters in length and enclosed in double quotes. The string sequence is "1234567890*#", and any keys that have not been pressed are replaced with an ASCII space.

Requirements: Unit must be up on a call

Response: <decimal integer>, "<12-character string>"

Example: CALL:PAGE; *WAI
MEAS:DTMF?
50,"1237"*
indicates the last DTMF key = decimal 50 = the '2' key
keys pressed since last reset: '1', '2', '3', '7', '*'(not necessarily in that order).

MEASure:DTMF:KEY

performs the same DTMF query response initialization as `MEAS:DTMF[:FULL]`, without beginning a DTMF test.

This command does not begin an overlapped command, nor does it set up the front panel screen to allow aborting the command from the sequencer.

Use this command prior to the initial `MEAS:DTMF:KEY?` command in order to prevent reading an outdated DTMF key sequence.

MEASure:DTMF:KEY?

reports the value of the first DTMF received following either the set or the query form of `MEAS:DTMF:KEY`.

If no key has been received, this command waits for the next key before allowing another remote command to be parsed. The command may be broken from this loop by one of three conditions:

- **a DTMF key is pressed**
- **the call is dropped**
- **a device/interface clear is received**

The return value is the integer value of the ASCII DTMF key pressed. ('0' to '9' = decimal 48 to 57, '*' = decimal 42, '#' = decimal 35).

If the call is dropped, the return value will be 0.

Response: <decimal integer>

MEASure:FREQuency:ERRor:RF?

performs an RF frequency measurement, and calculates and returns the frequency error in the units that are currently selected by `MEAS:FREQ:UNIT`.

See MEAS:FREQ:UNIT.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a call (AMPS, NAMPS, TDMA) or in service mode.

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values from CALC:LIM:FREQuency:RF.

Example: CALL:PAGE: *WAI
MEAS:FREQ:ERR:RF?
-30000
indicates a measured transmitter frequency that is 30000 Hz lower than the nominal value.

MEASure:FREQuency:ERRor:SAT?

performs a SAT frequency measurement, calculates, and returns the frequency error in the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

The phone may ring during this measurement. Do not answer the phone. Answering the phone could cause measurement errors.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values from CALC:LIM:FREQ:SAT.

Example: CALL:PAGE AMPS; *WAI
MEAS:FREQ:ERR:SAT?
-30

indicates a measured SAT frequency value that is 30 Hz lower than the nominal value.

MEASure:FREQuency:ERRor:ST?

performs an ST frequency measurement, calculates, and returns the frequency measurement in the units selected by `MEAS:FREQ:UNIT`.

See MEAS:FREQ:UNIT.

ST remains on after this query has been executed. Press the SEND or SND key on the mobile handset to turn the ST off.

The phone may ring during this measurement. Do not answer the phone. Answering the phone could cause measurement errors.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values from CALC:LIM:FREQuency:ST.

Example: CALL:PAGE AMPS; *WAI
MEAS:FREQ:ERR:ST?
-30
indicates a measured ST frequency value that is 30 Hz lower than the nominal value.

MEASure:FREQuency:RF?

performs a mobile transmitter RF frequency measurement and returns the frequency in the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a call (AMPS, NAMPS, TDMA) or in service mode.

Response: <numeric floating point>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values from CALC:LIM:FREQuency:RF. The nominal value from the limit entry is offset to reflect the current channel frequency.

MEASure:FREQuency:SAT?

performs a SAT frequency measurement and returns the frequency in the units that are selected by `MEAS:FREQ:UNIT`.

See MEAS:FREQ:UNIT.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values from CALC:LIM:FREQuency:SAT. The nominal value from the limit entry is offset to reflect the current SAT selection for the channel.

MEASure:FREQuency:ST?

turns on the signaling tone (if the unit is up on a call), performs an ST frequency measurement and returns the ST frequency in the units that are currently selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

The phone may ring during this measurement. Do not answer the phone. Answering the phone could cause measurement errors.

The ST remains on after this query has been executed. The SEND or SND key on the phone handset must be pressed following this query in order to turn the ST off.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values from CALC:LIM:FREQuency:ST. The nominal value from the limit entry is offset by +10000 Hz.

MEASure:FREQuency:UNITs <unit select>
<unit select>MHZ, KHZ, HZ

selects the units to be used in reporting the frequency and frequency error measurements.

This command also selects the units that are printed when `PRIN:ITEM DATA` is used to print the results of the last measurement if the last measurement command was one of the frequency (or frequency error) measurements.

Response: <character data> MHZ, KHZ, or HZ

Example: MEAS:FREQ:UNIT?
Hz
indicates that the current measurement frequency units selection is Hz.

MEASure:POWer:DC?

performs a DC input power measurement for the phone and reports the value in Watts.

Requirements: none

Response: <numeric floating point>

Resolution: 0.1

Units: Watt

Limit checking: none

Example: MEAS:POW:DC?
4.500000
indicates that the DC input power is 4.5 Watts.

MEASure:POWer:TRANsmitter:BURSt <burst select>
<burst select>NORMal or SHORt

selects the burst type to be measured by a subsequent modulation envelope measurement query.

The unit is capable of capturing either a normal burst (162 symbols) or a shortened burst (140 symbols).

This command affects the following modulation envelope measurement query:

MEASure:POWer:TRANsmitter:MENVelope:[IMMediate]?

This command is valid only if the DCM option is installed.

MEASure:POWer:TRANsmitter:DROOp?

measures the mobile transmitter power droop over the TDMA burst.

Droop is the average change in amplitude per symbol measured over symbols 6 - 162.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.001

Units: dB

Limit checking: uses the values from CALC:LIM:POWer:TRANsmitter:DROOp.

Example: CALL:PAGE TDMA; *WAI
MEAS:POW:TRAN:DROO?
0.004
indicates a 0.004 dB droop.

MEASure:POWer:TRANsmitter[:IMMediate]?

performs a mobile transmitter power measurement and reports the value in the units that are currently selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

If the phone is on a TDMA call, this command returns the average power of the modulation envelope.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a call.

Response: <numeric floating point>

Resolution: 0.1 dBW/dBm or 0.1 mW

Units: dBW, dBm, Watt, mW
Limit checking: uses the values from CALC:LIM:POWer:MAC, with the MAC index value (0 - 10) determined by the current mobile power selection.

Example: MEAS:POW:TRAN:UNIT dBW
CALL:PAFGE; *WAI
MEAS:POW:TRAN?

4.200000
indicates transmitter power of +4.2 dBW.

MEASure:POWer:TRANsmitter:MENVelope:[IMMediate]?

measures and reports the mobile transmitter modulation envelope.

This consists of the mobile transmitter power level at each symbol throughout one TDMA burst from symbol 1 through symbol 168 inclusive.

If the burst currently selected by `MEAS:POW:TRAN:BURS` is `SHORt`, only symbols 1 through 148 are valid power measurements.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: string of comma-separated entries

The first entry represents the type of burst measured.

The remaining 168 entries represent the power level for each of the 168 symbols of the modulation envelope.

<character data>, <numeric floating point #1>, <numeric floating point#2>,..., <numeric floating point # 168>

where:

<character data>= LONG or SHORt
<numeric floating point # 1> = power measured at symbol 1
<numeric floating point # 2> = power measured at symbol 2
<numeric floating point # 168> = power measured at symbol 168

If the first entry is SHORt, then the last valid power measurement is <numeric floating point # 148>.

Resolution: 0.1

Units: dBW

Limit checking: uses the values from CALC:LIM:POWer:MAC, with the MAC index value (0 - 10) determined by the current mobile power selection.

Symbols 1 through 3 and 165 through 168 (143 through 148 for Shortened burst) must be less than the minimum power which the unit can measure.

Symbols 4 through 164 (142 for Shortened burst) must be less than the maximum.

Symbols 6 through 162 (140 for Shortened burst) must be greater than the minimum.

Example: LONG, -40.0, -40.0, -40.0, -40.0, -37.1, -23.1,...
The normal (or long) burst has the following modulation envelope:

symbol 1 = -40.0 dBW
symbol 2 = -40.0 dBW
symbol 3 = -40.0 dBW
symbol 4 = -40.0 dBW
symbol 5 = -37.1dBW
symbol 6 = -23.1 dBW
etc. through symbol 168.

MEASure:POWer:TRANsmitter:PEAK?

measures the mobile transmitter power over one TDMA burst and reports the peak power measured in the units that are currently selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.1 dBW/dBm or 0.1 mW

Units: dBW, dBm, Watt, mW
Limit checking: uses the values from CALC:LIM:POWer:TRANsmitter:PEAK.

Example: MEAS:POW:TRAN:UNIT dBW
CALL:PAGE TDMA; *WAI
MEAS:POW:TRAN:PEAK?
5.2
indicates mobile transmitter power of +5.2 dBW.

MEASure:POWer:TRANsmitter:UNITs <unit select>
<unit select>DBW, DBM, WATT, MW

selects the units to be used in reporting the transmitter power measurement.

This command also selects the units that will be printed when `PRIN:ITEM DATA` is used to print the results of the last measurement (if the last measurement command was `MEAS:POW:TRAN?)`.

Response: <character data> DBM, DBW, WATT, MW, or UW

Example: MEAS:POW:TRAN:UNITS?
MW
indicates that the current power units selection = mW.

MEASure:RSSI:CURRent?

instructs the mobile phone to perform a Received Signal Strength Indicator (RSSI) measurement on the current forward traffic channel and returns the RSSI code reported by the mobile phone.

RSSI is a code from 0 to 31 that indicates the signal power, as estimated by the mobile phone.

RSSI values are defined by IS-54-B as follows:

TABLE 23.

RSSI Value	Estimated Signal Strength
0	-113 dBm or less
1	-111 dBm
2	-109 dBm
3	-107 dBm
4	-105 dBm
5	-103 dBm
6	-101 dBm
7	-99 dBm
8	-97 dBm
9	-95 dBm
10	-93 dBm
11	-91 dBm
12	-89 dBm
13	-87 dBm
14	-85 dBm

TABLE 23.

RSSI Value	Estimated Signal Strength
15	-83 dBm
16	-81 dBm
17	-79 dBm
18	-77 dBm
19	-75 dBm
20	-73 dBm
21	-71 dBm
22	-69 dBm
23	-67 dBm
24	-65 dBm
25	-63 dBm
26	-61 dBm
27	-59 dBm
28	-57 dBm
29	-55 dBm
30	-53 dBm
31	-51 dBm or greater

NOTE: This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a call with the mobile.

Response: <numeric integer>

Units: none

Limit checking: uses the values from CALC:LIM:RSSI.

Example: CALL:PAGE; *WAI
MEAS:RSSI:CURR?
4
estimates that the current channel's signal strength at -105 dBm.

MEASure:RSSI:MAHO?

instructs the mobile phone to perform an Received Signal Strength Indicator (RSSI) measurement on the second carrier and returns the RSSI code reported by the mobile phone.

RSSI is a code from 0 - 31 that indicates the signal power as estimated by the mobile phone.

RSSI values are defined by IS-54-B and summarized in the `MEASure:RSSI:CURRent?` description.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call with the phone. The second carrier must be on and at a channel which the mobile phone can access. See SOURce:SCHannel:POWer:LEVel? and SOURce:SCHannel:POWer:CHANnel?

This command indicates an error if not on a call or in service mode.

Response: <numeric integer>

Units: none

Limit checking: uses the values from CALC:LIM:RSSI.

Example: CALL:PAGE TDMA; *WAI
MEAS:RSSI:MAHO?
5
indicates that the mobile estimates the second carrier's signal strength at -103 dBm.

MEASure:RZERo:FORWard?

measures the waveform quality (Rho) value sent by the 4300 to the mobile on the forward TDMA traffic channel.

The Rho value is a unitless quantity from 0 - 31 that represents the average signal power in the received speech over a 20 millisecond interval. Both the forward and the reverse Rho measurement values are determined when this command is received.

Use `READ:RZERo:REVerse` after `MEASure:RZERo:FORWard` to access the reverse Rho value for the corresponding forward Rho.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be on a TDMA call or in TDMA service mode and sync detected.

A settings conflict error (-221) is reported if not in the proper mode. A hardware missing error (-241) is reported if the DCM option is not installed.

The measurement is only valid if the phone transmits a valid burst and the unit is not in SILENT mode.

See CALL:VOICe:TYPE.

If either of these conditions is true, the Rho measurement status bit is set in the **QuestionableStatus** register.

See STATus:QUEStionable.

Response: <numeric integer>

Units: none

Limit checking: none

MEASure:RZERo:REVerse?

measures the waveform quality (Rho) value sent by the mobile to the 4300 on the reverse TDMA traffic channel.

The Rho value is a unitless quantity from 0 - 31 that represents the average signal power in the input speech over a 20 millisecond interval. Both the forward and the reverse Rho measurement values are determined when this command is received.

Use `READ:RZERo:FORWard` after `MEASure:RZERo:REVerse` to access the forward Rho value for this corresponding reverse Rho.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be on a TDMA call or in TDMA service mode and sync detected for the query to be executed.

A settings conflict error (-221) is reported if not in the proper mode. A hardware missing error (-241) is reported if the DCM option is not installed.

The measurement is only valid if the phone transmits a valid burst and the unit is not in SILENT mode (see CALL:VOICe:TYPE).

If either of these conditions is true, the Rho measurement status bit is set in the QuestionableStatus register

See STATus:QUEStionable.

Response: <numeric integer>

Units: none

Limit checking: none

MEASure:SENSitivity?

performs the same receiver sensitivity measurement defined by MEAS:SENS? <base power level>, with the exception that the base power level is defined by the nominal limit selection for one of the following command headers:

for AMPS: CALCulate:LIMit:SENSitivity[: AMPS]
for NAMPS:CALCulate:LIMit:SENSitivity:NAMPs
for TDMA:CALCulate:LIMit:SENSitivity:TDMA

If the `STATe` entry for the command header is set to `OFF`, the test is not performed and the response is `NONE`.

In order to minimize dropping the call during this test, set the power control to All Up (`CALL:PCONtrol:SELect UP`) prior to performing the BER test at reduced base power levels.

Requirements: Unit must be up on a call (AMPS, NAMPS, TDMA) or in service mode.

Response: PASS, FAIL, NONE

Limit checking:
For AMPS & NAMPS: none
For TDMA: uses the values from CALC:LIM:SENS:BER.

MEASure:SENSitivity? <base power level>
<base power level>numeric value -125.0 to -50.0 dBm

For AMPS and NAMPS:

performs a receiver sensitivity measurement by temporarily setting the base power level to the value specified and checking whether SAT is still present.

The base power is restored after the test completes. If SAT is present, the test passes, if not, it fails. No value is returned if an error occurs with the command due to the base power level specified.

Limit checking: none

For TDMA:

performs a receiver sensitivity measurement by temporarily setting the base power level to the value specified and performing a BER MAHO (channel quality) measurement.

The base power is restored after the test completes. If the BER measurement passes the limit checks, the test passes, if not, it fails. No value is returned if an error occurs with the command due to the base power level specified.

It is assumed that the 4300 is set for no induced BER (`SOURce:SQUAlity:IBER`) prior to executing this command.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a call (AMPS, NAMPS, TDMA) or in service mode.

If the unit was unable to switch to the selected power output for any reason, the error -221, "Settings conflict" is indicated, and the test will not be performed.

Response: PASS/FAIL

Limit checking: uses the values from CALC:LIM:SENSitivity:BER.

MEASure:SNDRatio:RECeiver?

performs a receiver SINAD measurement.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Limit checking: uses the values from CALC:LIM:SINad:RECeive.

Example: CALL:PAGE AMPS; *WAI
MEAS:SNDR:REC?
0.1
indicates that the receiver SINAD is 0.01 dB.

MEASure:SNDRatio:TRANsmitter?

performs a transmitter SINAD measurement.

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Limit checking: uses the values from CALC:LIM:SINad:TRANsmit.

Example: CALL:PAGE AMPS; *WAI
MEAS:SNDR:TRAN?
13.500000
indicates that the transmitter SINAD is 13.5 dB.

Limit checking: NONE:no limit check
HANDoff:CALC:LIM:DUR:HANDoff
RELease:CALC:LIM:DUR:RELease
FLASh:CALC:LIM:DUR:FLASh

MEASure:SRVMode:SPECtrum[:IMMediate]?

causes a spectrum measurement to be made centered at the channel selected by `CALL:SRVMode:CHANnel`.

It returns a pass/fail indication to indicate whether the measurement was successful.

To retrieve spectrum results using the `READ:SRVMode:SPECtrum` command, ensure that the 4300 is in spectrum mode.

See `CALL:SRVMode:SPECtrum:STATe ON`.

If the 4300 is not in spectrum mode, a "Settings conflict" error results.

Returns: 0 if measurement was successful
1 if measurement was not successful

MEASure:SRVMode:SPECtrum:PEAK[:IMMediate]

performs a peak search within the data captured by the last `MEASure:SRVMode:SPECtrum[:IMMediate]?` command.

The search is conducted within the frequency range set by `CALL:SRVMode:SPECtrum:PEAKsearch:OFFSet` and `CALL:SRVMode:SPECtrum:PEAKsearch:BWIDth`.

It returns a pass/fail indication to indicate if the peak search was successful.

See `CALL:SRVMode:SPECtrum:STATe ON`.

Returns: 0 if measurement was successful
1 if measurement was not successful

MEASure:STATus:SAT?

indicates whether SAT is on or off (1 = ON, 0 = OFF).

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer 0 or 1>

Units: none

Example: CALL:PAGE AMPS; *WAI
MEAS:STAT:SAT?
1
indicates that the SAT is ON.

MEASure:STATus:ST:CONDition?

returns an indication of whether ST is on or off (1 = ON, 0 = OFF).

Requirements: Unit must be up on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer 0 or 1>

Units: none

Example: CALL:PAGE AMPS; *WAI
MEAS:STAT:ST:COND?
1
indicates that the Signaling Tone is ON.

MEASure:STATus:ST:DURation? <limit type>
<limit type>NONE, HANDoff, RELease, FLASh

returns the last ST duration measurement in milliseconds.

If `NONE` is selected, no limit check will be performed, just like the query without an argument. The other selections represent the `CALC:LIM:DURation` limit parameters that are used for determining if the test passed or failed.

Requirements: Unit must be on a call (AMPS or NAMPS only) or in service mode.

Response: <numeric integer>

Units: milliseconds

MEASure:TALignment?

performs a time alignment measurement and returns the mobile transmitter's time alignment.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.125

Units: symbols

Limit checking: uses the values from CALC:LIM:TALignment.

Example: CALL:PAGE TDMA; *WAI
MEAS:TAL?
4.75
indicates that the time alignment is at 4.75 symbols from the standard offset reference.

If the mobile was given a time alignment command of 9, then this value represents a mobile with a 1/4 symbol error, since the ideal mobile would have a time alignment of 4.5 symbols.

MEASure:TALignment:ERRor?

performs a time alignment measurement, calculates, and returns the mobile transmitter's time alignment error.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.001

Units: symbols

Limit checking: uses the values from CALC:LIM:TALignment.

Example: CALL:PAGE TDMA; *WAI
MEAS:TAL:ERR?
0.25
indicates that the time alignment is in error by 1/4 symbol.

MEASure:VERRor:IQ?

measures and returns the I-Q values of each of 157 symbols over one TDMA burst.

Each I-Q pair maps to one symbol in the burst from symbol 6 - 162.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting valid bursts.

Response: string of comma-separated entries.

The first 157 entries represent the I value for symbols 6 - 162.

The remaining 157 entries represent the Q value for symbols 6 -162.

<I for symbol 6>, <I for symbol 7>, <I for symbol 8>,... <I for symbol 162>, <Q for symbol 6>, <Q for symbol 7>,... <Q for symbol 161>, <Q for symbol 162>

Resolution: 0.001

Units: percentage

Limit checking: none

Example: 1.0, 0.0, 0.707,...

where: I for symbol 6 = 1.0
I for symbol 7 = 0.0
I for symbol 8 = 0.707

MEASure:VERRor:MAGNitude:PEAK?

measures and returns the peak magnitude component of vector errors over one TDMA burst (symbols 6 - 162).

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values from CALC:LIM:VERRor:MAGNitude:PEAK.

Example: CALL:PAGE TDMA;*WAI
MEAS:VERR:MAGN:PEAK?

16.5
indicates that the peak magnitude error is 16.5%.

MEASure:VERRor:MAGNitude:RMS?

measures and returns the RMS of the magnitude component of vector errors over one TDMA burst (symbols 6 -162).

This command indicates an error if not on a call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values from CALC:LIM:VERRor:MAGNitude:RMS.

Example: CALL:PAGE TDMA;*WAI
MEAS:VERR:MAGN:RMS?
3.5
indicates that the RMS magnitude error is 3.5%.

MEASure:VERRor:NORMalized:IQ?

measures and returns the I-Q values of each of first 10 symbols of a burst for 10 TDMA bursts within a 1 minute interval.

Each I-Q pair maps to one symbol in the normalized burst information.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting valid bursts.

Response: string of comma-separated entries.

The first 100 entries represent the I values.

The remaining 100 entries represent the Q values.

Each set of 10 entries represent the I or Q values for symbols 6 -15 of one burst, as specified as follow:

TABLE 24.

Entry number	Burst number	I or Q values
1 - 10	1	I
11 - 20	2	I
21 - 30	3	I
31 - 40	4	I
41 - 50	5	I
51 - 60	6	I
61 - 70	7	I
71 - 80	8	I
81 - 90	9	I
91 - 100	10	I
101 - 110	1	Q
111 - 120	2	Q
121 - 130	3	Q
131 - 140	4	Q
141 - 150	5	Q
151 - 160	6	Q
161 - 170	7	Q
171 - 180	8	Q
181 - 190	9	Q
191 - 200	10	Q

<I for symbol 6 of burst 1>, <I for symbol 7 of burst 1>,
<I for symbol 8 of burst 1>,... <I for symbol 15 of burst 1>,
<I for symbol 6 of burst 2>, <I for symbol 7 of burst 2>,...

<I for symbol 15 of burst 10>,
<Q for symbol 6 of burst 1>, <Q for symbol 7 of burst 1>,...
<Q for symbol 15 of burst 10>

Resolution: 0.001

Units: none

Limit checking: none

Example: CALL:PAGE TDMA; *WAI
MEAS:VERR:NORM:IQ?
1.0, 0.0, 0.707,...

where: I for symbol 6 = 1.0 of first burst
I for symbol 7 = 0.0 of first burst
I for symbol 8 = 0.707

MEASure:VERRor:NORMalized:PEAK?

measures and returns the peak vector error during the first 10 symbols of a burst for 10 TDMA bursts within a 1 minute interval.

This command indicates an error if not on a TDMA all or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting valid bursts.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values from CALC:LIM:VERRor:NORMalized:PEAK.

Example: CALL:PAGE TDMA; *WAI
MEAS:VERR:NORM:PEAK?
24.6
indicates that the peak normalized vector error of the first 10 symbols over 10 bursts is 24.6%.

MEASure:VERRor:NORMalized:RMS?

measures and returns the RMS normalized error vector magnitude during the first 10 symbols of a burst for 10 TDMA bursts within a 1 minute interval.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting valid bursts.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values from CALC:LIM:VERRor:NORMalized:RMS.

Example: CALL:PAGE TDMA; *WAI
MEAS:VERR:NORM:RMS?
12.1
indicates that the RMS normalized vector error of the first 10 symbols over 10 bursts is 12.1%.

MEASure:VERRor:OOFFset?

measures and returns the origin offset of a TDMA burst.

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting valid bursts.

Response: <numeric floating point>

Resolution: 0.1

Units: dBc

Limit checking: uses the values from CALC:LIM:VERRor:OOFFset.

Example: CALL:PAGE TDMA; *WAI
MEAS:VERR:OOFF?
-48.7
indicates that the origin offset is -48.7 dBc.

MEASure:VERRor:PEAK?

measures and returns the peak error vector over one TDMA burst (symbols 6 -162).

Peak vector error is also called Peak Error Vector Magnitude (EVM).

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values from CALC:LIM:VERRor:PEAK.

Example: CALL:PAGE TDMA; *WAI
MEAS:VERR:PEAK?
14.6
indicates that the peak vector error is 14.6%.

MEASure:VERRor:PHASe:PEAK?

measures and returns the peak phase component of vector errors over one TDMA burst (symbols 6 -162).

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.01

Units: degrees

Limit checking: uses the values from CALC:LIM:VERRor: PHASe: PEAK.

Example: CALL:PAGE TDMA; *WAI
MEAS:VERR:PHAS:PEAK?
9.51
indicates that the peak phase error is 9.51 degrees.

MEASure:VERRor:PHASe:RMS?

measures and returns the RMS of the phase component of vector errors over one TDMA burst (symbols 6 -162).

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.01

Units: degrees

Limit checking: uses the values from CALC:LIM:VERRor: PHASe: RMS.

Example: CALL:PAGE TDMA; *WAI
MEAS:VERR:PHAS:RMS?
2.69
indicates that the RMS phase error is 2.69 degrees.

MEASure:VERRor:RMS?

measures and returns the RMS vector error over one TDMA burst (symbols 6 - 162).

RMS vector error is also called RMS Error Vector Magnitude (EVM).

This command indicates an error if not on a TDMA call or in service mode, or if the DCM option is not installed.

Requirements: Unit must be up on a TDMA call or in TDMA service mode with the phone transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values from CALC:LIM:VERRor:RMS.

Example: CALL:PAGE TDMA; *WAI
MEAS:VERR:RMS?
5.8
indicates that the RMS vector error is 5.8%.

MEASure:VOLTage:AC?

measures the AC RMS voltage on the Audio In front panel connector.

Requirements: none

Response: <numeric floating point>

Resolution: 0.001

Units: volts (RMS)

Limit checking: none

MEASure:VOLTage[:DC]?

performs a DC input voltage measurement for the phone and reports the value in volts.

Requirements: none

Response: <numeric floating point>

Resolution: 0.01

Units: volt

Limit checking: none

Example: MEAS:VOLT?
12.1
indicates that the DC input voltage to the phone is 12.1 volts.

MEMory Subsystem

The *Memory* subsystem consists of the commands that modify buffers or tables in the system memory.

MEMory:COPY:[NAME] <source table>, <dest table>
<source table>NONE, EIA, MFC1, MFC2, MFC3, MFC4, MFC5
<dest table>MFC1, MFC2, MFC3, MFC4, MFC5

copies the specified limit table selection to one of the custom limit tables.

The source can be any of the custom tables, the `EIA` table (fixed values defined by the `EIA` standard), or `NONE`.

The `NONE` entry sets all parameter values to the `EIA` standard, but disables the limit values.

MEMory:NSTates?

returns the total number of stored settings locations in the test equipment.

This value is fixed at 10.

Response: <numeric integer> (always 10)

MEMory:STATe:CATalog?

returns the list of stored setting labels.

All 10 labels are listed, separated by commas.

Empty locations without labels use commas as place markers.

Response: <ASCII string, 160 characters maximum>

Example 1: MEM:STAT:CAT?
",,,,,,,,,"
indicates that there are no setting labels assigned.

Example 2: ",,Setting #2,,,,Setting #6,,,Setting #9"
indicates that only locations 2, 6, and 9 have settings with corresponding label names assigned.

MEMory:STATe:DEFine <stored setting name>, <setting index>
<stored setting name>ASCII string, up to15 characters, enclosed in double quotes
<setting index> numeric integer 0 - 9

assigns a label, or name, to a stored setting location.

It performs the same function as `MEMory:STATe:REGister[n]:LABel`, except that the stored setting location to be named is passed as a second argument (following the label name) rather than as part of the command header.

MEMory:STATe:DEFine? <stored setting name>
<stored setting name>ASCII string, up to 15 characters, enclosed in double quotes

performs a search for the specified name in the list of all stored settings, and returns the corresponding stored setting index value if found.

If not found, no response is generated and a -292, "Referenced name does not exist" error is placed in the error queue.

Response: <numeric integer>

MEMory:STATe:REGister[*n*]:CLEar
where: *n* = 0 - 9

clears the specified stored setting location and its associated label.

The stored setting index is the 0-9 value selected in the command.

MEMory:STATe:REGister[*n*]:CLEar?
where: *n* = 0 - 9

returns a value that indicates whether the specified stored setting location has valid data in it or is empty.

The stored setting index is the 0-9 value selected in the command.

Response: <numeric integer>

1 if the location is empty
0 if the location contains a stored setting

MEMory:STATe:REGister[*n*]:DATE?
where: *n* = 0 - 9

returns the date that the specified stored setting location was saved.

The stored setting index is the 0-9 value selected in the command.

The response is in the form of year, month, day.

Response: <numeric integer>, <numeric integer>, <numeric integer>

Example: MEM:STAT:REG5:DATE?
1995,10,16
indicates that location 5 was stored on Oct. 16, 1995.

MEMory:STATe:REGister[*n*]:LABel <name>
where: *n* = 0 - 9
<name>ASCII string, maximum length of 15 characters enclosed in double quotes

assigns a label to a stored setting location.

The stored setting index is the 0-9 value selected in the command. The default name that is chosen when a `*SAV` is executed is "Setting #X" where X is the number of the stored setting location.

To make it easier to identify the locations that have valid settings, do not label empty locations (no stored setting saved).

Example: MEM:STAT:REG3:LAB "Bob N. setup"
sets the label for stored setting location 3 to "Bob N. setup".

OUTPut Subsystem

The *Output* subsystem commands pertain to functions that condition the outgoing signal (from *Source* subsystem) after it has been generated.

Conditioning is a function of filtering, biasing, frequency conversion, and attenuation.

OUTPut:AOUT <output select>
<output select>SOURce or RECeive

selects the audio out source accessed on the Audio Out jack on the 4300 front panel.

`SOURce` selects the 4300 internal audio generator to be used as a tone generator.

`RECeive` selects the audio signal received from the mobile phone's transmitter demodulated from the RF input.

Example: OUT:AOUT REC
runs the mobile transmitter audio to the Audio Out jack.

PRINt Subsystem

The **Print** subsystem commands allow the user to customize the printed output file.

PRINt:BOLD <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the boldface setting.

All commands that output to the print buffer after this command is issued print data in the selected typeface setting.

Exceptions:

Any measurement failure indications for `DATA, MDATa, NOMLimit, MINLimit, MAXLimit, STATus, CSTatus, or TStatus` values are bolded when a failure occurs.

PRINt:CLEAr

initializes the print buffer to all ASCII spaces.

Do this prior to sending data to the print buffer.

This command cannot be queried.

PRINt[:DATA]:JUSTify <justify select>
<justify select>LEFT, RIGHt, or DECimal

positions the characters of specific numeric items to be printed.

Since the length of the data string is variable, based on its value, and not known in advance, this command is used to "straighten" columns of numeric data.

Selected commands are affected for `PRINt:ITEM` and `PRINt:SELected:ITEM: QUERy, MQUEry, DATA, MDATa,`
`NOMLimit, MINLimit, MAXLimit.`

The operation of the justification selections include:

LEFT- the numeric data is left-justified at the specified column.

RIGHT- the numeric data is right-justified at the specified column.

DECIMAL - the numeric data will be positioned such that the decimal point position of the data (whether one is printed or not) is placed at the specified column.

Example: For a last measured data value of 23.4567:

PRIN:SEL:LINE 20; COL 32
PRIN:JUST LEFT;
:SEL:ITEM DATA;
:LINE:INCR PRIN:JUST RIGHT;
:SEL:ITEM DATA PRIN:SEL:LINE:INCR...

PRIN:JUST DECIMAL:SEL:ITEM DATA
prints the following data (where the '|' indicates column 32 and is not part of the printout):

```
|
23.4567
23.4567
23.4567
```

PRINt[:DATA]:MFACtor <mult factor>
<mult factor> numeric value -10e6 to +10e6, resolution 10e-6

selects which multiplication factor to use when printing the `MDATa` and `MQUEry` selections.

This multiplication value is used for printing the measured data or queried parameter in another unit, as long as the unit conversion is a simple multiplication factor within the stated range and resolution.

This multiplied value adjusts the displayed resolution automatically; if a given data value is 12.2 with a resolution of 1 digit and the multiplication factor is 0.001, the displayed data has a resolution of 4 digits: 0.0122.

PRINt[:DATA]:PPRefix <prefix selection>
<prefix selection>NONE, PLUS, or SPACe

sets the character to prepend to positive values.

An added character affects the character location of the printed data only if LEFT justification is selected.

This command only pertains to the following `PRINt:ITEM` and `PRINt:SELected:ITEM` selections: `QUERy, MQUEry, DATA, MDATa, NOMLimit, MINLimit, MAXLimit.`

Example: For a last measured data value of 23.4567:

```
PRIN:SEL:LINE 20; COL 32
PRIN:JUST LEFT
PRIN:PPREFIX NONE;
:PRIN:SEL:ITEM DATA
PRIN:SEL:LINE:INCR
PRIN:PPREFIX PLUS;
:PRIN:SEL:ITEM DATA
```

prints the following:

```
23.4567
+23.4567
```

PRINt[:DATA]:RESolution:AUTO <boolean>
<boolean>OFF or ON (0=OFF, 1=ON)

sets the number of digits resolution (to the right of the decimal point) to display for the following print `PRINt:ITEM` and `PRINt:SELected:ITEM` selections: `QUERy, MQUEry, DATA, MDATa, NOMLimit, MINLimit, MAXLimit.`

PRINt[:DATA]:RESolution[:DIGits] <number of digits>
<number of digits>numeric value 0 - 8

sets the number of digits resolution (to the right of the decimal point) to display for the following print `PRInt:ITEM` and `PRINt:SELected:ITEM` selections: `QUERy, MQUEry, DATA, MDATa, NOMLimit, MINLimit, MAXLimit.`

This selection is ignored if `PRINt:DATA:RESolution:AUTO` is set to ON.

PRINt:DEVice:LANGuage<printer selection>
<printer selection>Epsom or PCL

selects the printer control language for all printer communications.

TABLE 25.

EPSon	Epson compatible printers (typically dot-matrix printers)
PCL	Hewlett-Packard Printer Control Language level 3 or greater compatible printers (typically LaserJet or DeskJet printers)

Many printers use either DIP switches or a setup menu to select between two or more different printer control languages. Ensure that either Epson or PCL mode is selected on your printer.

PRINt:DUMP:AGRaph

dumps the results of the last autograph test to the printer.

The autograph dump consists of 3 blocks separated by horizontal lines:

- the header information (defined by `PRINt:HEADer`)
- the time/date, phone information, phone service information, etc.

- the autograph power level measurement information in compressed text.

  Up to 11 columns of data are displayed, one for each MAC level tested, with a header at the top defining the MAC level for that column.

  The channels tested are listed at the left margin, along with the equivalent frequency, and the power measurements are placed in the corresponding locations in this grid.

  The measurement units are selected by `DISP:UNIT:POW:TRAN`.

  A "P" or an "F" is placed after the measurement value to indicate its pass/fail status, and measurements that fail are also printed in bold.

  If this command is executed when no autograph test has been run, both the first and second blocks are printed, but only the channel and MAC level header line in the third block are printed.

  Any header line that has no data is omitted from the printout.

This command cannot be queried.

PRINt:DUMP[:ALL]

dumps the entire contents of the print buffer to the printer.

The print buffer contains one full page (60 lines) of formatted text.

This command cannot be queried.

PRINt:DUMP:LIMits

dumps the current selected limits table settings to the printer.

The limits dump consists of:

- a header that contains the name of the limit table, the manufacturer code and ESN range defined for the table, and the limit table version number.
- limit parameters, one parameter per line.

  All limit parameters (name, units, nominal, lower offset, and upper offset values) are listed, whether enabled or not.

  Limit values that are enabled are bolded, and those that are disabled are printed in normal text.

This command cannot be queried.

PRINt:DUMP:LINE <start line>, <stop line>

dumps the selected lines from the print buffer to the printer.

PRINt:DUMP:LOG

dumps the contents of the sequencer log to the printer after a sequence completes.

If logging to the printer was not enabled prior to running the sequence or if the printer ran out of paper while the sequence was running, the user can get a printout of what remains in the buffer using this command.

The buffer holds up to 100 lines of sequencer information (most log entries are one line per measurement). If the buffer overflows, the oldest log entries are thrown out, and only the most recent 100 lines are kept.

The log buffer is automatically cleared out when a sequence is started.

This command cannot be queried.

PRINt:FORMat <format selection>
<format selection> HEXadecimal, DECimal, or BINary

selects which format to display the ESN and SCM data in when outputting these items to the print buffer.

The affected print `ITEM` selections are: `MCODe, RSVD, ESN1, ESN2, SCM.`

PRINt:HEADer[*n*] <header text>
[*n*] specifies which header line is being defined
<header text>the text information to place in the header.

defines one line of the header. (A maximum of 10 lines with up to 160 characters per line can be defined.)

Enclose the text information in double quotes. If fewer than 10 lines are needed, set any unneeded line to a null string to prevent it from being printed.

The header information is preserved from power down to power up.

To load the header definition from a disk file, use `DISK:LOAD:HEADer`.

Example:

Four lines of text are defined, and the other 6 lines are set to null.

prin:head1 “CustomerName________________________"
prin:head2 “Address ______________________________"
prin:head3 “Vehicle type __________________________"
prin:head4 “———————————————————"
prin:head5 “”
prin:head6 “”
prin:head7 “”
prin:head8 “”
prin:head9 “”
prin:head10 “”

PRINt:HLINe <line number>, <character position>, <length>
<line number> numeric integer 1 - 60
<character position> numeric integer 1 - 160
<length> numeric integer 1 - 160

draws an underline of a specified length on a specified line, and beginning at (and to the right of) a specified column.

This is similar to `PRINt:SELected:HLINe`, except that the line number and starting position are selected, as opposed to using the `PRIN:SEL:LINE` and `PRIN:SEL:COL` parameter values.

PRINt:ITEM <line number>, <character position>, <item selection>
<line number>numeric value from 1 - 6
<character position>numeric value from 1 - 132
<item selection>the item to be printed

places the specified data into a specific location in the printer buffer.

See `PRINt:SELected:ITEM`.

This buffer information is sent to the printer when `PRINt:DUMP` is issued.

This command does not affect the currently selected line number or column used by `PRINt:SELected`.

Example: PRIN:ITEM 23,1,DATE
prints the current date at the 1st character position of line 23.

PRINt:OVERwrite <boolean>
<boolean>OFF or ON (0= OFF, 1 = ON)

enables or disables the overwrite mode of `PRINt:STRing` and `PRINt:SELected:STRing`.

When overwrite is enabled, the string information is placed at the specified location in the page buffer, and overwrites any data that exists at that location.

When overwrite is disabled, only those characters that do not overwrite a previously written character are placed in the buffer, allowing data to be placed in the appropriate locations on the page with overlaid text around it, without overwriting the data.

Example: CALL:PAGE; *WAI
PRIN:OVER ON
MEAS:POW:TRAN?
PRIN:ITEM 6, 15, DATA
PRIN:STR 6, 1,"Power level dBm"
PRIN:STR overwrites the data placed by PRIN:ITEM and is either placed before PRIN:ITEM, or broken up into two commands: one to place "Power level" in front of the

data, and the other to place “dBm” after it, to prevent it from overwriting the data.

A better alternative is to disable overwrite mode prior to writing the string, if the string is written after the data has been placed.

By changing PRIN:OVER ON to PRIN:OVER OFF on the first line, the data remains intact and the text before and after it is still written.

PRINt:PRESet

initializes all of the 4300 printer parameters to their corresponding default values and clears the print dump buffer to prevent inconsistent operation in printer outputs due to uninitialized printer parameters.

This command also initializes new printer parameter commands to default values.

Use this command in all 4300 program sequences that use `PRINt:DUMP[:ALL] or PRINt:DUMP:LINE`.

Place this command in front of any other printer command to ensure that all printer parameters are set to a specific value.

The following printer parameters are initialized by this command:

TABLE 26.

SCPI command	Default Value
PRINt:SELected:LINE[:VALue]	1
PRINt:SELected:COLumn[:VALue]	1
PRINt:SELected:COLumn:TAB:DEFine	1, 9, 17, 25, 33, 41, 49, 57
PRINt[:DATA]:JUSTify	LEFT
PRINt[:DATA]:RESolution[:DIGits]	2
PRINt[:DATA]:RESolution:AUTO	ON
PRINt[:DATA]:MFACtor	1.0
PRINt[:DATA]:PPRefix	NONE
PRINt:SIZE	NORMAL
PRINt:BOLD	OFF
PRINt:UNDerline	OFF
PRINt:OVERwrite	ON
PRINt:FORMat	HEXADECIMAL

PRINt:SELected:COLumn:TAB:BACKward

shifts the current character position to the last defined tab stop.

This command operates only if the current character position is after the last defined tab stop.

This command cannot be queried.

Example: PRIN:SEL:COL:TAB:DEF 8, 16, 21, 50, 65
PRIN:SEL:COL 27
PRIN:SEL:COL:TAB:BACK
PRIN:SEL:COL:TAB:BACK
sets the tab stops, and moves the current character position from 27 to 21 and finally to 16.

PRINt:SELected:COLumn:TAB:DEFine <tab stop1> [, tab stop2...]
<tab stopN>numeric value 1 - 80

defines the tab stop character positions.

At least 1 tab stop must be defined, and up to 8 tab stops can be defined using this command.

Tab stops are defined in ascending order.

Example: PRIN:SEL:COL:TAB:DEF 8, 16, 21, 50, 65
defines tab stops at positions: 8, 16, 21, 50, and 65.

PRINt:SELected:COLumn:TAB[:FORWard]

advances the current character position to the next defined tab stop.

This command operates only if the last tab position has not been exceeded.

This command cannot be queried.

Example: PRIN:SEL:COL:TAB:DEF 8, 16, 21, 50, 65
PRIN:SEL:COL 17
PRIN:SEL:COL:TAB
PRIN:SEL:COL:TAB
sets the tab stops and moves the current character position from 17 to 21, and finally to 50.

PRINt:SELected:COLumn:TAB:RESet

sets the current character position to the first defined tab stop.

This command cannot be queried.

PRINt:SELected:COLumn[:VALue] <char position>
<char position>numeric value 1 - 80

defines the character position in the print buffer at which to place data from all subsequent `PRINt:SELected:STRing` and `PRINt:SELected:ITEM` commands.

Example: PRIN:SEL:LINE 27
PRIN:SEL:COL 14
PRIN:SEL:STR "Units"
sets the line number to 27, character position to 14, and prints 'Units' at that location.

PRINt:SELected:HLINe <length>
<length>numeric integer 1 - 160

draws an underline of the specified length on the line specified by `PRIN:SEL:LINE`, and beginning at (and to the right of) the column specified by the `PRIN:SEL:COL` command.

The specified length is in units of character positions; the actual size depends on `PRIN:SIZE`.

This command is equivalent to setting the `PRIN:UNDerline ON` command prior to writing data or a string, except that it allows the underline to be enabled after the string and/or data item combinations have been written.

If a line contains both data and text (and a combination of those is to be underlined), or if several non-sequential lines of text are to be underlined, it is easier to use this command than to turn the `PRIN:UNDerline` command `ON` and `OFF` several times.

If `PRIN:STR` or `PRIN:ITEM` follows this command, the underline under those written portions is disabled if `PRIN:UNDerline` is `ON`.

PRINt:SELected:ITEM <item selection>
<item selection>the item to be printed

places the specified data into the current line and character position of the printer buffer.

The line number is specified by `PRIN:SEL:LINE`; the character position by `PRIN:SEL:COL`.

This buffer information is sent to the printer when `PRINt:DUMP[:ALL]` or `PRINt:DUMP:LINE` is issued.

Example: PRIN:SEL:ITEM DATE
prints the current date at a specified line and character position.

TABLE 27.

Selections:

TABLE 27.

DATE	the current date in MM-DD-YY (or DD-MM-YY)* format. If SYSTem:DATE:FORMat US is specified, the date will be in MM-DD-YY format (April 17, 1992 formatted as 04-17-92). If SYSTem:DATE:FORMat INT is specified, the date will be in DD-MM-YY format (April 17, 1992 formatted as 17-04-92).
VDATe	the current date in Month DD, Year (or DD Month, Year)* format. If SYSTem:DATE:FORMat US is specified, the date will be in Month DD, Year format (e.g. Apr 17, 1992). If SYSTem:DATE:FORMat INT is specified, the date will be in DD Month, Year format (e.g. 17 Apr, 1992).
TIME	the current time in HH:MM format. If SYSTem:TIME:FORMat HR is specified, the time will be in 24-hour format (e.g. 13:47). If SYSTem:TIME:FORMat AMPM is specified, the time will be in 12-hour format (e.g. 1:47 PM).
MCODe	the 8-bit manufacturer code from the mobile registration ESN value. The data format is selected by the PRINt:FORMat command: HEX: the serial number is displayed as 2 hexidecimal digits DEC: the serial number is displayed as 3 decimal digits BIN: the serial number is displayed as 8 binary digits
MNAMe	string consisting of the manufacturer's name corresponding to the manufacturer code value (MCODe) of the mobile registration ESN value. The string may be up to 40 characters in length.
RSVD	the 6-bit reserved field from the mobile registration ESN value. The data format is selected by the PRINt:FORMat command: HEX: the serial number is displayed as 2 hexidecimal digits DEC: the serial number is displayed as 2 decimal digits BIN: the serial number is displayed as 6 binary digits

TABLE 27.

ESN1	the 18-bit serial number from the mobile registration ESN value. The data format is selected by the PRINt:FORMat command: HEX: the serial number is displayed as 5 hexidecimal digits DEC: the serial number is displayed as 6 decimal digits BIN: the serial number is displayed as 18 binary digits
ESN2	the 6-bit reserved field and 18-bit serial number from the mobile registration ESN value are combined into a 24-bit value, with the reserved field being most significant. The data format is selected by the PRINt:FORMat command: HEX: the serial number is displayed as 6 hexidecimal digits DEC: the serial number is displayed as 8 decimal digits BIN: the serial number is displayed as 24 binary digits
SCM	the 5-bit station class mark from the mobile registration. bits 4,1,0 represent the Power Class I - VIII bit 3 represents the Transmission (continuous/discontinuous) bit 2 represents the Bandwidth (20 or 25 MHz) The data format is selected by the PRINt:FORMat command: HEX: the value is displayed as 2 hexidecimal digits DEC: the value is displayed as 3 decimal digits BIN: the value is displayed as 8 binary digit
TYPE	a string that indicates the type of phone that has been registered. One of the following strings will be printed: AMPS for AMPS-only type phones NAMPS if the phone is capable of NAMPS service TDMA if the phone is capable of TDMA service ? if an unknown phone type
PClass	a string of up to 4 Roman numeral chars (I - VIII) representing the Power Class value derived from the Station Class Mark.

TABLE 27.

TRANsmission	a string of 17-20 characters indicating the transmission and bandwidth characteristics of the phone from the mobile registration station class mark (SCM) value. The possible bit values and corresponding string information is as follows: bits 3,2 = 00: "Continuous 20 MHz" 01: "Continuous 25 MHz" 10: "Discontinuous 20 MHz" 11: "Discontinuous 25 MHz"
OPERation	a string of 12 chars indicating the number of channels of operation defined for the phone from the mobile registration station class mark (SCM) value. The bandwidth indication (bit 2) defines this bit: bit 2 = 0: "666 Channels" (20 MHz bandwidth) bit 2 = 1: "832 Channels" (25 MHz bandwidth)
MIN	the 13-character string representing the 10-digit telephone number of the phone decoded from the 34-bit Mobile Identification Number from a mobile registration. The string is formatted for the US standards: (XXX) XXX-XXX
SID	the 5-digit System IDentification number from the last mobile registration.
DCC	the Digital Color Code value from the last time a forward control channel was established.
DIALed	the phone number entered from the mobile phone for the last origination performed. This string is initialized to ASCII spaces at power up and is always 17 characters (digits) in length. It prints the same string that is returned from the command CALL:ORIG:DIAL?

TABLE 27.

DTMF	a list of the DTMF keys entered from the mobile phone since the MEAS:DTMF command was last executed. This string is initialized to 12 ASCII spaces at power up and when the MEAS:DTMF command is first issued. As DTMF keys are pressed, the corresponding ASCII characters for the keys received are placed in the string in the following order: "1234567890*#". This string contains the keys that have been pressed, but not the order in which they were pressed. It prints the same string that is returned from the command MEAS:DTMF?
HFLash	(TDMA only) the calling party phone number. This string is initialized to NULL at power up can contain up to 18 characters (digits). It prints the same string that is returned from the command CALL:HFLash[RECeive]:DATA?
QUERy	the value of the last query command issued that returns a numeric value. The number of digits resolution and positioning of the numeric value can be adjusted by the PRINt[:DATA]:*XXXX* commands.
MQUEry	this is the same as QUERy, except that the value is multiplied by the value defined by PRINt[:DATA]:MFACtor.
QENum	the enumerated string value of the last query command issued that returns an enumerated value. (e.g. PRIN:ITEM) has enumerated values of QUERy, MQUEry, QENum, etc.).
QBOOl	the status of the last query command issued that returns a boolean value. The status is displayed as ON or OFF.
DATA	the value of the last measurement command that returns a numeric value. The number of digits resolution and positioning of the numeric value can be adjusted by the PRINt[:DATA]:*XXXX* commands. If the measurement was within the limits specified by the corresponding CALCulate:LIMit command, or was not tested against limits, this value will print in the typeface selection defined by the PRINt:BOLD command. If the measurement failed the limit specifications and the limits were enabled, the value will be printed in BOLD. (RESult can be used in place of DATA)

TABLE 27.

DATA2	this is the value of the second measurement value for measurement commands that return an (xi) coordinate as a value. The only measurement commands that functions this way are the READ:VERR:IQ? and the READ:VERR:NORM:IQ? commands. In both cases, the I value is reported using DATA, and the Q value is reported using DATA2.
MTESt	a string of characters (25 or less) representing the name of the last measurement performed (or read using the READ commands). The name uses the same nomenclature as the sequence log to printer.
MDATa	this is the same as DATA, except that the value is multiplied by the value defined by PRINt[:DATA]:MFACtor.
MUNit	the units defined for the last measurement value (DATA). This is a 1 to 4 character string of uppercase alpha characters. (e.g. DBM, MHZ, WATT)
LIMit	a string of characters representing the current limit table selection made by CALCulate:LIMit:TYPE:LOAD. The possible values are: NONE, EIA, Custom1, Custom2, Custom3, Custom4, Custom5
NOMLimit	the nominal value defined by the corresponding CALCulate:LIMit command for the last measurement value. The number of digits, resolution, and positioning of the numeric value can be adjusted using PRINt[:DATA]:*XXXX*.
MAXLimit	the absolute maximum limit value defined by the corresponding CALCulate:LIMIT command for the last measurement value. If no limits exist or the limit is disabled, no value will be printed. The number of digits resolution and positioning of the numeric value can be adjusted using PRINt[:DATA]:*XXXX*. If the measurement value was less than this limit, this value will print in the typeface selection defined by PRINt:BOLD. If the measurement failed the limit, the value is printed in bold.

TABLE 27.

MINLimit	the absolute minimum limit value defined by the corresponding CALCulate:LIMIT command for the last measurement value. If no limits exist or the limit is disabled, no value will be printed. The number of digits resolution and positioning of the numeric value can be adjusted by PRINt[:DATA]:*XXXX*. If the measurement value is greater than this limit, the value will print in the typeface selection defined by PRINt:BOLD. If the measurement failed the limit, the value will be printed in bold.
STATus	a string representing the PASS/FAIL status of the last measurement query. If the measurement failed the limit, the value will be printed in BOLD (ignores the PRINt:BOLD selection for this entry). The possible values are: Pass, Fail, Not tested.
CSTatus	a string representing a cumulative PASS/FAIL indication for all of the measurements since the last CALC:LIM:FAIL:CUMulative command. This latter command is used to reset the cumulative PASS/FAIL flag to NOT TESTED, so that a group of commands can be accumulated as a single PASS/FAIL status. It will indicate FAIL if any of the measurements failed, NOT TESTED if none of the measurements was verified against limits, and PASS otherwise. FAIL will always be printed in BOLD.
TSTatus	the same as CSTatus, except it is cumulative from the beginning of a test sequence. There is no way to reset the flag, except by restarting the test sequence. It will indicate if any measurements failed in the sequence.
HEADer	all non-NULL strings defined by PRINt:HEADer. Up to 10 lines of 160 characters can be defined.

PRINt:SELected:LINE:INCRement <increment value>
<increment value> numeric value -60 to +60

increments (or decrements) the current line number selection for the print buffer.

A positive value increases the line number, and a negative value decreases the line number to allow a test sequence loop to begin at a fixed line number, and each iteration of the loop print on sequential lines.

It also allows a subroutine to modify the current line number to print an entry offset from the current line and to restore the value before returning.

If <increment value> is omitted, a +1 is assumed.

Example: PRIN:SEL:LINE:INCR
PRIN:SEL:STR "Measurement"
PRIN:SEL:LINE:INCR -1
increments the current line number, print 'Measurement' at that location, then restores the line number to its original value.

PRINt:SELected:LINE[:VALue] <line position>
<line position> numeric value 1 - 60

defines the line number of the print buffer at which to place data from all subsequent `PRINt:SELected:STRing` and `PRINt:SELected:ITEM` commands.

Example: PRIN:SEL:LINE 27
PRIN:SEL:STR "Units"
sets the line number to 27 and print 'Units' at that location.

PRINt:SELected:STRing <text string>
<text string>ASCII text up to 160 characters, enclosed in double quotes

places a text string into the current line and character position of the printer buffer.

The line number is specified by the `PRIN:SEL:LINE` commands, and the character position by the `PRIN:SEL:COL` commands.

This buffer information is sent to the printer when `PRINt:DUMP[:ALL]` or `PRINt:DUMP:LINE` is issued.

Example: PRIN:SEL:STR "This text will be printed"
prints "This text will be printed" at the specified line and character position.

PRINt:SIZE <width selection>
<width selection>the character width selection: NORMal, SMALl, or LARGe

selects the font width (the character height remains constant at 60 lines per page).

All commands that output to the print buffer after this command is issued print data in the specified font width.

The character position (column) specified when outputting a string or item is the character position in `NORMAL` mode characters.

NORMAL standard mode(80 characters per line)
SMALL compressed mode(132 characters per line)
LARGEexpanded mode(40 characters per line)

PRINt:STRing <line number>, <character position>, <text string>
<line number>numeric value from 1 - 60
<character position>numeric value from 1 - 132
<text string>ASCII text up to 160 characters, enclosed in double quotes

places a text string into a specific location in the printer buffer.

This buffer information is sent to the printer when `PRINt:DUMP` is issued.

This command does not affect the selected line number or column that is used by `PRINt:SELected`.

Example: PRIN:STR 10,25,"This text will be printed"
prints 'This text will be printed' on line 10, at character position 25.

PRINt:UNDerline <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the underline setting for characters printed.

All commands that output to the print buffer after this command is issued print data based on this setting.

The underline is a part of the character and does not affect characters on the line beneath the current one.

PROGram Subsystem

The *Program* subsystem consists of commands that load, execute or direct a program sequence, autograph test, or time alignment test.

PROGram:AGRaph <mode select>
<mode select>RUN, STOP, PAUSe

begins, terminates, or pauses execution of the autograph test.

If the test is not running, PAUSe has no effect.

If the test is currently paused, RUN acts as a CONTinue by exiting the PAUSe mode and continuing execution at the current channel, but restarting at the lowest MAC level enabled.

This command is not valid from within a program. Use it only as a remote command.

PROGram:AGRaph:CHANnel:BAND <band select>
<band select>CELLular or PCS

selects the BAND (800 MHz cellular or 1900 MHz PCS) in which to perform the Autograph test.

The band selection is made for the entire autograph test. It cannot switch between PCS and cellular bands during testing.

If CELLular is selected, the channel selection range is from 1 - 799 and 991 - 1023.

If the start or stop channels are set outside the range of the cellular band (while PCS band is selected) and the band selection is then changed to cellular, the start channel will be set to 100 and the stop channel will be set to 650 (the default values).

If PCS is selected, the start and stop channel range = 1 - 1199.

The PCS band selection is only allowed if the PCS option is installed.

PROGram:AGRaph:CHANnel:STARt <channel number>
<channel number> numeric integer from 1 - 1999

specifies the STARt channel number for the Autograph test.

STARt is value-coupled with the STOP channel number and must be a lower frequency than the STOP channel.

In the cellular band, channels 991-1023 are lower than channel 1.

PROGram:AGRaph:CHANnel:STEP <channel number>
<channel number> numeric integer from 1 - 100

specifies the STARt channel step size for the Autograph test.

PROGram:AGRaph:CHANnel:STOP <channel number>
<channel number> numeric integer from 1 - 1999 (excluding 800 - 989, 1024 - 1999 for cellular)

specifies the `STOP` channel number for the Autograph test.

The `STOP` channel is value-coupled with the `START` channel number and must be a higher frequency than the `START` channel.

In the cellular band, channels 991-1023 are lower than channel 1.

PROGram:AGRaph:CHANnel:TYPE <voice type>
<voice type>AMPS, TDMA

specifies which type of voice channel type to use for the Autograph test.

PROGram:AGRaph:DLOG <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-disk for the autograph test.

This command is similar to `PROGram:CUSTom:DLOG`, except that it affects the Autograph test sequence instead of the Custom tests, and the filename extension is .agr instead of .log.

PROGram:AGRaph:PAUSe <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the autograph test pause on measurement failure.

If the test pause is enabled, the test equipment enters the paused mode if the power level measurement failed the limit tests.

PROGram:AGRAPH:PLOG <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-printer for the Autograph test sequences.

This command is similar to `PROGram:CUSTom:PLOG`, except that it affects the Autograph test sequence instead of the Custom tests.

PROGram:AGRaph:MAC[*n*] <boolean>
PROGram:AGRaph:VMAC[*n*] <boolean>
PROGram:AGRaph:DMAC[*n*]<boolean>
PROGram:AGRaph:CMAC[*n*] <boolean>
[*n*] numeric integer from 0 to 10
<boolean>OFF or ON (0 = OFF, 1 = ON)

specifies the MAC levels for the Autograph test.

MAC level entries 8-10 are ignored if the mobile phone has a power class < IV.

Example: PROG:AGR:VMAC0 1; VMAC1 0; VMAC2 0; VMAC3 1
PROG:AGR:VMAC4 0; VMAC5 1; VMAC6 0; VMAC7 1
enables MAC levels 0, 3, 5, 7 for the autograph test.

PROGram:AUTO:DLOG <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-disk for the Automatic test sequences.

It is similar to `PROGram:CUSTom:DLOG`, except that it affects the Automatic test sequence instead of the Custom tests.

PROGram:AUTO:PAUSe <pause selection>
<pause selection>NONE, ALWays, FAILure

selects which conditions cause the Automatic test sequence to pause.

It is similar to `PROGram:CUSTom:PAUSe`, except that it affects the Automatic test sequence instead of the Custom tests.

PROGram:AUTO:PLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-printer for the Automatic test.

This command is similar to `PROGram:CUSTom:PLOG`, except that it affects the Automatic test sequence instead of the Custom tests.

PROGram:CUSTom:DLOG <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-disk for the custom test sequences.

When enabled, each measurement or call processing command that executes places a corresponding entry in the disk log file containing: the name of the test performed and the measurement results, limits, pass/fail status, or other pertinent information as a result of the test.

When the log is opened, the filename is temp.log. After the sequence completes, the filename is updated to reflect the ESN of the mobile phone under test.

The updated filename includes the first 7 digits of the last recorded ESN from a mobile phone (using the format defined by `CALL:MDATa:ESN:FORMat`) plus a single character A-Z, chosen so as to not overwrite any previous log file with the same ESN already existing on the disk.

The updated filename has the .log extension.

PROGram:CUSTom:NUMBer <custom select #>
<custom select #>numeric integer from 1 - 3

specifies which custom program sequence to access when `PROGram:NAME` is set to Custom.

This command works in conjunction with `PROG:STATe`, `PROG:DEFine`, and `DISK:LOAD` for starting, reading from, or writing to a specific custom sequence.

PROGram:CUSTom:PAUSe <pause selection>
<pause selection>NONE, ALWays, FAILure

selects which conditions cause the custom test sequence to pause.

If `NONE` is selected, the pause mode is only entered if a command error is encountered (invalid syntax, etc.).

If `ALWays` is selected, the sequence will pause after every command is executed.

If `FAILure` is selected, it will pause when a measurement command fails its limit checks.

When paused, the soft key definitions on the front panel change to allow the user to continue with the next test, repeat the last test, abort the sequence, or enter manual mode.

PROGram:CUSTom:PLOG <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the log to printer for the custom test sequences.

When enabled, each measurement or call processing command that executes outputs a corresponding entry to the printer containing the name of the test performed, and the measurement results, limits, pass/fail status, or other pertinent information as a result of the test.

The format of the data is the same as what is saved when log to disk is enabled.

PROGram:QUICk:DLOG <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-disk for the quick test sequences.

This command is similar to `PROGram:CUSTom:DLOG`, except that it affects the Quick test sequence instead of the Custom tests.

PROGram:QUICk:PAUSe <pause selection>
<pause selection>NONE, ALWays, FAILure

selects which conditions cause the quick test sequence to pause.

This command is similar to `PROGram:CUSTom:PAUSe`, except that it affects the Quick test sequence instead of the Custom tests.

PROGram:QUICk:PLOG <boolean>
<boolean>OFF or ON (0 =OFF, 1 = ON)

enables or disables the log-to-printer for the quick test sequences.

This command is similar to `PROGram:CUSTom:PLOG`, except that it affects the Quick test sequence instead of the Custom tests.

PROGram[:SELected]:BUFFer:CLEAr

sets all entries in the program buffer to the following specifications:

TABLE 28.

value	0.0
resolution	1.0
valid flag	INVALID
pass/fail flag	NOT TESTED

The valid and pass/fail flags are only set when measurements are logged to the buffer, and only accessible by the `BranchIf` commands in program sequences.

The internal buffer index value is reset to 0 (the first location in the buffer) so that measurement logging is placed at the start of the buffer.

PROGram[:SELected]:BUFFer:DATA <value> [, resolution]
<value>numeric value -1.0 e10 to +1.0 e10
[resolution](optional) numeric value 1.0 e-6 to 1.0

sets the next location in the sequencer log buffer to a specific value (and optionally to a specific resolution).

If no resolution is specified, the current resolution value at this location is used. The resolution is 1.0 for all buffer locations following a `PROG:BUFF:CLEAr`, and is modified by this command and when a measurement is logged to the buffer.

The buffer location affected by this command can be set directly with `PROG:BUFF:INDex`, or modified from its current location by `PROG:BUFF:NEXT and PROG:BUFF:PREV`.

Following a `PROG:BUFF:CLEAr`, the buffer index is set to the first buffer location (index 0).

When `PROG:BUFF:DATA` is used to place data in the buffer and a measurement is logged to the buffer, the buffer index automatically increments so that the next write to the buffer does not overwrite the data previously written.

The program buffer commands are meant to be used by sequence programs for capturing measurement data for use later in the program, for use in another program, or to pass to the remote port.

To place data in the buffer manually:

Example: PROG:SEL:BUFF:CLEAR
PROG:SEL:BUFF:MLIST MEAS
PROG:SEL:BUFF:MLOG ON
MEAS:POW:TRAN:UNIT WATT; IMM?
1.0326
CALL:VMAC?

3
PROG:SEL:BUFF:DATA 3
PROG:SEL:BUFF:PREV; PREV
PROG:SEL:BUFF:DATA 1032.6, 0.1
PROG:SEL:BUFF:NEXT; NEXT

The first three commands initialize the buffer and enable automatic logging of measurement values read.

The power measurement is placed in the first buffer location, and has an associated resolution of 0.0001 Watts.

The current MAC value is read from the 4300 and is not logged automatically, since it is not a MEASure command. It is placed there manually using the PROG:BUFF:DATA command.

The resolution is omitted, since it defaults to 1.0 following the clear command. The measurement value in the first location is changed from Watts to mW using the PROG:BUFF:DATA command. This time, the resolution value is assigned, since it is less for the smaller units.

Finally, the buffer index is incremented back to the next available location.

After these commands complete, the buffer contains two entries: 1036.6 (res 0.1), and 3.0 (res 1.0).

PROGram[:SELected]:BUFFer:DATA? <buffer index>, [offset]
<buffer index> numeric integer 0 - 399
[offset](optional) numeric integer 0 - 399

returns the value contained in the sequence log buffer at a specified index location.

If the optional [offset] value is not sent, the program buffer is treated as a single dimensional array and the <buffer index> selects which value to return.

If the optional [offset] value is sent, the array is treated as a 2-dimensional array with one of the dimensions set to the number of measurement entries specified by the `PROG:BUFF:MLISt` command.

The <buffer index> now represents the measurement index and the [offset] represents the offset to a selected piece of information collected during that measurement.

The actual buffer index is calculated by multiplying <buffer index> by the number of entries selected to be logged and added to [offset]. This command affects the BranchIfInvalid, BranchIfNotTested, BranchIfPass, and BranchIfFail sequence directives by using the flags that have been set by the measurement logging.

See `PROG:BUFF:MLOG` for a description of how these flags are set.

The resulting buffer index value must be within the range specified for each parameter.

Response: <numeric floating point>

Resolution: depends upon the data stored

Units: depends upon the data stored

Example:

```
PROG:BUFF:CLEAR
PROG:BUFF:MLIST MIN, MAX, MEAS
PROG:BUFF:MLOG ON
FOR MacIndex = 0, 7
CALL:PLCH:VMAC <MacIndex>
MEAS:POW:TRAN?
NEXT

FOR MeasIndex = 0, 7
PROG:BUFF:DATA? <MeasIndex>, 2
PRIN:SEL:ITEM QUERY; COL:TAB
PROG:BUFF:DATA? <MeasIndex>, 0
PRIN:SEL:ITEM QUERY; COL:TAB
PROG:BUFF:DATA? <MeasIndex>, 1
PRIN:SEL:ITEM QUERY; COL:TAB:RES
PRIN:SEL:LINE:INCR
NEXT

FOR MeasIndex = 0, 23
PROG:BUFF:DATA? <MeasIndex>
PRIN:SEL:ITEM QUERY; COL:TAB
PROG:BUFF:DATA? <MeasIndex>
PRIN:SEL:ITEM QUERY; COL:TAB
PROG:BUFF:DATA? <MeasIndex>
PRIN:SEL:ITEM QUERY; COL:TAB:RES
PRIN:SEL:LINE:INCR
NEXT
```

The program buffer is first initialized and set to collect 3 parameters (minimum limit, maximum limit, and measured value) for each measurement command executed.

8 mobile power level changes and a power measurement at each level are performed.

The second loop queries the information at each measurement and prints them on successive lines. The measurement value is printed at the first tab stop (MEAS = offset of 2), the minimum limit value at the second stop (MIN = offset of

0), and the maximum limit at the third tab stop (MAX = offset of 1).

The third loop performs the same task without using the <offset> parameter in PROG:BUFF:DATA?.

PROGram[:SELected]:BUFFer:FREE?

returns the number of locations available for measurement storage rather than the number of measurements that can be stored.

To determine this value, divide by the number of `PROG:BUFF:MLIST` selections and truncating the fractional portion of the result.

If this command is issued immediately after `PROG:BUFF:CLEAR`, the response indicates the size of the program buffer.

Response: numeric integer

PROGram[:SELected]:BUFFer:INDex <buffer index>
<buffer index>numeric integer 0 - 399

sets the internal buffer index to the specified value.

The buffer index is used:

- when logging measurements to the program buffer
- when writing data directly to the buffer with `PROG:BUFF:DATA`.

This command modifies the results of an earlier entry, or a retested measurement.

The query form of this command responds with the current index value setting, rather than the last value sent by the command form.

If the buffer index has not been modified by `INDex`, `NEXT` or `PREV` commands, this value represents the number of entries written to the program buffer.

To safeguard the integrity of the index value, ensure that the index is returned to its former value after it has been changed.

This allows other programs or remote accesses of the program buffer to know how many values have been placed in the buffer.

PROGram[:SELected]:BUFFer:MLISt <select1>, [select2], [select3], [select4]
<select1>character data MEASurement, NOMinal, MINimum, MAXimum
[select2-4](optional) same as <select1> along with NONE

selects which measurement information is placed in the program buffer when measurement logging is enabled by `PROG:BUFF:MLOG`.

A minimum of one selection must be made and a maximum of 4 entries are allowed.

The measurement information consists of the measurement value (`MEASurement`) and all of the limit values (`NOMinal, MINimum, and MAXimum`).

They can be in any order and the same entry can be duplicated. Any omitted entries are set to `NONE`.

When a measurement is performed, this list is scanned and the corresponding values are placed in the program buffer in the specified order.

The entries placed in the buffer are terminated at the first occurrence of `NONE` found in the list.

If `MEASurement` is listed twice and the measurement being logged has a 2-parameter response (`READ:VERR:IQ?` and `READ:VERR:NORM:IQ?` are the only 2-parameter response commands), the first occurrence returns the first response parameter (the I value) and the second occurrence returns the second response parameter (the Q value).

If the measurement does not have a 2-parameter response, the same measurement value is logged twice.

See `READ:LAST[:DATA]?` for a list of the `MEASure` and `READ` commands that log results.

See `READ:LAST:MIN` and `READ:LAST:MAX` for instructions on handling measurements without limits.

Example 1: PROG:BUFF:MLIST MEAS
only logs the measurement values.

Example 2: PROG:BUFF:MLIST MEAS, MIN, MAX
logs, in order, the measurement value followed by its minimum and maximum limits.

Example 3: PROG:BUFF:MLIST NOM, NOM, NONE, MEAS
logs the nominal limit value twice in successive locations. Since NONE occurs in the list prior to MEAS, the measurement value is not placed in the program buffer.

PROGram[:SELected]:BUFFer:MLOG <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the automatic logging of MEAS and READ command results to the sequencer log buffer.

If disabled (OFF), measurement results are not logged.

If enabled (ON), the measurement information specified by PROG:BUFF:MLIST is placed in the program buffer along with the resolution of the value, whether the value was valid, and the pass/fail status of the measurement.

The valid flag indicates INVALID on measurements that had an error condition (an accurate measurement was prevented from being made).

See READ:LAST:VALid for a description of this flag.

For MINimum and MAXimum selections, the valid flag indicates INVALID if the corresponding limit was not enabled. The valid flag indicates VALID for NOMinal selections.

The pass/fail status records whether the measurement PASSED or FAILED the limit check, or indicates NOT TESTED if the limits were disabled.

These flags are accessible to the program sequence following PROG:BUFF:DATA? of a selected entry using the following BranchIf commands:

```
BranchIfInvalid
BranchIfNotTested
BranchIfPass
BranchIfFail
```

PROG:BUFF:DATA? sets the resolution used for printing the value (with PRIN:SEL:ITEM QUERY) to the value recorded in the program buffer for the selected entry, if PRIN:RES:AUTO is set to ON.

As measurements are recorded, an internal buffer index is incremented to keep track of the next location to write. This value can be modified by PROG:BUFF:INDEX, PROG:BUFF:NEXT and PROG:BUFF:PREV.

CAUTION: Ensure that measurement information is not lost by writing to the correct location.

The measurement information in the buffer is not removed when a front panel RETEST of a measurement is performed during the sequence.

Logging of measurement information is disabled on power up of the tester.

If the program buffer is full, no more measurements are logged until either the buffer is cleared or the buffer index value is decremented.

PROGram[:SELected]:BUFFer:NEXT

increments the internal buffer index to the next entry.

Use the buffer index when logging measurements to the program buffer and writing data directly to the buffer with `PROG:BUFF:DATA`.

Use this command after modifying the results of an earlier entry or a retested measurement.

PROGram[:SELected]:BUFFer:PREV

decrements the internal buffer index to the previous entry.

Use the buffer index when logging measurements to the program buffer and writing data directly to the buffer with `PROG:BUFF:DATA`.

Use this command to modify the results of an earlier entry or a retested measurement.

PROGram[:SELected]:DEFine <string data>
<string data>ASCII string data enclosed in double quotes

uploads a program sequence to the test equipment.

The sequence may be up to 5000 characters in length, including the newline characters. The entire string is enclosed in double quotes.

The sequence location for this information is defined by `PROG:NAME`.

The query form of this command returns the currently selected sequence data enclosed in double quotes.

This is equivalent to downloading a program sequence from the remote port.

PROGram[:SELected]:DELay <delay time>
<delay time>numeric floating point from 0.00 - 10.00 (resolution of 0.01)

introduces a delay time (0 -10 seconds) into the program execution sequence.

This command is not valid through the remote port. It can only be used from within a program.

PROGram[:SELected]:FCOUnt?

returns the number of total number of test failures for the last program sequence.

Response: <numeric integer>

Example: PROG:FCOU?
16
indicates that the last program sequence that was executed failed 16 measurement and/or call processing tests.

PROGram[:SELected]:NAME <program select>
<program select>AUTO, QUICk, CUSTom

selects a program sequence to be run, written to, or read from.

It does not run the program.

This command works in conjunction with `PROG:STATe`, `PROG:DEFine`, and `DISK:LOAD`.

If `CUSTom` is the selection for this parameter, `PROG:CUSTom:NUMBer` specifies which of the Custom sequences is accessed.

PROGram[:SELected]:PAUSe

causes the currently executing program to enter the `PAUSed` state, and stops executing commands from the program sequencer.

A special set of front panel soft keys is displayed to allow the user to select CONTINUE when appropriate.

When the CONTINUE soft key is pressed, execution continues at the next command in the program. This command is useful if user intervention is required at some point during the program execution.

This command is not valid through the remote port. It can only be used from within a program.

PROGram[:SELected]:STATe <mode select>
<mode select>RUN, STOP

begins or terminates execution of the program sequence selected by the `PROG:NAME` command.

Although `PROG:STAT STOP` can be issued to stop execution of a currently running sequence, any remote command has the same effect.

PROGram[:SELected]:STRing[1-2] <user message>
<user message>ASCII string enclosed in double quotes, maximum length = 30 characters

defines the text to place in two user-definable lines near the bottom of the LCD front panel display.

This message window is located just above the 3-line unit status window at the bottom of the display and just below the test results window.

The user-defined message window is erased when a program is first started, and when it completes. If a user message is defined, it is displayed in inverse video.

Specifying a null string erases any displayed user messages.

Example: PROG:STR1 “Make sure mobile has service”
PROG:STR2 “Press CONTINUE when ready”
PROG:PAUSe
PROG:STR1 “”

PROG:STR2 “”
This command is used in conjunction with the PROGram:PAUSe command to allow the user to read the message, take the required actions, and press the CONTINUE key.

The second set of definitions for the user-defined message clears the message out (assuming the user has pressed the CONTINUE key).

This command is not valid through the remote port. It can only be used from within a program.

PROGram[:SELected]:TCOUnt?

returns the total number of tests executed for the last program sequence that was run.

Response: <numeric integer>

Example: PROG:TCOU?
512
indicates that the last program sequence that was executed performed a total of 512 measurement and/or call processing tests.

PROGram:TALignment:COUNt?

queries the number of time alignment values that are defined in the time alignment test.

This command is only valid if the DCM option is installed.

Example: PROG:TAL:DATA 5, 3, 17, 3, 5
PROG:TAL:COUN?
5
indicates there are 5 time alignment values specified in the time alignment test sequence.

PROGram:TALignment:DATA <time align 1> [,.....<time align 32]
<time align 1> numeric integer value 0 - 31 for the 1st time alignment value
<time align 2> numeric integer value 0 - 31 for the 2nd time alignment value
<time align 32> numeric integer value 0 - 31 for the 32nd time alignment value

sets the time alignment sequence for the time alignment test.

The test sequences through the values specified, in the order they are defined.

The sequence can be 1 - 32 time alignment values in length.

Time alignment values do not have to be unique.

Example: PROG:TAL:DATA 5, 3, 17, 3, 5
sets up the time alignment test sequence to cycle through the 5 time alignment values listed.

PROGram:TALignment:DELay <time delay>
<time delay>numeric value from 0.0 - 5.0 (resolution 0.01) seconds

sets the amount of delay that the time alignment test waits between changing the time alignment value and making a time alignment measurement to allow for settling of the mobile phone hardware.

READ Subsystem

The *Read* subsystem commands obtain a previously made measurement value without performing another measurement.

These commands are in the same format as their *Measure* subsystem equivalents.

`PRINt:ITEM DATA, MDATa, and MUNit` reflect the `READ` command measurement values which allow a measurement to be made once, and the data to be read or printed in different units without taking another measurement.

The `READ` commands do not require the test equipment to be in any particular state in order to execute, since they are simply returning a previously measured parameter.

READ:BER:ANALog?

returns the last analog BER measurement performed by `MEAS:BER?` while on an AMPS call.

If the unit is not on an AMPS call when the `MEAS:BER?` command is performed, the analog BER measurement value is not updated.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric integer>

Resolution: 0.01

Units: percentage

Measured by: MEAS:BER

READ:BER:LOOPback?

returns the last loopback BER measurement performed by `MEAS:BER?` while in TDMA service mode.

If the test equipment is not in TDMA service mode when `MEAS:BER?` is performed, the loopback BER measurement value is not updated.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Requirement: Test equipment must be in TDMA service mode with the DCM option installed.

Response: <numeric floating point>

Resolution: 0.01

Units: percentage

Measured by: MEAS:BER

READ:BER:MAHO?

returns the last MAHO BER measurement performed by `MEAS:BER?` while on a TDMA call.

If the unit is not on a TDMA call when `MEAS:BER?` is performed, the MAHO BER measurement value is not updated.

The MAHO BER measurement is the 3-bit BER code reported by the mobile phone. Each of the 8 values represent a range of BER percentage values.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Requirement: Test equipment must be in TDMA service mode with the DCM option installed.

Response: <numeric integer> (mobile reported BER code: 0-7)

Units: none

Measured by: MEAS:BER
MEAS:RSSI:CURR
MEAS:RSSI:MAHO

READ:DEViation:PAUDio?

returns the last peak audio deviation measurement value.

The peak audio deviation value is reset to 0 at the beginning of `MEAS:DEV:AUDio?` and `MEAS:DEV:PAUD <value>` commands.

`MEAS:DEV:AUDio?` takes a single reading and sets the peak audio value to this value.

`MEAS:DEV:PAUDio` makes audio deviation measurements at approximately 300 millisecond intervals for the specified amount of time, updating the peak reading each time during this interval.

`READ:DEV:PAUDio?` can be issued at any time (even prior to completion) to retrieve this information.

To postpone reading the measurement until after the specified amount of time following `MEAS:DEV:PAUD <value>`, send `*WAI` or `*OPC?` prior to `MEAS:DEV:PAUD?` with the appropriate bit (`PEAK DEVIATION` value = bit 7 = 80 hexidecimal = 128 decimal) set in the `STATus:OPERation:COMPlete:ENABle` command.

Response: <numeric integer>

Units: Hz

Measured by: MEAS:DEV:PAUD

READ:FREQuency:ERRor:RF?

returns the last mobile transmitter RF frequency error measurement value set by either `MEAS:FREQ:RF?` or `MEAS:FREQ:ERR:RF?`, using the units that are currently selected by `MEAS:FREQ:UNIT`.

See MEAS:FREQ:UNIT.

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:RF
MEAS:FREQ:ERR:RF

READ:FREQuency:ERRor:SAT?

returns the last SAT frequency error measurement value set by either `MEAS:FREQ:SAT?` or `MEAS:FREQ:ERR:SAT?`, using the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT.`

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:SAT
MEAS:FREQ:ERR:SAT

READ:FREQuency:ERRor:ST?

returns the last ST frequency error measurement value set by either `MEAS:FREQ:ST?` or `MEAS:FREQ:ERR:ST?`, using the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:ST
MEAS:FREQ:ERR:ST

READ:FREQuency:RF?

returns the last mobile transmitter RF frequency measurement value set by either `MEAS:FREQ:RF?` or `MEAS:FREQ:ERR:RF?`, using the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT.`

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:RF
MEAS:FREQ:ERR:RF

READ:FREQuency:SAT?

returns the last SAT frequency measurement value set by either `MEAS:FREQ:SAT?` or `MEAS:FREQ:ERR:SAT?`, using the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT.`

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:SAT
MEAS:FREQ:ERR:SAT

READ:FREQuency:ST?

returns the last ST frequency measurement value set by either `MEAS:FREQ:ST?` or `MEAS:FREQ:ERR:ST?`, using the units selected by the `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT.`

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:ST
MEAS:FREQ:ERR:ST

READ:LAST[:DATA]?

returns the measured value of the last measurement performed.

All of the `MEASure` and `READ` commands update this value except for the following:

READ:LAST:xxxx (any of these commands)
MEASure:DTMF? (the query form only)
MEASure:FREQuency:UNITs
MEASure:POWer:TRANsmitter:UNITs
MEASure:POWer:TRANsmitter:BURSt
MEASure:BER:MINTerval
MEASure:BER:ANALog:SAMPles

Some of the measurement commands have only a pass/fail status associated with the test. For these commands the value is set to 0.0:

MEASure:STATus:SAT?
MEASure:STATus:ST:CONDition?

Some of the measurement commands return a non-numeric result, or multiple numeric values. The last measurement data value for these commands is as follows:

TABLE 29.

MEASure:DTMF (the command form only)	returns the value 0.0.
MEASure:SENSitivity? (for AMPS and NAMPS)	returns the base power level in dBm at which the test was performed.
(for TDMA)	returns the BER value (0 to 7) that was measured during the base power reduction.
MEASure:BER?	returns the BER measurement value without the type (ANAL, LOOP or MAHO).

TABLE 29.

MEASure:POWer:TRANsmitter: MENVelope[:IMMediate]?	returns the average burst power for symbols 6 to 162 (6 to 140 for shortened bursts).
MEASure:VERRor:IQ?	returns the value 0.0.
MEASure:VERRor:NORMalized:IQ?	returns the value 0.0.
READ:VERRor:IQ?	returns the value of I for the specified symbol.
READ:VERRor:NORMalized:IQ?	returns the value of I for the specified sample.

Response: <numeric floating point>

READ:LAST:MAXimum

returns the absolute maximum limit value associated with the last measurement performed, and whether the maximum limit was enabled or not.

If the maximum limit was disabled for the measurement, or if there is no limit or measurement value defined, the value returned is 0.0 and the status is `OFF`.

Response: <numeric floating point>, <character data>
where: <character data> OFF or ON

READ:LAST:MINimum

returns the absolute minimum limit value associated with the last measurement performed, and whether the minimum limit was enabled or not.

If the minimum limit was disabled for the measurement, or if there is no limit or measurement value defined, the value returned is 0.0 and the status is `OFF`.

Response: <numeric floating point>, <character data>
where:<character data> OFF or ON

Example: MEAS:DEV:SAT?
1734
READ:LAST:MIN?
1800,ON
indicates that the minimum SAT deviation is 1800 Hz, and that the
minimum limit was enabled.

READ:LAST:NAME

returns the name associated with the last measurement performed as follows:

TABLE 30.

MEAS:ATIM?	"Acq Time"
MEAS:BER?	one of the following, depending on mode: "Analog BER" "Bit Error Rate" "Loopback BER"
MEAS:CURR[:DC]?	"DC Current"
MEAS:DEV:AUD?	"Audio Deviation"
MEAS:DEV:DSAT?	"DSAT Deviation"
MEAS:DEV:DST?	"DST Deviation"
MEAS:DEV:PAUD?	"Peak Deviation"
MEAS:DEV:RES?	"Residual Deviation"
MEAS:DEV:SAT?	"SAT Deviation"
MEAS:DEV:ST?	"ST Deviation"
MEAS:DEV:WBAN?	"Wide Band Dev"
MEAS:DTMF	"DTMF Digits"
MEAS:FREQ:ERR:RF?	"Tx Freq Error" (for AMPS/NAMPS) "Freq Offset" (for TDMA)
MEAS:FREQ:ERR:SAT?	"SAT Freq Error"
MEAS:FREQ:ERR:ST?	"ST Freq Error"
MEAS:FREQ:RF?	"Tx Freq"
MEAS:FREQ:SAT?	"SAT Frequency"
MEAS:FREQ:ST?	"ST Frequency"
MEAS:POW[:DC]?	"DC Power"
MEAS:POW:TRAN:DROO?	"Droop"
MEAS:POW:TRAN[:IMM]?	"Tx Power (MAC *num*)"*
MEAS:POW:TRAN:MENV:ATT?	"Mod Envelope Attack"
MEAS:POW:TRAN:MENV[:IMM]?	"Mod Envelope"
MEAS:POW:TRAN:MENV:REL?	"Mod Envelope Release"
MEAS:POW:TRAN:PEAK?	"Peak Power (MAC *num*)"*

TABLE 30.

MEAS:POW[:DC]?	"DC Power"
MEAS:RSSI:CURR?	"RSSI (Current)"
MEAS:RSSI:MAHO?	"RSSI (MAHO)"
MEAS:RZER:FORW?	"Forward R0 (Mobile Rx)"
MEAS:RZER:REV?	"Reverse R0 (Mobile Tx)"
MEAS:SENS?	"Rx Sensitivity"
MEAS:SNDR:REC?	"Rx Sinad"
MEAS:SNDR:TRAN?	"Tx Sinad"
MEAS:STAT:ST:COND?	"ST Status"
MEAS:STAT:ST:DUR?	"ST Duration"
MEAS:STAT:SAT?	"SAT Status"
MEAS:TAL?	"Time Alignment"
MEAS:TAL:ERR?	"Time Align Error"
MEAS:VERR:IQ?	"Symbol Constellation"
MEAS:VERR:MAG:PEAK?	"Peak Mag Error"
MEAS:VERR:MAG:RMS?	"RMS Mag Error"
MEAS:VERR:NORM:PEAK?	"Norm Peak EVM"
MEAS:VERR:NORM:RMS?	"Norm RMS EVM"
MEAS:VERR:OOFF?	"Origin Offset"
MEAS:VERR:PEAK?	"Peak EVM"
MEAS:VERR:PHASE:PEAK?	"Peak Phase Error"
MEAS:VERR:PHASE:RMS?	"RMS Phase Error"
MEAS:VERR:NORM:PEAK?	"Norm Peak EVM"
MEAS:VERR:RMS?	"RMS EVM"
MEAS:VOLT[:DC]?	"DC Voltage"
MEAS:VOLT:AC?	"AC Voltage"
The `READ` commands that correspond to the above measure commands produce the same test names with the following exceptions:	
READ:POW:TRAN:MENV:ATT?	"Mod Env Att [*sym*, *sample*]"
READ:POW:TRAN:MENV[:IMM]?	"Mod Envelope [*sym*]"

TABLE 30.

READ:POW:TRAN:MENV:REL?	"Mod Env Rel [*sym*, *sample*]"
READ:VERR:IQ?	"Symbol I/Q [*sym*]"
READ:VERR:NORM:IQ?	"Normalized I/Q [*sample*, *sym*]"**

* *num* = mobile power setting at the time of the measurement (0-10)
*******sym* = symbol # for measurement (1 to 170)
sample = burst sample (1 to 16)
Response: <ASCII string up to 32 characters>

READ:LAST:NOMinal

returns the nominal limit value associated with the last measurement performed.

If there is no limit or measurement value defined, the value returned is 0.0.

Response: <numeric floating point>

READ:LAST:RESolution

returns the resolution of the measured value of the last measurement performed.

Response: <numeric floating point>

Example: MEAS:VOLT:DC?
11.8
READ:LAST:RES?
0.01
indicates that a DC voltage measurement of 11.8 volts with a resolution of 0.01 volts.

READ:LAST:STATus

returns the pass/fail status of the last measurement performed.

Response: <numeric integer>
0 = passed
1 = failed
2 = measurement was aborted by the user
3 = not tested (limits were not enabled)

READ:LAST:UNITs

returns the units associated with the value of the last measurement performed.

If there are no units associated with the measurement (or where there is no measurement value), `NONE` is returned.

A list of the unit identifiers returned along with the type of measurements each is associated with includes:

TABLE 31.

"AMP"	current
"DB"	SINAD, power droop
"DBC"	vector error origin offset
"DBM"	power, sensitivity
"DBW"	power
"DEG"	vector error phase
"HZ"	frequency, deviation
"KHZ"	frequency
"MHZ"	frequency
"MS"	time, duration
"MW"	power
"PCT"	vector error magnitude, BER
"SYM"	time alignment
"VOLT"	voltage
"WATT"	power

Response: <ASCII string up to 12 characters>

READ:LAST:VALid

returns a flag indicating if the last measurement performed was valid.

When a measurement begins, this value is set to 0 (indicating that the measurement value is not valid).

When a measurement completes, conditions are checked to determine if the measurement value is accurate.

If there are no questionable conditions, this value is set to a 1 (indicating that the data is ready to be read).

If a questionable condition exists, use `STATus:QUEStionable` to identify the cause of the measurement inaccuracy.

Response: <numeric integer, 0 or 1>
where:0 = invalid
1 = valid

Measured by: MEAS:DEV:PAUD

READ:POWer:TRANsmitter:DROOp?

returns the last mobile transmitter power `DROOp` measurement over one TDMA burst.

Droop is the average change in amplitude per symbol measured over symbols 6 - 162.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.001

Units: dB

Measured by: MEAS:POW:TRAN (if on a TDMA call)
MEAS:POW:TRAN:PEAK
MEAS:POW:TRAN:DROO
MEAS:POW:TRAN:MENV

READ:POWer:TRANsmitter:PEAK?

returns the last mobile transmitter power `PEAK` measurement over one TDMA burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.001

Units: dB

Measured by: MEAS:POW:TRAN (if on a TDMA call)
MEAS:POW:TRAN:PEAK
MEAS:POW:TRAN:DROO
MEAS:POW:TRAN:MENV

READ:POWer:TRANsmitter[:IMMediate]?

returns the last mobile transmitter power measurement value set by `MEAS:POW:TRAN?` in units selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

Response: <numeric floating point>

Resolution: 0.1 dB or 0.1 mW

Units: dBW, dBm, Watt, mW

Measured by: MEAS:POW:TRAN[:IMM]
MEAS:POW:TRAN:PEAK
MEAS:POW:TRAN:DROO
MEAS:POW:TRAN:MENV

READ:POWer:TRANsmitter:MENVelope[:IMMediate]? <symbol index>
<symbol index>numeric integer value from 0 - 167

reads the power level at a specific symbol in the modulation envelope.

The symbol index value is equal to the symbol number minus 1.

The modulation envelope must have been previously collected using `MEASure:POWer:TRANsmitter:MENVelope[:IMMediate]?`.

The response is in units selected by MEAS:POW:TRAN:UNIT.

See MEAS:POW:TRAN:UNIT.

Response: <numeric floating point>

Resolution: 0.1 dB or 0.1 mW

Units: dBW, dBm, Watt, mW

Measured by: MEAS:POW:TRAN[:IMM]
MEAS:POW:TRAN:PEAK
MEAS:POW:TRAN:DROO
MEAS:POW:TRAN:MENV[:IMM]

Example 1: MEAS:POW:TRAN:MENV?
responds with a string of comma-separated entries representing the power level for each of the 168 symbols of the modulation envelope.

Example 2: READ:POW:TRAN:MENV? 45
responds with a single numeric value that represents the power at symbol 46 from the modulation envelope data.

READ:RSSI:CURRent?

returns the previously saved current forward traffic channel Received Signal Strength Indicator (RSSI) measurement code reported by the mobile.

RSSI is a code from 0 - 31 that indicates the signal power as estimated by the mobile.

See `MEASure:RSSI:CURRent?` for a description of RSSI values.

See IS-54B for a definition of RSSI values.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric integer>

Units: none

Measured by: MEAS:BER (when measured on a TDMA call)
MEAS:RSSI:CURR
MEAS:RSSI:MAHO

READ:RSSI:MAHO?

returns the last second carrier Received Signal Strength Indicator (RSSI) measurement code reported by the mobile.

RSSI is a code from 0 - 31 that indicates the signal power as estimated by the mobile.

See `MEASure:RSSI:CURRent?` for a description of RSSI values.

See IS-54B for a definition of RSSI values.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric integer>

Units: none

Measured by: MEAS:BER (when measured on a TDMA call)
MEAS:RSSI:CURR
MEAS:RSSI:MAHO

READ:RZERo:FORWard?

returns the last forward waveform quality (Rho) value measured.

The Rho value is a unitless quantity from 0 – 31 that represents the energy of the burst.

Response: <numeric integer>

Units: none

Measured by: MEAS:RZER:FORW
MEAS:RZER:REV

READ:RZERo:REVerse?

returns the last reverse waveform quality (Rho) value measured.

The Rho value is a unitless quantity from 0 - 31 that represents the energy of the burst.

Response: <numeric integer>

Units: none

Measured by: MEAS:RZER:FORW
MEAS:RZER:REV

READ:SRVMode:SPECtrum:PEAK:POWer?

returns the power in dBm of last peak measured by the last `MEASure:SRVMode:SPECtrum:PEAK[:IMMediate]?` command.

READ:SRVMode:SPECtrum:PEAK:OFFSet?

returns the frequency offset from the center of the channel to the last peak measured by the last `MEASure:SRVMode:SPECtrum:PEAK[:IMMediate]?` command.

READ:SRVMode:SPECtrum:POWer? <offset>
<offset>numeric value –19.95 to +19.95, resolution 0.19

returns the power at the specified frequency offset (in kHz) as measured by the last `MEASure:SRVMode:SPECtrum[:IMMediate]?` command.

READ:TALignment:ERRor?

returns the results of last time alignment error measurement performed.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.001

Units: symbols

Measured by: MEAS:TAL
MEAS:TAL:ERR

READ:TALignment[:IMMediate]?

returns the results of last time alignment measurement performed.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.125

Units: symbols

Measured by: MEAS:TAL
MEAS:TAL:ERR

READ:VERRor:IQ? <symbol>
<symbol>numeric integer value from 6 - 162

returns the I-Q position of a selected symbol for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst. Printer access to the I and Q values and name of this measurement for the selected symbol are available following this command.

Limit checking is not performed on this parameter.

Response: <numeric floating point>, <numeric floating point>

Resolution: 0.001

Units: none

Measured by: MEAS:VERR:RMS
MEAS:VERR:PEAK
MEAS:VERR:OOFF
MEAS:VERR:MAGN:RMS
MEAS:VERR:MAGN:PEAK
MEAS:VERR:PHAS:RMS
MEAS:VERR:PHAS:PEAK
MEAS:VERR:IQ

READ:VERRor:MAGNitude:PEAK?

returns the peak magnitude component of vector errors for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:RMS
MEAS:VERR:PEAK
MEAS:VERR:OOFF
MEAS:VERR:MAGN:RMS
MEAS:VERR:MAGN:PEAK
MEAS:VERR:PHAS:RMS
MEAS:VERR:PHAS:PEAK
MEAS:VERR:IQ

READ:VERRor:MAGNitude:RMS?

returns the RMS of the magnitude component of vector errors for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:RMS
MEAS:VERR:PEAK
MEAS:VERR:OOFF
MEAS:VERR:MAGN:RMS
MEAS:VERR:MAGN:PEAK
MEAS:VERR:PHAS:RMS
MEAS:VERR:PHAS:PEAK
MEAS:VERR:IQ

READ:VERRor:NORMalized:IQ? <symbol>, <burst>
<symbol>numeric integer value from 6 - 15
<burst>numeric integer value from 1 - 10

returns the I-Q value of a selected symbol in a selected burst for the last normalized (multi-burst) vector error measurement performed.

The normalized measurement samples the first 10 symbols (symbols 6 - 15) of each burst for 10 TDMA bursts within a 1 minute interval.

Printer access to the I and Q values and name of this measurement for the selected symbol are available following this command.

Limit checking is not performed on this parameter.

Response: <numeric floating point>, <numeric floating point>

Resolution: 0.001

Units: none

Measured by: MEAS:VERR:NORM:RMS
MEAS:VERR:NORM:PEAK
MEAS:VERR:NORM:IQ

READ:VERRor:NORMalized:PEAK?

returns the peak vector error for the last normalized (multi-burst) vector error measurement performed.

The normalized measurement samples the first 10 symbols (symbols 6 - 15) of each burst for 10 TDMA bursts within a 1 minute interval.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:NORM:RMS
MEAS:VERR:NORM:PEAK
MEAS:VERR:NORM:IQ

READ:VERRor:NORMalized:RMS?

returns the RMS normalized error vector magnitude for the last normalized (multi-burst) vector error measurement performed.

The normalized measurement samples the first 10 symbols (symbols 6 - 15) of each burst for 10 TDMA bursts within a 1 minute interval.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:NORM:RMS
MEAS:VERR:NORM:PEAK
MEAS:VERR:NORM:IQ

READ:VERRor:OOFFset?

returns the origin offset for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: dBc

Measured by: MEAS:VERR:RMS
MEAS:VERR:PEAK
MEAS:VERR:OOFF
MEAS:VERR:MAGN:RMS
MEAS:VERR:MAGN:PEAK
MEAS:VERR:PHAS:RMS
MEAS:VERR:PHAS:PEAK
MEAS:VERR:IQ

READ:VERRor:PEAK?

returns the peak error vector for the last single-burst vector error measurement performed.

Peak error vector is also known as the peak EVM (Error Vector Magnitude).

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:RMS
MEAS:VERR:PEAK
MEAS:VERR:OOFF
MEAS:VERR:MAGN:RMS
MEAS:VERR:MAGN:PEAK

MEAS:VERR:PHAS:RMS
MEAS:VERR:PHAS:PEAK
MEAS:VERR:IQ

READ:VERRor:PHASe:PEAK?

returns the peak phase component of vector errors for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.01

Units: degrees

Measured by: MEAS:VERR:RMS
MEAS:VERR:PEAK
MEAS:VERR:OOFF
MEAS:VERR:MAGN:RMS
MEAS:VERR:MAGN:PEAK
MEAS:VERR:PHAS:RMS
MEAS:VERR:PHAS:PEAK
MEAS:VERR:IQ

READ:VERRor:PHASe:RMS?

returns the RMS of the phase component of vector errors for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.01

Units: degrees

Measured by: MEAS:VERR:RMS
MEAS:VERR:PEAK
MEAS:VERR:OOFF
MEAS:VERR:MAGN:RMS
MEAS:VERR:MAGN:PEAK
MEAS:VERR:PHAS:RMS

MEAS:VERR:PHAS:PEAK
MEAS:VERR:IQ

READ:VERRor:RMS?

returns the RMS vector error for the last single-burst vector error measurement performed.

RMS vector error is also known as the RMS EVM (Error Vector Magnitude).

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter is available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:RMS
MEAS:VERR:PEAK
MEAS:VERR:OOFF
MEAS:VERR:MAGN:RMS
MEAS:VERR:MAGN:PEAK
MEAS:VERR:PHAS:RMS
MEAS:VERR:PHAS:PEAK
MEAS:VERR:IQ

SOURce Subsystem

The *Source* subsystem commands pertain to functions that affect the signal generation sources of the instrument.

SOURce:AUDio:DEViation <deviation value>
<deviation value>numeric value from 0.0 - 25575.0 Hz (resolution of 25 Hz)

sets the amount of deviation in the output to the mobile phone due to the audio generator source.

This parameter is also expressed as the amplitude (level) of the audio generator level therefore, setting the deviation value causes a corresponding change in the `SOUR:AUD:LEVel` parameter.

See `SOUR: AUD:LEVel`.

Allowable units are Hz, KHz, MHz (Hz is the default).

Example 1: SOUR:AUD:DEV 1.27 KHZ
sets the audio deviation to 1270 Hz.

Example 2: SOUR:AUD:DEV 15000
sets the audio deviation to 15000 Hz.

SOURce:AUDio:FREQuency <frequency value>
<frequency value> numeric integer value from 1 - 100000 Hz

sets the frequency of the internal audio generator.

Example: SOUR:AUD:FREQ 1000
sets the internal audio frequency to 1000 Hz.

SOURce:AUDio:LEVel <amplitude>
<amplitude> numeric value from 0.0 - 7990.0 mV (resolution of 7.81 mV)

sets the amplitude level (in mV) of the audio generator output to the Audio Out jack on the front panel.

This parameter can also be expressed as the amount of deviation in the output to the mobile phone due to the audio generator source therefore setting this amplitude value causes a corresponding change in the `SOUR:AUD:DEViation` parameter.

See `SOUR:AUD:DEViation.`

The relationship is expressed by: `DEV` = `LEV` x 3.201.

Example: SOUR:AUD:LEV 1600
sets the internal audio generator to an amplitude of 1600mV.

This level is found at the Audio Out jack on the front panel if the OUTPut:AOUT parameter is set to the SOURce selection.

SOURce:AUDio[:SOURce]:EXTernal <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the external audio source.

SOURce:AUDio[:SOURce]:INTernal <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the internal audio source.

SOURce:POWer:LEVel[:IMMediate] <amplitude>
<amplitude> numeric value from -125.0 to 10.0 dBm (resolution of 0.1)

sets the power level of the RF output carrier to a given value in dBm.

This can also be accessed through the front panel base power level increment and decrement hard keys.

This command allows the user to set the output power to a higher level than the maximum output of -23.00 dBm, and then adjusts the output to the highest available power.

Example: SOUR:POW:LEV:IMM -72.5
sets the output power level to -72.5 dBm.

SOURce:POWer:LEVel:MAXimum?

query-only command which reports the maximum allowable base power.

SOURce:POWer:LEVel:MINimum?

query-only command which reports the minimum allowable base power.

SOURce:POWer:SCHannel:CHANnel?

reads the channel selection of the second channel.

Cellular:

For carrier channels 1 - 33, the value returned is the carrier channel + 989.

For channels > 33 and ≤ 799, the value returned is the carrier channel - 33.

For channels > 799, a value of 0 is returned, indicating there is no valid second channel.

PCS:

For carrier channels 1 - 1966, the value returned is 0.

For carrier channels 1967 - 1999, the value returned is the carrier channel +3.

This command is valid only if the DCM option is installed.

Response: <numeric value>

Units: none

Example: SOUR:POW:SCH:CHAN?
237
indicates that the second channel is 237.

SOURce:POWer:SCHannel:LEVel?

reads the power level of the second channel in dBm.

The response value is the carrier power value plus the second channel power offset.

This command is valid only if the DCM option is installed.

Response: <numeric value>

Units: dBm

Example: SOUR:POW:LEV -72.5
SOUR:SCH:POW:OFFS -14.5
SOUR:SCH:POW:LEV?
-87.0
indicates that the second channel power level is -87.0 dBm, which is equal to (-72.5) + (-14.5) dBm.

SOURce:POWer:SCHannel:OFFSet <delta amplitude>
<delta amplitude>numeric value from -20 - 0 dB (resolution of 2.0)

sets the power level of the second channel on the RF output to a given value, relative to the carrier power.

The second channel power level cannot exceed the carrier level, and can be no more than 20 dB less than it.

This command is only valid if the DCM option is installed.

Example: SOUR:SCH:POW:OFFS -14.5
sets the second channel output power level to 14.5 dB below the carrier.

If the carrier power is -72.5 dBm, the second channel power level is -87.0.

SOURce:POWer:SCHannel:STATe <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

enables or disables the second channel.

This command is valid only if the DCM option is installed.

SOURce:POWer:STATe <boolean>
<boolean>OFF or ON (0 = OFF, 1 = ON)

turns the RF output to the front panel jack on or off.

To turn off both carrier and modulation, either set the RF output to `OFF`, or toggle the base power switch on the front panel on and off.

SOURce:SQUAlity:DINTerval <delay>
<delay>numeric value from 0.0 - 1.0 symbols (resolution of 0.25)

sets the delay interval when measuring signal quality.

The units are assumed to be symbols.

This parameter is valued-coupled with `SOURce:SQUAlity:VSPeed` to set the delay interval to 0 if the vehicle speed is set to 0.

See `SOURce:SQUAlity:VSPeed.`

If the vehicle speed is 0 and the delay interval is set to a non-zero value, the vehicle speed is set to 0.1 km/h.

This command is valid only if the DCM option is installed.

This command is not available in PCS band.

Example: SOUR:SQUAlity:DINT 0.75
sets the delay interval to 0.75 symbols.

SOURce:SQUAlity:IBER <BER value>
<BER value>numeric value from 0.0 - 11.0% (resolution of 0.01)

sets the induced BER value (in percent) when measuring signal quality.

This command is valid only if the DCM option is installed.

Example: SOUR:SQUAlity:IBER 3.45
sets the induced BER to 3.45%.

SOURce:SQUAlity:VSPeed <speed>
<speed> numeric value from 0.0 - 100.0 (resolution of 0.1)

sets the simulated vehicle speed (in kph) when measuring signal quality.

This parameter is valued-coupled with `SOURce:SQUAlity:DINTerval` to set the delay interval to 0 if the vehicle speed is set to 0.

If the vehicle speed is 0 and the delay interval is set to a non-zero value, the vehicle speed is set to 0.1 kph.

See `SOURce:SQUAlity:DINTerval.`

This command is valid only if the DCM option is installed.

This command is not available in PCS band.

Example: SOUR:SQUAlity:VSPeed 50
sets the simulated vehicle speed to 50 km/h.

STATus Subsystem

The *Status* subsystem commands control the reporting of test equipment status conditions.

Several layers of status registers can be accessed. Some are at the lowest level and detect specific conditions, others are more general, grouping many conditions into a single status event.

The summary condition detects only changes in register conditions and selects whether a positive-going or negative-going change (or neither) is of interest.

All status registers report a summary bit to a parent register, resulting in a final status byte register (accessed by `*STB?` as well as a serial poll).

For GPIB remote operation, the status byte register issues a service request to the controller, indicating when a selected event(s) occurs.

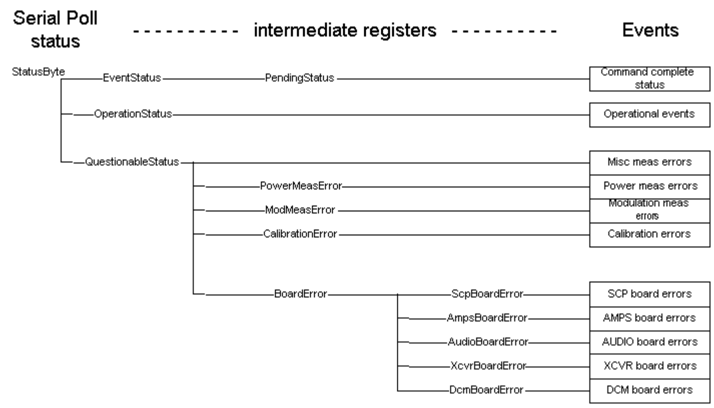
For each grouping of status events, five registers exist which conform to IEEE 488.2 specifications and include: a condition register, an event register, an enable register, and positive and negative transition filters.

These groups report to each other in a hierarchical fashion; the upper level registers indicate general conditions, and the register groups that report to them indicate specific conditions.

There are three basic groups of status registers:

- event status which are defined by IEEE 488.2 (including overlapped command complete status)
- operational status which report normal operating conditions of the instrument
- questionable status which report possible error conditions that have occurred in the instrument.

The following illustrates the reporting structure for each of these register groups:



REGISTER NAME	BIT POS (VALUE)	BIT DESCRIPTION
Status Byte	bit 0 (dec 1)	(not used)
	bit 1 (dec 2)	(not used)
	bit 2 (dec 4)	Error queue contains error messages
	bit 3 (dec 8)	summary: **QuestionableStatus**
	bit 4 (dec 16)	Response message available for reading
	bit 5 (dec 32)	summary: **EventStatus**
	bit 6 (dec 64)	summary: **StatusByte**
	bit 7 (dec 128)	summary: **OperationStatus**
EventStatus	bit 0 (dec 1)	(not used)
	bit 1 (dec 2)	Request control (not used)
	bit 2 (dec 4)	Query error
	bit 3 (dec 8)	Device error
	bit 4 (dec 16)	Execution error
	bit 5 (dec 32)	Command error
	bit 6 (dec 64)	Front panel EXIT REMOTE
	bit 7 (dec 128)	Power on
PendingStatus	bit 0 (dec 1)	Call Registration command
	bit 1 (dec 2)	Call Origination command
	bit 2 (dec 4)	Call Page command
	bit 3 (dec 8)	Base or mobile release
	bit 4 (dec 16)	Hook flash command
	bit 5 (dec 32)	DTMF test command
	bit 6 (dec 64)	Autograph test command
	bit 7 (dec 128)	Peak audio dev measurement
	bit 8 (dec 256)	Printer command
PendingStatus (cont.)	bit 9 (dec 512)	Disk command
	bit 10 (dec1024)	Program Sequence in process
	bit 11 -15	(not used)
OperationStatus	bit 0 - 3	(not used)
	bit 4 (dec 16)	Measurement in progress
	bit 5 -9	(not used)
	bit 10 (dec 1024)	Call processing in progress
	bit 11 (dec 2048)	TDMA mobile sync acquired
	bit 12 -15	(not used)
QuestionableStatus	bit 0 - 2	(not used)
	bit 3 (dec 8)	summary: **Power Meas Error**
	bit 4 -6	(not used)
	bit 7 (dec 128)	summary: **ModMeasError**
	bit 8 (dec 256)	summary: **CalibrationError**
	bit 9 (dec 512)	summary: **BoardError**
	bit 10 (dec 1024)	BER measurement accuracy

REGISTER NAME	BIT POS (VALUE)	BIT DESCRIPTION
	bit 11 (dec 2048)	SINAD measurement accuracy
	bit 12 (dec 4096)	Reverse R0 measurement
	bit 13 (dec 8192)	Forward R0 measurement questionable
	bit 14- 15	(not used)
CalibrationError	bit 0 (dec 1)	Calibration table checksum failure
	bit 1- 15	(not used)
ModMeasError	bit 0 (dec 1)	Mobile ST ordered ON during manual mode
	bit 1 (dec 2)	Fading enabled in base station modulation
ModMeasError (cont.)	bit 2 -15	(not used)
Power Meas Error	bit 0 (dec 1)	Calibration table bad
	bit 1 (dec 2)	Tx power meas low w/ base power high
	bit 12 -15	(not used)
BoardError	bit 0 (dec 1)	summary: **ScpBoardError**
	bit 1 (dec 2)	summary: **AmpsBoardError**
	bit 2 (dec 4)	summary: **DcmBoardError**
	bit 3 (dec 8)	summary: **AudioBoardError**
	bit 4 (dec 16)	summary: **XcvrBoardError**
	bit 5 -15	(not used)
ScpBoardError	bit 0 (dec 1)	Generic failure
	bit 1 (dec 2)	Static RAM memory failure
	bit 2 (dec 4)	Printer failure
	bit 3 (dec 8)	Disk drive failure
	bit 4 (dec 16)	Remote port failure
	bit 5 -15	(not used)
AmpsBoardError	bit 0 (dec 1)	Generic failure
	bit 1 -15	(not used)
AudioBoardError	bit 0 (dec 1)	Generic failure
	bit 1 -15	(not used)
XcvrBoardError	bit 0 (dec 1)	Generic failure
	bit 1 -15	(not used)
DcmBoardError	bit 0 (dec 1)	Generic failure
	bit 1 (dec 2)	Restart failure
	bit 2 (dec 4)	Version ID failure
	bit 3 (dec 8)	Dual Port ram failure on SCP
DcmBoardError (cont.)	bit 4 (dec 16)	Dual Port ram failure on DCM
	bit 5 (dec 32)	Loopback diagnostics failure
	bit 6 (dec 64)	DCM is dead
	bit 7 (dec 128)	68302 checksum failure

REGISTER NAME	BIT POS (VALUE)	BIT DESCRIPTION
	bit 8 (dec 256)	DSP checksum failure
	bit 9 (dec 512)	SW/HW mismatch (software does not support hardware)
	bit 10 -15	(not used)
* Multiple events can occur in any register, and their values are cumulative.		

Each of the status register groups defined above with the exception of **StatusByte, EventStatus,** and **PendingStatus** consist of 2 distinct read-only registers and 3 read-write mask registers (filters).

The status events presented to each register (either a specific event or the summary status from a child register set) are reported directly to a condition register which reflects the actual status of events at any given time. It may be read at any time without affecting its own value, or the value of any other register.

The purpose of the event register is to store the occurrence of certain events, in order to guarantee that the event is captured.

When the condition register value changes, its change (not its value) is passed through the transition filters, capturing the specified event changes, to the event register.

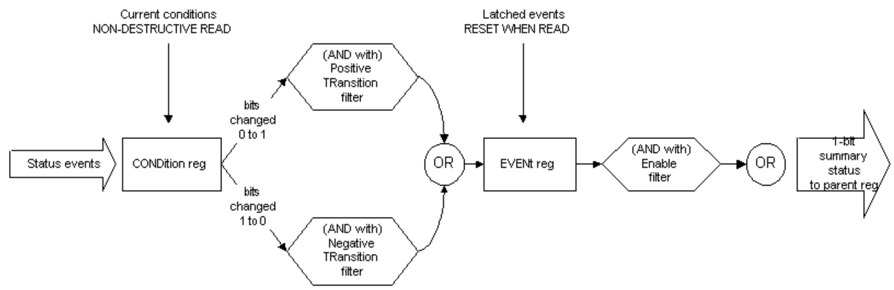
The **StatusByte**, **EventStatus**, and **PendingStatus** register groups consist of a condition register and an enable register.

The enable register on the **PendingStatus** register group specifies which bits will set or clear an internal 1-bit flag **OpcStatus**. This result is passed on to the OPC bit in the **EventStatus** condition register if the *OPC command has been received. The **EventStatus** group is 8-bits instead of 16, and reports its summary bit to the **StatusByte** condition register if a binary ANDing of its condition and enable registers is non-zero.

The **StatusByte** group is also 8-bits. Bit 7 and bits 0 - 5 function like the other condition and enable register bits. Bit 6 is the summary bit for this register. It is set if the binary ANDing of the other 7 bits in the condition and enable registers is non-zero, and cleared otherwise. Bit 6 of the enable register is used to enable/disable sending a service request (SRQ) to the controller based on the **StatusByte** summary bit value.

This function is only implemented for GPIB remote operation.

The following illustrates this operation:



STATus:OPERation:COMPlete:CONDition?

returns a value that represents all overlapped processes that have been started, but have not yet completed.

Response: <numeric integer value> (range 0x0000 to 0x7FFF).

Example: STAT:OPER:COMP:COND?
48
indicates both the DTMF and hook flash tests are in progress.

STATus:OPERation:COMPlete:ENABle <numeric value>

sets the mask to determine which overlapped processes are evaluated in determining an **PendingStatus** bit value.

Only those processes with corresponding bits enabled in this register cause a change in the **PendingStatus** value.

When any enabled overlapped process starts, **PendingStatus** is set to 1 where it remains until all enabled overlapped processes have completed, at which point it is set to 0.

This value is used by three commands to pause processing of any more commands until pending commands have completed:

- the `*WAI` command waits until PendingStatus is 0 before executing the next command.
- the `*OPC` command causes the OPC bit in the EventStatus register to be set to 1 when the PendingStatus value is set to 0. The controller monitors this value to determine when the command has completed.
- the `*OPC?` command waits for the PendingStatus value to become 0, and will then place a “1” response message in the output queue. This causes the Message Available bit to become set in the StatusByte register. The control-

ler monitors the status of this bit to determine when the command has completed.

The `*OPC and *OPC?` commands are improved by performing a serial poll to query for status, or by issuing an SRQ to notify the controller when the command(s) have completed.

This requires GPIB, and cannot be accomplished using RS-232.

STATus:OPERation:COMPlete:ERRor?

returns the bit value of the sum of all of the overlapped commands that completed with an error.

These bits are initially 0 when the overlapped command begins, and are set when the command completes with an error.

A `*CLS` command and a device clear (GPIB only) reset this register to 0.

Response: <numeric integer (decimal value)>

Example: STAT:OPER:COMP:ERR?
2
indicates that a CALL:ORIGination command failed.

STATus:OPERation:CONDition?

returns the value of the condition register associated with the 16-bit **OperationStatus** register group.

The condition register reflects the current status condition of the test equipment. The returned value is a decimal value that represents the sum of all current conditions.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:OPERation:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **OperationStatus** summary bit should be set or cleared in the **StatusByte** register group's condition register.

STATus:OPERation[:EVENt]?

returns the value of the event register associated with the 16-bit **OperationStatus** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transitional filters determine the importance of events occurring (bit value changing from 0 to 1) or being cleared out (bit value changing from 1 to 0).

The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:OPERation:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **OperationStatus** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:OPERation:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **OperationStatus** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

Example: STAT:OPER:PTR 4096
STAT:OPER:NTR 16
positive transition register value: 4096 (bit 11 set)
negative transition register value:16 (bit 4 set)
condition register value: 0
event register value: 0

1. TDMA sync is acquired(bit 11 = 1)
condition register value:4096(bit 11 set)
event register value:4096(bit 11 set)

2. measurement begins (bit 4 = 1)
condition register value:4112(bits 4, 11 set)
event register value:4096(bit 11 set)

3. measurement completes (bit 4 = 0)
condition register value:4096(bit 11 set)
event register value:4112(bits 4, 11 set)

4. event register read (value returned = 4112)
condition register value:4096(bit 11 set)
event register value:0

The configuration is set to capture only positive-going bit-11 events and negative-going bit-4 events.

This translates into detecting when TDMA sync is acquired and when a measurement command completes. (The values in the condition and event registers are not necessarily the same.)

The event register captures changes that may be missed by reading the condition register.

For instance, the condition register has the same value prior to the measurement beginning as it does after it completes. If a polling operation on the condition register was used to detect when the measurement completes, it might have missed this event if the time between queries was too long. The event register ensures the capture this condition.

If using GPIB, avoid multiple queries of the event register by setting the enable register to 4112, so that the **OperationStatus** summary bit of the **StatusByte** register is set when both of these conditions exist.

The **StatusByte** register value can be read by performing serial polls, which executes much faster than a query. The **StatusByte** can be setup to issue a SRQ to the controller when the conditions exist, eliminating the need to perform multiple serial polls.

STATus:PRESet

sets all of the *Status* subsystem enable and transition registers to a known state.

The values for STATus:OPERation and STATus:QUEStionable register group are set as follows:

ENABle registers > 0 (off)
Positive TRanstion regs> 1 (on)
Negative TRanstion regs> 0 (off)

For all other register groups (those that report to STATus:QUEStionable) the values are set as follows:

ENABle registers > 1 (on)
Positive TRanstion regs > 1 (on)
Negative TRanstion regs> 0 (off)

STATus:QUEStionable:BOARd:AMPS:CONDition?

returns the value of the condition register associated with the 16-bit **AmpsBoardError** register group.

The condition register always reflects the current status condition of the instrument. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:AMPS:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **AmpsBoardError** summary bit should be set or cleared in the **BoardError** register group's condition register.

STATus:QUEStionable:BOARd:AMPS [:EVENt]?

returns the value of the event register associated with the 16-bit **AmpsBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:AMPS:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **AmpsBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:AMPS:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **AmpsBoardError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:AUDio:CONDition?

returns the value of the condition register associated with the 16-bit **AudioBoardError** register group.

The condition register reflects the current status condition of the audio measurement board. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers is affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:AUDio:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **AudioBoardError** summary bit should be set or cleared in the **BoardError** register group's condition register.

STATus:QUEStionable:BOARd:AUDio [:EVENt]?

returns the value of the event register associated with the 16-bit **AudioBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:AUDio:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **AudioBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:AUDio:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **AudioBoardError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:CONDition?

returns the value of the condition register associated with the 16-bit **BoardError** register group.

The condition register reflects the current status condition of all the boards. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:DCM:CONDition?
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

returns the value of the condition register associated with the 16-bit **DcmBoardError** register group.

The condition register reflects the current status condition of the DCM board. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:DCM:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **DcmBoardError** summary bit should be set or cleared in the **BoardError** register group's condition register.

STATus:QUEStionable:BOARd:DCM [:EVENt]?

returns the value of the event register associated with the 16-bit **DcmBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:DCM:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **DcmBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:DCM:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **DcmBoardError** register group to determines which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **BoardError** summary bit should be set or cleared in the **QuestionableStatus** register group's condition register.

STATus:QUEStionable:BOARd [:EVENt]?

returns the value of the event register associated with the 16-bit **BoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **BoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **BoardError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:SCP:CONDition?

returns the value of the condition register associated with the 16-bit **ScpBoardError** register group.

The condition register reflects the current status condition of the SCP board. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:SCP:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **ScpBoardError** summary bit should be set or cleared in the **BoardError** register group's condition register.

STATus:QUEStionable:BOARd:SCP [:EVENt]?

returns the value of the event register associated with the 16-bit **ScpBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:SCP:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **ScpBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:SCP:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **ScpBoardError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:XCVR:CONDition?

returns the value of the condition register associated with the 16-bit **XcvrBoardError** register group.

The condition register reflects the current status condition of the transceiver board. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:XCVR:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **BoardError** summary bit should be set or cleared in the **XcvrBoardError** register group's condition register.

STATus:QUEStionable:BOARd:XCVR [:EVENt]?

returns the value of the event register associated with the 16-bit **XcvrBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:XCVR:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **XcvrBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:XCVR:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **XcvrBoardError** register group. It determines which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:CALibration:CONDition?

returns the value of the condition register associated with the 16-bit **CalibrationError** register group.

The condition register reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:CALibration:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **CalibrationError** summary bit should be set or cleared in the **QuestionableStatus** register group's condition register.

STATus:QUEStionable:CALibration [:EVENt]?

returns the value of the event register associated with the 16-bit **CalibrationError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:CALibration:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **CalibrationError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:CALibration:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **CalibrationError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:CONDition?

returns the value of the condition register associated with the 16-bit **QuestionableStatus** register group.

The condition register reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **QuestionableStatus** summary bit should be set or cleared in the **StatusByte** register group's condition register.

STATus:QUEStionable[:EVENt]?

returns the value of the event register associated with the 16-bit **QuestionableStatus** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:MODulation:CONDition?

returns the value of the condition register associated with the 16-bit **ModMeasError** register group.

The condition register reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:MODulation:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **ModMeasError** summary bit should be set or cleared in the **QuestionableStatus** register group's condition register.

STATus:QUEStionable:MODulation [:EVENt]?

returns the value of the event register associated with the 16-bit **ModMeasError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:MODulation:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **ModMeasError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:MODulation:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **ModMeasError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:NTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **QuestionableStatus** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **QuestionableStatus** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:POWer:CONDition?

returns the value of the condition register associated with the 16-bit **PowerMeasError** register group.

The condition register reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:POWer:ENABle <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **PowerMeasError** summary bit should be set or cleared in the **QuestionableStatus** register group's condition register.

STATus:QUEStionable:POWer [:EVENt]?

returns the value of the event register associated with the 16-bit **PowerMeasError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:POWer:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **PowerMeasError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:POWer:PTRansition <enable pattern>
<enable pattern>numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **PowerMeasError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

SYSTem Subsystem

The *System* subsystem commands, though not related to instrument performance, include general housekeeping functions and setting up global configurations (remote communication, time, date, etc.).

SYSTem:COMMunicate:GPIB:ADDRess <primary >, [secondary]
<primary > numeric value 0 - 30
[secondary] numeric value 0 - 30 (optional)

sets up the GPIB address to be used.

The primary address, which selects both the talk and listen address of the instrument, is required.

The secondary address which is used only when the controller sends secondary addressing with the commands, is optional.

The secondary addressing is disabled on power up and whenever this command is received without the secondary address parameter, and enabled when this command is received with the secondary address specified.

Example 1: SYST:COMM:GPIB:ADDR 9
sets the GPIB address to 09, and disables secondary addressing.

Example 2: SYST:COMM:GPIB:ADDR 14, 23
sets the GPIB primary address to 14, the secondary address to 23, and enables secondary addressing.

SYSTem:COMMunicate:GPIB:TERMinator <terminator selection>
<terminator selection>CR, LF, or CRLF

sets the output terminator character selection to be used for the remote GPIB port.

This does not affect the remote serial terminator selection or the input termination characters for the GPIB port.

The input command terminates when any of the following is received: a carriage return character, a linefeed character, or an END signal (EOI asserted with the data).

The output terminator selection determines how response data (sent by the 4300 to the controller) is terminated.

An END bus signal accompanies the last character sent, regardless of the selection.

Example: SYST:COMM:GPIB:TERM CRLF
terminates all GPIB response messages with a carriage return followed by a linefeed which asserts the EOI.

SYSTem:COMMunicate:SERial:BAUD <baud rate>
<baud rate>numeric value (300, 600, 1200, 2400, 4800, 9600, 19200)

sets the baud rate for the remote serial port to one of seven standard rates.

This selection is used for both receiving commands and transmitting responses.

SYSTem:COMMunicate:SERial:BITS <bits per character>
<bits per character>numeric value 7 or 8

sets the number of bits per character to be used for the remote serial port.

This selection is used for both receiving commands and transmitting responses.

SYSTem:COMMunicate:SERial:PACE <pace selection>
<pace selection>NONE or XON

enables or disables the XON/XOFF data flow control used for the remote serial port.

This selection is used for both receiving commands and transmitting responses.

When NONE is selected, the unit ignores the XON and XOFF characters if they are received, and does not issue either character to control data flow. If this mode is selected, the controller must implement some other way of preventing data loss. This can be done by enabling echo and verifying the data character by character.

When XON is selected, the unit stops sending response characters after the XOFF character has been received, and does not continue until the XON character has been received.

The unit also sends an XOFF character if its input buffer is nearly full (less than 100 bytes available) and during disk drive accesses. This is to indicate to the remote controller device that the unit is unable to accept further input until it is allowed time to process the data.

An XON character is sent when more buffer space is available (200 bytes or more), and it is ready to accept more input.

The input buffer character length is 8192.

SYSTem:COMMunicate:SERial:PARity[:TYPE] <parity selection>
<parity selection>NONE, EVEN, ODD, MARK, or SPACe

sets the parity selection to be used for the remote serial port.

This selection is used for both receiving commands and transmitting responses.

SYSTem:COMMunicate:SERial:SBITs <stop bits>
<stop bits> numeric value 1 or 2

sets the number of stop bits to be used for the remote serial port.

This selection is used for both receiving commands and transmitting responses.

SYSTem:COMMunicate:SERial:TRANsmit:CREPort <status>
<status>OFF or ON (0 = OFF, 1 = ON)

enables or disables generation of a command report response on the serial port.

When enabled, if the command is a query that generates a response message without executing any errors, only the response message will be output

If the command does not generate a response message without executing without any errors, the command report response is:

```
>OK
```

If an error occurs in executing the command, the command report response is:

```
>ERROR -XXX
```

where: *XXX* is the 3-digit SCPI error code

See `SYSTem:ERRor?` for error code descriptions.

This command has no effect on GPIB responses.

SYSTem:COMMunicate:SERial:TRANsmit:ECHO <status>
<status>OFF or ON (0 = OFF, 1 = ON)

enables or disables echo of the input characters received at the serial port.

When enabled, the characters are output to the serial port as they are parsed.

This is different than echoing in that the data is contained in the input buffer waiting to be parsed.

Input terminator characters are converted to the current serial output terminator character selection prior to echoing which allows the user to treat the echo response as any other response on termination.

This command has no effect on GPIB operation.

SYSTem:COMMunicate:SERial:TRANsmit:TERMinator <selection>
<selection>CR, LF, or CRLF

sets the output terminator character selection used for the remote serial port.

This does not affect the remote GPIB terminator selection or the input termination characters for the serial port.

The input command terminates when either a carriage return or a linefeed character is received. The output terminator selection determines how response data sent by the 4300 to the controller is terminated.

Example: SYST:COMM:SER:TRAN:TERM LF
terminates all serial response messages with a linefeed character.

SYSTem:COMMunicate:SOURce <remote select>
<remote select>SERial, GPIB, AUTO

selects the source of the remote input.

If `SERial` is selected, the RS-232 port is used for remote communication and the GPIB port is disabled.

If `GPIB` is selected, the IEEE-488.2 connector is used and RS-232 is disabled.

If `AUTO` is selected, the first remote port (RS-232 or GPIB) that sends a command to the instrument following power up is selected as the remote communication port. Once a port has been selected in the `AUTO` mode, it does not re-set if commands are issued to the other port.

Example: SYST:COMM:SOUR GPIB
selects the GPIB (IEEE488.2) to be used for remote communication.

SYSTem:DATE[:DATA] <year>, <month>, <day>
<year>numeric value 1989 - 2089
<month>numeric value 1 - 12(representing month of the year)
<day>numeric value 1 - 31(representing day of month)

sets the current date for the real-time clock in the tester.

It is used when files are saved-to-disk and can be included in printouts.

SYSTem:DATE[:DATA]?

returns the current date read from the real-time clock in the tester.

The response is always year, month, day regardless of the format set by `SYSTem:DATE:FORMat`.

Response: <numeric integer>, <numeric integer>, <numeric integer>

The values represent the year (1989 - 2089), the month (1 - 12), and the day of the month (1 - 31).

Units: none

Example: 1994,10,23
represents the date October 23, 1994.

SYSTem:DATE:FORMat <format select>
<format select>US or INTer

sets the LCD display format for date information and selects the format for printing `DATE` and `VDATe` selections for the `PRINt:ITEM` and `PRINt:SELected:ITEM` commands.

This has no effect on the format of the remote query command.

The formats are defined as:

USmonth, day, year

`INTer` day, month, year

Example: SYST:DATE:FORMAT US
The date is displayed on the LCD as month, day, year.

SYST:DATE:FORMAT INT
The date is displayed on the LCD as day, month, year.

SYSTem:ERRor?

returns the oldest entry in the error queue.

The error queue is empty on powerup, and is cleared when a device clear or interface clear is received from the GPIB port or the control-C character is received from the serial port.

The error queue holds up to 10 errors and deletes them (using `SYSTem:ERRor?` as they are read.

The error information returned from this query consists of a numeric error code, followed by an ASCII string describing the error.

If the error queue is empty when this command is received, the following message is returned:

0,“No error”

If the error queue is full and another error occurs prior to reading an error, the most recent error is discarded and replaced with:

-350,“Queue overflow”

The numeric portion of the response is always a negative value (-100 to -499) or 0 if there are no errors.

The string consists of an English language translation of the error, as defined by SCPI, and contains additional information.

If there is additional information, it is displayed as:

“English description; (return code) offending command plus ˆ mark”

A semicolon and a numeric value enclosed in parenthesis are displayed after the English description. The numeric value is a return code that gives additional details about the cause of the error.

If no additional details are available, the numeric value is 0; otherwise, it is a negative value.

The command on which the error occurred follows the return code. A ‘ˆ’ character is placed after the character in the command at which the error occurred to help pinpoint the cause of command errors (error codes -100 to -199).

Response: <numeric integer>, <ASCII string>

Example 1: CALL:REGI:IMM
SYST:ERR?
-113,"Undefined header;(0)CALL:REGI:^IMM"
indicates that the intended command was CALL:REG:IMM. REGI is not a valid subnode of CALL.

NOTE: The '^' was placed after the REGI: indicating this is the node that could not be found.

Example 2: *RCL 7
SYST:ERR?
*-314,"Save/recall memory lost;(-744)*RCL 7^"*
indicates an error in retrieving stored settings location #7.

Return code -744 indicates that the location had an invalid checksum due to a memory failure or recalling a location that had never been stored.

SYSTem:KEY:DEBounce <debounce time>
<debounce time>numeric value 10 - 200

sets the debounce time (in milliseconds) for the front panel keys.

Adjust it only if there is a problem with the keypad. Use larger values for noisy keypads.

SYSTem:KEY:DELay <delay time>
<delay time>numeric value 10 - 10000

sets the amount of delay time (in milliseconds) to wait after a front panel keypress before entering the repeat mode.

Any key held down for less than this duration will be accepted as a single keypress. If a key is held down longer than this time, the key will be repeated multiple times, at a rate given by `SYSTem:KEY:REPeat`.

SYSTem:KEY:REPeat <repeat time>
<repeat time> numeric value 10 - 2000

sets the amount of delay time (in milliseconds) before repeating a front panel key that has been held down longer than the repeat delay time.

See `SYSTem:KEY:DELay`.

SYSTem:PRESet

initializes many of the parameters of the 4300 to their corresponding default values.

This is more comprehensive than `*RST`; it performs all of the actions of `*RST` in addition to initializing the following parameters:

TABLE 32.

SCPI command	Default Value
CALibration:ALL:AUTO	ON
CALibration:CABLe:COMPensation	OFF
CALibration:CABLe:SELection	1
CALL:MDATa:ESN:FORMat	STANDARD2
CALL:PAGE:MIN1	0
CALL:PAGE:MIN2	0
CALL:SSD:AKEY:DATA	“00000000000000000000000000”
CALL:SSD:SSDA	“00000000000000000000”
DISPlay:RATio	SNDRATIO
DISPlay:UNITs:FREQuency:RF	KHZ
DISPlay:UNITs:POWer:TRANsmitter	WATT
MEASure:FREQuency:UNITs	HZ
MEASure:POWer:TRANsmitter:UNITs	WATT
PRINt:BOLD	OFF
PRINt:FORMat	HEXADECIMAL
PRINt:JUSTify	LEFT
PRINt:MFACtor	1.0
PRINt:OVERwrite	ON
PRINt:PPRefix	NONE
PRINt:RESolution:AUTO	ON
PRINt:RESolution:DIGits	2
PRINt:SELected:COLumn:TAB:DEFine	1, 9, 17, 25, 33, 41, 49, 57
PRINt:SELected:COLumn:VALue	1
PRINt:SELected:LINE:VALue	1
PRINt:SIZE	NORMAL
PRINt:UNDerline	OFF
PROGram:AGRaph:CHANnel:STARt	100
PROGram:AGRaph:CHANnel:STEP	10
PROGram:AGRaph:CHANnel:STOP	650

TABLE 32.

SCPI command	Default Value
PROGram:AGRaph:CHANnel:TYPE	AMPS
PROGram:AGRaph:DLOG	OFF
PROGram:AGRaph:PLOG	OFF
PROGram:AGRaph:VMAC0	ON
PROGram:AGRaph:VMAC1	ON
PROGram:AGRaph:VMAC2	ON
PROGram:AGRaph:VMAC3	ON
PROGram:AGRaph:VMAC4	ON
PROGram:AGRaph:VMAC5	ON
PROGram:AGRaph:VMAC6	ON
PROGram:AGRaph:VMAC7	ON
PROGram:AGRaph:VMAC8	ON
PROGram:AGRaph:VMAC9	ON
PROGram:AGRaph:VMAC10	ON
PROGram:AGRaph:PAUSe	OFF
PROGram[:SELected]:BUFFer:MLISt	MEAS, NONE, NONE, NONE
PROGram[:SELected]:BUFFer:MLOG	OFF
PROGram:TALignment:DATA	5, 9, 16, 19, 24, 30, 25, 18, 11, 4, 0
PROGram:TALignment:DELay	0.0 sec
SYSTem:KEY:DEBounce	20 msec
SYSTem:KEY:DELay	800 msec
SYSTem:KEY:REPeat	50 msec

These commands have no associated accessible parameters (they have internal parameters or buffers that are accessible and used to initialize them):

PRINt:PRESet

PROGram[:SELected]:BUFFer:CLEAr

This command places the 4300 (and mobile phone) on the control channel selection defined by `CALL:CONTrol:XXX`:

CALL:FCC ON

If the mobile phone was up on a call, the call is dropped.

SYSTem:SNUMber?

returns the serial number of the test equipment.

The serial number is established when the calibration table is loaded into the test equipment from disk.

Response: <ASCII string (maximum length 21 characters)>

Units: none

Example: SYST:SNUM?
045769
indicates the serial number for the test equipment is 045769.

SYSTem:TIME[:DATA] <hour>, <minute>, <second>
<hour>numeric value 0 - 23 (24-hour clock format)
<minute>numeric value 0 - 59
<second>numeric value 0 - 59

sets the current time for the real-time clock in the instrument.

It is used when files are saved-to-disk and can be included in printouts.

SYSTem:TIME[:DATA]?

returns the current time read from the real-time clock in the instrument.

The response is given in the 24-hour format regardless of the `SYSTem:DATE:FORMat` command setting.

Response: <numeric integer>, <numeric integer>, <numeric integer>

The values represent the hour (0 - 23), the minute (0 - 59), and the second (0 - 59).

Units: none

Example: 17,22,56
represents the time 17:22:56 or 5:22:56 PM.

SYSTem:TIME:FORMat <format select>
<format select>AMPM or HR

sets the LCD display format for the time information and selects the format for printing the `TIME` selection for `PRINt:ITEM` and `PRINt:SELected:ITEM`.

This has no effect on the format of the remote query command.

The formats are defined as:

AMPMtime in 12-hour format with an AM/PM suffix
HRtime in 24-hour format

Example 1: SYST:TIME:FORMAT HR
The time is displayed on the LCD as hour, minute, second.

Example 2:SYST:TIME:FORMAT AMPM
The time is displayed on the LCD as hour, minute, second.

SYSTem:TSTamp?

returns a timestamp value that is equivalent to the number of seconds from the beginning of the current calendar year.

The day of the year is multiplied by 86400 (to account for leap year), the hour of the day is multiplied by 3600 (24-hour clock format), the minutes are multiplied by 60, and all of these are added to the seconds.

The result is a range of values from 0.0 - 31622399.0, with 0.0 representing 0:00:00 (midnight) on January 1, and 31622399.0 representing 23:59:59 (1 second before midnight) on December 31 on a leap year. If there is no leap year, this value is 31535999.0. The decimal portion of seconds is truncated, so each reading may have an error by as much as 1.0 second.

Response: <numeric floating point>

Resolution: 1.0

Units: seconds

SYSTem:TSTamp? <reference timestamp>
<reference timestamp>numeric value from 0.0 - 31622399.0

returns the time that has elapsed since the reference timestamp value.

The timestamp value should be a value that was returned by the `SYSTem:TSTamp?` query without any arguments.

Since this timestamp value does not include information on the year, the maximum elapsed time that can be determined is one year, however it does compensate for end-of-year rollover. If the reference timestamp is collected on December 31 and this
command is used on January 1 (the following day), the correct elapsed time will be returned.

Due to the timestamp truncation of fractional seconds, the elapsed time may have a 1.0 second error.

Response: <numeric floating point>

Resolution: 1.0

Units: seconds

Example: SYST:TST?
4031183.0
CALL:REG; *WAI
SYST:TST? 4031183.0
14.0
The initial SYST:TST? command gets a timestamp value that can be used as a reference for obtaining elapsed time.

A registration is initiated and completed before the second SYST:TST? query is executed.

The final result indicates that the registration took ~ 14 seconds.

SYSTem:VERSion:AMPS?

returns the software version for the AMPS microprocessor.

Response: <string>

The string is composed of a numeric value representing the major release version, a decimal point, another numeric value representing the minor revision for the release, and an optional lowercase alpha character representing the maintenance revision.

SYSTem:VERSion:DCM[:MICR]?

returns the software version for the DCM microprocessor.

This command produces the error -708,"Hardware missing" if the DCM card is not installed in the unit.

This command is valid only if the DCM option is installed.

Response: <string>

The string is composed of a numeric value representing the major release version, a decimal point, another numeric value representing the minor revision for the release, and an optional lowercase alpha character representing the maintenance revision.

SYSTem:VERSion:DCM:RXDSp?

returns the software version for the Rx DSP on the DCM card.

This command produces the error -708,"Hardware missing" if the DCM card is not installed in the unit.

This command is valid only if the DCM option is installed.

Response: <string>

The string is composed of a numeric value representing the major release version, a decimal point, another numeric value representing the minor revision for the release, and an optional lowercase alpha character representing the maintenance revision.

SYSTem:VERSion:DCM:TXDSp?

returns the software version for the Tx DSP on the DCM card.

This command produces error -708,"Hardware missing" if the DCM card is not installed in the unit.

This command is valid only if the DCM option is installed.

Response: <string>

The string is composed of a numeric value representing the major release version, a decimal point, another numeric value representing the minor revision for the release, and an optional lowercase alpha character representing the maintenance revision.

SYSTem:VERSion:DCM:VCDSp?

returns the software version for the Voice Codec DSP on the DCM card.

This command produces the error -708,"Hardware missing" if the DCM card is not installed in the unit.

This command is valid only if the DCM option is installed.

Response: <string>

The string is composed of a numeric value representing the major release version, a decimal point, another numeric value representing the minor revision for the release, and an optional lowercase alpha character representing the maintenance revision.

SYSTem:VERSion:SCP?

returns the software version for the SCP microprocessor.

Response: <string>

The string is composed of a numeric value representing the major release version, a decimal point, another numeric value representing the minor revision for the release, and an optional lowercase alpha character representing the maintenance revision.

Units: none

Example: "2.1b"

SYSTem:VERSion[:SCPI]?

returns the SCPI version for which the software is compliant.

Response: <numeric value>

represents the SCPI version

Units: none

Example: 1994.000000
indicates the software is compliant with SCPI version 1994.0.

The value proceeding the decimal point is the year in which the version was released, and the first digit following the decimal point indicates the minor revision made for that year and is 0 for the initial release for the year.

SYSTem:VOLume <level>
<level>numeric value 0 - 1023
(0 = softest audio level, 1023 = loudest audio level)

sets the audio speaker volume level.

CDMA SCPI Commands

5

This chapter describes the functionality of the instrument. Topics discussed in this chapter are as follows:

CALCulate Subsystem

CALCulate:LIMit:FAIL:CUMulative

resets the cumulative pass/fail flag reported by the `CALC:LIM:FAIL:CUM?` query.

This allows the flag to be used to represent a group of sequential measurement results.

This is a sequencer-only command.

CALCulate:LIMit:FAIL:CUMulative?

determines if all of the measurements performed since either the beginning of the test sequence or the last `CALC:LIM:FAIL:CUM` command passed or if any of them failed the specified limits.

This is a sequencer-only command.

0 - indicates all measurements passed.
1 - indicates at least one measurement failed or the limits were not enabled.

Response: 0 or 1

Units: none

Example: MEAS:DEV:SAT?
5000
MEAS:DEV:ST?
10000
MEAS:DEV:RES?
5600
CALC:LIM:FAIL:CUM?
0
indicates that all measurements passed the limit checks.

CALCulate:LIMit:FAIL[:LAST]?

determines if the last measurement performed passed or failed the specified limits.

Response: 0 or 1

where: 0 - indicates that the measurement passed.
1 - indicates that the limits were not enabled.

Units: none

Example: MEAS:DEV:SAT?
5000
CALC:LIM:FAIL?
0
indicates that the measurement passed the limit check.

CALCulate:LIMit:MANufacturer:CODE <mfr code>
<mfr code> numeric integer value from 0 - 255

defines the manufacturer code for the custom limit table.

This value is used in determining the custom table selection when the `AUTO` select is enabled.

See `CALC:LIM:TYPE[:LOAD]:AUTO`.

Example: CALC:LIM:MAN:CODE 87
defines the manufacturer code to be 87 for the current custom limit table selection.

CALCulate:LIMit:MANufacturer:ESN:LOWer <esn value>
<esn value> numeric integer value from 0 - 16777215 (0xFFFFFF)

defines the lower ESN value for the custom limit table.

This value is used in determining the custom table selection when the `AUTO` select in enabled. This parameter is valued-coupled with `CALC:LIM:MAN:ESN:UPPer`. It cannot be set lower than the value specified by `UPPer`.

See `CALC:LIM:TYPE[:LOAD]:AUTO`.

Example: CALC:LIM:MAN:ESN:LOW #h39999
defines the lower ESN value to be 0x39999 for the current custom limit table selection.

CALCulate:LIMit:MANufacturer:ESN:UPPer <esn value>
<esn value> numeric integer value from 0 - 16777215 (0xFFFFFF)

defines the upper ESN value for the custom limit table.

This value is used in determining the custom table selection when the `AUTO` select in enabled

This parameter is valued-coupled with `CALC:LIM:MAN:ESN:LOWer`. It cannot be set lower than the value specified by `LOWer`.

See `CALC:LIM:TYPE[:LOAD]:AUTO`.

Example: CALC:LIM:MAN:ESN:UPP #h99999
defines the upper ESN value to be 0x99999 for the current custom limit table selection.

CALCulate:LIMit:MANufacturer:NAME <mfr name>
<mfr name> a string of 1 - 21 characters enclosed in double quotes

defines a name for the custom limit table.

Example: CALC:LIM:MAN:NAME "My limit table"
defines the name "My limit table" for the current custom limit table selection.

CALCulate:LIMit:TYPE[:LOAD]:AUTO <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables auto selection of the custom limit table, based on the manufacturer code and ESN range for the five custom tables defined.

When disabled, no change occurs in the limit table selection.

When enabled, the limit table selection is immediately modified whenever a registration or origination of a mobile is performed.

The registration determines the manufacturer and ESN for the mobile, which determines if a different limit table selection should be made.

If more than one custom limit table matches the registration information, the first one found (searching from `MFC1` to `MFC5`) will be selected. If none of the custom limit tables matches the registration information, `MFC1` is used.

Example: CALC:LIM:TYPE:AUTO ON
enables auto selection of the limit table.

CALCulate:LIMit:TYPE[:LOAD][:SELect] <limit table select>
<limit table select> NONE, EIA, MFC1, MFC2, MFC3, MFC4, MFC5

selects the limit table to use when performing measurements.

The limit parameters reflect the table selected by this command.

- If `NONE` is selected, all limits will be disabled and limit parameter changes will not be allowed.
- If `EIA` is selected, the limits are set to values specified by the EIA standard, and changes will not be allowed.
- Limit parameter values can only be modified if one of the Manufacturer/Custom tables is selected (MFC1-MFC5).

These tables are stored in non-volatile static memory, allowing them to be modified, retaining their values when the power to the 4300 is turned off.

The factory settings for the custom tables is the same as the EIA table.

Example 1: CALC:LIM:TYPE EIA
selects the EIA limits and disables the limit parameter commands.

Example 2: CALC:LIM:TYPE MFC3
selects custom table 3 limits and enables the limit parameter commands.

CALCulate:LIMit:TYPE:STORe <limit table>
<limit select> MFC1, MFC2, MFC3, MFC4, MFC5

saves the current limit table setup to the specified user definable limit table.

Limit table value changes can only be made if one of these user-definable selections is the current limit table selection.

CALCulate:LIMit: {*parameter*}:LOWer[:DATA] <value>
<value> numeric value specified by {*lower*} range in Table 33.

sets the lower limit value for the selected parameter.

This is a relative value that is added to the `NOMinal` value to determine the absolute minimum limit that the measurement is compared to.

The current custom limit table selection is automatically updated with this parameter value change.

CALCulate:LIMit: {*parameter*}:LOWer:STATe <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the minimum limit when limit checking the selected parameter.

The current custom limit table selection is automatically updated with this parameter value change.

CALCulate:LIMit: {*parameter*}:NOMinal <value>
<value> numeric value specified by {*nominal*} value in Table 33.

sets the nominal limit value for the selected parameter.

This affects both the upper and the lower absolute limits that the measurement is compared to. The absolute maximum and maximum limits are derived by adding the `NOMinal` value to the `LOWer` and `UPPer` values respectively. The current custom limit table selection is automatically updated with this parameter value change.

CALCulate:LIMit: {*parameter*}:STATe <boolean>
<boolean> OFF or ON (0 = OFF,1 = ON)

enables or disables limit checking for the selected parameter.

The current custom limit table selection is automatically updated with this parameter value change.

CALCulate:LIMit: {*parameter*}:UPPer[:DATA] <value>
<value> numeric value specified by {*upper*} range in Table 33.

sets the upper limit value for the selected parameter.

This is a relative value that is added to the `NOMinal` value to determine the absolute maximum limit that the measurement is compared to.

The current custom limit table selection is automatically updated with this parameter value change.

CALCulate:LIMit: {*parameter*}:UPPer:STATe <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the maximum limit when limit checking the selected parameter.

The current custom limit table selection is automatically updated with this parameter value change.

Table 33 and Table 34 contain the limit parameter selections that are substituted into the limit table parameter command.

TABLE 33. Limit Parameter Description

{parameter}	{description}
ATIMe	Acquisition time measurements for 2nd channel power < -103 dBm
BER:ANALog	Analog BER measurements
DEViation:SAT	SAT deviation measurements
DEViation:ST	ST deviation measurements
DEViation:AUDio[:AMPS]	audio and peak audio deviation measurements in AMPS mode
DEViation:AUDio:NAMPs	audio and peak audio deviation measurements in NAMPS mode
DEViation:NAMPs	DSAT and DST deviation measurements
DEViation:WBND	wideband data deviation measurements
DEViation:RESidual	Tx residual deviation measurements
DISTortion:RECeive	Rx distortion measurements
DISTortion:TRANsmit	Tx distortion measurements
DURation:FLASh	ST duration measurements for a hook flash
DURation:HANDoff	ST duration measurements for handoffs
DURation:RELease	ST duration measurements for a release
FREQuency:RF[:AMPS]	Tx frequency (and frequency error) measurements in AMPS mode
FREQuency:RF:NAMPs	Tx frequency (and frequency error) measurements in NAMPS mode

TABLE 33. Limit Parameter Description

{parameter}	{description}
FREQuency:RF:CDMA [:CELLular]	Tx frequency (and frequency error) measurements in CDMA mode in the cellular band
FREQuency:RF:CDMA:PCS	Tx frequency (and frequency error) measurements in CDMA mode in the PCS band.
FREQuency:SAT	SAT frequency (and frequency error) measurements
FREQuency:ST	ST frequency (and frequency error) measurements
POWer:MAC*n*	Mobile power measurements for MAC level *n* = 0 to 7
POWer:TRANsmitter: CDMA: APRobe:CHANge:LARGe	CDMA access probe output power measurements for power step > 0
POWer:TRANsmitter: CDMA: APRobe:CHANge:SMALl	CDMA access probe output power measurements for power step < 0
POWer:TRANsmitter: CDMA: APRobe: CHANge:TEST1	CDMA access probe output power measurements for power step = 0
POWer:TRANsmitter: CDMA:CLOSed[:RELative]	CDMA closed loop output power response measurements.
POWer:TRANsmitter:CDMA :CLOSed:PERCentage	CDMA closed loop output power response measurements.
POWer:TRANsmitter: CDMA :CLOSed: RANGe:MAXimum	CDMA closed loop output power response range measurements.
POWer:TRANsmitter: CDMA: CLOSed: RANGe:MINimum	CDMA closed loop output power response range measurements.
POWer:TRANsmitter:CDMA :CLOSed:DELay	CDMA closed loop output power response time measurements.
POWer:TRANsmitter:CDMA :GATed:ON	CDMA gated on power output measurements
POWer:TRANsmitter: CDMA: GATed: OFF [RELative:]	CDMA gated off output power measurements relative to powerr output measurements
POWer:TRANsmitter: CDMA: GATed: OFF: FLOor	CDMA gated off output power measurements relative to transmitter noise floor
POWer:TRANsmitter: CDMA: GATed: TIME: FALL	CDMA gated output power fall time measurements
POWer:TRANsmitter: CDMA: GATed: TIME: RISE	CDMA gated output power rise time measurements
POWer:TRANsmitter:CDMA :MAXimum[:CELLular]: CLASs 1:3	Maximum power measurements for CDMA Mobile Station Classes I, II, and III
POWer:TRANsmitter:CDMA :MAXimum: PCS:CLASs 1:5	Maximum power measurements for CDMA Mobile Station Classes I through V

TABLE 33. Limit Parameter Description

{parameter}	{description}
POWer:TRANsmitter:CDMA :MINimum	CDMA minimum controlled output power measurements
POWer:TRANsmitter:CDMA :OPEN:ESTimate	CDMA open loop output power estimate.
POWer:TRANsmitter: CDMA: OPEN: CHANge:SMALl	CDMA open loop output power response measurements.
POWer:TRANsmitter: CDMA: OPEN: CHANge:LARGe	CDMA open loop output power response measurements.
POWer:TRANsmitter: CDMA: OPEN: CHANge:OFFSet	CDMA open loop output power response measurements.
POWer:TRANsmitter:CDMA :STANdby	CDMA standby power output measurements
SENSitivity[:AMPS]	Sensitivity measurements for AMPS mode (used for base power level)
SENSitivity:CDMA	Sensitivity measurements for CDMA mode (used for base power level)
SENSitivity:CONFidence	Confidence for CDMA sensitivity
SENSitivity:FER	Error rate for CDMA sensitivity
SENSitivity:NAMPs	Sensitivity measurements for NAMPS mode (used for base power level)
SINad:RECeive	Rx SINAD measurements
SINad:TRANsmit	Tx SINAD measurements
STNoise:TRANsmit	Tx Signal to Noise measurements
STNoise:RECeive	Rx Signal to Noise measurements
TIME	Time error measurements in CDMA mode
VERRor:CFEedthrough	Carrier Feedthrough vector error measurements
VERRor:IQIMbalance	IQ imbalance vector error measurements
WQUality	Waveform Quality measurements in CDMA mode

TABLE 34. Command Range and Units

	Factory Settings			
{parameter}	{nominal }	{upper}	{lower}	{res}{units}
ATIMe	0, 5000	0, 5000	-5000, 0	1 msec
BER:ANALog	0, 100	0, 100	-100, 0	0.01%
BER:MAHO	0 (fixed)	0, 7	-7, 0	1 (none)

TABLE 34. Command Range and Units

DEViation:AUDio[:AMPS]	0, 20000	0, 20000	-20000, 0	1Hz
DEViation:AUDio:NAMPs	0, 20000	0, 20000	-20000, 0	1Hz
DEViation:NAMPs	0, 20000	0, 20000	-20000, 0	1Hz
DEViation:RESidual	0, 20000	0, 20000	-20000, 0	1Hz
DEViation:SAT	0, 20000	0, 20000	-20000, 0	1Hz
DEViation:ST	0, 20000	0, 20000	-20000, 0	1Hz
DEViation:WBND	0, 20000	0, 20000	-20000, 0	1Hz
DISTortion:RECeive	0, 40	0, 40	-40, 0	0.1dB
DISTortion:TRANsmit	0, 40	0, 40	-40, 0	0.1dB
DURation:FLASh	0, 5000	0, 5000	-5000, 0	1msec
DURation:HANDoff	0, 5000	0, 5000	-5000, 0	1msec
DURation:RELease	0, 5000	0, 5000	-5000, 0	1msec
FREQuency:RF[:AMPS]	-10000, 10000	0, 10000	-10000, 0	1Hz
FREQuency: RF: CDMA [:CELLular]	-10000, 10000	0, 10000	-10000, 0	1Hz
FREQuency: RF: CDMA: PCS	-10000, 10000	0, 10000	- 10000, 0	1Hz
FREQuency:RF:NAMPs	-10000, 10000	0, 10000	-10000, 0	1Hz
FREQuency:SAT	-10000, 10000	0, 10000	-10000, 0	1Hz
FREQuency:ST	-10000, 10000	0, 10000	-10000, 0	1Hz
POWer:TRANsmitter:CDMA: APRobe:CHANge :LARGe	0, 100	0, 100	-100, 0	0.1 dB
POWer:TRANsmitter: CDMA:APRobe:CHANge: SMALl	-30, 30	0, 30	-30, 0	0.1 dB
POWer:TRANsmitter: CDMA:CLOSed:DELay	-10, 10	0, 10	-10, 0	1
POWer:TRANsmitter: CDMA:CLOSed:PERCentage	-30, 30	0, 30	-30, 0	0.1 dB
POWer:TRANsmitter: CDMA:CLOSed:RANGe: MAXimum	-30, 30	0, 30	-30, 0	0.1dB

TABLE 34. Command Range and Units

POWer:TRANsmitter: CDMA:CLOSed:RANGe: MINimum	–30, 30	0, 30	–30, 0	0.1dB
POWer:TRANsmitter:CDMA: CLOSed [:RELative]	–30, 30	0, 30	–30, 0	0.1 dB
POWer:TRANsmitter :CDMA:GATed:OFF:FLOor	–90, 10	0, 30	–30, 0	0.1dB
POWer:TRANsmitter:CDMA: GATed:OFF [RELative:]	–30, 30	0, 30	–30, 0	0.1dB
POWer:TRANsmitter: CDMA :GATed:ON	–30, 30	0, 10	–10, 0	0.1dB
POWer:TRANsmitter: CDMA :GATed:TIME:FALL	0, 1280	0, 30	–30, 0	1 usec
POWer:TRANsmitter:CDMA: GATed:TIME:RISE	–30, 30	0, 30	–30, 0	1 usec
POWer:TRANsmitter: CDMA :MAXimum[:CELLular]: CLASs 1:3	–60, 10	0, 30	–30, 0	0.1dBW
POWer:TRANsmitter: CDMA :MAXimum: PCS: CLASs 1:5	–60, 10	0, 30	–30, 0	0.1 dBW
POWer:TRANsmitter: CDMA:MINimum	–90, 10	0, 30	–30, 0	0.1 dBW
POWer:TRANsmitter:CDMA: OPEN:CHANge:LARGe	0, 100	0, 100	–100, 0	0.1 dB
POWer:TRANsmitter: CDMA:OPEN:CHANge: OFFSet	–30, 30	0, 30	–30, 0	0.1 dB
POWer:TRANsmitter: CDMA:OPEN:CHANge: SMALl	–30, 30	0, 30	–30, 0	0.1 dB
POWer:TRANsmitter: CDMA :OPEN:ESTimate	–30, 30	0, 30	–30, 0	0.1 dB
POWer:TRANsmitter: CDMA :STANdby	–90, 10	0, 30	–30, 0	0.1 dBW
SENSitivity[:AMPS]	–130, -50	0 (fixed)	0 (fixed)	0.1 dBW
SENSitivity:NAMPs	–130, -50	0 (fixed)	0 (fixed)	0.1 dBW
SINad:RECeive	0, 40	0, 40	–40, 0	0.1dB
SINad:TRANsmit	0, 40	0, 40	–40, 0	0.1dB
STNoise:RECeive	0, 40	0, 40	–40, 0	0.1dB
STNoise:TRANsmit	0, 40	0, 40	–40, 0	0.1dB
TIME	–100, 100	0, 100	–100, 0	0.001 usec
VERRor:CFEedthrough	–30, 0	–30, 0	–30,0	0.1 dBc
VERRor:IQIMbalance	–30, 0	–30, 0	–30, 0	0.1 dBc
WQUality	0, 1	0, 1	–1, 0	0.0001

CALibration Subsystem

The *Calibration* subsystem commands allow the test equipment to compensate for various factors.

CALibration[:ALL]:AUTO <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the auto-calibration for the CDMA Control Module (CCM).

The calibration is performed every 10 minutes for the first 30 minutes of operation, and every 30 minutes thereafter.

The calibration takes 15 minutes and pre-empts all other tasks in order to prevent disk accesses, sequencer operation, and remote operation during this time period.

This command is valid only if the CCM option is installed.

CALibration[:ALL]:DATE?

query-only command which returns the date of the last factory calibration.

CALibration[:ALL][:IMMediate]

performs a calibration required for the CCM.

This command is valid only if the CCM option is installed.

CALibration:CABLe:COMPensation <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the cable loss factor.

CALibration:CABLe:DATA[*n*] <cable loss>
[*n*] numeric integer value 1 - 3
<cable loss> numeric value from 0.0 - 5.0 dB (resolution of 0.1)

defines the cable loss value for up to three cables.

The cable loss value is selected using `CAL:CABL:SEL` only when the `CAL:CABL:COMP` parameter is set to `ON`.

CALibration:CABLe:SELection <cable select>
<cable select> numeric integer value 1 - 3

selects which cable loss factor to use.

The cable loss value is selected using `CAL:CABL:SEL` only when the `CAL:CABL:COMP` parameter is set to `ON`.

Call Processing Subsystem

The *Call Processing* subsystem serves several purposes.

One purpose of the call processing subsystem is to allow the user to control the parameters used during call processing. Some of the parameters that can be controlled are the type of processing (analog or digital), the channel and band that the unit used to communicate with the mobile, and the system parameters that are transmitted to the mobile through overhead messages.

The call processing subsystem is also used to initiate any call processing activity: ordered registrations, pages and originations, handoffs, flashes and alerts, transmit power changes, and any other order that might be sent to the mobile.

The call processing subsystem can determine the status of call processing or the mobile that is being tested including: current call processing type channel, completion status of call processing commands, and mobile identity.

CALL:ALERt:CNI1:NPLan <number plan>
CALL:ALERt:CNI2:NPLan <number plan>
<number plan> numeric integer value 0 - 15

sets the Numbering Plan Identification, as defined in ANSI T1.607, used for the caller ID for Alert With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters setup for `CNI1`.

where:

0	Unknown
1	ISDN/Telephony numbering plan (CCITT E.164 and E.163)
2	Reserved
3	Data Numbering Plan (CCITT X.121)
4	Telex numbering plan (CCITT F.69)
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Private numbering plan
10	Reserved
11	Reserved
12	Reserved
13	Reserved

14	Reserved
15	Reserved for extension

See CALL:HFLash:TRANsmit:CNI1:NPLan.

CALL:ALERt:CNI1:NTYPe <number type>
CALL:ALERt:CNI2:NTYPe <number type>
<number type> numeric integer value 0 - 7

selects the Type of Number, as defined in ANSI T1.607, used for the caller ID for Alert With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters set up for `CNI1`.

where:

0	Unknown
1	International number
2	National number
3	Network-specific number
4	Subscriber number
5	Reserved
6	Abbreviated number
7	Reserved for extension

See `CALL:HFLash:TRANsmit:CNI1:NTYPe.`

CALL:ALERt:CNI1:NUMBer <caller id info>
CALL:ALERt:CNI2:NUMBer <caller id info>
<caller id info> ASCII string, 15 character maximum enclosed in double quotes

defines the text (mobile number, name) used for the calling number indicator (caller ID) for Alert With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the `CNI1` parameters.

CALL:ALERt:CNI1:PI <presentation indicator>
CALL:ALERt:CNI2:PI <presentation indicator>
<presentation indicator> numeric integer value 0 - 3

sets the Presentation Indicator (PI) selection, as defined in ANSI T1.607, used for the caller ID for Alert With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the parameters set up for `CNI1`.

where:

0	Presentation allowed
1	Presentation restricted
2	Number not available
3	Reserved

See `CALL:HFLash:TRANsmit:CNI1:PI`.

CALL:ALERt:CNI1:SI <screening indicator>
CALL:ALERt:CNI2:SI <screening indicator>
<screening indicator> numeric integer value 0 - 3

sets the Screening Indicator (`SI`) selection, as defined in ANSI T1.607, used for the caller ID for Alert With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the `CNI1` parameters.

where:

0	User-provided, not screened
1	User-provided, verified and passed
2	User-provided, verified and failed
3	Network-provided

See `CALL:HFLash:TRANsmit:CNI1:SI`.

CALL:ALERt[:IMMediate]

sends the Alert With Info message to the mobile, with the information specified by `CALL:ALERt:SELect`.

If the first parameter is `NONE`, an Alert message is sent to the mobile, instead of the Alert With Info message.

The mobile must be up on a call when this command is executed, or an error is indicated.

This command cannot be queried.

CALL:ALERt:INFOrmation:AMPS <boolean>
<boolean> OFF, ON (0 = OFF, 1 = ON)

enables or disables the Alert with Info and Flash with Info messages sent to the mobile (if the mobile protocol indicates it is EIA 553 or IS-54A).

These messages are not supported in this protocol.

CALL:ALERt:SELect <info select1>, [info select2], [info select3]
<info select> NONE, SIGNal, CNI1, CNI2

selects which information is sent to the mobile when `CALL:ALERt[:IMMediate]` is executed.

On an AMPS traffic channel, only one parameter is allowed, and it can only be `CNI1` or `NONE`.

On a CDMA traffic channel, up to 3 selections may be made and can be any of the following choices for <info select>:

NONE no information sent
SIGNal alert signal (for distinctive ring)
CNI1 calling number indicator (caller ID info)
CNI2 second calling number indicator (second caller ID info).

CALL:ALERt:SIGNal:CADence <cadence select>
<cadence select> numeric integer value 0 - 63

sets the cadence of the ring used for the Alert With Info messages sent to the mobile.

where:

0	No tone
1	long
2	short-short
3	short-short-long
4	short-short-2
5	short-long-short
6	short-short-short-short
7	PBX long
8	PBX short-short
9	PBX short-short-long
10	PBX short-long-short
11	PBX short-short-short-short
12-63	Reserved

See `CALL:HFLash:TRANsmit:SIGNal:CADence`.

CALL:ALERt:SIGNal:PITCh <pitch select>
<pitch select> MEDium, HIGH, LOW, REServed

sets the pitch of the ringing tone used for Alert With Info messages sent to the mobile.

CALL:BRELease

This overlapped command uses the Call Processing resources to perform a base release, terminates the current call, and returns to the current setup for the forward control channel specified by the `CALL:CONTROL` parameters.

See `STAT:OPER:COMP`.

When on an AMPS call, the test equipment performs an ST duration measurement upon completion. By following this command with the ST duration measurement query, the ST duration of the base release is reported.

The mobile must be up on a call when this command is executed, or an execution error is indicated.

This command cannot be queried.

CALL:CONTrol:BAND <band select>
<band select> CELLular, PCS or KORean

selects which band to use for the control channel.

The selection is activated by the `CALL: FCC ON` command.

The PCS band selection is allowed only if the PCS option is installed.

CALL:CONTrol:CCCHannel:ACCess:PROBe:POWer:INITial <power >
<power > numeric value -16 to 15 dB

selects the `INIT_PWR` parameter in the access parameter message sent to the mobile over the paging channel, which effects the mobile's open loop power estimate.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:ACCess:PROBe:POWer:NOMinal <power >
<power > numeric value -24 to 7 dB (excluding -24 to -9 for cellular)

sets the NOM_PWR parameter in the access parameter message sent to the mobile over the paging channel, which effects the mobile's open loop power estimate.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:ACCess:PROBe:POWer:STEP <power >
<power> numeric value 0 - 7 dB

sets the `PWR_STEP` parameter in the access parameter message sent to the mobile over the paging channel, which effects the mobiles output power step per access probe in an access sequence.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:ACCess:PROBe:PREamble <preamble >
<preamble> numeric value 1 - 16

sets the preamble length (in access channel frames) output by the mobile on the Access channel.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:ACCess:PROBe:REQuests <seq>
<seq> numeric value 1 - 15

sets the MAX_REQ_SEQ parameter in the access parameter message sent to the mobile over the paging channel, which sets the maximum number of probe sequences output by the mobile in an access attempt.

The selection is activated by `CALL: FCC ON` .

CALL:CONTrol:CCCHannel:ACCess:PROBe:RESPonses <seq>
<seq> numeric value 1 - 15

sets the MAX_RSP_SEQ parameter in the access parameter message sent to the mobile over the paging channel, which sets the maximum number of access sequences the mobile should attempt when responding to a base station message.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:ACCess:PROBe:STEPs <step select>
<step select> numeric value 1 - 16

sets the `NUM_STEP` parameter in the access parameter message sent to the mobile over the paging channel, which sets the number of probes output by the mobile per access sequence.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:ACCess:PROBe:TIMeout <timeout>
<timeout> numeric value from 0.160 - 1.360 (resolution of 0.080)

sets the timing between access probes of an access probe sequence (the amount of time a mobile waits for an acknowledgement before the next access probe is transmitted).

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:BID <BID value>
<BID value> numeric value 0 - 65535

selects a Base station IDentification (`BID`) code for the CDMA control channel.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:IMSI:MCC <country code>
<country code> numeric integer 0 - 999

selects the Mobile Country Code (`MCC)` used as part of the IMSI addressing of the mobile on a digital control channel.

CALL:CONTrol:CCCHannel:IMSI:MNC <network code>
<network code> numeric integer 0 - 99

selects a Mobile Network Code (`MNC)` as part of the IMSI addressing of the mobile on a digital control channel.

CALL:CONTrol:CCCHannel:LATitude <latitude value>
<latitude value> numeric value -1296000 to 1296000

selects a base station latitude (`LAT)` for the CDMA control channel where positive numbers represent northern latitudes.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:LONGitude <longitude value>
<longitude value> numeric value -2592000 to 2592000

selects a base station longitude (`LONG`) for the CDMA control channel where positive numbers represent eastern longitudes.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:NID <NID value>
<NID value> numeric value 0 - 65535

selects a Network IDentification (`NID`) code for the CDMA control channel.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:PNOFfset <PN offset select>
<PN offset select> numeric value 0 - 511

selects a PN Offset (Pilot PN Sequence Offset Index) for the CDMA control channel.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:RATE <rate select>
<rate select> FULL or HALF

selects a rate for the CDMA control channel.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:REGistration:TIMer <boolean>
<boolean> 0 or 1 (0 = OFF, 1 = ON)

enables or disables timed registrations on the mobile.

If enabled (Boolean = 1), the mobile performs periodic registrations while on a CDMA control channel. The timer period is set to a value of 29 (~ 14 seconds).

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:SEARch:CRITeria:ADD <level>
<level> numeric value –31.5 to 0.0 dB (default = -14.0 dB)

selects the T_ADD threshold value used by the mobile to compare to the strengths of the pilot signals it is tracking.

If the Ec/Io value of one of the pilots (not in the Active Set) exceeds this threshold, a pilot strength measurement report is sent to the base station, to inform it that the pilot is a candidate for a handoff.

This corresponds to the "Add Threshold" soft key on the CDMA Search Criteria screen.

This parameter is value-coupled with the corresponding `CALL:CONTrol:CCCH:SEARch:CRITeria:DROP` command.

To avoid a -221 "Settings Conflict" error, ensure that T_ADD ≥ T_DROP.

This command sets a parameter that must be sent to the mobile before it takes effect.

This occurs in the System Parameters message over the CDMA paging channel when the next `CALL:FCC ON` command is issued or in the Handoff Direction message over the CDMA traffic channel when the next `CALL:HOFF[:IMM]` or `CALL:HOFF:SOFTer` commands are issued.

CALL:CONTrol:CCCHannel:SEARch:CRITeria:COMP <level>

<level> numeric value 0.0 - 7.5 dB (default = 2.5 dB)

selects the T_COMP threshold value used by the mobile to compare to the strengths of the pilot signals it is tracking.

If the Ec/Io value of one of the pilots not in the Active Set exceeds this threshold plus 0.5 dB, it sends a pilot strength measurement report to the base station to inform it that the pilot is a candidate for a handoff.

This command corresponds to the "Comp Threshold" soft key on the CDMA Search Criteria screen.

This command sets a parameter that must be sent to the mobile before it takes effect.

This occurs in the System Parameters message over the CDMA paging channel when the next `CALL:FCC ON` command is issued, or in the Handoff Direction message over the CDMA traffic channel when the next `CALL:HOFF[:IMM]` or `CALL:HOFF:SOFTer` commands are issued.

CALL:CONTrol:CCCHannel:SEARch:CRITeria:DROP <level>

<level> numeric value –31.5 to 0.0 dB (default = -16.0 dB)

selects the T_DROP threshold value used by the mobile to compare to the strengths of the pilot signals it is tracking.

If the Ec/Io value of one of the pilots in the Active or Candidate Sets is less than this threshold, it starts the drop timer for that pilot.

When the timer expires, another pilot strength measurement is sent to the base station.

This corresponds to the "Drop Threshold" soft key in the CDMA Search Criteria screen.

This parameter is value-coupled with the corresponding `CALL:CONTrol:CCCH:SEARch:CRITeria:ADD` command.

To avoid a (-221) "Settings Conflict" error, ensure that T_ADD ≥ T_DROP.

This command sets a parameter that must be sent to the mobile before it takes effect.

This occurs in the System Parameters message over the CDMA paging channel when the next `CALL:FCC ON` command is issued, or in the Handoff Direction message over the CDMA traffic channel when the next `CALL:HOFF[:IMM]` or `CALL:HOFF:SOFTer` commands are issued.

CALL:CONTrol:CCCHannel:SEARch:CRITeria:TIMer <value>

<value> numeric value 0 - 15 (default = 3)

This command sets the T_TDROP timer value.

The pilot drop timer starts when its signal strength falls below the T_DROP value.

When the timer expires, a pilot strength measurement is sent to the base station and, if the pilot was not in the Active Set, it is moved to the Neighbor Set.

This command corresponds to the "Drop Timer" soft key in the CDMA Search Criteria screen.

The timer values correspond with the following time delays:

T_TDROP value	Time (seconds)
0	≤0.1
1	1
2	2
3	4
4	6
5	9
6	13
7	19
8	27
9	39
10	55
11	79
12	112
13	159

T_TDROP value	Time (seconds)
14	225
15	319

This command sets a parameter that must be sent to the mobile before it takes effect.

This occurs in the System Parameters message over the CDMA paging channel when the next `CALL:FCC ON` command is issued, or in the Handoff Direction message over the CDMA traffic channel when the next `CALL:HOFF[:IMM]` or `CALL:HOFF:SOFTer` commands are issued.

CALL:CONTrol:CCCHannel:SEARch:WINDow:ACTive <window select>
<window select> numeric value 0 - 15

selects the search window size for the Active and Candidate sets (SRCH_WIN_A) used by the mobile.

Window select	Window size (PN chips)
0	4
1	6
2	8
3	10
4	14
5	20
6	28
7	40
8	60
9	80
10	100
11	130
12	160
13	226
14	320
15	452

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:SEARch:WINDow:AGE <age select>
<age select> numeric value 0 - 15

sets the maximum age (NGHBR_MAX_AGE) used by the mobile to determine when to remove a pilot channel from the neighbor set.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:SEARch:WINDow:NEIGhbor <neighbor set select>
<neighbor set select> numeric value 0 - 15

selects the search window size for the Neighbor set (SRCH_WIN_N) used by the mobile.

Window select	Window size (PN chips)
0	4
1	6
2	8
3	10
4	14
5	20
6	28
7	40
8	60
9	80
10	100
11	130
12	160
13	226
14	320
15	452

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:SEARch:WINDow:REMaining <remaining set select>
<remaining set select> numeric value 0 - 15

selects the search window size for the Remaining set (SRCH_WIN_R) used by the mobile.

Window select	Window size (PN chips)
0	4
1	6
2	8
3	10
4	14
5	20
6	28
7	40
8	60
9	80
10	100
11	130
12	160
13	226
14	320
15	452

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:CCCHannel:ZID <ZID value>
<ZID value> numeric value 0 - 4095

selects a registration Zone IDentification (`ZID`) code for the CDMA control channel.

The selection is activated by `CALL: FCC ON.`

CALL:CONTrol:CHANnel <channel select>
<channel select> numeric value 1 - 1999

sets the channel number to use for systems A or B.

The selection is activated by `CALL: FCC ON`.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:CONTrol:DCC <code value>
<code value> numeric value 0 - 3

selects a Digital Color Code (`DCC`) for systems A or B.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:MAC <mobile power level>
<mobile power level> numeric value 0 - 7 (0 - 10 for digital control channel)

selects a digital Mobile Attenuation Code (`MAC`) level for the analog control channel.

The selection is activated by `CALL: FCC ON`.

If the mobile power level is > 7 when `CALL:CONTrol:TYPE` is changed to `ANALog`, then the mobile power level is set to 7.

CALL:CONTrol:SID <SID value>
<SID value> numeric value 0 - 32767

selects a System IDentification (`SID`) code for systems A or B.

The selection is activated by `CALL: FCC ON`.

CALL:CONTrol:SYSTem <system select>
<system select> A or B

selects whether system A or B is used.

CALL:CONTrol:TYPE <channel type>
<channel type> where:ANALog, CDMA
ANAlog = AMPS operation
CDMA = CDMA operation

selects the type of control channel to use.

The selection is activated by `CALL: FCC ON`.

CDMA operation requires that the CCM option is installed.

CALL:ERRor?

returns the status of the last call processing execution command, and is used to detect when a mobile fails to complete a call processing task (handoff, registration, origination, page).

Response: <numeric integer>

0 = passed
1 = failed
2 = aborted
3 = overlapped command is still in process

Units: none

Example: CALL:PAGE AMPS;*WAI
CALL:ERR?
0
indicates that the page completed successfully.

CALL:FCC <boolean>

<boolean> 0 or 1 (0 = OFF, 1 = ON)

When set to `OFF`, the control channel is disabled.

When set to `ON`, this command sets the 4300 and mobile to the control channel parameters specified by `CALL: CONTrol`.

Setup the `CALL: CONTrol` parameters prior to executing this command.

CALL:HFLash[:RECeive]:DATA?

returns the information sent by the mobile when it performed a hook flash with information.

Response: <15 character string>

Example: CALL:PAGE CDMA: *WAI
CALL:HFL:DATA?
"1234567"

Requirements: Mobile must perform a hook flash with information to obtain valid data.

Resources required: none

CALL:HFLash[:RECeive][:IMMediate]

instructs the test equipment that a hook flash ST duration measurement is about to be made.

This is an overlapped command with an enable code of 0010 hexidecimal (16 decimal) requiring no exclusive resources and does not conflict with any other overlapped command.

See `STAT:OPER:COMP`.

Press the proper key on the handset to perform the hook flash, which sets the operation complete flag for a `HOOKflash` (used by the `*WAI` command).

Follow this command with an ST duration measurement query in order to return the measurement value.

The mobile must be up on a call when this command is executed, or an execution error is indicated.

This command cannot be queried.

CALL:HFLash:TRANsmit:CNI1:NPLan <number plan>
CALL:HFLash:TRANsmit:CNI2:NPLan <number plan>
<number plan> numeric integer value 0 - 15

sets the Numbering PLan identification (`NPL`), as defined in ANSI T1.607, used for the caller ID for Flash With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the `CNI1` parameters.

The definitions for <number plan> include:

0	Unknown
1	ISDN/Telephony numbering plan (CCITT E.164 and E.163)
2	Reserved
3	Data Numbering Plan (CCITT X.121)
4	Telex numbering plan (CCITT F.69)
5	Reserved
6	Reserved
7	Reserved
8	Reserved
9	Private numbering plan
10	Reserved
11	Reserved
12	Reserved
13	Reserved
14	Reserved
15	Reserved for extension

CALL:HFLash:TRANsmit:CNI1:NTYPe <number type>
CALL:HFLash:TRANsmit:CNI2:NTYPe <number type>
<number type> numeric integer value 0 - 7

set the Type of Number, as defined in ANSI T1.607, used for the caller ID for Flash With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the `CNI1` parameters.

The definitions for <number type> include:

0	Unknown
1	International number
2	National number
3	Network-specific number

0	Unknown
4	Subscriber number
5	Reserved
6	Abbreviated number
7	Reserved for extension

CALL:HFLash:TRANsmit:CNI1:NUMBer <caller id info>
CALL:HFLash:TRANsmit:CNI2:NUMBer <caller id info>
<caller id info> ASCII string 15 character maximum enclosed in double quotes

defines the text used for the calling number indicator (caller ID) for Flash With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the `CNI1` parameters.

CALL:HFLash:TRANsmit:CNI1:PI <presentation indicator>
CALL:HFLash:TRANsmit:CNI2:PI <presentation indicator>
<presentation indicator>numeric integer value 0 - 3

sets the Presentation Indicator selection (`PI`), as defined in ANSI T1.607, used for the caller ID for Flash With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the `CNI1` parameters.

The definitions for the <presentation indicator> include:

0	Presentation allowed
1	Presentation restricted
2	Number not available
3	Reserved

CALL:HFLash:TRANsmit:CNI1:SI <screening indicator>
CALL:HFLash:TRANsmit:CNI2:SI <screening indicator>
<screening indicator>numeric integer value 0 - 3

sets the Screening Indicator selection (`SI`), as defined in ANSI T1.607, used for the caller ID for Flash With Info messages sent to the mobile.

CDMA allows two caller IDs; the `CNI1` and `CNI2` selection in the command allows access to each of these.

AMPS allows one caller ID; the 4300 uses the `CNI1` parameters.

The definitions for the <screening indicator> include:

0	User-provided, not screened
1	User-provided, verified and passed
2	User-provided, verified and failed
3	Network-provided

CALL:HFLash:TRANsmit[:IMMediate]

sends the Flash With Info message to the mobile, with information specified by `CALL:HFLash:TRANsmit:SELect`.

If the first parameter specified is `NONE`, a Hook Flash message is sent to the mobile instead of the Flash With Info.

If the first parameter specified is `MWI` and the mobile is on an AMPS call (dual-mode mobile only), the Message Waiting message is sent to the mobile instead of the Flash With Info.

The mobile must be up on a call when this command is executed, or an error is indicated.

This command cannot be queried.

CALL:HFLash:TRANsmit:MWAIting <number of msgs>
<number of msgs> numeric integer value 0 - 255

sets the number of messages waiting for the message waiting feature.

A value of 0 indicates `NONE.` (This can be used to clear the indicator on the mobile.)

A value of 255 indicates some undetermined number > 0.

CALL:HFLash:TRANsmit:SELect < sel1>[, sel2] [, sel3] [, sel4]
<sel 1-4> NONE, SIGNal, MWI, CNI1, CNI2

selects the information to be sent to the mobile when the `CALL:HFLash:TRANsmit[:IMMediate]` command is executed.

On an AMPS channel, only the first parameter is allowed, and must be either `MWI`, `CNI1`, or `NONE`.

On a CDMA traffic channel, up to four selections may be made and can be any of the following choices for <sel>:

NONEno information sent
SIGNalalert signal (for distinctive ring)
MWImessage waiting indicator
CNI1calling number indicator (caller ID info)
CNI2second calling number indicator (second caller ID info)

CALL:HFLash:TRANsmit:SIGNal:CADence <cadence select>
<cadence select> numeric integer value 0 - 63

sets the cadence of the ring for Flash With Info messages sent to the mobile.

Each of the following patterns are repeating:

Value	Description
0	No tone
1	long
2	short-short
3	short-short-long
4	short-short-2
5	short-long-short
6	short-short-short-short
7	PBX long
8	PBX short-short
9	PBX short-short-long
10	PBX short-long-short
11	PBX short-short-short-short
12-63	Reserved

CALL:HFLash:TRANsmit:SIGNal:PITCh <pitch select>
<pitch select> MEDium, HIGH, LOW, or REServed

sets the pitch of the ringing tone for Flash With Info messages sent to the mobile.

CALL:HOFF:ASET1?

returns the sector specified as the first active channel in the last softer handoff.

Return value: SECT1 or SECT2

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:ASET2?

returns the sector specified as the second active channel in the last softer handoff.

`NONE` is the return value if a handoff to both sectors was performed.

Return value: SECT1, SECT2 or NONE

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:BAND <band select>
<band select> CELLular, PCS, or KORean

selects the `BAND` used for the next handoff.

If a band change is dictated by the handoff, a Dedicated Digital Traffic Channel Handoff message is sent to the mobile instead of the normal Handoff message.

The PCS or KORean band selection is allowed only if the PCS or KORean option is installed.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the handoff voice channel number.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:FOFFset <frame offset select>
<frame offset> numeric value 0 - 15

sets the frame offset value of the traffic channel to be used when the next hard handoff to a CDMA traffic channel is performed.

The frame offset is a time-skewing of the traffic channel frames from the system time.

The programmed value is in units of power control groups; a power control group is defined as 1.25 msec or 1/16 of a frame.

The maximum frame offset is 15, which = 18.75 milliseconds.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF[:IMMediate]

performs a handoff to the analog voice channel or digital traffic channel, as specified by the handoff parameter setup commands.

Set up the handoff parameters prior to executing this command.

This command indicates an error if `CALL:HOFF:TYPE` is CDMA and the CCM option is not installed.

CALL:HOFF:MAC <mobile power level>
<mobile power level> numeric value 0 - 10 (AMPS = 0 - 7)

selects the handoff mobile access code (`MAC`) level.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:NOMinal <nominal power select>
<nominal power select> numeric value -24 to 7 dB (excluding -24 to -9 for cellular)

selects the nominal power level for a handoff.

This command is valid only if the CCM option is installed.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:OFFset <offset>
<offset> MID, LOW, or HIGH

selects the handoff channel offset.

This applies only to the NAMPS channel type.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:SAT <SAT select>

<SAT select> numeric value (0 - 2 for AMPS, 0 - 6 for NAMPS)

where <SAT select> represents:

SAT frequency vector for AMPS

where:0 = 5970 Hz
1 = 6000 Hz
2 = 6030 Hz

DSAT vector for NAMPS

where:0 = 2556CB
1 = 255B2B
2 = 256A9B
3 = 25AD4D
4 = 26AB2B
5 = 26B2AD
6 = 2969AB

selects the SAT or DSAT vector code to handoff to.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:SOFTer <type>

<type > SECTor1, SECTor2, or BOTH (optional)

issues the handoff direction message to the mobile, specifying the sectors to place in the Active Set.

This corresponds to the Handoff to Both, Handoff Sector 1, and Handoff Sector 2 soft keys in the CDMA Softer Handoff screen.

If this command is issued without specifying the type, a softer handoff to the current Active Set specification is performed.

This is useful for sending new search criteria information without affecting the Active Set.

See `CALL:CONTrol:CCCH:SEARch:CRITeria`.

If `BOTH` is specified, the Active Set consists of both Sector 1 and Sector 2 pilots; the first pilot in the list is unchanged. That is, if handing off from `SECTor1` to `BOTH`, the first pilot specified in the Active Set is Sector 1 pilot. If handing off from `SECTor2` to `BOTH`, the first pilot is sector 2 pilot.

The order of the Active Set pilots affects the order the pilots are reported in the mobile power report.

See `CALL:PREPort:PILot[n]:POWer?`.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:TCODe <traffic code select>
<traffic code select> numeric value 2 - 63 (excluding 32)

selects the CDMA traffic channel (Walsh Code) to handoff to.

This command is valid only if the CCM option is installed.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:HOFF:TYPE <voice select>
<voice select> AMPS, NAMPs, or CDMA

selects the voice channel type to handoff to.

The CDMA selection is valid only if the CCM option is installed.

Issue the `CALL:HOFF` command on the next line or, add `:IMM` to the end of the command string.

CALL:INITial:AMPS:AVAIlable <boolean>
<boolean> ON, OFF (1 = ON, 0 = OFF)

determines if a voice channel is available when originating a call from an AMPS control channel.

If no voice channel is available, the mobile generates a system (fast) busy signal.

CALL:INITial:AMPS:BAND <band select>
<band select> CELLular, PCS, or KORean

sets which `BAND` is used for the next page or origination to an AMPS voice channel.

The PCS or KORean band selection is allowed only if the PCS or KORean option is installed.

CALL:INITial:AMPS:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the initial voice channel number to be used for the AMPS system.

If the channel is not defined (800-989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:INITial:AMPS:MAC <mobile power level>
<mobile power level>numeric value 0 - 7

selects the MAC level to be used for the AMPS voice channel.

CALL:INITial:AMPS:SAT <SAT frequency select>
<SAT frequency select>numeric value 0 - 2
where: 0 = 5970 Hz
1 = 6000 Hz
2 = 6030 Hz

selects the SAT frequency code to be used for the AMPS system.

CALL:INITial:CDMA:BAND <band select>
<band select> CELLular, PCS, or KORean

selects which BAND to use for the next page or origination to a CDMA traffic channel.

The PCS or KORean band selection is allowed only if the PCS or KORean option is installed.

CALL:INITial:CDMA:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects which initial traffic channel number to use for the CDMA system.

If the channel is not defined (800-989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:INITial:CDMA:FOFFset <frame offset>
<frame offset> numeric value 0 - 15

selects which frame offset value of the traffic channel to use when the next page or origination to a CDMA traffic channel is performed.

The frame offset skews the traffic channel frames from the system time.

The programmed value is in units of power control groups; a power control group is defined as 1.25 msec or 1/16 of a frame.

The maximum frame offset is 15, which = 18.75 milliseconds.

CALL:INITial:CDMA:SCONfig:RCONfig[:FORWard] <integer>
<integer> integer value 1-5

selects which forward radio configuration to use on the next CDMA2000 call (page only).

If the current service option is not valid for the new RC value, the service option is changed to the lowest valid value.

See CALL:INIT:CDMA:SCON:SOPT.

If a CDMA call is initiated on a non CDMA2000-capable mobile, this parameter has no effect on the call.

This command returns a -241, "Hardware missing error" if the IS-2000 option is not installed.

Example: CALL:INIT:CDMA:SCON:RCON:FORW 5
CALL:PAGE CDMA

CALL:INITial:CDMA:SCONfig:RCONfig:REVerse?

selects which reverse radio configuration to use on the next mobile page.

This command returns a -241, "Hardware missing error" if the IS-2000 option is not installed.

Returns: integer value 1-4

Example: CALL:INIT:CDMA:SCON:RCON:FORW 5
CALL:INIT:CDMA:SCON:RCON:REV?
4

CALL:INITial:CDMA:SCONfig:SOPTion <integer>
<integer> integer value 1-55

selects which Service Option to use on the next CDMA2000 call (page only).

If a CDMA call is initiated on a non CDMA2000-capable mobile, this parameter is ignored and the `CALL:INITial:CDMA:SOPTion` value is used.

See `CALL:INITial:CDMA:SOPTion.`

This command returns a -241, "Hardware missing error" if the IS-2000 option is not installed.

Example: CALL:INIT:CDMA:SCON:SOPT 55; RCON 3
CALL:PAGE CDMA

CALL:INITial:CDMA:SOPTion <service option>
<service option> numeric integer value

where:1 voice mode using 8k rate
2 loopback mode using 8k rate
3 voice mode using 8k EVRC
9 loopback mode using 13k rate
32768voice using 13k rate

selects which Service Option to use on the CDMA traffic channel when a page is performed.

The mobile selects the Service Option on origination.

CALL:INITial:CDMA:TCODE <traffic code>
<traffic code> numeric value 2 - 63 (excluding 32)

selects which traffic (Walsh) code channel to use for the CDMA system.

Walsh Code 32 is reserved for the sync channel.

CALL:INITial:NAMPs:BAND <band select>
<band select> CELLular, PCS, or KORean

sets which `BAND` to use for the next page or origination to an NAMPS voice channel.

`CALL:INITial:NAMPS:CHANNEL <channel select>`

`<channel select>numeric value 1 - 1999`

selects which initial voice channel number to use for the NAMPS system.

If the channel is not defined (800-989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:INITial:NAMPs:DSAT <DSAT vector select>
<DSAT vector select>numeric value 0 - 6
where: 0 = 2556CB
1 = 255B2B
2 = 256A9B
3 = 25AD4D
4 = 26AB2B
5 = 26B2AD
6 = 2969AB

selects which DSAT vector code to use for the NAMPS system.

CALL:INITial:NAMPs: MAC <mobile power level>
<mobile power level>numeric value 0 - 7

selects the `MAC` level to be used for the NAMPS voice channel.

CALL:INITial:NAMPs:OFFSet <channel offset level>
<channel offset select>MID, LOW, or HIGH

selects the channel offset to be used for the NAMPS system.

CALL:INITial:SYSA:BAND <band select>
CALL:INITial:SYSB:BAND <band select>
<band select> CELLular, PCS, or KORean

sets which `BAND` to use when establishing a control channel.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:BID <BID value>
CALL:INITial:SYSB:CCCHannel:BID <ID value>
<BID value> numeric value 0 - 65535

selects which Base station IDentification (`BID`) code to use for systems A or B.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:IMSI:MCC <country code>
CALL:INITial:SYSB:CCCHannel:IMSI:MCC <country code>
<country code> numeric integer 0 - 999

sets the Mobile Country Code (`MCC`) used as part of the International Mobile Station Identifier (IMSI) addressing of the mobile for systems A or B.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:IMSI:MNC <network code>
CALL:INITial:SYSB:CCCHannel:IMSI:MNC <network code>
<network code> numeric integer 0 - 99

sets the Mobile Network Code (`MNC`) used as part of the IMSI addressing of the mobile for systems A or B.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:LATitude <latitude value>
CALL:INITial:SYSB:CCCHannel:LATitude <latitude value>
<latitude value> numeric value -1296000 to 1296000

selects which base station latitude (`LAT`) to use for systems A or B.

Positive numbers represent northern latitudes.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:LONGitude <longitude value>
CALL:INITial:SYSB:CCCHannel:LONGitude <longitude value>
<longitude value> numeric value -2592000 to 2592000

selects which Base Station Longitude to use for systems A or B.

Positive numbers represent eastern longitudes.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:NID <ID value>
CALL:INITial:SYSB:CCCHannel:NID <ID value>
<ID value> numeric value 0 - 65535

selects the Network IDentification (`NID`) code for systems A or B.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:PNOFfset <PN Offset index>
CALL:INITial:SYSB:CCCHannel:PNOFfset <PN Offset index>
<PN Offset index> numeric value 0 - 511

selects the Pilot PN Sequence Offset Index for the CDMA control channel for systems A or B.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:RATE <paging rate>
CALL:INITial:SYSA:CCCHannel:RATE <paging rate>
<paging rate> FULL or HALF
where: full rate = 9600 bps; half rate = 4800 bps

selects which data rate to use on the paging channel for the CDMA control channel for systems A or B.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CCCHannel:ZID <ZID value>
CALL:INITial:SYSB:CCCHannel:ZID <ID value>
<ZID value> numeric value 0 - 4095

selects the Zone IDentification (`ZID`) code for systems A or B.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:CHANnel <channel select>
CALL:INITial:SYSB:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the channel number used for systems A or B.

If the channel is not defined (800 - 989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:DCC <code value>
CALL:INITial:SYSB:DCC <code value>
<code value> numeric value 0 - 3

selects the Digital Color Code (`DCC`) for systems A or B.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:MAC <mobile power level>
CALL:INITial:SYSB:MAC <mobile power level>
<mobile power level> numeric value 0 - 10

selects the Mobile Attenuation Code level (`MAC`) to use for control channel A or B.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:SID <SID value>
CALL:INITial:SYSB:SID <SID value>
<SID value> numeric value 0 - 32767

selects the System IDentification (`SID`) for systems A or B.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSA:TYPE <channel type>
CALL:INITial:SYSB:TYPE <channel type>

<channel type>	ANALog, CDMA
where:	ANALog = AMPS operation
	CDMA = CDMA operation

selects the type of control channel to use for systems A or B.

The CDMA selection is valid only if the CCM option is installed.

This command sets only the front panel control channel selection. It does not affect the control channel unless the user exits the Remote or Sequencer mode.

This selection is automatically activated with the next `CALL:FCC ON` command, when returning to the manual mode screen.

CALL:INITial:SYSTem <system select>
<system select> A or B

selects whether systems A or B is used.

CALL:INITial:VCHannel <voice type>
<voice type> AMPS, NAMPs, or CDMA

selects which voice channel type to use.

The CDMA selection is valid only if the CCM option is installed.

CALL:MDATa:CCCHannel:DEDicated:FORward:RCONfig?

indicates which radio configurations are supported on the forward dedicated control channel.

Response: <numeric integer>

bit 0	RC 1
bit 1	RC 2
bit 2	RC 3
bit 3	RC 4
bit 4	RC 5
bit 5	RC 6
bit 6	RC 7

bit 7	RC 8
bit 8	RC 9

Example: CALL:MDAT:CCCH:DED:FOR:RCON?

CALL:MDATa:CCCHannel:DEDicated:REVerse:RCONfig?

indicates which radio configurations are supported on the reverse dedicated control channel.

Response: <numeric integer>

bit 0	RC 1
bit 1	RC 2
bit 2	RC 3
bit 3	RC 4
bit 4	RC 5
bit 5	RC 6

Example: CALL:MDAT:CCCH:DED:REV:RCON?

CALL:MDATa:CCCHannel:DEDicated[:SUPPort]?

indicates whether the mobile supports a dedicated control channel.

Response: <boolean> 1 = supported, 0 = not supported

Example: CALL:MDAT:CCCH:DED:SUPP?
1

CALL:MDATa:CCCHannel:PVERsion?

returns the mobile's protocol version.

Perform a registration or origination on the mobile's CDMA control channel prior to this query to obtain valid information.

Response: <numeric integer>

Units: none

CALL:MDATa:CCCHannel:RTYPe?

returns the mobile's registration type.

Perform a registration on a digital control channel on the mobile prior to this query to obtain valid information.

Response: <numeric integer>

Range: 0 - 16

Numeric Integer	Registration Type
0	Timer Based
1	Power-up
2	Zone Based
3	Power down
4	Parameter Change
5	Ordered
6	Distance Based
7-15	undefined
16	Implicit (Origination)

Units: none

Example: CALL:REG; *WAI
CALL:MDAT:CCCH:RTYP?
5
indicates that an ordered registration occurred.

CALL:MDATa:CCCHannel:SCINdex?

returns the preferred slot cycle index for mobiles configured for slotted mode.

A value of “0” is returned for mobiles that are not configured for slotted mode.

Perform a registration or origination on a the mobile’s CDMA control channel prior to this query to obtain valid information.

Response: <numeric integer>

Units: none

CALL:MDATa:CCCHannel:TCALls?

returns the status of the mobile’s configuration to accept calls with the current roaming status.

This bit is set to “1” if the mobile is configured to accept calls with the current roaming status.

This bit is set to “0” if the mobile is not configured to accept calls with the current roaming status.

Perform a registration or origination on the mobile’s CDMA control channel prior to this query to obtain valid information.

Response: <numeric integer>

CALL:MDATa:ESN:FORMat <format select>
<format select> STANdard1 or STANdard2

selects whether to include the 6-bit reserved field of the ESN as part of the serial number or as a separate entity.

This only affects `CALL:MDATa:ESN[:IMMediate]` and `CALL:MDATa:ESN:SNUMber` Responses.

`STANdard1` considers them to be separate values, and `STANdard2` combines them into a single serial number value.

CALL:MDATa:ESN[:IMMediate]?

returns the Electronic Serial Number (`ESN`) of the mobile as a string of ASCII numeric characters separated into groups by ASCII spaces.

The number of groups and the value of the numbers contained in those groups depends on whether `STANdard1` or `STANdard2` format setting is chosen by `CALL:MDATa:ESN:FORMat,` and whether or not the mobile supports expanded ESN.

The response formats include:

Expanded ESN?	**Format Selection**	**Groups**	**String Length**	**String Sample**
YES	STANdard1	5	23	"MMM RR SSSSSS XXX CCCCC"
YES	STANdard2	4	22	"MMM NNNNNNNN XXX CCCCC"
NO	STANdard1	3	13	"MMM RR SSSSSS"
NO	STANdard2	2	12	"MMM NNNNNNNN"

where: MMM= 3-digit (8-bit) manufacturer's code
RR = 2-digit (6-bit) reserved field
SSSSSS = 6-digit (18-bit) serial number field
NNNNNNNN = 8-digit (24-bit) combined serial number and reserved
XXX = 3-digit (8-bit) expanded reserved field
CCCCC = 5-digit (16-bit) expanded manufacturer code

Example 1: CALL:REG; *WAI
CALL:MDAT:ESN:FORM STAN1; IMM?
"162 02 188303"
indicates a mobile that is not using expanded ESN and is displayed in STANdard1 format.

Example 2: CALL:REG; *WAI
CALL:MDAT:ESN:FORM STAN2; IMM?
"162 00712591 000 00011"
indicates a mobile that is using the expanded ESN and is displayed in STANdard2 format.

Units: none

CALL:MDATa:ESN:MCODe?

returns the mobile manufacturer's code contained within the ESN.

Perform a registration or origination on the mobile prior to this query to obtain valid information.

Response: <numeric integer>

Example: CALL:REG;*WAI?
CALL:MDAT:ESN:MCOD?
130
indicates the manufacturer's code for Motorola.

CALL:MDATa:ESN:REServed?

returns the 6-bit reserved field portion (bits 18-23) of the mobile ESN.

Response: <numeric integer>

CALL:MDATa:ESN:SNUMber?

returns the serial number portion of the mobile ESN.

If `CALL:MDATa:ESN:FORMat` is set to `STANdard1`, the serial number consists of the 18 least-significant bits (bits 0-17) of the 32-bit ESN value.

If `CALL:MDATa:ESN:FORMat` is set to `STANdard2`, it consists of the 24 least significant bits (bits 0-23).

The extra six bits added in `STANdard2` are defined as a serial number extension reserved for the FCC.

Response: <numeric integer>

The differences are as follows:

Standard Format	Number of Groups	String Length	Response
STANdard1	3	13	"MMM RR SSSSSS"
STANdard2	2	12	"MMM NNNNNNNN"
MMM	= 8-bit manufacturer's code field(bits 31-24)		
RR	= 6-bit reserved field (bits 23-18)		
SSSSSS	= 18-bit serial number field (bits 17-0)		
NNNNNNNN	= 24-bit reserved plus serial number field(bits 23-0)		

Response: <13 character string>

Example: CALL:MDAT:ESN:FORM STAN1
CALL:MDAT:ESN?
"162 02 188303"
188303
CALL:REG; *WAI
CALL:MDAT:ESN:FORM STAN2
CALL:MDAT:ESN?
"162 00712591"
CALL:MDAT:ESN:SNUM?
0071259

CALL:MDATa:FCHannel:FORWard:RCONfig?

indicates which radio configurations are supported on the forward fundamental channel.

Response: <integer value>

bit 0	RC 1
bit 1	RC 2
bit 2	RC 3
bit 3	RC 4
bit 4	RC 5
bit 5	RC 6
bit 6	RC 7
bit 7	RC 8
bit 8	RC 9

Example: CALL:MDAT:FCH:FOR:RCON?
4

CALL:MDATa:FCHannel:REVerse:RCONfig?

indicates which radio configurations are supported on the reverse fundamental channel.

Response: <integer value>

bit 0	RC 1
bit 1	RC 2
bit 2	RC 3
bit 3	RC 4
bit 4	RC 5
bit 5	RC 6

Example: CALL:MDAT:FCH:REV:RCON?
3

CALL:MDATa:FCHannel[:SUPPort]?

indicates whether the mobile supports a fundamental channel.

Response: <boolean> 1 - supported; 0 = not supported

Example: CALL:MDAT:FCH:SUPP?

CALL:MDATa:MANufacturer?

returns the mobile's station class mark.

Perform a registration or origination on the mobile prior to this query to obtain valid information.

Response: <35 character string> the manufacturer's name corresponding to the manufacturer code value (MCODe) of the mobile's ESN.

Example: CALL:REG;*WAI?
CALL:MDAT:ESN:MAN?
"*Motorola*"
indicates the manufacturer's name corresponding to the manufacturer's code (in this example, Motorola).

CALL:MDATa:MIN?

returns the mobile's mobile identification number (`MIN`).

Perform a registration or origination on the mobile prior to this query to obtain valid information.

Response: <13 or 15 character string or a 15 digit number in IMSI format, if MCC and MNC are reported by the mobile>
10 digit telephone number in the US standard format

Example 1: CALL:REG;*WAI?
CALL:MDAT:MIN?
"*(618) 623-9404*"

Example 2: CALL:REG;*WAI?
CALL:MDAT:MIN?
"*000016186239404*"
indicates that this mobile number is for a mobile that does not report the MCC and MNC information.

NOTE: The mobile number is shown in 15 digit IMSI format, indicating that the mobile did send the MCC and MNC information.

The MCC for this mobile is 000 and the MNC is 01.

CALL:MDATa:ORIGinate:EMERgency?

indicates whether a mobile placed an E-9-1-1 call.

Return <boolean> 1 = emergency call; 0 = non-emergency call

Example: CALL:MDAT:ORIG:EMER?
0

CALL:MDATa:PCLass[:CELLular]?

returns the power class for the cellular band of the mobile.

Perform a registration or origination on the mobile prior to this query to obtain valid information.

Response: numeric integer value or -1 if the mobile is not cellular band-capable.

CALL:MDATa:PCLass:PCS?

returns the power class for the PCS band of the mobile.

Perform a registration or origination on the mobile prior to this query to obtain valid information.

Response: numeric integer value or -1 if the mobile is not PCS band-capable.

CALL:MDATa:QPCHannel[:SUPPort]?

indicates whether the mobile supports a quick paging channel.

Return <boolean> 1 - supported; 2 = not supported

Example: CALL:MDAT:QPCH:SUPP?

CALL:MDATa:SCM?

returns the mobile's station class mark (`SCM`).

Perform a registration or origination on the mobile prior to this query, to obtain valid information.

Response: <numeric integer>

Example: CALL:REG;*WAI?
CALL:MDAT:SCM?
3
indicates that the mobile's station class mark is 3.

CALL:MDATa:RCONfig:ENHanced[:SUPPort]?

indicates whether the mobile supports enhanced radio configurations (RC values > 2).

Response: <boolean>1 - supported; 2 = not supported

Example: CALL:MDAT:SOUR?

CALL:MDATa:SOURce?

the response returns an enumerated value indicating the mobile message that last updated the mobile information.

Response: ORIG, PAGE, REG

Example: CALL:MDAT:SOUR?
REG

CALL:MDATa:SRATe3x[:SUPPort]?

indicates whether the mobile supports spreading rate 3 common channels.

Response: <boolean> 1 - supported; 2 = not supported

Example: CALL:MDAT:SRAT:SUPP?

CALL:MDATa:STD?

the response string indicates that the protocols, received during a registration, are supported by the mobile.

Perform a registration or origination on the mobile prior to this query, to obtain valid information.

Response: <20 character string> (the protocol standard)

Example: CALL:REG; *WAI
CALL:MDAT:STD?
IS-95

CALL:MDATa:TDIVersity:ORTHogonal[:SUPPort]?

indicates whether the mobile supports orthogonal transmit diversity.

Response: <boolean>1 = supported; 0 = not supported

Example: CALL:MDAT:TDIV:ORTH:SUPP?
1

CALL:MDATa:TDIVersity:STSPreading[:SUPPort]?

indicates whether the mobile supports space time spreading transmit diversity.

Response: <boolean>1 = supported; 0 = not supported

Example: CALL:MDAT:TDIV:STSP:SUPP?
1

CALL:MDATa:TYPE:CELLular:AMPS? <boolean>

returns an indication of the AMPS capabilities of the mobile.

Perform an registration or origination on the mobile prior to this query, to obtain valid information.

Response: 1 = the mobile is AMPS-capable in the cellular (800 MHz) band.

0 = the mobile is not AMPS-capable in the cellular (800 MHz) band.

CALL:MDATa:TYPE:CELLular:CDMA?

returns an indication of the CDMA capabilities of the mobile.

Perform an ordered registration or origination on the mobile prior to this query, to obtain valid information.

Response: 1 = the mobile is CDMA-capable in the cellular (800 MHz) band.

0 = the mobile is not CDMA-capable in the cellular (800 MHz) band.

CALL:MDATa:TYPE:CELLular:NAMPs?

returns an indication of the NAMPS capabilities of the mobile.

Perform an ordered registration or origination on the mobile prior to this query, to obtain valid information.

Response: 1 = the mobile is NAMPS-capable in the cellular (800 MHz) band.

0 = the mobile is not NAMPS-capable in the cellular (800 MHz) band.

CALL:MDATa:TYPE:CELLular:TDMA?

returns an indication of the TDMA capabilities of the mobile.

Perform an ordered registration or origination on the mobile prior to this query, to obtain valid information.

Response: 1 = the mobile is TDMA-capable in the cellular (800 MHz) band.

0 = the mobile is not TDMA-capable in the cellular (800 MHz) band.

CALL:MDATa:TYPE [:IMMediate]?

returns the mobile type indicating the type(s) of voice/ traffic channels the mobile is capable of using.

Perform an ordered registration or origination on the mobile prior to this query to obtain valid information.

Response: <character data> AMPS, NAMP, TDMA, CDMA.

Example: CALL:REG; *WAI
CALL:MDATA:TYPE?
CDMA
indicates that the mobile is capable of using CDMA traffic channels.

CALL:MDATa:TYPE:PCS:AMPS?

returns an indication of the AMPS capabilities of the mobile.

Perform an ordered registration or origination on the mobile prior to this query to obtain valid information.

Response: 1 = the mobile is AMPS-capable in the PCS (1900 MHz) band.

0 = the mobile is not AMPS-capable in the PCS (1900 MHz) band

CALL:MDATa:TYPE:PCS:CDMA?

returns an indication of the CDMA capabilities of the mobile.

Perform an ordered registration or origination on the mobile prior to this query to obtain valid information.

Response: 1 = the mobile is CDMA-capable in the PCS (1900 MHz) band.

0 = the mobile is not CDMA-capable in the PCS (1900 MHz) band

CALL:MDATa:TYPE:PCS:NAMPs?

returns an indication of the `NAMPs` capabilities of the mobile.

Perform an ordered registration or origination on the mobile prior to this query to obtain valid information.

Response: 1 = the mobile is NAMPs-capable in the PCS (1900 MHz) band.

0 = the mobile is not NAMPs-capable in the PCS (1900 MHz) band

CALL:MDATa:TYPE:PCS:TDMA?

returns an indication of the `TDMA` capabilities of the mobile.

Perform an ordered registration or origination on the mobile prior to this query to obtain valid information.

Response: 1 = the mobile is TDMA-capable in the PCS (1900 MHz) band.

0 = the mobile is not TDMA-capable in the PCS (1900 MHz) band

CALL:MRELease

tells the test equipment that a mobile release ST duration measurement is about to be made.

This overlapped command requires no exclusive resources and does not conflict with any other overlapped commands.

See `STAT:OPER:COMP`.

Press the proper key on the handset to perform the mobile release. This sets the operation complete flag for a `RELease` command (used by `*WAI`).

Follow this command with an ST duration measurement query, to return the measurement.

When the call is terminated, the 4300 returns to the forward control channel parameters specified by `CALL:CONTrol`.

The unit must be up on a call when this command is executed, or an execution error is indicated.

This command cannot be queried.

CALL:ORIGination:DIALed?

returns the called party number sent by the mobile during an origination process.

This number corresponds to the digits dialed on the mobile's keypad when a call is made.

Response: <16 character string> (the digits dialed as sent by the mobile)

Requirements: Unit must have successfully completed an origination for the information to be valid.

Example: CALL:ORIG;*WAI
CALL:ORIG:DIAL?
623-9404

indicates that the mobile reported *623-9404* as the digits dialed to place the call.

CALL:ORIGination[:IMMediate] <voice type>
<voice type> AMPS, NAMPs, or CDMA (optional)

This overlapped command uses the Call Processing resources to set the unit up from the initial system data for the selected type of call to be originated from the mobile.

See `STAT:OPER:COMP`.

If voice type parameter is omitted, the type is specified by `CALL: INITial:VCHannel`.is used.

This command cannot be queried.

The CDMA selection is valid only if the CCM option is installed.

CALL:PAGE[:IMMediate] <voice type>
<voice type> AMPS, NAMPs, or CDMA (optional)

pages the mobile to the selected voice channel using the parameters set in the initial system.

When the mobile receives the page, press the proper key on the handset to answer the call.

Either an Alert or an Alert With Info message is sent to the mobile when it is on a voice or traffic channel.

Use the parameters in `CALL:ALERt:XXX`.

This overlapped command requires an enable code of 0004 hexidecimal and uses the *Call Processing* resources.

See `STAT:OPER:COMP`.

If voice type parameter is omitted, the type is specified by `CALL: INITial:VCHannel`.

This command cannot be queried.

The CDMA selection is valid only if the CCM option is installed.

CALL:PAGE:MIN1 <lower MIN value>
<lower MIN value> numeric integer value 0 - 9999999

sets the lower 7 digits of the Mobile Identification Number (MIN) of the mobile to be paged using `CALL:PAGE[:IMMediate]`.

These digits are designated as `MIN1`.

CALL:PAGE:MIN2 <upper MIN value>
<upper MIN value> numeric integer value 0 - 999

sets the upper 3 digits of the Mobile Identification Number (MIN) of the mobile to be paged using `CALL:PAGE[:IMMediate]`.

These digits are designated as `MIN2`.

CALL:PCONtrol:ONCE <direction>, <bits>
<direction> UP or DOWN
<bits> numeric value 1 - 100

causes the 4300 to insert a one-time burst of up or down power control correction bits into the alternating pattern it is normally sending.

Each power control bit corresponds to ~ 1 dB of mobile power change.

Using this command causes a corresponding shift in mobile power measurements.

The mobile's behavior is more predictable in loopback service options than in voice service options.

This command is only allowed if the power control mode is set to alternating or active and currently on a CDMA traffic channel.

See `CALL:PCONtrol:SELect`.

If the power control mode is not set to alternating or active and currently on a CDMA traffic channel, a -221,"Settings Conflict" error is issued, with a return code value of –603.

CALL:PCONtrol:RESet[:IMMediate]

causes the 4300 to perform a one-time correction to the power control by successively measuring the mobile power and issuing the appropriate number of up or down power control correction bits to adjust the mobile power to be within ± 2 dB of the current expected power.

This command does not allow the next command to be executed until the reset has completed, which may take as much as 6 seconds. No correction is performed if the error is less than 2 dB prior to the reset.

This command is only allowed if the power control mode is set to alternating or active and currently on a CDMA traffic channel.

See `CALL:PCONtrol:SELect`.

If the command is not set to alternating or active and currently on a CDMA traffic channel, a -221,"Settings Conflict" error is issued, with a return code value of –603.

CALL:PCONtrol:RESet:OFFSet?

returns the adjustment error (in dB) of the last power control reset performed.

Typically, the reset brings the mobile power to within ±2 dB of the expected value, although many factors could cause this value to be larger. If the error is greater than ±10 dB, perform another reset.

CALL:PCONtrol:SELect <power control mode>
<power control mode> ACTive, ALTernating, UP, DOWN, or FDOWn

selects the operation of closed loop power control.

If set to `ACTive`, power control bits are sent to the mobile as needed to keep the mobile close to the expected output power.

If set to `ALTernating`, an alternating pattern of 0's and 1's is sent to the mobile.

If set to `UP`, only 0's are sent.

If set to `FDOWn`, only 1's are sent.

If set to `DOWN`, all 1's are issued to the mobile, the same as the `FDOWn` selection. In this mode, however, if the mobile output power drops too low (< -60 dBm) for the 4300 to detect, it occasionally sends a few 0's to increase the power (which prevents dropping the call) and returns to an all `DOWN` (1's) pattern.

If the unit does not have valid calibration data (after a software upgrade), an attempt to set the mode to `ACTive` results in a -221, "Settings Conflict" error.

CALL:PLCH:MAC <mobile power level>
<mobile power level> numeric value 0 - 10 (AMPS = 0 - 7)

sets the current mobile power to the value specified if a voice channel (AMPS, NAMPS, CDMA) is assigned.

CALL:PREPort:DELay <integer>
<integer> 0 - 31

specifies the delay between sending a power measurement report and beginning the next report period.

The delay (0 - 31) can be specified with each value representing a delay of 4 frames or 80 milliseconds.

This parameter is used for both threshold and periodic reporting.

The value specified for this parameter does not take effect until the next `CALL:PREPort[:IMMediate]` command is issued.

CALL: PREPort:FER?

returns the Frame Error Rate calculated from the number of frame errors divided by the number of total frames from the last power report received.

This value is not updated until a new report is received and `CALL:PREPort:STATus?` is issued.

Response **:**floating point value 0.00 to 100.00% with a resolution of 0.01%.

CALL: PREPort:FRAMes:ERRor?

returns the number of frame errors found in the power report.

This value will not be updated until a new report is received and `CALL:PREPort:STATus?` is issued.

Response: integer value 0 - 905

CALL:PREPort:FRAMes:TRANsmitted?

returns the number of frames included in the power report.

This value matches the value specified by `CALL:PREPort:PERiodic:FRAMes` (converted to frames) prior to the last `CALL:PREPort[:IMMediate]` command.

This value is not updated until a new report is received and `CALL:PREPort:STATus?` is issued.

Response: integer value 0 - 905

CALL:PREPort[:IMMediate]

issues the new power report settings which help the mobile determine when to send a power report.

This command is always successful (no response is expected from the mobile) and does not set the status flag used by `CALL:ERR?`.

CALL: PREPort:NPILots?

returns the number of pilots in Active Set that the mobile has reported.

This value will be 1 if on Sector 1 or Sector 2 only, and 2 if on both Sectors 1 and 2.

If external sectors are seen by the mobile, they may affect this value.

This value is not updated until a new report is received and `CALL:PREPort:STATus?` is issued.

Response :integer value 1 - 2

CALL:PREPort:PERiodic:ENABle <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables periodic reporting by the mobile.

The value specified for this parameter does not take effect until the next `CALL:PREPort[:IMMediate]` command is issued.

The delay time between periodic reports is specified by `CALL:PREPort:DELay`.

CALL:PREPort:PERiodic:FRAMes <integer>
<integer> 0 - 15

specifies the number of continuous frames for the mobile to include in a power report (each frame is 20 msec).

The actual number of frames is specified by:

frames = $5 * 2^{(\text{ReportPeriod} / 2)}$

where ReportPeriod is the value of this parameter.

The corresponding minimum and maximum frames are 5 and 905 respectively.

The value specified for this parameter does not take effect until the next `CALL:PREPort[IMMediate]` command is issued.

CALL: PREPort:PILot[*n*]:POWer?
where:*n* = the pilot selection, a value from 1 - 15

returns the equivalent Ec/Io value (in dB) of the Nth pilot reported by the mobile.

This value is not updated until a new report is received and `CALL:PREPort:STATus?` is issued.

Response: floating point value range (0.0 to –31.5 in steps of 0.5 dB).

CALL: PREPort:PILot[*n*]:STRength?
[*n*] the pilot selection, a value from 1 - 2

returns pilot strength of the Nth pilot reported by the mobile.

This value is not updated until a new report is received and `CALL:PREPort:STATus?` is issued.

Response: integer value 0 - 63

CALL: PREPort:RINDex?

returns an index value indicating the order of the power report.

When a call is first established, this value should be 0. Each time a new report is received from the mobile, the index value for the report is incremented.

A new report is not available for reading until `CALL:PREPort:STATus?` is issued.

Response: 0 - 32767

CALL: PREPort:STATus?

indicates if a new power report was received from mobile.

If a new power report was received from mobile, the report data is saved for query access.

This command uses the **OperationStatus** flag in the *Status* subsystem for determining when a new report is received.

If the flag is set when this command is issued, the flag is cleared upon completion.

Response: 0 or 1
1 indicates that a report was received.

CALL:PREPort:THReshold:ENABle <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables threshold reporting by the mobile.

Threshold reports are sent when a the number of frame errors received in a specified group of continuous frames exceeds a threshold.

The number of continuous frames defined for the group is specified by `CALL:PREPort:PERiodic:FRAMes` and the frame error threshold is specified by `CALL:PREPort:THReshold:FRAMes`.

The value specified for this parameter does not take effect until the next `CALL:PREPort[:IMMediate]` command is issued.

CALL:PREPort:THReshold:FRAMes <integer>
<integer> 1 - 31

specifies the number of frame errors to exceed in a report period before the mobile is to send a report.

The value specified for this parameter does not take effect until the next `CALL:PREPort[:IMMediate]` command is issued.

CALL:PREPort:VALid?

indicates if there is valid data in the power report structure.

The data is marked invalid when a call has been dropped and when first coming up on a call.

It is marked as valid when a report is received.

Returns: 0 or 1
1 indicates valid report data.

CALL:PSTRength[:IMMediate]

issues a pilot measurement request order (to the mobile) to retrieve a new pilot strength report, and saves it for query access.

If a report is received within 7 seconds, the data is saved in a buffer using `CALL:PSTRength:XXX`, and the operational status flag is cleared.

See the **OperationStatus** flag description.

If the report is not received in this time frame, the test is flagged as a failure.

The pass/fail status is determined using `CALL:ERR?`.

CALL:PSTRength:NPILots?

returns the number of pilots that the mobile is tracking in Active and Candidate Sets.

Response: integer value from 0 - 15

CALL:PSTRength:PILot[*n*]:KEEP?
[*n*] the pilot selection, a value from 1 - 15

returns the keep pilot indicator (0 or 1) of the *n*th pilot reported by the mobile.

Response: 0 or 1
0 indicates that the drop timer has expired for the pilot.

CALL:PSTRength:PILot[*n*]:PNCHip?
[*n*] the pilot selection, a value from 1 - 15

returns phase error in chips of the *n*th pilot reported by the mobile.

This is the PN phase (511 * PN offset) calculated using `CALL:PSTRength:PILot[N]:PNOFfset?`.

Response: integer value –31 to +32

CALL:PSTRength:PILot[*n*]:PNOFfset?
[*n*] the pilot selection, a value from 1 - 15

returns the pilot PN offset of the *n*th pilot reported by the mobile.

This is the PN phase value divided by 63 and rounded to the nearest integer.

Response: integer value from 0 - 511

CALL:PSTRength:PILot[*n*]:PNPHase?
[*n*] the pilot selection, a value from 1 - 15

returns the pilot PN phase of the *n*th pilot reported by the mobile.

This is the phase offset in chips from a PN offset of 0.

Response: integer value from 0 - 32767

CALL:PSTRength:PILot[*n*]:POWer?
[*n*] the pilot selection, a value from 1 - 15

returns the equivalent Ec/Io value (in dB) of the Nth pilot reported by the mobile.

Response: floating point value with a range 0.0 to –31.5 in steps of 0.5 dB.

CALL:PSTRength:PILot[*n*]:STRength?
[*n*] the pilot selection, a value from 1 - 15

returns pilot strength of the Nth pilot reported by the mobile.

This is the value reported by the mobile and represents steps of -0.5 dB.

Response: integer value 0 - 63

where:

0	represents 0
1	represents -0.5 dB
...	
63	represents -31.5 dB

CALL:PSTRength:RINDex?

returns an index value indicating the order of the pilot strength report.

When a call is first established, this value should be 0. Each time a new report is received from the mobile, the index value for the report is incremented.

Response: 0 - 32767

CALL:PSTRength:RPILot:KEEP?

returns the keep pilot indicator (0 or 1) of the pilot being used as the time reference for the mobile.

Response: 0 or 1
0 indicates the drop timer has expired for the pilot.

CALL:PSTRength:RPILot:PNOFfset?

returns PN offset of the pilot being used as the time reference for the mobile.

Response: integer value from 0 - 511

CALL:PSTRength:RPILot:POWer?

returns the equivalent Ec/Io value (in dB) of the pilot being used as the time reference for the mobile.

Response: floating point value range 0.0 to –31.5 in steps of 0.5 dB.

CALL:PSTRength:RPILot:STRength?

returns the pilot strength of the pilot being used as the time reference for the mobile.

This is the value reported by the mobile and represents steps of –0.5 dB.

Response: integer value 0 to 63

where:

0	represents 0
1	represents –0.5 dB
...	
63	represents –31.5 dB

CALL:PSTRength:STATus?

indicates if a new pilot strength report was received from mobile.

If a new pilot strength report was received from mobile, the report data is saved for query access.

This command uses the **OperationStatus** flag in the *Status* subsystem for determining when a new report is received.

If the flag is set when this command is issued, the flag is cleared upon completion.

Response: 0 or 1
1 indicates a report was received.

CALL:PSTRength:VALid?

indicates if there is valid data in the pilot strength report structure.

The data is marked invalid when a call has been dropped and when first coming up on a call.

It is marked valid when a report is received.

Response: 0 or 1
1 indicates valid report data.

CALL:REGistration

performs an ordered registration and saves the results internally.

After this command completes, with no errors, the data is accessible using `CALL:MDAT:XXX`.

Save the data in the print buffer using `PRIN:ITEM` and `PRIN:SEL:ITEM`.

This command cannot be queried.

CALL:SERVice:OPTion[:IMMediate]

instructs the mobile to switch to the service option specified in `CALL:SERVice:OPTion:NUMBer`, if a CDMA channel is assigned.

The unit must be on a CDMA traffic channel.

CALL:SERVice:OPTion:NUMBer <service option>

<service option>	numeric integer value
where:	1 = voice mode using 8k rate
	2 = loopback mode using 8k rate
	3 = voice mode using 8k EVRC
	9 = loopback mode using 13k rate
	32768 = voice using 13k rate

selects the Service Option to be used on the CDMA traffic channel when `CALL:SERVice:OPTion[:IMMediate]` is executed.

This allows changing the service option when up on a traffic channel.

If the mobile does not support the specified service option, the previous service option is in effect.

To verify if the service option was accepted after sending this command, perform a query of its parameter.

Example: CALL:SERV:OPT:NUMB 2; IMM; NUMB?
2
CALL:SERV:OPT:NUMB 32768; IMM; NUMB?
32768
CALL:SERV:OPT:NUMB 3; IMM; NUMB?

32768
indicates that the service option was changed while up on a CDMA call.

NOTE: In each command, the desired service option number is specified, followed by issuing CALL:SERV:OPT:IMM to issue the service option request to the mobile, followed by a query to verify the new service option.

The first two requests of service option 2 and 32768 were accepted, verified by the response messages.

The last request of service option 3 (EVRC) was rejected by the mobile, as verified by the response returning the previous service option value.

NOTE: Some mobiles will reject service option changes that involve changing from voice to loopback or from loopback to voice. If this occurs, establish a new call by paging to the desired service option.

CALL:SERVice:RATE <select rate>
<select rate> FULL, HALF, QUARter, or EIGHth

selects the data rate to be used while on a loopback service option (i.e. service option 2 or 9).

CALL:SRVMode:BAND <band select>
<band select> CELLular, PCS, or KORean

selects the `BAND` for the Service Mode.

The PCS or KORean band selection is allowed only if the PCS or KORean option is installed.

CALL:SRVMode:CHANnel <channel select>
<channel select> numeric value 1 - 1999

selects the voice channel number for service mode.

If the channel is not defined (800-989, 1024 - 1999 for cellular), the command is not executed and an error is generated.

CALL:SRVMode:ESN:MCODe <mfr code>
<mfr code> integer value 0 - 225 (0xFF)

specifies which 8-bit manufacturer's code to use when entering the service mode.

This value is initialized to 0 on power up and overwritten by the ESN manufacturer's code value on a registration or origination attempt. The new setting takes affect after the next CALL:SRVM:IMM command.

See CALL:SRVM:IMM.

Do not include this parameter in stored settings.

Example: CALL:SRVM:ESN:MCOD 179

CALL:SRVMode:ESN:SNUMber <serial number>
<serial number> integer value 0 - 16777215 (0xFFFFFF)

specifies which 24-bit ESN serial number and reserved field to use when entering the service mode.

This value is initialized to 0 on power up and overwritten by the ESN reserved field value on a registration or origination attempt. The new setting takes affect after the next CALL:SRVM:IMM command.

See CALL:SRVM:IMM.

Do not include this parameter in stored settings.

Example: CALL:SRVM:ESN:MCOD 123; SNUM 12345678
CALL:SRVM:IMM

CALL:SRVMode:FOFFset <frame offset select>
<frame offset select>numeric value 0 - 15

selects the Frame Offset of the Traffic channel for service mode.

CALL:SRVMode[:IMMediate]

places the 4300 in service mode, using the parameters specified by the various Service Mode Parameter Setup commands.

This allows certain measurements to be performed without establishing a traffic or voice channel.

The RF hardware of the 4300 is set as if it is up on a call and some of the measurement commands are permitted. It is assumed that the mobile is set to the same RF settings (i.e. band, channel, channel type, etc.) so that the measurements can be performed.

If the `CALL:SRVMode:TYPE` is set to CDMA, only the pilot, sync and paging channels are transmitted; a forward traffic channel is not.

CALL:SRVMode:MAC <mobile power level>
<mobile power level>numeric value 0 - 10(AMPS = 0 - 7)

selects the MAC level for AMPS and NAMPS service mode.

CALL:SRVMode:OFFSet <channel offset>
<channel offset> MID, LOW, or HIGH

selects the channel offset to use for the NAMPS service mode.

CALL:SRVMode:PNOFfset <pn offset>
<pn offset> numeric value from 0 - 511

selects the PN Offset value to use when in service mode and `CALL:SRVMode:TYPE` is set to CDMA.

CALL:SRVMode:SAT <SAT select>
<SAT select> numeric value (AMPS 0 - 2, NAMPS 0 - 6, TDMA 0 - 255)

This command selects the SAT or DSAT vector code for service mode,
where <SAT select> represents:

SAT frequency vector for AMPS

where:0 = 5970 Hz
1 = 6000 Hz
2 = 6030 Hz

DSAT vector for NAMPs

where:
0= 2556CB
1 = 255B2B
2 = 256A9B
3 = 25AD4D
4 = 26AB2B
5 = 26B2AD
6 = 2969AB

CALL:SRVMode:SCONfig:RCONfig[:FORWard] <integer>
<integer> integer value 1-5

selects which forward radio configuration to use in CDMA service mode.

This command returns a -241, "Hardware missing error" if the IS-2000 option is not installed.

Example: CALL:SRVM:SCON:RCON:FORW 4
CALL:SRVM:IMM

CALL:SRVMode:SCONfig:RCONfig:REVerse?

returns an integer value indicating which reverse radio configuration to use the next time CDMA service mode is selected.

This command returns a -241, "Hardware missing error" if the IS-2000 option is not installed.

Returns: <integer value> 1-4

Example: CALL:SRVM:SCON:RCON:FORW 3; rev?
3

CALL:SRVMode:TYPE <voice type>
<voice type> AMPS, NAMPs, or CDMA

selects the voice channel type for service mode.

CALL:SSD:AKEY:CHECksum

adjusts the last 6-digits in the `AKEY` string (the checksum digits) for a valid checksum.

This does not guarantee that the `AKEY` number is valid or that it matches the mobile.

This is useful when the `AKEY` of the mobile (except for the checksum digits) is known or when the `AKEY` value has been set to 0 and the mobile has not yet been assigned an `AKEY`.

CALL:SSD:AKEY[:DATA] <auth key string>
<auth key string> string of 26 digits (0-9) enclosed in double quotes

selects the `AKEY` value used in the SSD Update test which matches the `AKEY` value in the mobile being updated.

If an `AKEY` has not been assigned to the mobile, enter the default `AKEY` value string "00000000000000000000000000" and send the `CALL:SSD:AKEY:CHECksum` command, to calculate the checksum digits (the last 6 digits of the string).

CALL:SSD:AUTHbs?

returns the `AUTHBS` value calculated by the 4300 (from the `RANDBS` value sent by the mobile) for the last SSD Update test performed.

Response: <18-bit numeric integer> with a value from 0 - 262143.

CALL:SSD[:IMMediate]?

performs a Shared Secret Data (SSD) Update test which requires that a valid `AKEY` value has been entered into the 4300 (using `CALL:SSD:AKEY[:DATA]`) and that it matches the `AKEY` value in the mobile being updated.

This update procedure consists of first verifying the `AKEY` checksum digits. If the `AKEY` checksum is valid, the 4300 sends the SSD Update Order (including the `RANDSSD` value) to the mobile, and then waits for the Base Station Challenge Order message sent by the mobile.

The 4300 then uses the `RANDBS` value returned by the mobile message to calculate the `AUTHBS` value, and passes it back to the mobile with the Base Station Challenge Order Confirmation message.

The mobile determines if the `AUTHBS` value is valid, and updates its SSD value if it is.

It returns an SSD Update Order Confirmation message containing the pass/fail status. This pass/fail status is returned as a response message.

The mobile must be up on an AMPS or CDMA call or on a CDMA control channel for this command to work.

If an error occurs (the mobile not responding to the SSD Update Order) an error is placed in the error queue, and no response is returned.

Response: 1= PASS, 0 = FAIL

CALL:SSD:RANDbs?

returns the `RANDBS` value received from the mobile during the last SSD Update test.

Since this value is a 32-bit value and some remote controllers may have difficulty with an unsigned 32-bit value, it is returned as an ASCII string with 10 decimal digits enclosed in double quotes.

Response: <10-digit decimal numeric string>

Example: CALL:SSD:RAND?
"1298283512"

CALL:SSD:RANDssd:AUTO <auto select>
<auto select> ONCE, OFF, or ON

enables or disables auto-selection of the `RANDSSD` value used in the SSD Update test.

If `ONCE` is selected, a new `RANDSSD` value is chosen immediately and remains in effect until this command is again executed, or until `CALL:SSD:RANDssd[:DATA]` changes the value.

The `OFF` selection does not change the `RANDSSD` value, and disables the auto selection each time an SSD Update is performed.

If `ON` is selected, no change is made to the `RANDSSD` value; a new value is selected whenever `CALL:SSD[:IMMediate]?` begins a new SSD Update test.

CALL:SSD:RANDssd[:DATA] <random number string>
<random number string>string of 17 decimal digits (0-9) enclosed in double quotes

selects the `RANDSSD` value to use in the SSD Update test.

`CALL:SSD:RANDssd:AUTO` must be `OFF` to prevent the auto selection from overwriting this value.

CALL:SSD:SSDA <shared data string>
<shared data string> string of 20 decimal digits enclosed in double quotes

or string of 16 hexadecimal digits preceded by the ‘H’ or ‘h’ character and entire string enclosed in double quotes.

allows direct access to modify the SSDA value.

Example: CALL:SSD:SSDA “12345678901234567890”
CALL:SSD:SSDA “HAB54A98CEB1F0AD2”
sets the SSDA to the same value, since 12345678901234567890 decimal is equal to AB54A98CEB1F0AD2 hexadecimal.

CALL:SSD:SSDA? [format]
[format] (optional) DECimal or HEXadecimal

reports back the current SSDA value.

The SSDA value is set automatically when an SSD update is performed using `CALL:SSD[:IMMediate]?` or when set manually using `CALL:SSD:SSDA`.

The optional parameter specifies whether the response is formatted as a 20-digit decimal value or a 16-digit hexadecimal value.

The default is decimal format.

Response: <20-digit decimal or 16-digit hexadecimal string>

Example: CALL:SSD:SSDA "12345678901234567890"
CALL:SSD:SSDA?
"12345678901234567890"
CALL:SSD:SSDA? HEX
"AB54A98CEB1F0AD2"

CALL:SSD:SSDB? [format]
[format] (optional) DECimal or HEXadecimal

reports back the current SSDB value.

The SSDB value is set automatically when an SSD update is performed using `CALL:SSD[:IMMediate]?`.

The optional parameter specifies whether the response is formatted as a 20-digit decimal value or a 16-digit hexadecimal value.

The default is decimal format.

Response: <20-digit decimal or 16-digit hexadecimal string>

Example: CALL:SSD:SSDB?
"11111111111111111111"
CALL:SSD:SSDB? HEX
"9A3298AFB5AC71C7"

CALL[:STATus]:BAND?

returns the `BAND` selection for the voice or control channel currently in use.

Response: <character data>CELL, PCS, or KORean

Example: CALL:BAND?
PCS
CALL:HOFF:BAND CELLULAR; IMM
CALL:BAND?
CELL

CALL[:STATus]:CHANnel?

returns the channel number through which the instrument and the mobile are communicating.

Response: <numeric integer>

Units: none

Example: CALL:CHAN?
330
indicates that the current channel is 330.

CALL[:STATus]:FOFFset?

returns the frame offset of the CDMA traffic channel used to maintain communication with the mobile.

Frame offsets are 1.25 ms each; the maximum frame offset is 15 (18.75 ms).

Response: <numeric integer>

Range: 0 - 15

Units: 1.25 ms intervals

CALL[:STATis]:MAC?

returns the Mobile Attenuation Code level (MAC) at which the mobile is transmitting while on an analog control or voice channel.

Response: <numeric integer>

Channel Type	Response Code	Range of Values
control (A or B)	CMAC	0-7
voice (AMPS or NAMPs)	VMAC	0-7

Units: none

Example: CALL:MAC?
0
indicates that the current MAC level = 0 (highest mobile transmitter power).

CALL[:STATus]:MEM?

returns an indication of whether Message Encryption Mode (MEM) is on or off.

An on is represented by a value of 1, and an off is represented by the value 0.

Response: <numeric integer 0 or 1>

Units: none

Example: CALL:MEM?
0
indicates that the message encryption mode is off.

The 4300 does not enable message encryption mode.

CALL[:STATus]:OFFSet?

returns the NAMPs channel offset through which the instrument and the mobile are communicating.

Response: <character data> MID, LOW, or HIGH

Example: CALL:PAGE NAMPs;*WAI?
CALL:OFFS?
LOW
indicates that the current NAMPs channel offset is LOW.

CALL[:STATus]:PM?

returns an indication of whether privacy mode (PM) is on or off.

An on is represented by a value of 1, and an off is represented by the value 0.

Response: <numeric integer 0 or 1>

Units: none

Example: CALL:PAGE TDMA; *WAI
CALL:PM?
0
indicates that the privacy mode is off.

The 4300 does not enable privacy mode.

CALL[:STATus]:PNOFFset?

returns the PN Offset of the Traffic Channel being used to maintain communication with the mobile while on a CDMA control or traffic channel.

Response: <numeric integer>

Range: none

Units: 0 - 511

CALL[:STATus]:PREVision?

returns an integer value indicating the current protocol used for communicating with the mobile.

Response: numeric integer

PREV_IN_USE	**Description**
1	IS-95 (cellular band), JSTD-008 (PCS band)
2	IS-95A
3	TSB-74
4	IS-95B
5	IS-95B+
6	IS-2000 rev 0
7	IS-2000 rev A

Example**:** CALL:PREV?
7
indicates that the current protocol for the mobile is the new IS-2000.

CALL[:STATus]:RCONfig:FORWard?

indicates the current forward radio configuration selection.

This query has no meaning if the mobile is not on a CDMA call or in service mode.

This command returns a -241, "Hardware missing" error, is the IS-2000 option is not installed.

Response: <numeric integer> 1-5

Units: none

Example: CALL:STAT:RCON:REV?

CALL[:STATus]:RCONfig:REVerse?

indicates the current forward radio configuration selection.

This query has no meaning if the mobile is not on a CDMA call or in service mode.

This command returns a -241, "Hardware missing" error, is the IS-2000 option is not installed.

Response: <numeric integer> 1-4

Example: CALL:STAT:RCON:REV?

CALL[:STATus]:SAT?

returns the SAT, DSAT or DCC being used to maintain communication with the mobile, while on analog control or voice channel.

Response: <numeric integer>

Channel Type	Response Code	Range of Values
Control (A or B)	DCC	0-3
AMPS	SAT	0-2 where: 0 = 5970 Hz 1 = 6000 Hz 2 = 6030 Hz
NAMPS	DSAT	0-6

Units: none

Example: CALL:SAT?
2
indicates that the SAT frequency is 6030 Hz.

CALL[:STATus]:SID?

returns the System IDentification (SID) used on a control channel.

This query has no meaning if the current type is not a control channel (system A or B).

Response: <numeric integer>

Units: none

Example: CALL:SID?
2000
indicates that the current system identification code is 2000.

CALL[:STATus]:SOPTion?

returns the Service Option used on the CDMA traffic channel.

The unit must be on a CDMA traffic channel.

Response: <numeric integer>

Range: 0 - 65535

Service options supported:

where:1 = voice mode using 8k rate
2 = loopback mode using 8k rate

3 = voice mode using 8k EVRC
9 = loopback mode using 13k rate
32768 = voice using 13k rate

CALL[:STATus]:TCODe?

returns the CDMA Traffic Channel Walsh Code used to maintain communication with the mobile.

Response: <numeric integer>

Units: none

Range: 2 - 31
33 - 63

Walsh Code 0 is reserved for the pilot channel.
Walsh Code 1 is reserved for the paging channel.
Walsh Code 32 is reserved for the sync channel.

CALL[:STATus]:TYPE?

returns the type of channel through which the instrument and the mobile are communicating.

The channel type is either a control channel (system A or B), a CDMA control channel (system A or B), an AMPS or NAMPS voice channel, or a CDMA traffic channel.

Response: <character data> A, B, CCCA, CCCB, AMPS, NAMPs, CDMA

A	analog control channel on system A
B	analog control channel on system B
CCCA	CDMA control channel on system A
CCCB	CDMA control channel on system B
AMPS	AMPS voice channel (Advanced Mobile System)
NAMPS	NAMPS (narrow-AMPS) voice channel
CDMA	CDMA traffic channel (Code-Domain, Multiple-Access)

Example: CALL:TYPE?
A
indicates that the current channel type is a control channel on system A.

CALL:UNIQue:AUTHu:BASE?

returns the AUTHU value calculated by the 4300 for the last Unique Challenge test performed.

Response: <18-bit numeric integer> (value from 0 - 262143)

CALL:UNIQue:AUTHu[:MOBile]?

returns the `AUTHU` value sent back from the mobile for the last Unique Challenge test performed.

Response: <18-bit numeric integer> (value from 0 - 262143)

CALL:UNIQue[:IMMediate]?

performs the Unique Challenge test, which consists of the 4300 sending the Unique Challenge Order (including the `RANDU` value) to the mobile.

The 4300 waits for the Unique Challenge Order Confirmation message response from the mobile, and reads the `AUTHU` value sent with the message.

The `AUTHU` value is compared to the value calculated by the 4300 using the Cellular Authentication and Voice Encryption (CAVE) algorithm.

The pass/fail status is returned as a response message.

The mobile must be up on an AMPS or CDMA call, or on a CDMA control channel for this command to work.

If an error occurs (the mobile does not respond to the Unique Challenge Order) an error is placed in the error queue, and no response is returned.

Response: 1 = PASS, 0 = FAIL

CALL:UNIQue:RANDu:AUTO <auto select>
<auto select> ONCE, OFF, or ON

decides whether or not to allow the 4300 to automatically pick a random number when the Unique Challenge test is performed.

If `ONCE` is selected, a new `RANDU` value is chosen immediately. which remains in effect until this command is again executed, or until `CALL:UNIQue:RANDu[:DATA]` changes the value.

The `OFF` selection does not change the `RANDU` value; it disables the auto selection each time a Unique Challenge is performed.

If `ON` is selected, no change is made to the current `RANDU` value; a new value is selected whenever `CALL:UNIQue[:IMMediate]?` begins a new Unique Challenge test.

CALL:UNIQue:RANDu[:DATA] <random number>
<random number> numeric integer value from 0 - 16777215

selects the `RANDU` value to use in the Unique Challenge test.

CAUTION: `CALL:UNIQue:RANDu:AUTO` must be `OFF` to prevent the auto selection from overwriting this value.

CALL:VOICe:DELay <voice delay>
<voice delay> numeric value 0 - 5, resolution 0.001 seconds

selects the delay interval when the `LOOPback` voice mode is selected.

See `CALL:VOICe:TYPE`.

CALL:VOICe:TYPE <voice type>
<voice type> LOOPback, NORMal, RECeiver, or SILent

selects the voice mode to be used while on a voice service option.

where:

NORMal	The user speaks into the handset microphone and the transmitted audio is heard on the 4300's speaker.
LOOPback	The user speaks into the handset microphone and hear his speech retransmitted to the phone with a delay. The delay is selectable. See CALL:VOICE:DELAY.
RECeiver	The 4300 transmits a pre-stored speech sequence to the phone, thereby testing the mobile's receive audio path.
SILent	No audio processing is performed by the unit.

Only `LOOPback` mode is allowed.

CONFigure Subsystem

The *Configure* subsystem commands perform any setup conditions required to make measurements.

CONFigure:CDOMain:POWer:ACTive <threshold>
<threshold> floating point value -30.0 to 0.0

selects the code channel power measurement for the active/inactive channel classificationt.

If the threshold is set to a value ≤ the inactive channel power limit, this limit is disabled in service mode. See `CALC:LIM:CDOM:POW:INAC`.

Requirement: Must be in service mode.

Unit: dB (default = -23.0 dB)

CONFigure:DEViation <detect select>
<detect select> PEAK, POSitive, NEGative, or RMS

selects the method of measurement to use when taking a deviation measurement.

Example: CONF:DEV POS
MEAS:DEV:WBAN?
1111
configures the unit to use the positive peak wideband method, and measures the wideband deviation.

DIAGnostic Subsystem

The *Diagnostic* subsystem commands verify the test equipment is operating properly.

DIAGnostic[:ALL]?

performs an overall diagnostic test of the unit.

This command is the same as the `*TST?` command defined by IEEE-488.2.

The response consists of the bit values of the tests that failed.

A zero indicates that all tests passed.

The bit values for the boards are the same as for the `STATus:QUES:BOARd` register, which is set when this command is executed.

The board bit values are:

bit 5	(dec 32)	CCM board
bit 4	(dec 16)	transceiver board
bit 3	(dec 8)	audio measurement board
bit 1	(dec 2)	AMPS board
bit 0	(dec 1)	`SCP board`

Response: <numeric integer>

Example: DIAG?
33
indicates that the CCM and SCP boards failed self-test.

DIAGnostic:BOARd? <board select>
<board select> SCP, AMPS, CCM, AMEas, or XCVR

performs a self-test on the specified board only.

The bit values for the boards is the same as for the `STATus:QUES:BOARd` register, which is also set when this command is executed.

CCM

bit 13	(dec 8192)	CCM DSP Health Check failure
bit 12	(dec 4096)	CCM DSP RAM failure
bit 11	(dec 2048)	CCM MIC RAM failure
bit 8	(dec 256)	CCM DSP Check Sum failure
bit 7	(dec 128)	CCM MIC Check Sum failure
bit 6	(dec 64)	CCM is not responding to commands
bit 0	(dec 1)	generic failure

SCP		
bit 4	(dec 16)	GPIB port failures
bit 1	(dec 2)	static RAM memory failure
bit 0	(dec 1)	generic failure
All other cards		
bit 0		set for a failure, and cleared otherwise

Response: <numeric integer>

Example: DIAG:BOAR? CCM
64
indicates that the CCM is not responding to commands.

DIAGnostic:OPTion[:DATA]?

returns a list of all options installed in the system.

The list of options is comma-separated and includes:

String Entry	Option Description
"FEX"	PCS band
"Kor_PCS"	Korean PCS band only
"Dual_PCS"	US and Korean PCS band
"CCM-CDMA"	Old or new CCM board for CDMA option
"IS-2000"	New CCM board that uses the CSM5000 chip and supports CDMA2000

Response: <ASCII string data>

Example: DIAG:OPT?
"CCM-CDMA, IS-2000"
indicates that the CCM card is installed for CDMA operation.

DIAGnostic:OPTion:STATus?

returns the value of the sum of all bitmap system options.

Response: <numeric integer (decimal value)

Value	Short Name	Description
1	"DCM-TDMA"	TDMA support board is installed
2	"IS-136"	Digital control channel option is enabled for TDMA unit
4	"CUSTOM-136"	Custom-136 control channel option - allows user to specify control channel information - is enabled for TDMA unit

8	"FEX"	PCS option is installed
16	"GPIB"	GPIB communication is supported - always available in 4300
32	"CCM-CDMA"	CDMA support board is installed
64	"VSELP"	VSELP audio processing option is installed for TDMA unit
128	"ACELP"	ACELP audio processing option is installed for TDMA unit
256	"HI_AUDIO"	Hi output audio measurement board is installed - always available in 4300
512	"HBAND_HO"	CDMA hyperband handoffs support is enabled - always available in 4300
1024	"Dual_PCS"	unit is capable of operating in both the US PCS and Korean PCS bands
2048	"Alt_PCS"	unit is capable of operating in the Korean PCS band, but not the US PCS band

Example: DIAG:OPT:STAT?
32
indicates that the CCM card is installed for CDMA operation.

DISK Subsystem

The *Disk* subsystem commands access the floppy disk to perform specific disk-related activities including: formatting the disk, reading a file, writing to a file, deleting a file, modifying a filename, and listing a directory of the disk.

DISK:DELete <filename>
<filename> string containing the filename to delete enclosed in double quotes

deletes the specified file from the disk directory.

If the file does not exist, error -256 “File not found” is issued.

DISK:DIRectory

produces a directory listing output of the disk currently inserted in the disk drive.

It may require several seconds to build up the response string.

The response returned is ASCII text enclosed in double quotes, and contains the filenames of each file found on the disk, along with the size of the file in bytes, the date stamp, and the time stamp.

Filename entries are separated by newline characters, and the data for each file is separated by a comma.

DISK:DIRectory? <extension filter>
<extension filter> ALL, AGR, BIN, CAL, HDR, LIM, LOG, PAG, or PRG

produces a directory listing output of the disk currently inserted in the disk drive.

The extension filter value placed after the command indicates which type of files to list on the directory.

If `ALL` is used, the command produces a directory of all files found on the disk.

This is the same response as when no extension filter is used.

For any of the other selections, only the files having the specified extension are included in the directory listing.

Extension	File type
AGR	the output of an Autograph test when log-to-disk is enabled
BIN	the executable program for the 4300 (CDMA-cnt.bin, etc.)
CAL	a calibration table file
HDR	a header file
LIM	a limit table that has been saved to disk
LOG	the output of a sequence when log-to-disk is enabled.

Extension	File type
PAG	a page buffer that has been saved to disk
PRG	a program sequence file

DISK:ERRor?

returns the status of the last disk command completed.

A value of 0 indicates that the command completed with no errors.

A negative code indicates an error code and does not indicate that the last disk command has completed.

This condition is true when the Disk Command bit in the PendingStatus register is not set.

See chapter 5 for a list of error codes.

DISK:EXISts? <filename>
< filename> string containing the name of file to find enclosed in double quotes

determines whether the specified filename is contained on the installed disk.

It returns a value that indicates whether the file was found or not, or if an error occurred.

Disk errors are not tagged as SCPI errors for this command; they are returned as a negative response value.

Response: numeric integer value (-999 to +1)
0 = file found on disk
1 = file not found (no errors)
<0 = error occurred

DISK:FORMat

formats the floppy disk in the disk drive.

CAUTION: Any data on the disk is erased during a disk format.

DISK:FNAMe <file spec>
< file spec> file name specification enclosed in double quotes

finds a unique file name (one that does not exist on the disk) that has the filename characteristics given by the file specification.

The file specification is a filename with selected characters of the filename replaced by special characters to indicate that these characters may be replaced by any of a set of characters.

This permits a filename format to be specified; the 4300 returns the next filename having that pattern, which does not currently exist on the disk.

The special characters and their representations are as follows:

#	represents a character that can be replaced by a single numeric character (0-9)
@	represents a character that can be replaced by a single alpha character (a-z)
?	represents a character that can be replaced by a single alphanumeric character (a-z, 0-9, dash or underscore)

The specified file format must contain the same number of characters as the filename.

Response: <ASCII string, maximum length 12 characters>

Example:DISK: FNAM? "file ###.pag"
"file 004.pag"
indicates that files named "file 000.pag", "file 001.pag" and "file 002.pag" already exist on the disk.

DISK:LOAD:HEADer <filename>
<filename> string containing name of header file enclosed in double quotes

loads the specified header file from the disk into internal memory.

This file consists of up to 10 lines of ASCII text that are included at the top of log files, and can be inserted into the page buffer using `PRINt:HEADer1-10` commands.

Header filenames have an .hdr extension.

DISK:LOAD:LIMit <limit table select>, <filename>
<limit table select>MFC1, MFC2, MFC3, MFC4, or MFC5
<filename> string containing name of file enclosed in double quotes

loads the specified limit table file from the disk into internal memory.

The limit table is loaded into the custom limit table specified by the first parameter.

Limit table filenames have a.lim extension.

DISK:LOAD:PAGE <filename>
<filename> string containing name of page buffer file enclosed in double quotes

loads the specified page buffer file from the disk into internal memory.

This file consists of up to 60 lines of ASCII text (up to 160 characters per line) that can include several standard control characters for producing special fonts (boldface, underscore, expanded and compressed).

The page buffer can be modified with the *Print* subsystem commands and output to the printer using `PRINt:DUMP`.

Page buffer filenames have a .pag extension.

DISK:LOAD:PROGram <filename>
<filename> string containing name of program file enclosed in double quotes

loads the specified program sequence file from the disk into internal memory.

If the command is issued from the remote port, the program is placed in the (non-volatile) location specified by `PROGram[:SELected]:NAME` (`AUTO`, `QUICK`, or `CUSTOM`).

If `CUSTOM` is selected, the custom sequence location is specified by `PROGram:CUStom:NUMBer`.

These parameters must be set up prior to issuing `DISK:LOAD:PROGram`.

If the command is issued from the sequencer (that is, if a sequence program is currently running and this command is executed from within the program), the specified program file is loaded into a temporary location that can only be accessed by the sequencer using the `RUNPROG` program directive command to nest programs.

This allows one program to pass control to another, and return to the original program following the `RUNPROG` directive when the nested program completes.

Only one level of nesting is allowed, therefore an -284, "Program currently running" error is flagged, if this command is executed while running a nested program.

CAUTION: The temporary location for the nested program is not saved when the power to the test equipment is removed.

DISK:REName <current filename>, <new filename>
<current filename> string containing current filename enclosed in double

quotes
<new filename> string containing new filename enclosed in double quotes

renames one filename to another.

The filenames must include an extension that identifies the file type, and the extensions for both filenames must be the same.

See `DISK:DIR` for a list of file extensions.

DISK:STORe:AGRaph

stores the last autograph log buffer to disk after the autograph test has completed.

This is useful if the log-to-disk feature was not enabled when the autograph was run.

The filename to store to is generated internally by taking the first 7 digits of the last recorded ESN from a mobile (using the format defined by `CALL:MDATa:ESN:FORMat`) plus a single character A-Z, chosen so as to not overwrite a previous log file with the same ESN already existing on the disk.

If no unique filename can be created in this manner (26 autograph files with the same ESN code), no file is generated and an error code is returned.

Autograph log filenames have an .agr extension.

DISK:STORe:LIMit <limit table select>, <filename>
<limit table select> EIA, MFC1, MFC2, MFC3, MFC4,or MFC5
<filename> string containing name of file to save to enclosed in double quotes

stores the specified limit table data into the specified file.

The limit table filenames have a .lim extension.

CAUTION: If the filename already exists when this command is executed, it is overwritten.

DISK:STORe:LOG

stores the last program sequence log buffer (up to 500 lines) to disk after the sequence has completed.

This is useful if the log-to-disk feature was not enabled when the sequence was run, and only the last 500 test results are required to be reported.

The filename to store to is generated internally by taking the first 7 digits of the last recorded ESN from a mobile (using the format defined by `CALL:MDATa:ESN:FORMat`) plus a single character A-Z, chosen so as to not overwrite a previous log file with the same ESN already existing on the disk.

If no unique filename can be created in this manner (26 log files with the same ESN code), no file is generated and an error code is returned.

Log filenames have a .log extension.

DISK:STORe:PAGE <filename>, [start line], [end line]
<filename> string containing name of file to save to enclosed in double quotes
<start line> (optional) numeric value 1 - 60
<end line> (optional) numeric value 1 - 60

stores the current page buffer text into the specified file.

This buffer consists of 60 lines of ASCII text (up to 160 characters per line) that includes several standard control characters for producing special fonts (boldface, underscore, expanded and compressed).

An optional start and end line number of the buffer can be specified to only copy a portion of the page buffer to the disk. If these parameters are omitted, the entire 60 lines are copied. If only one numeric parameter is given, only that one line will be copied.

If both parameters are given, all lines beginning with the start line up to, and including, the end line will be copied. The start line must be less than or equal to the end line. If the filename already exists when this command is executed, it is appended to the end of the file.

DISK:STORe:PROGram <program select>, <filename>
<program select> AUTO, QUICk, CUSTom
<filename> string containing name of file to save enclosed in double quotes

stores the specified program sequence into the specified file.

If `CUSTOM` is selected, the custom sequence location (1-3) stored is specified by `PROG:CUST:NUMB`.

Program sequence filenames have a .prg extension.

CAUTION: If the selected filename already exists when this command is executed, it is overwritten.

DISPlay Subsystem

DISPlay:CONTrast <contrast>
<contrast> numeric value 0 - 31

adjusts the contrast level of the LCD display.

DISPlay:INVerse <boolean>
<boolean> OFF, ON (0= OFF, 1= ON)

enables or disables the inverse video of the LCD Display.

DISPlay:RATio [:IMMediate] <mode>
<mode> RECeive or TRANsmit

selects whether Rx or Tx measurements are displayed on the AMPS Summary and AMPS RF Tests LCD screens during front panel operation.

`RECeive` selects the Rx (receiver tests) and `TRANsmit` selects the Tx (transmitter) tests.

DISPlay:RATio:RECeiver <type select>
<type select> SNDRatio or DISTortion

selects which type of audio measurement is displayed on the AMPS Summary and AMPS RF tests screens (on the LCD) during front panel operation.

When `DISPlay: RATio [IMMediate]` is set to `RECeive`, `SNDRatio` selects Rx SINAD measurements and `DISTortion` selects Rx distortion measurements.

DISPlay:RATio:TRANsmitter <type select>
<type select> SNDRatio or DISTortion

selects which type of audio measurement is displayed on the AMPS Summary and AMPS RF tests screens (on the LCD) during front panel operation.

When `DISPlay :RATio [IMMediate]` is set to `TRANsmit`, `SNDRatio` selects Tx SINAD measurements and `DISTortion` selects Tx distortion measurements.

DISPlay:UNITs:FREQuency:RF <unit select>
<unit select> HZ or KHZ

selects which units to use when displaying the RF frequency measurements (on the LCD) during front panel operation.

This does not set the units returned on Remote measurements.

Response: <character data> HZ or KHZ

DISPlay:UNITs:PHASe <unit select>
<unit select> DEGRee or RADian

selects the units to be used when displaying phase measurements on the LCD during front panel operation.

This does not set the units returned on Remote measurements.

Response: <character data> DEGR or RAD

DISPlay:UNITs:POWer:TRANsmitter <unit select>
<unit select> WATT, DBW, or DBM

selects which units to use when displaying the transmitter power measurements (on the LCD) during front panel operation.

When `WATT` is selected, the unit automatically selects between Watts (W) or milliWatts (mW) for transmitter power measurements.

This does not set the units returned on Remote measurements.

Response: <character data> WATT, DBW, or DBM

Example: DBW
indicates that the current power unit selection is dBW.

IEEE Common Commands

*CLS

resets the status reporting structure.

See *Status* subsystem commands.

When the `*CLS` command is executed:

- the Status Byte Register (`*STB?`) is cleared,
- the Event Status Register (`*ESR?`) is cleared,
- all `STAT:OPER and STAT:QUES EVENT` registers are cleared,
- all `STAT:OPER:COMP` registers are cleared, and
- the error queue is cleared.

*ESE <enable bits>
<enable bits> numeric integer value from 0 - 255

sets the mask for the Event Status Register to produce an ESR summary bit that is set in the Status Byte Register.

See `*ESR?` and `*STB?`.

This register is bitwise ANDed to the ESR register to produce the summary bit.

This command can be queried (`*ESE?`), and responds with the current mask value set (decimal format).

*ESR?

reports back the 8-bit Event Status Register defined by IEEE 488.2.

Four of the bits are used to indicate the occurrence of a specific type of error, while the remaining bits indicate other miscellaneous events.

The **Power On** bit is only set at power up, and is used to determine if the power has been recycled.

The **User Request** bit is set when the Exit Remote front panel key is used to exit remote mode.

The **Command Error** bit is set when a command error occurs (SCPI error code = -100 to -199).

The **Execution Error** bit is set when an execution error occurs for a command (SCPI error code = -200 to -299).

The **Device Dependent Error** bit is set when a device dependent error occurs for a command (SCPI error = -300 to -399).

The **Query Error** bit is set when a query error occurs for a command (SCPI error code = -400 to -499).

The **Request Control** bit is not used.

The **Operation Complete** bit is set when any overlapped command that has its corresponding `STAT:OPER:COMP:ENAB` bit set has completed.

bit 7:	(dec 128)	Power On
bit 6:	(dec 64)	User Request
bit 5:	(dec 32)	Command Error
bit 4:	(dec 16)	Execution Error
bit 3:	(dec 8)	Device Dependent Error
bit 2:	(dec 4)	Query Error
bit 1:	(dec 2)	Request Control
bit 0:	(dec 1)	Operation Complete

Response: <numeric integer (decimal value)>

Example: *ESR?
17
indicates that an execution error has occurred and an overlapped command has reported completion.

*IDN?

returns an ASCII string response that identifies the test equipment.

There are four response string fields, separated by a comma.

The first field identifies the test equipment manufacturer.

The second field indicates the model number of the test equipment.

The third field indicates the serial number of the test equipment.

The fourth field indicates the software revision installed in the test equipment's SCP microprocessor board.

Response: <ASCII string>

Example: *IDN?
"WVTK, 4300,12345678,2.4"
indicates that the test equipment is a Wavetek 4300 with SCP software
revision 2.4 installed and that the serial number of the test equipment is 12345678.

*OPC

enables the Operation Complete bit in the Event Status Register when all pending commands have completed.

See `*ESR?`.

It is also used for sending a service request signal to the GPIB controller when overlapped commands have completed.

The service request signal is not available for serial remote mode.

Prior to sending this command, setup the `STATus:OPERation:COMPlete:ENABle` register with the enable bits set for the overlapped commands desired, to signal their completion.

For the service request to be sent upon completion, the Operation Complete (bit 0) of the Event Status Enable register (`*ESE`) must be set and the Event Status Register Summary (bit 5) and My Summary Status (bit 6) must both be set in the Status Register Enable (`*SRE`).

*OPC?

is similar to `*OPC` and `*WAI`, but does not require the service request signal, which allows it to be used in GPIB and serial remote modes.

Setup the `STATus:OPERation:COMPlete:ENABle` register with the enable bits set for the overlapped commands desired to signal their completion.

Neither the `*ESE` or the `*SRE` registers need to be setup.

This command does not complete until the selected overlapped commands have signaled completion. The response message, an ASCII '1', is returned when this command completes.

The Message Available bit is set in the Status Byte Register when the message is available for reading.

Response: <numeric integer> (always a 1)

Example: STAT:OPER:COMP:ENAB 4
CALL:PAGE
*OPC?
1
indicates that the CALL:REG command has completed.

*RCL <setting index>
<setting index> numeric integer 0 - 9

sets the 4300 parameters to those defined in the specified stored setting location.

The list of parameters that are modified is the same as for `*RST`.

If the stored setting location is empty or has invalid data in it, the current parameters do not change and the appropriate error is placed in the error queue.

*RST

initializes many 4300 parameters to their default values, in order to prevent inconsistent operation (due to uninitialized parameters) between remote sessions.

Use this command if a remote program performed various CDMA measurements (including BER) but the BER measurement interval was not setup beforehand.

CAUTION: Set any parameters that affect the operation of a program, once at the start of each session, as the program may not behave consistently if parameters are modified between uses.

Sending `*RST` is the most effective method of initializing the parameters of current and future commands.

The parameters initialized with this command are the same as those that are saved and recalled using `*SAV` and `*RCL`.

The list of remote command parameters set by `*RST`, along with their default values include:*

SCPI Command	Default Value
CALCulate:LIMit:TYPE[:LOAD]:AUTO	OFF
CALCulate:LIMit:TYPE[:LOAD][:SELect]	EIA
CALL:ALERt:CNI1:NPLan	0
CALL:ALERt:CNI1:NTYPe	0
CALL:ALERt:CNI1[:NUMBer]	"8008511198"
CALL:ALERt:CNI1:PI	0
CALL:ALERt:CNI1:SI	0
CALL:ALERt:CNI2:NPLan	0
CALL:ALERt:CNI2:NTYPe	0
CALL:ALERt:CNI2[:NUMBer]	"8008511198"
CALL:ALERt:CNI2:PI	0
CALL:ALERt:CNI2:SI	0
CALL:ALERt:INFormation:AMPS	OFF
CALL:ALERt:SELect	NONE, SIGNal, CNI1, CNI2
CALL:ALERt:SIGNal:CADence	1
CALL:ALERt:SIGNal:PITCh	MEDium
CALL:CONTrol:BAND	CELLular
CALL:CONTrol:CCCHannel:ACCess:PROBe: POWer:INITial	0
CALL:CONTrol:CCCHannel:ACCess:PROBe: POWer:NOMinal	0
CALL:CONTrol:CCCHannel:ACCess:PROBe: POWer:STEP	0
CALL:CONTrol:CCCHannel:ACCess:PROBe: PREamble	1
CALL:CONTrol:CCCHannel:ACCess:PROBe: REQuests	15
CALL:CONTrol:CCCHannel:ACCess:PROBe: RESPonses	15
CALL:CONTrol:CCCHannel:ACCess:PROBe: STEPs	4

SCPI Command	Default Value
CALL:CONTrol:CCCHannel:ACCess:PROBe: TIMeout	1.360
CALL:CONTrol:CCCHannel:BID	0
CALL:CONTrol:CCCHannel:LATitude	517920
CALL:CONTrol:CCCHannel:LONGitude	–1239360
CALL:CONTrol:CCCHannel:NID	0
CALL:CONTrol:CCCHannel:PNOFfset	0
CALL:CONTrol:CCCHannel:RATE	FULL
CALL:CONTrol:CCCHannel:REGistration: TIMer	OFF
CALL:CONTrol:CCCHannel:SEARch:CRITeria: ADD	–14.0
CALL:CONTrol:CCCHannel:SEARch:CRITeria: COMParison	2.5
CALL:CONTrol:CCCHannel:SEARch:CRITeria: DROP	–16.0
CALL:CONTrol:CCCHannel:SEARch:CRITeria: TIMer	3
CALL:CONTrol:CCCHannel:SEARch:WINdow: ACTive	8
CALL:CONTrol:CCCHannel:SEARch:WINdow: AGE	0
CALL:CONTrol:CCCHannel:SEARch:WINdow: NEIGhbor	8
CALL:CONTrol:CCCHannel:SEARch:WINdow: REMaining	8
CALL:CONTrol:CCCHannel:ZID	0
CALL:CONTrol:CHANnel	330
CALL:CONTrol:DCC	0
CALL:CONTrol:MAC	2
CALL:CONTrol:SID	19
CALL:CONTrol:SYSTem	A
CALL:CONTrol:TYPE	ANALog
CALL:HFLash:TRANsmit:CNI1:NPLan	0
CALL:HFLash:TRANsmit:CNI1:NTYPe	0
CALL:HFLash:TRANsmit:CNI1[:NUMBer]	"8008511198"
CALL:HFLash:TRANsmit:CNI1:PI	0
CALL:HFLash:TRANsmit:CNI1:SI	0
CALL:HFLash:TRANsmit:CNI2:NPLan	0
CALL:HFLash:TRANsmit:CNI2:NTYPe	0
CALL:HFLash:TRANsmit:CNI2[:NUMBer]	"8008511198"

SCPI Command	Default Value
CALL:HFLash:TRANsmit:CNI2:PI	0
CALL:HFLash:TRANsmit:CNI2:SI	0
CALL:HFLash:TRANsmit:MWAIting	1
CALL:HFLash:TRANsmit:SELect	NONE, SIGNal, MWI, CNI1, CNI2
CALL:HFLash:TRANsmit:SIGNal:CADence	1
CALL:HFLash:TRANsmit:SIGNal:PITCh	MEDium
CALL:HOFF:ATIMe:ENABle	OFF
CALL:HOFF:ATIMe[:IMMediate]	0.08
CALL:HOFF:BAND	CELLular
CALL:HOFF:CHANnel	100
CALL:HOFF:FOFFset	0
CALL:HOFF:MAC	2
CALL:HOFF:NOMinal	0
CALL:HOFF:OFFSet	MID
CALL:HOFF:SAT	0
CALL:HOFF:TCODe	8
CALL:HOFF:TYPE	AMPS
CALL:INITial:AMPS:BAND	CELLular
CALL:INITial:AMPS:CHANnel	330
CALL:INITial:AMPS:MAC	2
CALL:INITial:AMPS:SAT	0
CALL:INITial:CDMA:BAND	CELLular
CALL:INITial:CDMA:CHANnel	283
CALL:INITial:CDMA:PNOFfset	0
CALL:INITial:CDMA:TCODe	8
CALL:INITial:CDMA:FOFFset	0
CALL:INITial:CDMA:SOFTion	2
CALL:INITial:NAMPs:BAND	CELLular
CALL:INITial:NAMPs:CHANnel	330
CALL:INITial:NAMPs:DSAT	2
CALL:INITial:NAMPs:MAC	2
CALL:INITial:NAMPs:OFFSet	MID
CALL:INITial:SYSA:BAND	CELLular
CALL:INITial:SYSA:CCCHannel:BID	0
CALL:INITial:SYSA:CCCHannel:LATitude	571920
CALL:INITial:SYSA:CCCHannel:LONGitude	–1239360
CALL:INITial:SYSA:CCCHannel:NID	0
CALL:INITial:SYSA:CCCHannel:PNOFfset	0

SCPI Command	Default Value
CALL:INITial:SYSA:CCCHannel:RATE	FULL
CALL:INITial:SYSA:CCCHannel:ZID	0
CALL:INITial:SYSA:CHANnel	330
CALL:INITial:SYSA:DCC	0
CALL:INITial:SYSA:MAC	2
CALL:INITial:SYSA:SID	19
CALL:INITial:SYSA:TYPE	ANALog
CALL:INITial:SYSB:BAND	CELLular
CALL:INITial:SYSB:CCCHannel:BID	0
CALL:INITial:SYSB:CCCHannel:LATitude	571920
CALL:INITial:SYSB:CCCHannel:LONGitude	-1239360
CALL:INITial:SYSB:CCCHannel:NID	0
CALL:INITial:SYSB:CCCHannel:PNOFfset	0
CALL:INITial:SYSB:CCCHannel:RATE	FULL
CALL:INITial:SYSB:CCCHannel:ZID	0
CALL:INITial:SYSB:CHANnel	334
CALL:INITial:SYSB:DCC	0
CALL:INITial:SYSB:MAC	2
CALL:INITial:SYSB:SID	80
CALL:INITial:SYSB:TYPE	ANALog
CALL:INITial:SYSTem	A
CALL:INITial:VCHannel	AMPS
CALL:PCONTrol:SELect	ACTive
CALL:SERVice:OPTion:NUMBer	2
CALL:SERVice:RATE	FULL
CALL:SRVMode:BAND	CELLular
CALL:SRVMode:CHANnel	100
CALL:SRVMode:FOFFset	0
CALL:SRVMode:MAC	2
CALL:SRVMode:OFFSet	MID
CALL:SRVMode:SAT	0
CALL:SRVMode:TCODE	8
CALL:SRVMode:TYPE	AMPS
CALL:SSD:RANDssd:AUTO	OFF
CALL:SSD:RANDssd[:DATA]	"00000000000000000"
CALL:UNIQue:RANDu:AUTO	OFF
CALL:UNIQue:RANDu[:DATA]	0
CALL:VOICe:TYPE	LOOPback
CALL:VOICe:DELay	1.0 sec

SCPI Command	Default Value
CONFigure:DEViation	PEAK
INPut:AUDio:CMESsage	ON
INPut:AUDio:COMPressor	OFF
INPut:AUDio:DEViation	1000 Hz
INPut:AUDio[:SOURce]	RECeive
INPut:CMESsage	ON
INPut:EXPandor	OFF
INPut:FILTer	Bandpass, 300, 3000
INPut:RF	DIRect
MEASure:BER:ANALog:SAMPles	100
MEASure:FER:CONFidence	95.0
MEASure:FER:FRAMes[:MAXimum]	10000
MEASure:FER:LIMit:ENABle	ON
MEASure:FER:LIMit[:MAXimum]	0.5
MEASure:FREQency:UNITs	Hz
MEASure:POWer:TRANsmitter:GATed:COUNt	100
MEASure:POWer:TRANsmitter:MAXimum:SOURce:POWer	-104.0
MEASure:POWer:TRANsmitter:MINimum:SOURce:POWeR	-25.0
MEASure:POWer:TRANsmitter:OPEN:STEP	20.0
MEASure:POWer:TRANsmitter:UNITs	WATT
MEASure:VERRor:PHASe:UNITs	DEGRee
OUTPut:AOUT	SOURce
SOURce:AUDio:DEViation	8000 Hz
SOURce:AUDio:FREQuency	1000 Hz
SOURce:AUDio[:SOURce]:EXTernal	OFF
SOURce:AUDio[:SOURce]:INTernal	ON
SOURce:AWGN[:LEVel]	0.0
SOURce:AWGN:STATe	OFF
SOURce:CDMA:PAGing[:LEVel]	-12.0
SOURce:CDMA:PILot1[:LEVel]	-7.0
SOURce:CDMA:PILot2[:LEVel	-7.0
SOURce:CDMA:SSECtor[:LEVel]	-3.0

SCPI Command	Default Value
SOURce:CDMA:SSECtor:STATe	OFF
SOURce:CDMA:SSECtor:TCODe	8
SOURce:CDMA:SYNC[:LEVel]	-16.0
SOURce:CDMA:TRAFfic1[:LEVel]	-7.4
SOURce:CDMA:TRAFfic2[:LEVel]	-7.4
SOURce:POWer:LEVel	-70.0 dBm
SOURce:POWer:STATe	ON

*System parameters (those that affect the 4300 and the user/controller interface) are excluded from this list.

*SAV <setting index>
<setting index> numeric integer 0 - 9

saves the current parameter values at the specified stored setting location.

A default label is automatically assigned to the location: “Setting #N” where N is the <setting index> value.

If the specified setting location contains a stored setting prior to executing this command, the original setting information is overwritten.

The stored setting data is contained in non-volatile RAM, which retains these settings when power is removed from the test equipment.

The list of parameters that are stored is the same as for `*RST`.

*SRE <enable bits>
<enable bits> numeric integer value from 0 - 255

sets the mask for the Status Byte Register (`*STB?`) to produce a My Summary Status bit which is placed in bit 6 of the Status Byte Register.

If bit 6 of the SRE register is set and the My Summary Status bit in the STB register is also set, a service request is issued (GPIB mode only).

This command can be queried (`*SRE?`), and responds with the current mask value set (decimal format).

*STB?

reports back the 8-bit Status Byte register defined by IEEE 488.2.

This command consists of summary bits from 3 other register structures (Event Status, Operational Status, and Questionable Status).

The My Summary Status bit represents a summary of the rest of the bits in this register, masked by the Status Register Enable.

See `*SRE`.

For GPIB mode, this register can be read by performing a serial poll of the instrument where bit 6 (My Summary Status) represents the service request line status.

The definitions of the Status Byte bits are:

bit 7	(dec 128)	OPERational status summary
bit 6	(dec 64)	My Summary Status
bit 5	(dec 32)	Event Status Register summary
bit 4	(dec 16)	Message Available
bit 3	(dec 8)	QUEStionable status summary
bit 2	(dec 4)	Error Queue Status
bit 1	(dec 2)	Remote Command Completed
bit 0	(dec 1)	-not used-

The **Message Available** bit is set when a query response has been generated and is ready for reading. This bit is used to determine when to read a query response after the query command has been issued. Some measurement responses may take more than 1 second to generate the response.

The **Error Queue Status** bit is set when the error queue contains at least one error. Up to 10 errors can be accumulated in the error queue; this bit uses `SYSTem:ERRor?` to determine when all of the errors have been read.

The **Remote Command Completed** bit is cleared when a remote command is being parsed and set after the terminator is received and the command has been executed or if the command generated an error.

This command is self-destructive. After the command returns the current value of the register, the register is cleared to 0.

Response: <numeric integer (decimal value)>

Example: *STB?
80
indicates that a response message is available for reading,

and the service request was issued (if GPIB remote). The service request is negated after this command is executed.

If a serial poll had been performed on the test equipment, the response would have been a value of 80 decimal, and the service request would have been negated after the serial poll had been conducted.

*TST?

performs a complete self-test and reports the result as pass/fail.

See `DIAG?`.

Response: <numeric 0 or 1>

Example: *TST?
0
indicates that the self-test diagnostics passed.

*WAI

This command is similar to `*OPC` and `*OPC?`, but does not require the service request signal and does not produce a response message that must be read.

The `STATus:OPERation:COMPlete:ENABle` register must be setup with the enable bits set for the overlapped commands to signal their completion.

Neither the `*ESE` or the `*SRE` registers need to be setup.

This command will not complete until the selected overlapped commands have signaled completion.

Example: STAT:OPER:COMP:ENAB 2
CALL:REG; *WAI
CALL:MDAT:ESN?
By placing the *WAI command after the CALL:REG command, the command has been converted from an overlapped command into a sequential command. The next command, reading the ESN data acquired by the registration, will not execute until the registration has completed.

INPut Subsystem

The *Input* subsystem commands condition the incoming signal prior to measurement.

INPut:AUDio:CMESsage <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

inserts or removes the `CMESsage` filter in the Audio In path prior to the Rx SINAD/distortion measurements.

Example: INP:AUD:CMES ON
inserts the CMESSAGE filter for SINAD measurements.

INPut:AUDio:COMPressor <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

inserts or removes the `COMPressor/PRE-emphasis` from the path between the external Audio In jack and the Tx modulation.

This has no effect unless the `INPut:AUDio` selection is `EXTERNAL`.

INPut:AUDio:DEViation <dev value>
<dev value> numeric value from 0.0 - 20000.0 Hz with a resolution of 25.0

sets the audio input deviation in units of Hz, KHz, or MHz.

The units are assumed to be Hz if not otherwise specified.

Example: INP:AUD:DEV 1500
sets the audio input deviation to 1500 Hz.

INPut:AUDio[:SOURce] <input select>
<input select> RECeive or EXTernal

selects the source of the input modulation.

`RECEIVE` is used for SINAD and other measurements.

`EXTERNAL` selects an external modulation source that is connected to the Audio In jack on the front panel.

Example: INP:AUD REC
selects audio from the mobile, allowing measurements to be made.

INPut:CMESsage <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

inserts or removes the `CMESsage` filter in the RF path prior to the Tx SINAD/distortion measurements.

INPut:EXPandor <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

inserts or removes the `EXPandor` from the receive audio measurement path for performing deviation measurements; may be cascaded with input filters.

See `INPut:FILTer`.

Example: INP:AUD:EXP ON
inserts the EXPandor.

INPut:FILTer NONE
INPut:FILTer HP <hi-pass freq>
INPut:FILTer LP <lo-pass freq>
INPut:FILTer BP <hi-pass freq>, <lo-pass freq>
<filter type>, <hi-pass freq>, <lo-pass freq>
where: <filter type>= NONE, HP, LP, BP
<hi-pass freq> = numeric value: 50, 300
<lo-pass freq>= numeric value: 3000, 15000, 30000

selects the input filters used between the receiver demodulator and the deviation meter.

The four filter type selections are:

- none,
- high pass only,
- low pass only, and
- bandpass.

There are 2 high-pass filter selections and 3 low-pass selections. The bandpass filter is a combination of any high and low pass filter selections.

The query form of this command returns all 3 parameters.

Units: Hz

Example 1: INP:FILT BP, 300, 30000
selects bandpass filtering with a 300 Hz high-pass cutoff frequency and a 30000 Hz low-pass cutoff frequency.

Example 2: INP:FILT?
LP,50,15000
indicates that the 15 kHz low-pass filter is currently selected.

INPut:RF <input select>

<input select> DIRect or OFFair

selects whether the 20 Watt 20 dB pad is inserted in the RF input line.

`DIRect` inserts the pad, allowing a direct connection to be made from the mobile transmitter to the RF input. The base transmitter power range for this selection is -50 to -125 dBm. This selection is required for making power measurements.

`OFFair` removes this pad to allow greater sensitivity. This selection is used if the RF input is connected to an antenna to receive the mobile transmitter RF signal. Base transmitter power for this selection is -30 to -105 dBm.

Power measurements cannot be made in this mode.

This selection does not affect the CDMA mode as the RF pad is automatically inserted or removed based on the power received from the mobile at the RF input.

MEASurement Subsystem

The *Measurement* subsystem commands initiate and report measurements, and perform limit checking as specified by the current limit table parameters.

See CALCulate:LIMit.

Information about the measurement (value, units, limits) may be output to the printer buffer using `PRINt:ITEM`. Many of these commands require the unit to be in a particular state in order to execute. See the **`Requirements:`** section of the command description.

Certain measurements have a corresponding `READ` subsystem command which reports the previous measurement value taken. This allows a measurement to be made once, and the data can be read or printed in different units, without taking another measurement.

MEASure:BER?

performs an analog `BER` measurement using the number of samples specified by `MEASure:BER:ANALog:SAMPles`.

The measured value is returned along with an indication of the measurement performed.

Requirements: Unit must be up on an AMPS call.

Response: <BER type>, <numeric floating point>

where:<BER type> = analog BER measurement
<numeric floating point> = percentage BER

Resolution: 0.01

Units: percentage

Limit checking: uses the values in CALC:LIM:BER:ANALog.

Example: CALL:PAGE AMPS; *WAI
MEAS:BER
ANAL,2.5
indicates that the analog BER is 2.5% (2.5% of the audit messages were not acknowledged).

MEASure:BER:ANALog:SAMPles <number of samples>
<number of samples> numeric integer value 1 - 1000

selects the number of audit messages to use when an analog BER measurement is performed using `MEASure:BER?` while the unit is in AMPS call mode.

A larger number of samples provides a more accurate analog BER measurement.

MEASure:CDOMain[:IMMediate]?

performs a code domain measurement and returns a pass/fail indication.

If this command is executed when the unit is not in RC 3-5, error -603, "Call Process Invalid" is indicated and the measurement is not performed. This is reported to the remote as error -221, "Settings conflict".

Requirements: RC > 2

Response: PASS, FAIL, NONE

Limit checking: uses the values in CALculate:LIMit:CDOMain.

MEASure:CURRent[:DC]?

performs a DC input current measurement for the mobile and reports the value in Amps.

Requirements: none

Response: <numeric floating point>

Resolution: 0.01

Units: amps

Limit checking: none

Example: MEAS:CURR?
1.300000
indicates that the DC input current to the mobile is 1.3 amps.

MEASure:DEViation:AUDio?

performs an audio deviation measurement and returns the deviation in Hz.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values in CALC:LIM:DEViation:AUDio.

MEASure:DEViation:DSAT?

performs a DSAT deviation measurement and returns the deviation in Hz.

Requirements: Unit must be up on a call or in service mode (NAMPs only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values in CALC:LIM:DEViation:NAMPs.

MEASure:DEViation:DST?

performs a DST deviation measurement and returns the deviation in Hz.

Requirements: Unit must be up on a call or in service mode (NAMPs only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values in CALC:LIM:DEViation:NAMPs.

MEASure:DEViation:PAUDio <duration>
<duration> numeric integer value 1 - 60 seconds

initiates a peak audio deviation measurement for the specified amount of time but does not return a measurement value.

Use `READ:DEViation:PAUDio?` following completion of the measurement to obtain measurement value.

This overlapped command allows other commands to be executed while the measurement is in progress; delay the `READ` command until the measurement completes.

Bit 7 (decimal value 128) of the **PendingStatus** register is set upon completion of the peak measurement.

See *Status* subsystem commands and “Overlapped and Sequential Commands” for an explanation of how the **PendingStatus** flag provides notification to the controller of when the measurement completes and is ready to be read.

This command requires the exclusive use of one of the measurement tasks in the 4300; any other measurement command that uses the same measurement task must wait until the task is available.

The following commands, when placed after `MEAS:DEV:PAUD`, execute only after the peak audio measurement is complete:

MEASure:POWer:TRANsmit[:IMMediate]?
MEASure:POWer:DC?
MEASure:CURRent[:DC]?
MEASure:VOLTage[:DC]?
MEASure:VOLTage:AC?
MEASure:SNDRatio:RECeiver?
MEASure:SNDRatio:TRANsmitter?
MEASure:DISTortion:REceiver:TRANsmitter
MEAS:DEViation:*XXXX*(all of the deviation measurements)

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

MEASure:DEViation:PAUDio? <duration>

<duration> numeric integer value 0 - 60 seconds

functions the same as the non-query form of the command except that it is non-overlapped (next command is not processed until this command completes) and it returns the peak deviation measurement value.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPS only).

This command is not valid through the remote port. It can only be used from within a program.

MEASure:DEViation:RESidual?

performs a residual deviation measurement and returns the deviation in Hz.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values in CALC:LIM:DEViation:RESidual.

MEASure:DEViation:SAT?

performs a SAT deviation measurement and returns the deviation in Hz.

Requirements: Unit must be up on a call or in service mode (AMPS only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values in CALC:LIM:DEViation:SAT.

Example: CALL:PAGE AMPS; *WAI
MEAS:DEV:SAT?
1880
indicates that the measured SAT deviation is 1880 Hz.

MEASure:DEViation:ST?

turns the signaling tone ON, performs an ST deviation measurement, and returns the deviation in Hz.

The ST remains on after this query has been executed. Pressing the SEND or SND key on the mobile handset turns the ST off. Doing so too soon may cause invalid results.

Requirements: Unit must be up on a call or in service mode (AMPS only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values in CALC:LIM:DEViation:ST.

MEASure:DEViation:WBANd?

performs a wideband deviation measurement and returns the deviation in Hz units.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric integer>

Units: Hz

Limit checking: uses the values in CALC:LIM:DEViation:WBANd.

MEASure:DISTortion:RECeiver?

performs a receiver distortion measurement.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values in CALC:LIM:DISTortion:RECeive.

Example: CALL:PAGE AMPS; *WAI
MEAS:DIST:REC?
3.2
indicates that the Rx distortion is 3.2%.

MEASure:DISTortion:TRANsmitter?

performs a transmitter distortion measurement.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values in CALC:LIM:DISTortion:TRANsmit.

MEASure:DTMF[:FULL]

initializes the DTMF key string prior to sending `MEAS:DTMF?` from the remote.

The string is initialized to 12 ASCII spaces, and the last character read is initialized to 0. The PASS/FAIL status is set to PASS when all 12 keys have been pressed.

The operation pending flag (see `Status` subsystem) for DTMF is set when this command is received, and is cleared when all 12 DTMF keys have been received.

This overlapped command uses an enable code of 0020 hexidecimal (32 decimal) and requires no exclusive resources (does not conflict with any other overlapped command).

Requirements: Unit must be up on a call (AMPS, NAMPs, CDMA), not in service mode.

MEASure:DTMF[:FULL]?

reports the value of the last DTMF key pressed (the ASCII code represented in decimal format), along with a string indicating which of the 12 DTMF keys have been pressed since the last `MEAS:DTMF` command (not query) was received.

The string is 12 characters in length and enclosed in double quotes. The string sequence is “1234567890*#”, and any keys that have not been pressed are replaced with an ASCII space.

Requirements: none

Response: <decimal integer>, "<12-character string>"

Example: 50,"123 7 * "
indicates that the last DTMF key pressed was decimal 50 (the '2' key) and that the keys pressed since last reset are: '1', '2', '3', '7', '*'(not necessarily in that order).

MEASure:DTMF:KEY

performs the same DTMF query response initialization as `MEAS:DTMF[:FULL]`, without beginning a DTMF test.

This command does not begin an overlapped command or set up the front panel screen, to allow aborting the command from the sequencer.

Use this command prior to the initial `MEAS:DTMF:KEY?` in order to prevent reading an outdated DTMF key sequence.

MEASure:DTMF:KEY?

reports the value of the first DTMF received following either the set or the query form of `MEAS:DTMF:KEY`.

If no key has been received, this command waits for the next key before allowing another remote command to be parsed. The command may be broken from this loop by one of three conditions:

- a DTMF key is pressed,
- the call is dropped, or
- a device/interface clear is received.

The return value is the integer value of the ASCII DTMF key pressed. ('0' to '9' = decimal 48 to 57, '*' = decimal 42, '#' = decimal 35).

If the call is dropped, the return value is 0.

Response: <decimal integer>

MEASure:FER:CONFidence <% confidence>
<% confidence> 90.0 - 100.0% (resolution of 0.5)

sets the confidence level used during a Frame Error Rate measurement using `MEASure:FER[:IMMediate]?`.

This command is valid only if the CCM option is installed.

MEASure:FER:FRAMes:MAXimum <number of frames>
<number of frames> numeric integer value 1 - 10000

sets the number of frames to be received by the mobile during Frame Error Rate measurements using
`MEASure:FER[:IMMediate]?`.

This command is valid only if the CCM option is installed.

MEASure:FER:[IMMediate]

This overlapped command version functions in the same way as `MEASure:FER[:IMMediate]?` with 2 exceptions: it does not return a response and, it allows other commands to run while this one is in process.

Use `READ:FER?` to read the result when this command completes. Bit 3 (decimal value 8) of the **PendingStatus** register is set upon completion of the FER measurement.

See the *Status* subsystem commands and "Overlapped and Sequential Commands" for an explanation of how the **PendingStatus** flag provides notification to the controller of when the measurement completes and a result is ready to be read.

To avoid dropping the call during this test, set the power control to All Up (`CALL:PCONtrol:SELect UP`) prior to performing the FER test at reduced base power levels.

MEASure:FER[:IMMediate]?

begins a frame error rate measurement, waits for it to complete, and returns the FER, expressed as a percentage, for the completed test.

The measurement is performed by putting the mobile up on a CDMA call in a loopback service option (2 or 9) so that the mobile is returning the same frame data that the 4300 is transmitting.

The frame data transmitted consists of a pseudo-random bit stream. A frame is considered an error if it is a valid frame but does not match the frame data that was transmitted.

The user defines the duration of the test by specifying the maximum number of frames transmitted and the confidence factor. The test completes no later than when the maximum number of frames has been reached.

This value is set from 1 to 10000, each frame taking 20 msec to complete. The maximum duration for this test is 200 sec.

A confidence factor of 100% forces the test to run to the maximum number of frames. Specifying a confidence factor of <100% uses statistical probability to determine that the FER does not exceed the maximum FER limit.

If the number of frame errors is small, this can result in the test terminating before reaching the maximum number of frames.

To avoid dropping the call during this test, set the power control to All Up (`CALL:PCONtrol:SELect UP`) prior to performing the FER test at reduced base power levels.

Requirements: Unit must be up on a CDMA call (loopback service option 2 or 9).

Response: <numeric floating point>

Resolution: 0.0001

Units: percentage

Limit checking: uses the value in MEAS:FER:LIMit if test does not pass with confidence.

Example:

MEAS:FER:LIM 0.3; CONF 100.0; FRAM:MAX 5000
MEAS:FER?
0.0230

MEASure:FER:LIMit:ENABle <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the Frame Error Rate Maximum limit set by `MEASure:FER:LIMit[:MAXimum]`.

This command is valid only if the CCM option is installed.

MEASure:FER:LIMit[:MAXimum] <maximum frame error rate>
<maximum frame error rate> 0.0 - 5.0% with a resolution of 0.1

sets the limit value for a Frame Error Rate measurement using `MEASure:FER[:IMMediate]?`.

This command is valid only if the CCM option is installed.

MEASure:FREQuency:ERRor:RF?

performs an RF frequency measurement and calculates and returns the frequency error in the units that are currently selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

This command indicates an error if not on a call or not in service mode.

Requirements: Unit must be up on a call or in service mode (AMPS, NAMPs, CDMA).

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values in CALC:LIM:FREQuency:RF.

Example: CALL:PAGE; *WAI
MEAS:FREQ:ERR:RF?
-30000
indicates a measured transmitter frequency that is 30000 Hz below the nominal value.

MEASure:FREQuency:ERRor:SAT?

performs a SAT frequency measurement and calculates and returns the frequency error in the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values in CALC:LIM:FREQ:SAT.

Example: CALL:PAGE AMPS; *WAI
MEAS:FREQ:ERR:SAT?
-30
indicates a measured SAT frequency value that is 30 Hz lower than the nominal value.

MEASure:FREQuency:ERRor:ST?

performs an ST frequency measurement and calculates and returns the frequency measurement in the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

ST remains on after this query has been executed. Pressing the `SEND` or `SND` key on the mobile handset turns the ST off. Doing so too soon may cause invalid results.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values in CALC:LIM:FREQuency:ST.

MEASure:FREQuency:RF?

performs a mobile transmitter RF frequency measurement and returns the RF frequency in the units selected by `MEAS:FREQ:UNIT`.

See MEAS:FREQ:UNIT.

This command indicates an error if not on a call or in service mode.

Requirements: Unit must be up on a call or in service mode (AMPS, NAMPs, CDMA).

Response: <numeric floating point>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values in CALC:LIM:FREQuency:RF. The nominal value from the limit entry is offset to reflect the current channel frequency.

MEASure:FREQuency:SAT?

performs a SAT frequency measurement and returns the frequency in the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values in CALC:LIM:FREQuency:SAT. The nominal value from the limit entry is offset to reflect the current SAT selection for the channel.

MEASure:FREQuency:ST?

turns on the signaling tone (if the unit is up on a call), performs an ST frequency measurement and returns the ST frequency in the units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

The ST remains on after this query has been executed.

Pressing the `SEND`/`SND` key on the mobile handset following this query turns the ST off. Doing so too soon may cause invalid results.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric integer>

Resolution: 1.0

Units: Hz, kHz, MHz

Limit checking: uses the values in CALC:LIM:FREQuency:ST.
The nominal value from the limit entry is offset by +10000 Hz.

MEASure:FREQuency:UNITs <unit select>
<unit select> MHZ, KHZ, or HZ

selects the units to be used in reporting the frequency and frequency error measurements.

This command also selects the units that are printed when `PRIN:ITEM DATA` prints the results of the last measurement, if the last measurement command was one of the frequency (or frequency error) measurements.

Response: <character data> MHZ, KHZ, or HZ

MEASure:POWer:DC?

performs a DC input power measurement for the mobile.

Requirements: none

Response: <numeric floating point>

Resolution: 0.1

Units: Watt

Limit checking: none

Example: MEAS:POW:DC?
4.5
indicates that the DC input power to the mobile is 4.5 Watts.

MEASure:POWer:TRANsmitter:APRobe[:IMMediate] [<test type>]
<test type> TEST1, TEST2, or NORMal (default)

initiates the access probe power OPC test to test the mobile's ability to acquire the access channel.

Bit 1 of the **PendingStatus** register clears when this test completes.

See *Status* Subsystem commands.

This command indicates an error if on a CDMA call or if the CCM option is not installed.

This test may take more than one minute to complete, depending on the conditions set by `CALL: CONT:CCCH:ACC:PROB`.

Requirements: Unit must be on a CDMA control channel.

Response: PASS or FAIL: if running from a sequence
NONE:otherwise

Limit checking:
uses the values in: CALC:LIM:POW:CDMA:APR:TEST1
CALC:LIM:POW:CDMA:APR:SMALl
CALC:LIM:POW:CDMA:APR:LARGe.

MEASure:POWer:TRANsmitter:CLOSed[:IMMediate]?

measures the closed loop power response of a mobile resulting from 100 consecutive power up commands followed by 100 power down commands followed by 100 power up commands to the mobile.

The results of the test are the mobile's response to the 100 power down commands and the second set of 100 up commands.

This command indicates an error if not on a CDMA call or if the CCM option is not installed.

Requirements: Unit must be up on a CDMA call (CDMA service option 2 or 9 selected).

Response: PASS or FAILif limits are enabled
NONEif limits not enabled

Limit checking: uses the values in:
CALC:LIM:POW:CDMA:CLOS[:REL]
CALC:LIM:POW:CDMA:CLOS:PERC
CALC:LIM:POW:CDMA:CLOS:RANG:MAX
CALC:LIM:POW:CDMA:CLOS:RANG:MIN.

MEASure:POWer:TRANsmitter:CLOSed:INITial?

measures and reports the initial open loop estimate RF output power of a mobile.

See MEAS:POW:TRAN:UNIT.

This command indicates an error if not on a CDMA call or if the CCM option is not installed.

Requirements: Unit must be up on a CDMA call (CDMA service option 2 or 9 selected).

Response: <numeric floating point>

Resolution: 0.1 dB or 0.1 mW

Units: dBW, dBm, Watt, or mW

Limit checking: none

MEASure:POWer:TRANsmitter:CLOSed:MAXimum?

measures and reports the maximum output power relative to the initial power of the mobile response to the test described in `MEASure:POWer:TRANsmitter:CLOSed[:IMMediate]?`
.

This command indicates an error if not on a CDMA call or if the CCM option is not installed.

Requirements: Unit must be up on a CDMA call (CDMA service option 2 or 9 selected).

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Limit checking: uses the values in CALC:LIM:POW:CDMA:CLOS:RANG:MAX.

MEASure:POWer:TRANsmitter:CLOSed:MINimum?

measures and reports the minimum output power relative to the initial power of the mobile response to the test described in `MEASure:POWer:TRANsmitter:CLOSed[:IMMediate]?`
.

This command indicates an error if not on a CDMA call or if the CCM option is not installed.

Requirements: Unit must be up on a CDMA call (CDMA service option 2 or 9 selected).

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Limit checking: uses the values in CALC:LIM:POW:CDMA:CLOS:RANG:MIN.

MEASure:POWer:TRANsmitter:CLOSed:RATE[:FALL]?

measures and reports the fall rate of the mobile response to the test described in `MEASure:POWer:TRANsmitter:CLOSed[:IMMediate]?`.

This command indicates an error if not on a CDMA call or if the CCM option is not installed.

Requirements: Unit must be up on a CDMA call (CDMA service option 2 or 9 selected).

Response: <numeric floating point>

Resolution: 0.1

Units: dB/20ms

Limit checking: none

MEASure:POWer:TRANsmitter:CLOSed:RATE[:RISE]?

measures and reports the rise rate of the mobile response described in `MEASure:POWer:TRANsmitter:CLOSed[:IMMediate]?`.

This command indicates an error if not on a CDMA call or if the CCM option is not installed.

Requirements: Unit must be up on a CDMA call (CDMA service option 2 or 9 selected).

Response: <numeric floating point>

Resolution: 0.1

Units: dB/20ms

Limit checking: none

MEASure:POWer:TRANsmitter:GATed:AVERage?

measures and reports the average mobile transmitter RF power over the number of gated-on power control groups set in `MEAS:POW:TRAN:GAT:COUN` .

See `MEAS:POW:TRAN:UNIT.`

This command indicates an error if not on a CDMA call or in CDMA service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call with data rate set to half, quarter or eighth.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Limit checking: uses the values in CALC:LIM:POW:TRAN:CDMA:GAT:ON.

Example: CALL:PAGE CDMA; *WAI
MEAS:POW:TRANGAT:AVER?
1.3
indicates that the average mobile transmitter power is 1.3 dBW.

MEASure:POWer:TRANsmitter:GATed:COUNt <average count>
<average count> numeric integer value 1 - 200

sets the number of power control groups used while performing gated transmitter power measurements using:

```
MEASure:POWer:TRANsmitter:GATed[:IMMediate]:AVERage
MEASure:POWer:TRANsmitter:GATed[:IMMediate]:PEAK
MEASure:POWer:TRANsmitter:GATed[:IMMediate]:DROOp?.
```

A larger number of power control groups provides a more accurate gated transmitter power measurement.

This command is valid only if the CCM option is installed.

MEASure:POWer:TRANsmitter:GATed[:IMMediate]?

measures and reports the mobile transmitter RF power throughout the number of gated-on power control groups, set in `MEAS:POW:TRAN:GAT:COUN`.

This command indicates an error if not on a CDMA call or in CDMA service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call with data rate set to half, quarter or eighth for the query.

Response: PASS or FAIL: if limits are enabled.
NONE: if limits are not enabled.

Limit checking:
uses the values in:CALC:LIM:POW:TRAN:CDMA:GAT:ON
CALC:LIM:POW:TRAN:CDMA:GAT:OFF:REL
CALC:LIM:POW:TRAN:CDMA:GAT:OFF:FLO
CALC:LIM:POW:TRAN:CDMA:GAT:TIME:RISE
CALC:LIM:POW:TRAN:CDMA:GAT:TIME:FALL.

MEASure:POWer:TRANsmitter:GATed:PEAK?

measures and reports the average peak (largest) RF power over the number of gated-on power control groups set in `MEAS:POW:TRAN:GAT:COUN`.

See MEAS:POW:TRAN:UNIT.

This command indicates an error if on a CDMA call or in service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call with data rate set to half, quarter or eighth for the query to be executed.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Limit checking:
uses the values in:
CALC:LIM:POW:TRAN:CDMA:GAT:ON:LOW.

MEASure:POWer:TRANsmitter[:IMMediate]?

performs a mobile transmitter power measurement and reports the value in the units that are currently selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

If the mobile is on a CDMA call, this command returns the average power of the modulation envelope.

This command indicates an error if not on a call or in service mode.

Requirements: none (if not in AMPS, NAMPs, or CDMA mode, defaults to AMPS power measurement).

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Limit checking: analog uses the values in CALC:LIM:POWer:MAC with the MAC index value (0 - 7) determined by the current mobile power selection. For CDMA, no limit check is performed.

Example: CALL:PAGE; *WAI
MEAS:POW:TRAN?
4.200000
indicates mobile transmitter power of +4.2 dBW.

MEASure:POWer:TRANsmitter:MAXimum?

measures the mobile transmitter maximum RF output power and reports the maximum power measured, in units selected by `MEAS:POW:TRAN:UNIT`. See MEAS:POW:TRAN:UNIT.

The input power to the mobile is set to the `MEAS:POW:TRAN:MAX:SOUR:POW` level, and returned to its previous value after the report.

This command indicates an error if not on a CDMA call and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Limit checking: uses the values in
CALC:LIM:POW:TRAN:CDMA:MAX[:CELL:]CLASs 1:3
CALC:LIM:POW:TRAN:CDMA:MAX:PCS:CLASs 1:5.

Example: CALL:PAGE CDMA; *WAI
MEAS:POW:TRAN:MAX?
6.3
indicates that the maximum RF output is +6.3 dBW.

MEASure:POWer:TRANsmitter:MAXimum:SOURce:POWer <power>

<power> numeric value -125.0 to 10.0 with a resolution of 0.1 dBm

sets the power level of base power to a given value (in dBm) to use during a maximum transmitter power measurement using `MEASure:POWer:TRANsmitter:MAXimum?`.

The RF carrier level returns to its previous value after the measurement is performed.

This command is valid only if the CCM option is installed.

This command allows the user to set the output power to a higher level than the maximum output of -23.00 dBm, and then adjusts the output to the highest available power.

MEASure:POWer:TRANsmitter:MINimum?

measures the mobile transmitter minimum RF output power and reports the minimum power measured in units selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

The input power to the mobile is set to the `MEAS:POW:TRAN:MIN:SOUR:POW` level, and returned to its previous value after the report.

This command indicates an error if not on a CDMA call and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call.

Response: <numeric floating point>

Resolution: 0.1dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Limit checking: uses the values in CALC:LIM:POW:TRAN:CDMA:MIN.

Example: CALL:PAGE CDMA; *WAI
MEAS:POW:TRAN:MIN?
-50.4
indicates that the minimum RF output is -50.4 dBW.

MEASure:POWer:TRANsmitter:MINimum:SOURce:POWer <power>
<power> numeric value -125.0 to 10.0 with a resolution of 0.1 dBm

sets the power level of base power to a given value (in dBm) to use during a minimum transmitter power measurement using `MEASure:POWer:TRANsmitter:MINimum?`.

The RF carrier level returns to its previous value after the measurement is performed.

This command is valid only if the CCM option is installed.

This command allows the user to set the output power to a higher level than the maximum output of -23.00 dBm, and then adjusts the output to the highest available power.

MEASure:POWer:TRANsmitter:OPEN[:IMMediate]?

measures the open loop output power response of the mobile resulting from the input power step change set in `MEAS:POW:TRAN:OPEN:STEP`.

This command indicates an error if not on a CDMA call and the CCM option is not installed.

For best results, perform this test while using a `LOOPback` mode (service option 2 or 9) and `FULL` data rate.

Requirements: Unit must be up on a CDMA call.

Response: PASS or FAIL:if limits are enabled.
NONE: if limited are not enabled.

Limit checking:
uses the values in:CALC:LIM:POW:CDMA:OPEN:CHAN:SMAL
CALC:LIM:POW:CDMA:OPEN:CHAN:LARG
CALC:LIM:POW:CDMA:OPEN:CHAN:OFFS

MEASure:POWer:TRANsmitter:OPEN:INITial?

measures and reports the open loop estimate RF output power of a mobile.

See MEAS:POW:TRAN:UNIT.

This command indicates an error if not on a CDMA call and the CCM option is not installed.

For best results, perform this test while using a `LOOPback` mode (service option 2 or 9) and `FULL` data rate.

Requirements: Unit must be up on a CDMA call.

Response: <numeric floating point>

Units: dBW, dBm, Watt, or mW

Limit checking: none

MEASure:POWer:TRANsmitter:OPEN:STEP <delta amplitude>
<delta amplitude> -30.0 to 30.0 dB with a resolution of 0.1

sets the step size of base power during CDMA mobile open loop transmitter power measurements.

This command is valid only if the CCM option is installed.

MEASure:POWer:TRANsmitter:STANdby?

measures the mobile transmitter standby RF output power (mobile transmitter disabled) and reports the standby power measured in the units selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

This command indicates an error if on a CDMA call or in service mode, and the CCM option is not installed.

Requirements: Unit must be on a CDMA control channel.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Limit checking: uses the values in CALC:LIM:POW:TRAN:CDMA:STAN.

Example: MEAS:POW:TRAN:STAN?
-81.5
indicates that the standby RF output is -81.5 dBW.

MEASure:POWer:TRANsmitter:UNITs <unit select>
<unit select> DBW, DBM, WATT, MW, or UW

selects the units to be used in reporting the transmitter power measurement.

This command also selects the units that will be printed when `PRIN:ITEM DATA` prints the results of the last measurement, if the last measurement command was `EAS:POW:TRAN?`.

Response: <character data> DBM, DBW, WATT, MW, or UW

MEASure:SENSitivity?

performs the same receiver sensitivity measurement as `MEASure:SENSitivity? <base power level>`, with one exception: the base power level is defined by the nominal limit selection for one of the following command headers:

for AMPS CALCulate:LIMit:SENSitivity[: AMPS]
for NAMPsCALCulate:LIMit:SENSitivity:NAMPs
for CDMACALCulate:LIMit:SENSitivity:CDMA

If the `STATe` entry for the command header is set to `OFF`, the test is not performed and the response is `NONE`.

In order to minimize dropping the call during this test, set the power control to All Up (`CALL:PCONtrol:SELect UP`) prior to performing the FER test at reduced base power levels.

Requirements: Unit must be up on a call (AMPS, NAMPS, CDMA).

Response: PASS, FAIL, NONE

Limit checking:
For AMPS/NAMPs: none
For CDMA: uses the values in CALC:LIM:SENS:FER.

MEASure:SENSitivity? <base power level>
<base power level> numeric value -125.0 to -50.0 dBm

For AMPS and NAMPS:

performs a receiver sensitivity measurement by temporarily setting the base power level to the value specified and checking whether SAT is still present.

The base power is restored after the test completes. If SAT is present, the test passes, if not, it fails. No value is returned if an error occurs with the command due to the base power level specified.

Limit checking: none

For CDMA:

performs a receiver sensitivity measurement by temporarily setting the base power level to the value specified and performing a FER measurement.

The base power is restored after the test completes. If the FER measurement passes the limit checks, the test passes, if not, it fails. No value is returned if an error occurs with the command due to the base power level specified.

This command indicates an error if not on a CDMA call (loopback service option 2 or 9) and the CCM option is not installed.

In order to minimize dropping the call during this test, set the power control to All Up (`CALL:PCONtrol:SELect UP`) prior to performing the FER test at reduced base power levels.

Requirements: Unit must be up on a call (AMPS, NAMPs, CDMA).

If the unit was unable to switch to the selected power output for any reason, a -221, "Settings conflict" error is returned, and the test is not performed.

Response: PASS/FAIL

Limit checking: uses the values in CALC:LIM:SENSitivity:FER.

MEASure:SNDRatio:RECeiver?

performs a receiver SINAD measurement.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Limit checking: uses the values in CALC:LIM:SINad:RECeive.

Example: CALL:PAGE AMPS; *WAI
MEAS:SNDR:REC?
0.1
indicates a receiver SINAD of 0.01 dB.

MEASure:SNDRatio:TRANsmitter?

performs a transmitter SINAD measurement.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Limit checking: uses the values in CALC:LIM:SINad:TRANsmit.

Example: CALL:PAGE AMPS; *WAI
MEAS:SNDR:TRAN?
13.5
indicates that the transmitter SINAD is 13.5 dB.

MEASure:STATus:SAT?

indicates whether SAT is on or off: ON = 1, OFF = 0.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPs only).

Response: <numeric integer 0 or 1>

Units: none

Example: CALL:PAGE AMPS; *WAI
MEAS:STAT:SAT?
1
indicates that the SAT is ON.

MEASure:STATus:ST:CONDition?

returns an indication of whether ST is on or off: ON = 1, OFF = 0.

Requirements: Unit must be up on a call or in service mode (AMPS or NAMPS only).

Response: <numeric integer 0 or 1>

Units: none

Example: CALL:PAGE AMPS; *WAI
MEAS:STAT:ST:COND?
1
indicates that the signaling tone is ON.

MEASure:STATus:ST:DURation? <limit type>
<limit type> NONE, HANDoff, RELease, FLASh

returns the last ST duration measurement in milliseconds.

If `NONE` is selected as the limit type, no limit check is performed.

`HANDoff`, `RELease`, and `FLASh` represent `CALC:LIM:DURation` limit parameters that determine the test pass/fail status.

Requirements: Unit must be on a call or in service mode (AMPS or NAMPs only).
Response: <numeric integer>
Units: msec

Limit checking:NONE:no limit check
HANDoff:CALC:LIM:DUR:HANDoff
RELease:CALC:LIM:DUR:RELease
FLASh:CALC:LIM:DUR:FLASh

MEASure:TIME:ERRor?

measures and reports the time error estimate obtained in the computation of Rho and transmitter frequency error.

This command indicates an error if on a CDMA call or in service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call transmitting a synchronized valid burst, or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.001

Units: microsecond

Limit checking: uses the values in CALC:LIM:TIME:ERR.

MEASure:VERRor:IQ?

measures and returns the I-Q position of each of 157 symbols over one CDMA burst.

Each I-Q pair maps to one symbol in the burst from symbols 6 - 162.

This command indicates an error if not on a CDMA call or in CDMA service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call or in CDMA service mode with the mobile transmitting a valid burst.

Response: PASS or FAIL: if limits are enabled.
NONE: if limits are not enabled

MEASure:VERRor:IQIMbalance?

measures and reports the I/Q imbalance due to an amplitude difference between the input signals to the mobile transmitter I/Q modulator.

This command indicates an error if on a CDMA call or in CDMA service mode and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call transmitting a synchronized valid burst, or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: dBc

Limit checking: uses the values in CALC:LIM:VERR:IQIM.

MEASure:VERRor:MAGNitude:PEAK?

measures and returns the peak magnitude component of vector errors over one CDMA burst (symbols 6 - 162).

This command indicates an error if not on a CDMA call or in service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call transmitting a synchronized valid burst or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values in CALC:LIM:VERRor:MAGNitude:PEAK.

Example: CALL:PAGE CDMA; *WAI
MEAS:VERR:PHAS:PEAK?
16.5
indicates that the peak magnitude error is 16.5%.

MEASure:VERRor:MAGNitude:RMS?

measures and returns the RMS of the vector error magnitude over one TDMA burst (symbols 6 - 162).

This command indicates an error if not on a CDMA call or in service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call transmitting a synchronized valid burst or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values in CALC:LIM:VERRor:MAGNitude:RMS.

Example: CALL:PAGE CDMA; *WAI
MEAS:VERR:MAGN:RMS?
3.5
indicates that the RMS magnitude error is 3.5%.

MEASure:VERRor:OOFFset?

measures and returns the origin offset of a CDMA burst.

This command indicates an error if not on a call or in service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call transmitting a synchronized valid burst or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: dBc

Limit checking: uses the values in CALC:LIM:VERRor:OOFFset.

Example: CALL:PAGE CDMA; *WAI
MEAS:VERR:OOFF?
-48.7
indicates that the origin offset is -48.7 dBc.

MEASure:VERRor:PEAK?

measures and returns the peak error vector over one CDMA burst (symbols 6 - 162).

Peak vector error is also called Peak Error Vector Magnitude (EVM).

This command indicates an error if not on a CDMA call or in service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call transmitting a synchronized valid burst or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values in CALC:LIM:VERRor:PEAK.

Example: CALL:PGAE CDMA; *WAI
MEAS:VERR:PEAK?
14.6
indicates that the peak vector error is 14.6%.

MEASure:VERRor:PHASe:PEAK?

measures and returns the peak phase component of vector errors over one CDMA burst (symbols 6 - 162).

This command indicates an error if not on a CDMA call or in CDMA service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call transmitting a synchronized valid burst or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.01

Units: degrees

Limit checking: uses the values in CALC:LIM:VERRor:PHASe:PEAK.

Example: CALL:PAGE CDMA; *WAI
MEAS:VERR:PHAS:PEAK?
9.51
indicates that the peak phase error is 9.51 degrees.

MEASure:VERRor:PHASe:RMS?

measures and returns the RMS of the vector error phase over one CDMA burst (symbols 6 - 162).

This command indicates an error if not on a CDMA call or in CDMA service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA transmitting a synchronized valid burst or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.01

Units: degrees

Limit checking: uses the values in CALC:LIM:VERRor:PHASe:RMS.

Example: CALL:PAGE CDMA; *WAI
MEAS:VERR:PHAS:RMS?

2.69
indicates that the RMS vector error is 2.69 degrees.

MEASure:VERRor:PHASe:UNITs <unit select>
<unit select> DEGRee, RADian

selects the units to be used in reporting the vector error phase measurement and also selects the units that print when `PRIN:ITEM:DATA` is used, if the last measurement command was `MEAS:VERR:PHASE?`

MEASure:VERRor:RMS?

measures and returns the RMS vector error over one CDMA burst (symbols 6 - 162).

RMS vector error is also called RMS Error Vector Magnitude (EVM).

This command indicates an error if not on a call or in service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call or in CDMA service mode with the mobile transmitting a valid burst.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Limit checking: uses the values in CALC:LIM:VERRor:RMS.

Example: CALL:PAGE CDMA; *WAI
MEAS:VERR:RMS?
5.8
indicates that the RMS vector error is 5.8%.

MEASure:VOLTage:AC?

measures the AC RMS voltage on the Audio In front panel connector.

Requirements: none

Response: <numeric floating point>

Resolution: 0.001

Units: volt (RMS)

Limit checking: none

MEASure:VOLTage[:DC]?

performs a DC input voltage measurement and reports the value in volts.

Requirements: none

Response: <numeric floating point>

Resolution: 0.01

Units: volt

Limit checking: none

Example: MEAS:VOLT?
12.1
indicates that the DC input voltage is 12.1 volts.

MEASure:WQUality?

measures and reports the Waveform QUality (Rho) of the mobile transmitter.

This command indicates an error if not on a CDMA call or in service mode, and the CCM option is not installed.

Requirements: Unit must be up on a CDMA call transmitting a synchronized valid burst or in CDMA service mode.

Response: <numeric floating point>

Resolution: 0.0001

Units: none

Limit checking: uses the values in CALC:LIM:WQU.

MEMory Subsystem

The *Memory* subsystem consists of the commands that modify buffers or tables in the system memory.

MEMory:COPY:[NAME] <source table>, <destination table>
<source table> NONE, EIA, MFC1, MFC2, MFC3, MFC4, or MFC5
<destination table> MFC1, MFC2, MFC3, MFC4, or MFC5

copies the specified limit table selection to one of the custom limit tables.

The source can be any of the custom tables, the `EIA` table (fixed values defined by the `EIA` standard), or `NONE`.

The `NONE` entry sets all parameter values to the `EIA` standard, but disables the limit values.

MEMory:DATA:FLOAt <numeric integer>, < numeric floating point >
<numeric integer> value from 1-20
< floating point value>

allows the user to save a floating point value in non-volatile memory.

Response: <numeric floating point>

MEMory:DATA:FLOAt? <numeric integer>
<numeric integer> value from 1-20

allows the user to query a floating point value previously stored in non-volatile memory.

MEMory:DATA:INTeger <numeric integer>,<numeric integer>
<numeric integer> value from 1-20

allows the user to save an integer value in non-volatile memory.

MEMory:DATA:INTeger? <numeric integer>
<numeric integer> value from 1-20

allows the user to query a previously saved integer value in non-volatile memory.

MEMory:DATA:LONG <numeric integer>, <numeric integer>
<numeric integer> value from 1-20

allows the user to save a long integer value in non-volatile memory.

MEMory:DATA:LONG? <numeric integer>
<numeric integer> value from 1-20

allows the user to query a previously saved long integer value in non-volatile memory.

MEMory:DATA:SHORt <numeric integer>, <numeric integer>
<numeric integer> value from 1-20

allows the user to save a short integer value in non-volatile memory.

MEMory:DATA:SHORt? <numeric integer>
<numeric integer> value from 1-20

allows the user to query a previously saved short integer value in non-volatile memory.

MEMory:DATA:STRing <numeric integer>, <"string">
<numeric integer> value from 1-20

allows the user to save an ASCII string (of up to 31 characters) in non-volatile memory.

MEMory:DATA:STRing? <numeric integer>
<numeric integer> value from 1-20

allows the user to query a previously saved ASCII string in non-volatile memory.

MEMory:NSTates?

returns the total number of stored settings locations in the test equipment.

This value is fixed at 10.

Response: <numeric integer> (always 10)

MEMory:STATe:CATalog?

returns the list of all 10 stored setting labels, separated by commas.

Empty locations without labels use commas as place markers.

Response: <ASCII string, 160 chars maximum>

Example 1: “,,,,,,,,,”

indicates that there are no setting labels assigned.

Example 2: “,,Setting #2,,,,Setting #6,,,Setting #9”

indicates that only locations 2, 6, and 9 have settings with corresponding assigned label names.

MEMory:STATe:DEFine < setting name>, <setting index>
<setting name> ASCII string, maximum length of 15 characters, enclosed in

double quotes
<setting index> numeric integer 0 - 9

assigns a label, or name, to a stored setting location.

It performs the same function as `MEMory:STATe:REGister[n]: LABel`, except that the stored setting location to be named is passed as a second argument (following the label name) rather than as part of the command header.

MEMory:STATe:DEFine? <setting name>
<setting name> ASCII string, maximum length of 15 characters, enclosed in double quotes

performs a search for the specified name in the list of all stored settings, and returns the corresponding stored setting index value if found.

If not found, no response is generated and a -292, "Referenced name does not exist" error is placed in the error queue.

Response: <numeric integer>

MEMory:STATe:REGister[*n*]:CLEar
[*n*] 1-9

clears the specified stored setting location and its associated label.

MEMory:STATe:REGister[*n*]:CLEar?
[*n*] 1-9

returns a value that indicates whether the specified stored setting location has valid data in it or is empty.

Response: <numeric integer>

where:1 = location is empty
0 = location contains a stored setting

MEMory:STATe:REGister[*n*]:DATE?
[*n*] 1-9

returns the date that the specified stored setting location was saved.

The response is in the form of year, month, day.

Response: <numeric integer>, <numeric integer>, <numeric integer>

Example: MEM:STAT:REG5:DATE?
1995,10,16
indicates that location 5 was stored on Oct. 16, 1995.

MEMory:STATe:REGister[*n*]:LABel < name>

[*n*] 1-9

<name> ASCII string, maximum length of 15 characters, enclosed in double quotes

assigns a label, or name, to a stored setting location.

The stored setting index is the 0-9 value selected in the command. The default name that is chosen when a `*SAV` is executed is "Setting #X" where X is the number of the stored setting location.

To make it easier to identify the locations that have valid settings, do not label empty locations (no stored setting saved).

Example: MEM:STAT:REG3:LAB "Bob N. setup"
sets the label for stored setting location 3 to "Bob N. setup".

OUTPut Subsystem

The *Output* subsystem commands pertain to functions that condition the outgoing signal (from *Source* subsystem) after it has been generated.

Conditioning is a function of filtering, biasing, frequency conversion, and attenuation.

OUTPut:AOUT <output select>
<output select> SOURce, RECeive, or OFF

selects the audio out source accessed on the Audio Out jack on the 4300 front panel.

`SOURce` selects the 4300 internal audio generator to be used as a tone generator.

`RECeive` selects the audio signal received from the mobile mobile's transmitter demodulated from the RF input.

Example: OUT:AOUT REC
runs the mobile transmitter audio to the Audio Out jack.

PRINt Subsystem

The Print subsystem commands allow the user to customize the printed output file.

PRINt:BOLD <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the boldface setting.

All commands that output to the print buffer after this command is issued, print data in that typeface setting.

Exceptions:

Any measurement failure indications for `DATA, MDATa, NOMLimit, MINLimit, MAXLimit, STATus, CSTatus, or TStatus` values are bolded when a failure occurs.

PRINt:CLEAr

initializes the print buffer to all ASCII spaces prior to sending data to the print buffer.

This command cannot be queried.

PRINt[:DATA]:JUSTify <justify select>
<justify select> LEFT, RIGHt, or DECimal

positions the characters of specific numeric items to be printed.

Since the length of the data string is variable, based on its value, and not known in advance, this command is used to "straighten" columns of numeric data.

Selected `PRINt:ITEM` and `PRINt:SELected:ITEM` commands are affected:`QUERy, MQUEry, DATA, MDATa, NOMLimit, MINLimit, MAXLimit.`

The operation of the justification selections include:

`LEFT` the numeric data is left-justified at the specified column.

`RIGHt` the numeric data is right-justified at the specified column

`DECimal` the numeric data will be positioned such that the decimal point position of the data (whether one is printed or not) is placed at the specified column.

Example: For a last measured data value of 23.4567:

PRIN:SEL:LINE 20; COL 32
PRIN:JUST LEFT;
:SEL:ITEM DATA;
:LINE:INCR PRIN:JUST RIGHT;
:SEL:ITEM DATA PRIN:SEL:LINE:INCR...

PRIN:JUST DECIMAL:SEL:ITEM DATA
prints the following data (where the '|' indicates column 32 and is not part of the printout):

```
|
23.4567
23.4567
23.4567
```

PRINt[:DATA]:MFACtor <multiplication factor>
<multiplication factor> numeric value -10e6 to +10e6 with a resolution of 10e-6

selects which multiplication factor to use when printing `MDATa` and `MQUEry` selections.

This multiplication value is used for printing the measured data or queried parameter in another unit, as long as the unit conversion is a simple multiplication factor within the stated range and resolution.

This multiplied value adjusts the displayed resolution automatically; if a given data value is 12.2 with a resolution of 1 digit and the multiplication factor is 0.001, the displayed data has a resolution of 4 digits: 0.0122.

PRINt[:DATA]:PPRefix <prefix selection>
<prefix selection> NONE, PLUS, or SPACe

sets the character to prepend to positive values.

An added character affects the character location of the printed data only if `LEFT` justification is selected.

Selected `PRINt:ITEM` and `PRINt:SELected:ITEM` commands are affected:`QUERy, MQUEry, DATA, MDATa, NOMLimit, MINLimit, MAXLimit.`

Example: For a last measured data value of 23.4567:

```
PRIN:SEL:LINE 20; COL 32
PRIN:JUST LEFT
PRIN:PPREFIX NONE;
:SEL:ITEM DATA
PRIN:SEL:LINE:INCR
PRIN:PPREFIX PLUS;
:SEL:ITEM DATA
```

prints the following:

```
23.4567
+23.4567
```

PRINt[:DATA]:RESolution:AUTO <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

sets the number of digits resolution (to the right of the decimal point) to display for the following print `PRINt:ITEM` and `PRINt:SELected:ITEM` selections:`QUERy, MQUEry, DATA, MDATa, NOMLimit, MINLimit, MAXLimit.`

PRINt[:DATA]:RESolution[:DIGits] <number of digits>
<number of digits> numeric value 0 - 8

sets the number of digits (to the right of the decimal point) to display for the following print `PRInt:ITEM` selections:QUERy, MQUEry, DATA, MDATa, NOMLimit, MINLimit, MAXLimit.

This selection is ignored if `PRINt:RESolution:AUTO` is set to `ON`.

PRINt:DEVice:LANGuage<printer selection>
<printer selection> EPSon or PCL

selects the printer control language for all printer communications by defining the control codes that mark special fonts.

EPSon	Epson compatible printers (typically dot-matrix printers)
PCL	Hewlett-Packard Printer Control Language level 3 or greater compatible printers (typically LaserJet or DeskJet printers)

Many printers use either DIP switches or a setup menu to select between two or more different printer control languages. Ensure that either Epson or PCL mode is selected on your printer.

PRINt:DUMP:AGRaph

dumps the results of the last autograph test to the printer.

The autograph dump consists of 3 blocks separated by horizontal lines:

- the header information (defined by `PRINt:HEADer`),
- the time/date, mobile information, mobile service information, etc. and the autograph power level measurement information in compressed text.

Up to 11 columns of data are displayed, one for each MAC level tested, with a header at the top defining the MAC level for that column.

The channels tested are listed at the left margin, along with the equivalent frequency, and the power measurements are placed in the corresponding locations in this grid.

The measurement units are selected by `DISP:UNIT:POW:TRAN`.

A "P" or an "F" is placed after the measurement value to indicate its pass/fail status; measurements that fail are printed in bold.

If this command is executed when no autograph test has been run, both the first and second blocks are printed, but only the channel and MAC level header line in the third block are printed.

Any header line that has no data is omitted from the printout.

This command cannot be queried.

PRINt:DUMP[:ALL]

dumps the entire contents of the print buffer to the printer.

The print buffer contains one full page (60 lines) of formatted text.

This command cannot be queried.

PRINt:DUMP:LIMits

dumps the current selected limits table settings to the printer.

The limits dump consists of:

- a header that contains the name of the limit table, the manufacturer code and ESN range defined for the table, and the limit table version number, and
- limit parameters, one parameter per line.

All limit parameters (name, units, nominal, lower offset, and upper offset values) are listed, whether enabled or not.

Limit values that are enabled are bolded, and those disabled are printed in normal text.

This command cannot be queried.

PRINt:DUMP:LINE <start line>, < stop line>

dumps the selected lines from the print buffer to the printer.

If a stop line is not specified, the end of the buffer (line 60) is assumed to be the stop line.

PRINt:DUMP:LOG

dumps the contents of the sequencer log to the printer after a sequence completes.

If logging to the printer was not enabled prior to running the sequence or if the printer ran out of paper while the sequence was running, the user can get a printout of what remains in the buffer using this command.

The buffer holds up to 100 lines of sequencer information (most log entries are one line per measurement). If the buffer overflows, the oldest log entries are thrown out, and only the most recent 100 lines are kept.

The log buffer is automatically cleared out when a sequence is started.

This command cannot be queried.

PRINt:FORMat <format selection>
<format selection> HEXadecimal, DECimal, or BINary

selects the format to display the ESN and SCM data in when outputting these items to the print buffer.

Selected `PRINt:SELected:ITEM` selections are affected: `MCODe, RSVD, ESN1, ESN2, SCM`.

PRINt:HEADer[*n*] <header text>
[*n*] specifies which header line is being defined
<header text> the text information to place in the header.

defines one line of the header. (A maximum of 10 lines with up to 160 characters per line can be defined.)

Enclose the text information in double quotes. If fewer than 10 lines are needed, set any unneeded line to a null string to prevent it from being printed.

The header information is preserved from power down to power up.

Load the header definition from a disk file using `DISK:LOAD:HEADer`.

Example:

Four lines of text are defined, and the other 6 lines are set to null.

prin:head1 “CustomerName_________________________”
prin:head2 “Address _______________________________”
prin:head3 “Vehicle type ___________________________”
prin:head4 “——————————————————”
prin:head5 “”
prin:head6 “”

prin:head7 ""
prin:head8 ""
prin:head9 ""
prin:head10 ""

PRINt:HLINe <line number>, <character position>, <length>

<line number>	numeric integer 1 - 60
<character position>	numeric integer 1 - 160
<length>	numeric integer 1 - 160

draws an underline of the specified length on the specified line, and beginning at (and to the right of) the specified column.

This is similar to `PRINt:SELected:HLINe`, except that the line number and starting position are specified, rather than using the `PRIN:SEL:LINE` and `PRIN:SEL:COL` parameter values.

PRINt:ITEM <line number>, <character position>, <item selection>

<line number>	numeric value from 1 - 60
<character position>	numeric value from 1 - 132
<item selection>	the item to be printed

places the specified data into a specific location in the printer buffer.

See `PRINt:SELected:ITEM`.

This buffer information is sent to the printer when `PRINt:DUMP` is issued.

This command does not affect the currently selected line number or column used by `PRINt:SELected`.

Example: PRIN:ITEM 23,1,DATE
prints the current date at the 1st character position of line 23.

PRINt:OVERwrite <boolean>

<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the overwrite mode of `PRINt:STRing` and `PRINt:SELected:STRing`.

When overwrite is enabled, the string information is placed at the specified location in the page buffer, and overwrites any data that exists at that location.

When overwrite is disabled, only those characters that do not overwrite a previously written character are placed in the buffer, allowing data to be placed in the appropriate locations on the page with overlaid text around it, without overwriting the data.

Example: PRIN:OVER ON
MEAS:POW:TRAN?
PRIN:ITEM 6, 15, DATA

PRIN:STR 6, 1,"Power level dBm"
PRIN:STR overwrites the data placed by PRIN:ITEM and is either placed before PRIN:ITEM, or broken up into two commands: one to place "Power level" in front of the data, and the other to place "dBm" after it, to prevent it from overwriting the data.

A better alternative is to disable overwrite mode prior to writing the string, if the string is written after the data has been placed.

By changing PRIN:OVER ON to PRIN:OVER OFF on the first line, the data remains intact and the text before and after it is still written.

PRINt:PRESet

initializes all of the 4300 printer parameters to their corresponding default values and clears the print dump buffer to prevent inconsistent operation in printer outputs due to uninitialized printer parameters.

This command also initializes new printer parameter commands to default values.

Use this command in all 4300 program sequences that use `PRINt:DUMP[:ALL] or PRINt:DUMP:LINE`.

Place this command in front of any other printer command to ensure that all printer parameters are set to a specific value.

The following printer parameters are initialized by this command:

SCPI command	Default Value
PRINt:BOLD	OFF
PRINt[:DATA]:JUSTify	LEFT
PRINt[:DATA]:MFACtor	1.0
PRINt[:DATA]:PPRefix	NONE
PRINt[:DATA]:RESolution:AUTO	ON
PRINt[:DATA]:RESolution[:DIGits]	2
PRINt:FORMat	HEXADECIMAL
PRINt:OVERwrite	ON
PRINt:SELected:COLumn:TAB:DEFine	1, 9, 17, 25, 33, 41, 49, 57
PRINt:SELected:COLumn[:VALue]	1
PRINt:SELected:LINE[:VALue]	1
PRINt:SIZE	NORMAL
PRINt:UNDerline	OFF

PRINt:SELected:COLumn:TAB:BACKward

shifts the current character position to the last defined tab stop.

This command operates only if the current character position is after the last defined tab stop.

This command cannot be queried.

Example 1: PRIN:SEL:COL:TAB:DEF 8, 16, 21, 50, 65
PRIN:SEL:COL 27
sets character position to 27.
PRIN:SEL:COL:TAB:BACK
sets character position to 21 by using the back tab.
PRIN:SEL:COL:TAB:BACK
sets character position to 16 by using the back tab.

PRINt:SELected:COLumn:TAB:DEFine <tab stop1> [, tab stop2...]
<tab stop*n*> numeric value 1 - 132

defines the tab stop character positions.

At least 1 tab stop must be defined, and up to 8 tab stops can be defined using this command.

Define tab stops in ascending order.

Example: PRIN:SEL:COL:TAB:DEF 8, 16, 21, 50, 65
defines tab stops at positions 8, 16, 21, 50, and 65.

PRINt:SELected:COLumn:TAB[:FORWard]

advances the current character position to the next defined tab stop.

This command operates only if the last tab position has not been exceeded.

This command cannot be queried.

Example 1: PRIN:SEL:COL:TAB:DEF 8, 16, 21, 50, 65
PRIN:SEL:COL 17
sets character position to 17.
PRIN:SEL:COL:TAB
sets character position to 21 by using the tab.
PRIN:SEL:COL:TAB
sets character position to 50 by using the tab.

PRINt:SELected:COLumn:TAB:RESet

sets the current character position to the first defined tab stop.

This command cannot be queried.

PRINt:SELected:COLumn[:VALue] <character position>
<character position>numeric value 1 - 132

defines the character position in the print buffer at which to place data from all subsequent `PRINt:SELected:STRing` and `PRINt:SELected:ITEM` commands.

Example: PRIN:SEL:LINE 27
PRIN:SEL:COL 14
PRIN:SEL:STR “Units”
sets the line number to 27, character position to 14, and prints ‘Units’ at that location.

PRINt:SELected:HLINe <length>
<length> numeric integer 1 - 160

draws an underline of the specified length on the line specified by `PRIN:SEL:LINE`, and beginning at (and to the right of) the column specified by `PRIN:SEL:COL`.

The specified length is in units of character positions; the actual size depends on `PRIN:SIZE`.

This command is equivalent to setting `PRIN:UNDerline ON` prior to writing data or a string, except that it allows the underline to be enabled after the string and/or data item combinations have been written.

If a line contains both data and text (and a combination of those is to be underlined), or if several non-sequential lines of text are to be underlined, it is easier to use this command than to turn `PRIN:UNDerline ON` and `OFF` several times.

If `PRIN:STR` or `PRIN:ITEM` follows this command, the underline under those written portions is disabled if `PRIN:UNDerline` is `OFF`.

PRINt:SELected:ITEM <item selection>
<item selection> the item to be printed

places the specified data into the current line and character position of the printer buffer.

The line number is specified by `PRIN:SEL:LINE`; the character position by `PRIN:SEL:COL`.

This buffer information is sent to the printer when either `PRINt:DUMP[:ALL]` or `PRINt:DUMP:LINE` is issued.

Example: PRIN:SEL:ITEM DATE
prints the current date at a specified line and character position.

Selections:

DATE	the current date in MM-DD-YY (or DD-MM-YY) format. If SYSTem:DATE:FORMat US is specified, the date is in MM-DD-YY format (April 17, 1992 formatted as 04-17-92). If SYSTem:DATE:FORMat INT is specified, the date is in DD-MM-YY format (April 17, 1992 formatted as 17-04-92).
VDATe	the current date in Month DD, Year (or DD Month, Year) format. If SYSTem:DATE:FORMat US is specified, the date is in Month DD, Year format (e.g. Apr 17, 1992). If SYSTem: DATE:FORMat INT is specified, the date is in DD Month, Year format (e.g. 17 Apr, 1992).
TIME	the current time in HH:MM format. If SYSTem:TIME:FORMat HR is specified, the time is in 24-hour format (e.g. 13:47). If SYSTem:TIME:FORMat AMPM is specified, the time is in 12-hour format (e.g. 1:47 PM).
MCODe	the 8-bit manufacturer code from the mobile registration ESN value. The data format is selected by PRINt:FORMat: HEX: the serial number is displayed as 2 hexidecimal digits DEC: the serial number is displayed as 3 decimal digits BIN: the serial number is displayed as 8 binary digits.
MNAMe	string consisting of the manufacturer's name corresponding to the manufacturer code value (MCODe) of the mobile registration ESN value. The string may be up to 40 characters in length.
RSVD	the 6-bit reserved field from the mobile registration ESN value. The data format is selected by PRINt:FORMat: HEX: the serial number is displayed as 2 hexidecimal digits DEC: the serial number is displayed as 3 decimal digits BIN: the serial number is displayed as 6 binary digits.
ESN1	the 18-bit serial number from the mobile registration ESN value. The data format is selected by PRINt:FORMat: HEX: the serial number is displayed as 5 hexidecimal digits DEC: the serial number is displayed as 6 decimal digits BIN: the serial number is displayed as 18 binary digits.
ESN2	the 6-bit reserved field and 18-bit serial number from the mobile registration ESN value are combined into a 24-bit value, with the reserved field being most significant. The data format is selected by PRINt:FORMat: HEX: the serial number is displayed as 6 hexidecimal digits DEC: the serial number is displayed as 8 decimal digits BIN: the serial number is displayed as 24 binary digits.

SCM	the 5-bit station class mark from the mobile registration. bits 4,1,0 = the power class I - VIII bit 3 = the transmission (continuous/discontinuous) bit 2 = the bandwidth (20 or 25 MHz) The data format is selected by PRINt:FORMat: HEX: the value is displayed as 2 hexidecimal digits DEC: the value is displayed as 3 decimal digits BIN: the value is displayed as 8 binary digit.
TYPE	a string that indicates the type of mobile that has been registered. One of the following strings is printed: AMPS for AMPS-only type mobiles NAMPs if the mobile is capable of NAMPS service CDMA if the mobile is capable of CDMA service ? if an unknown mobile type.
PClass	a string of up to 4 roman numeral chars (I - VIII) representing the Power Class value derived from the Station Class Mark.
TRANsmission	a string of 17-20 characters indicating the transmission and bandwidth characteristics of the mobile from the mobile registration Station Class Mark (SCM) value. Possible bit values and corresponding string information are: bits 3,2 = 00: "Continuous 20 MHz" 01: "Continuous 25 MHz" 10: "Discontinuous 20 MHz" 11: "Discontinuous 25 MHz"
OPERation	a 12-characters string indicating the number of channels of operation defined for the mobile from the mobile registration Station Class Mark (SCM) value. The bandwidth indication (bit 2) defines this bit: bit 2 = 0: "666 Channels" (20 MHz bandwidth) bit 2 = 1: "832 Channels" (25 MHz bandwidth)
MIN	a 13-character string indicating the 10-digit telephone number of the mobile decoded from the 34-bit Mobile Identification Number from a mobile registration. The string is formatted for the US standards: (XXX) XXX-XXX.
SID	the 5-digit System IDentification number from the last mobile registration.
DCC	the Digital Color Code value from the last time a forward control channel was established.
DIALed	the mobile number entered from the mobile for the last origination performed. This string is initialized to ASCII spaces at power up and is always 17 characters (digits) in length. It prints the same string that is returned from ALL:ORIG:DIAL?

DTMF	a list of the DTMF keys entered from the mobile since MEAS:DTMF was last executed. This string is initialized to 12 ASCII spaces at power up and when MEAS:DTMF is first issued. As DTMF keys are pressed, the corresponding ASCII characters for the keys received are placed in the string in the following order: "1234567890*#". This string contains the keys that have been pressed, but not the order in which they were pressed. It prints the same string that is returned from MEAS:DTMF?
QUERy	the value of the last query command issued that returns a numeric value. The number of digits resolution and positioning of the numeric value can be adjusted by the PRINt[:DATA]:*XXXX* commands.
MQUEry	this is the same as QUERy, except that the value is multiplied by the value defined by PRINt[:DATA]:MFACtor.
QENum	the enumerated string value of the last query command issued that returns an enumerated value. (e.g. PRIN:ITEM has enumerated values of QUERy, MQUEry, QENum, etc.).
QBOOl	the status of the last query command issued that returns a boolean value. The status is displayed as ON or OFF.
DATA	the value of the last measurement command that returns a numeric value. The number of digits resolution and positioning of the numeric value can be adjusted by PRINt[:DATA]:*XXXX*. If the measurement was within the limits specified by the corresponding CALCulate:LIMit command, or was not tested against limits, this value prints in the typeface (NORMAL/BOLD) selection defined by PRINt:BOLD. If the measurement failed the limit specifications and the limits were enabled, the value is printed in BOLD. (RESult can be used in place of DATA.)
DATA2	this is the value of the second measurement value for measurement commands that return an (xi) coordinate as a value. The only measurement commands that functions this way are READ:VERR:IQ? and READ:VERR:NORM:IQ?. In both cases, the I value is reported using DATA, and the Q value is reported using DATA2.
MTESt	a string of characters (25 or less) representing the name of last measurement performed (or read using READ). These always match the values selected by DATA, DATA2, MUNit, NOMLimit, MINLimit, and MAXLimit. The name uses the same nomenclature as the sequence log to printer.
MDATa	this is the same as DATA, except that the value is multiplied by the value defined by PRINt[:DATA]:MFACtor.

MUNit	the units defined for the last measurement value (DATA). This is a 1 to 4 character string of uppercase alpha characters. (e.g. DBM, MHZ, WATT)
LIMit	a string of characters representing the current limit table selection made by CALCulate:LIMit:TYPE:LOAD. The possible values are: NONE, EIA, Custom1, Custom2, Custom3, Custom4, or Custom5.
NOMLimit	the nominal value defined by the corresponding CALCulate:LIMit command for the last measurement value. The number of digits resolution and positioning of the numeric value can be adjusted using PRINt[:DATA]:*XXXX*.
MAXLimit	the absolute maximum limit value defined by the corresponding CALCulate:LIMIT command for the last measurement value. If no limits exist or the limit is disabled, no value is printed. The number of digits resolution and positioning of the numeric value can be adjusted using PRINt[:DATA]. If the measurement value is less than this limit, this value prints in the typeface (NORMAL/BOLD) selection defined by PRINt:BOLD. If the measurement failed the limit, the value is printed in bold.
MINLimit	the minimum limit value defined by the corresponding CALCulate:LIMIT command for the last measurement value. If no limits exist or the limit is disabled, no value is printed. The number of digits resolution and positioning of the numeric value can be adjusted using PRINt[:DATA]:*XXXX*. If the measurement value is greater than this limit, the value prints in the typeface (NORMAL/BOLD) selection defined by PRINt:BOLD. If the measurement failed the limit, the value is printed in bold.
STATus	a string representing the PASS/FAIL status of the last measurement query. If the measurement failed the limit, the value is printed in bold (ignores the PRINt:BOLD selection for this entry). The possible values are: PASS, FAIL, or NOT TESTED.
CSTatus	a string representing a cumulative PASS/FAIL indication for all of the measurements since the last CALC:LIM:FAIL:CUMulative command. This command resets the cumulative PASS/FAIL flag to NOT TESTED, so that a group of commands can be accumulated as a single PASS/FAIL status. It indicates FAIL if any of the measurements failed, NOT TESTED if none of the measurements was verified against limits, and PASS otherwise. FAIL is printed in bold.

TSTatus	the same as CSTatus, except it is cumulative from the beginning of a test sequence and indicates if any measurements failed in the sequence. Reset the flag by restarting the test sequence.
HEADer	all non-NULL strings defined by PRINt:HEADer. Up to 10 lines of 160 characters can be defined.

PRINt:SELected:LINE:INCRement <increment value>
<increment value> numeric value -60 - +60

increments (or decrements) the current line number selection for the print buffer.

A positive value increases the line number, and a negative value decreases the line number to allow a test sequence loop to begin at a fixed line number, and each iteration of the loop print on sequential lines.

It also allows a subroutine to modify the current line number to print an entry offset from the current line and to restore the value before returning.

If <increment value> is omitted, a +1 is assumed.

Example: PRIN:SEL:LINE:INCR
PRIN:SEL:STR “Measurement”
PRIN:SEL:LINE:INCR -1
increments the current line number, prints measurement at that location, and restores the line number to its original value.

PRINt:SELected:LINE[:VALue] <line position>
<line position> numeric value 1 - 60

defines the line number of the print buffer at which to place data from all subsequent `PRINt:SELected:STRing` and `PRINt:SELected:ITEM` commands.

Example: PRIN:SEL:LINE 27
PRIN:SEL:STR “Units”
sets the line number to 27 and print ‘Units’ at that location.

PRINt:SELected:STRing <text string>
<text string> the text (up to 160 characters) enclosed in double quotes

places a text string into the current line and character position of the printer buffer.

The line number is specified by `PRIN:SEL:LINE`, and the character position by `PRIN:SEL:COL`.

This buffer information is sent to the printer when either `PRINt:DUMP[:ALL]` or `PRINt:DUMP:LINE` is issued.

Example: PRIN:SEL:STR "This text will be printed"
prints "This text will be printed" at the current line and character position.

PRINt:SIZE <width selection>
<width selection> character width selection: NORMal, SMALl, or LARGe

selects the font width (the character height remains constant at 60 lines per page).

All commands that output to the print buffer after this command is issued, print data in the specified font width.

The character position (column) specified when outputting a string or item is the character position in `NORMAL` mode characters.

NORMAL standard mode(80 characters per line)
SMALL compressed mode(132 characters per line)
LARGEexpanded mode(40 characters per line)

PRINt:STRing <line number>, <character position>, <text string>
<line number> numeric value from 1 - 60
<character position> numeric value from 1 - 132
<text string> the text to be printed is enclosed in double quotes

places a text string into a specific location in the printer buffer.

This buffer information is sent to the printer when `PRINt:DUMP` is issued.

This command does not affect the selected line number or column that is used by `PRINt:SELected`.

Example: PRIN:STR 10,25,"This text will be printed"
prints 'This text will be printed' on line 10, at character position 25.

PRINt:UNDerline <boolean>

<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the underline setting for characters printed.

All commands that output to the print buffer after this command is issued print data based on this setting.

The underline a part of the character and does not affect characters on the line beneath the current one.

PROGram Subsystem

The *Program* subsystem commands load, execute or direct a program sequence, autograph test, or time alignment test.

PROGram:AGRaph <mode select>
<mode select> RUN, STOP, or PAUSe

begins, terminates, or pauses execution of the Autograph test.

If the test is not running, `PAUSe` has no effect.

If the test is currently paused, `RUN` acts as a `CONTinue` by exiting the `PAUSe` mode and continuing execution at the current channel, but restarting at the lowest MAC level enabled.

This command is not valid from within a program. Use it only as a remote command.

PROGram:AGRaph:CHANnel:BAND <band select>
<band select> CELLular or PCS

selects the `BAND` in which to perform the Autograph test.

If `PCS` is selected, the start and stop channel range = 1 - 1199.

The PCS band selection is only allowed if the PCS option is installed.

If `CELLular` is selected, the cellular band the channel selection range = 1 - 799 and 991 - 1023.

If the start or stop channels are set outside the range of the cellular band (while PCS band is selected) and the band selection is changed to cellular, the start channel is set to 100 and the stop channel is set to 650 (the default values).

The band selection is made for the entire Autograph test and cannot switch between PCS and cellular bands during testing.

PROGram:AGRaph:CHANnel:STARt <channel number>
<channel number> numeric integer1 - 1199 (excluding 800-989, 1024-119 for cellular)

specifies the start channel number for the Autograph test.

`STARt` is value-coupled with the `STOP` channel number and must be a lower frequency than the `STOP` channel.

Channels 991-1023 are lower than channel 1 in the cellular band.

PROGram:AGRaph:CHANnel:STEP <channel number>
<channel number> numeric integer from 1 - 100

specifies the channel step size for the Autograph test.

PROGram:AGRaph:CHANnel:STOP <channel number>
<channel number> numeric integer1 - 1199 (excluding 800-989, 1024-119 for cellular)

specifies the stop channel number for the Autograph test.

It is value-coupled with the `START` channel number and must be a higher frequency than the `START` channel.

Channels 991-1023 are lower than channel 1 in the cellular band.

PROGram:AGRaph:CHANnel:TYPE <voice type>
<voice type> AMPS or CDMA

specifies which type of voice channel type to use for the Autograph test.

PROGram:AGRaph:DLOG <boolean>
<boolean. OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-disk for the Autograph test.

This command is similar to `PROGram:CUSTom:DLOG`, except that it affects the Autograph test sequence instead of the Custom tests, and the filename extension is .agr instead of .log.

PROGram:AGRaph:MAC[*n*] <boolean>
[*n*] numeric integer from 0 - 7
<boolean> OFF or ON (0 = OFF, 1 = ON)

specifies the MAC levels for the Autograph test.

Example: PROG:AGR:MAC0 1; MAC1 0; MAC2 0; MAC3 1
PROG:AGR:MAC4 0; MAC5 1; MAC6 0; MAC7 1
enables MAC levels 0, 3, 5, 7 for the Autograph test.

PROGram:AGRaph:PAUSe <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the Autograph test pause on measurement failure.

If the test pause is enabled, the test equipment enters the paused mode if the power level measurement failed the limit tests.

See `PROG:CUSTom:PAUSe`.

PROGram:AGRaph:PLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-printer for the Autograph test sequences.

This command is similar to `PROGram:CUSTom:PLOG`, except that it affects the Autograph test sequence instead of the Custom tests.

PROGram:AUTO:DLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables and disables the log-to-disk for the Automatic test sequences.

It is similar to `PROGram:CUSTom:DLOG`, except that it affects the Automatic test sequence instead of the Custom tests.

PROGram:AUTO:PAUSe <pause selection>
<pause selection> NONE, ALWays, or FAILure

selects which conditions cause the Automatic test sequence to pause.

If `NONE` is selected, the pause mode is only entered if a command error is encountered (invalid syntax, etc.).

If `ALWays` is selected, the sequence pauses after every command is executed.

If `FAILure` is selected, it pause when a measurement command fails its limit checks.

When paused, the soft key definitions on the front panel change to allow the user to continue with the next test, repeat the last test, abort the sequence, or enter manual mode.

This command is similar to `PROGram:CUSTom:PAUSe`, except that it affects the Automatic test sequence instead of the Custom tests.

PROGram:AUTO:PLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-printer for the Automatic test.

This command is similar to `PROGram:CUSTom:PLOG`, except that it affects the Automatic test sequence instead of the Custom tests.

PROGram:CUSTom:DLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-disk for the Custom test sequences.

When enabled, each measurement or call processing command that executes places a corresponding entry in the disk log file containing the name of the test performed and the measurement results, limits, pass/fail status, or other pertinent information as a result of the test.

When the log is first opened, a temp.log filename is created. When the sequence completes, the filename is changed to reflect the ESN of the mobile that was tested.

It does so by taking the first 7 digits of the last recorded ESN from a mobile (using the format defined by `CALL:MDATa:ESN:FORMat`) plus a single character A-Z, chosen so as to not overwrite a previous log file with the same ESN already existing on the disk.

The filename still has the .log extension.

PROGram:CUSTom:NUMBer <custom select #>
<custom select #> numeric integer from 1 - 3

specifies which custom program sequence to access when `PROGram:NAME` is set to Custom.

This command works in conjunction with `PROG:STATe`, `PROG:DEFine`, and `DISK:LOAD` for starting, reading from, or writing to a specific Custom sequence.

PROGram:CUSTom:PAUSe <pause selection>
<pause selection> NONE, ALWays, or FAILure

selects which conditions cause the Custom test sequence to pause.

If `NONE` is selected, the pause mode is only entered if a command error is encountered (invalid syntax, etc.).

If `ALWays` is selected, the sequence pauses after every command is executed.

If `FAILure` is selected, it pauses when a measurement command fails its limit checks.

When paused, the soft key definitions on the front panel change to allow the user to continue with the next test, repeat the last test, abort the sequence, or enter manual mode.

PROGram:CUSTom:PLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the log to printer for the Custom test sequences.

When enabled, each measurement or call processing command that executes outputs a corresponding entry to the printer containing the name of the test performed and the measurement results, limits, pass/fail status, or other pertinent information as a result of the test.

The format of the data is the same as what is saved when log-to-disk is enabled.

PROGram:QUICk:DLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-disk for the Quick test sequences.

This command is similar to `PROGram:CUSTom:DLOG`, except that it affects the Quick test sequence instead of the Custom tests.

PROGram:QUICk:PAUSe <pause selection>
<pause selection> NONE, ALWays, or FAILure

selects which conditions cause the Quick test sequence to pause.

This command is similar to `PROGram:CUSTom:PAUSe`, except that it affects the Quick test sequence instead of the Custom tests.

PROGram:QUICk:PLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the log-to-printer for the Quick test sequences.

This command is similar to `PROGram:CUSTom:PLOG`, except that it affects the Quick test sequence instead of the Custom tests.

PROGram[:SELected]:BUFFer:CLEAr

sets all entries in the program buffer to the following specifications:

value	0.0
resolution	1.0
valid flag	INVALID
pass/fail flag	NOT TESTED

The valid and pass/fail flags are only set when measurements are logged to the buffer, and only accessible by the `BranchIf` commands in the program sequences.

The internal buffer index value is reset to 0 (the first location in the buffer) so that measurement logging is placed at the start of the buffer.

PROGram[:SELected]:BUFFer:DATA <value>, [resolution]

<value> numeric value-1.0 e10 to +1.0 e10

[resolution] (optional) numeric value1.0 e-6 to 1.0

sets the next location in the sequencer log buffer to a specific value (and optionally to a specific resolution).

If no resolution is specified, the current resolution value at this location is used. The resolution is 1.0 for all buffer locations following `PROG:BUFF:CLEAr`, and is modified by this command and when a measurement is logged to the buffer.

The buffer location affected by this command can be set directly with `PROG:BUFF:INDex`, or modified from its current location by `PROG:BUFF:NEXT and PROG:BUFF:PREV`.

Following `PROG:BUFF:CLEAr`, the buffer index is set to the first buffer location (index 0). When `PROG:BUFF:DATA` is used to place data in the buffer and a measurement is logged to the buffer, the buffer index automatically increments so that the next write to the buffer does not overwrite the data previously written.

The program buffer commands are meant to be used by the sequence programs for capturing measurement data for use later in the program, for use in another program, or to pass to the remote port.

To place data in the buffer manually:

Example:

PROG:BUFF:CLEAR
PROG:BUFF:MLIST MEAS
PROG:BUFF:MLOG ON
MEAS:POW:TRAN:UNIT WATT; IMM?
1.0326
CALL:VMAC?
3
PROG:BUFF:DATA 3
PROG:BUFF:PREV; PREV
PROG:BUFF:DATA 1032.6, 0.1
PROG:BUFF:NEXT; NEXT

The first three commands initialize the buffer and enable automatic logging of measurement values read.

The power measurement is placed in the first buffer location, and has an associated resolution of 0.0001 Watts.

The current MAC value is read from the 4300 and is not logged automatically, since it is not a MEASure command. It is placed there manually using PROG:BUFF:DATA.

The resolution is omitted, since it defaults to 1.0 following the clear command. The measurement value in the first location is changed from Watts to mW using PROG:BUFF:DATA. This time, the resolution value is assigned, since it is less for the smaller units.

Finally, the buffer index is incremented back to the next available location.

After these commands complete, the buffer contains two entries: 1036.6 (res 0.1), and 3.0 (res 1.0).

PROGram[:SELected]:BUFFer:DATA? <buffer index>, [offset]

<buffer index>	numeric integer0 - 399
[offset]	(optional) numeric integer 0 - 399

returns the value contained in the sequence log buffer at a specified index location.

If the optional [offset] value is not sent, the program buffer is treated as a single-dimensional array and the <buffer index> selects which value to return.

If the optional [offset] value is sent, the array is treated as a 2-dimensional array with one of the dimensions set to the number of measurement entries specified by `PROG:BUFF:MLISt`.

The <buffer index> now represents the measurement index and the [offset] represents the offset to a selected piece of information collected during that measurement.

The actual buffer index is calculated by multiplying <buffer index> by the number of entries selected to be logged and added to [offset]. This command affects the `BranchIfInvalid`, `BranchIfNotTested`, `BranchIfPass`, and `BranchIfFail` sequence directives by using the flags that have been set by the measurement logging.

See `PROG:BUFF:MLOG` for a description of how these flags are set.

The resulting buffer index value must be within the range specified for each parameter.

Response: <numeric floating point>

Resolution: depends upon the data stored

Units: depends upon the data stored

Example:

```
PROG:BUFF:CLEAR
PROG:BUFF:MLIST MIN, MAX, MEAS
PROG:BUFF:MLOG ON

FOR MacIndex = 0, 7
CALL:PLCH:VMAC <MacIndex>
MEAS:POW:TRAN?
NEXT

FOR MeasIndex = 0, 7
PROG:BUFF:DATA? <MeasIndex>, 2
PRIN:SEL:ITEM QUERY; COL:TAB
PROG:BUFF:DATA? <MeasIndex>, 0
PRIN:SEL:ITEM QUERY; COL:TAB
PROG:BUFF:DATA? <MeasIndex>, 1
PRIN:SEL:ITEM QUERY; COL:TAB:RES
PRIN:SEL:LINE:INCR
NEXT

FOR MeasIndex = 0, 23
PROG:BUFF:DATA? <MeasIndex>
PRIN:SEL:ITEM QUERY; COL:TAB
PROG:BUFF:DATA? <MeasIndex>
PRIN:SEL:ITEM QUERY; COL:TAB
PROG:BUFF:DATA? <MeasIndex>
PRIN:SEL:ITEM QUERY; COL:TAB:RES
PRIN:SEL:LINE:INCR
NEXT
```

The program buffer is first initialized and set to collect 3 parameters (maximum limit, maximum limit, and measured value) for each measurement command executed.

Then 8 mobile power level changes and a power measurement at each level are performed.

The second loop queries the information at each measurement and prints them on successive lines. The measurement value is printed at the first tab stop (MEAS = offset of 2), the minimum limit value at the second stop (MIN = offset of 0), and the maximum limit at the third tab stop (MAX = offset of 1). The third loop performs the same task without using the <off-set> parameter in PROG:BUFF:DATA?.

PROGram[:SELected]:BUFFer:FREE?

returns the number of locations available for measurement storage rather than the number of measurements that can be stored.

To determine this value, divide by the number of `PROG:BUFF:MLIST` selections and truncating the fractional portion of the result.

If this command is issued immediately after `PROG:BUFF:CLEAR`, it indicates the size of the program buffer.

Response: numeric integer

PROGram[:SELected]:BUFFer:INDex <buffer index>
<buffer index> numeric integer 0 - 399

sets the internal buffer index to the specified value.

The buffer index is used:

- when logging measurements to the program buffer, or
- when writing data directly to the buffer using `PROG:BUFF:DATA`.

This command modifies the results of an earlier entry, or a retested measurement.

The query form of this command responds with the current index value setting, rather than the last value sent by the command form.

If the buffer index has not been modified by `PROG:BUFF:INDex`, `PROG:BUFF:NEXT`, or `PROG:BUFF:PREV`, this value represents the number of entries written to the program buffer.

To safeguard the integrity of the index value, ensure that it is returned to its former value after it has been changed. This allows other programs or remote accesses of the program buffer to know how many values have been placed in the buffer.

PROGram[:SELected]:BUFFer:MLISt <sel1>[, sel] [, sel3] [, sel4]
<sel1> character data (MEASurement, NOMinal, MINimum, or MAXimum)
[sel2-4] (optional) same as <sel1>, plus NONE

selects which measurement information is placed in the program buffer when measurement logging is enabled by `PROG:BUFF:MLOG`.

A minimum of one selection must be made; a maximum of 4 entries are allowed.

The measurement information consists of the measurement value (`MEASurement`) and all of the limit values (`NOMinal, MINimum, and MAXimum`).

They can be in any order and the same entry can be duplicated.

Any omitted entries are set to `NONE`.

When a measurement is performed, this list is scanned and the corresponding values are placed in the program buffer in the specified order.

The entries placed in the buffer are terminated at the first occurrence of `NONE` found in the list.

If `MEASurement` is listed twice and the measurement being logged has a 2-parameter response (`READ:VERR:IQ?` and `READ:VERR:NORM:IQ?` are the only 2-parameter response commands), the first occurrence returns the first response parameter (the I value) and the second occurrence returns the second response parameter (the Q value).

If the measurement does not have a 2-parameter response, the same measurement value is logged twice.

See `READ:LAST[:DATA]?` for a list of the `MEASure` and `READ` commands that log results.

See `READ:LAST:MIN` and `READ:LAST:MAX` on how to handle measurements without limits.

Example 1: PROG:BUFF:MLIST MEAS
only logs the measurement values.

Example 2: PROG:BUFF:MLIST MEAS, MIN, MAX
logs, in order, the measurement value followed by it minimum and maximum limits.

Example 3: PROG:BUFF:MLIST NOM, NOM, NONE, MEAS
logs the nominal limit value twice in successive locations.

Since NONE occurs in the list prior to MEAS, the measurement value is not placed in the program buffer.

PROGram[:SELected]:BUFFer:MLOG <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the automatic logging of `MEAS` and `READ` results to the sequencer log buffer.

If disabled (`OFF`), measurement results are not logged.

If enabled (`ON`), the measurement information specified by `PROG:BUFF:MLIST` is placed in the program buffer along with: the resolution of the value, whether the value was valid, and the pass/fail status of the measurement.

The valid flag indicates `INVALID` on measurements that had an error condition (an accurate measurement was prevented from being made).

See `READ:LAST:VALid` for a description of this flag.

For `MINimum` and `MAXimum` selections, the valid flag indicates `INVALID` if the corresponding limit was not enabled. The valid flag indicates `VALID` for `NOMinal` selections.

The pass/fail status records whether the measurement `PASSED` or `FAILED` the limit check, or indicates `NOT TESTED` if the limits were disabled.

These flags are accessible to the program sequence following `PROG:BUFF:DATA?` of a selected entry using the following `BranchIf` commands:

```
BranchIfInvalid
BranchIfNotTested
BranchIfPass
BranchIfFail
```

`PROG:BUFF:DATA?` sets the resolution used for printing the value (with `PRIN:SEL:ITEM QUERY`) to the value recorded in the program buffer for the selected entry, if `PRIN:RES:AUTO` is set to `ON`.

As measurements are recorded, an internal buffer index is incremented to keep track of the next location to write. This value can be modified using `PROG:BUFF:INDEX`, `PROG:BUFF:NEXT` and `PROG:BUFF:PREV`.

CAUTION: To avoid losing measurement information, ensure that information is written to the correct location.

The measurement information in the buffer is not removed when a front panel `RETEST` of a measurement is performed during the sequence.

Logging of measurement information is disabled on power up of the tester.

If the program buffer is full, no more measurements are logged until either the buffer is cleared or the buffer index value is decremented.

PROGram[:SELected]:BUFFer:NEXT

increments the internal buffer index to the next entry.

Use the buffer index when logging measurements to the program buffer and writing data directly to the buffer using `PROG:BUFF:DATA`.

Use this command to modify the results of an earlier entry or a retested measurement.

PROGram[:SELected]:BUFFer:PREV

decrements the internal buffer index to the previous entry.

Use the buffer index when logging measurements to the program buffer and writing data directly to the buffer with `PROG:BUFF:DATA`.

Use this command to modify the results of an earlier entry or a retested measurement.

PROGram[:SELected]:DEFine <string data>
<string data> ASCII string data enclosed in double quotes

uploads a program sequence to the test equipment.

The sequence may be up to 5000 characters in length, including the newline characters. The entire string is enclosed in double quotes.

The sequence location for this information is defined by `PROG:NAME`.

The query form of this command returns the currently selected sequence data enclosed in double quotes. This is equivalent to downloading a program sequence to the remote port.

PROGram[:SELected]:DELay <delay time>
<delay time> numeric floating point from 0.0 - 10.0 (resolution of 0.01)

introduces a delay time (0 to 10 seconds) into the program execution sequence.

This command is not valid through the remote port. It can only be used from within a program.

PROGram[:SELected]:FCOUnt?

returns the total number of test failures for the last program sequence.

Response: <numeric integer>

Example: PROG:FCOU?
16
indicates that the last executed program sequence failed 16 measurement and/or call processing tests.

PROGram[:SELected]:NAME <program select>
<program select> AUTO, QUICk, or CUSTom

selects a program sequence to be run, written to, or read from.

This command does not run the program; it works in conjunction with `PROG:STATe, PROG:DEFine`, and `DISK:LOAD`.

If `CUSTom` is the selection for this parameter, `PROG:CUSTom:NUMBer` specifies which of the Custom sequences is being accessed.

PROGram[:SELected]:PAUSe

causes the currently executing program to enter the `PAUSed` state, and stops executing commands from the program sequencer.

A special set of front panel soft keys is displayed to allow the user to select CONTINUE when appropriate.

When the CONTINUE soft key is pressed, execution continues at the next command in the program. This command is useful if user intervention is required at some point during the program execution.

This command is not valid through the remote port. It can only be used from within a program.

PROGram[:SELected]:STATe <mode select>
<mode select> RUN or STOP

begins or terminates execution of the program sequence selected by `PROG:NAME`.

Use `PROG:STAT STOP` (or any remote command) to stop execution of a currently running sequence.

PROGram[:SELected]:STRing[*n*] <user message>
[*n*] 1 or 2
<user message> ASCII string enclosed in double quotes, with a maximum length of 30 characters

defines the text to place in two user-definable lines near the bottom of the LCD front panel display.

This message window is located just above the 3-line unit status window at the bottom of the display and just below the test results window.

The user-defined message window is erased when a program is first started, and when it completes. If a user message is defined, it is displayed in inverse video.

Example: PROG:STR1 “Make sure mobile has service”
PROG:STR2 “Press CONTINUE when ready”
PROG:PAUSe
PROG:STR1 “”
PROG:STR2 “”

This command is used in conjunction with PROGram:PAUSe to allow the user to read the message, take the required actions, and press the **CONTINUE** key.

The second set of definitions for the user-defined message clears the message out (assuming the user has pressed the **CONTINUE** key).

This command is not valid through the remote port. It can only be used from within a program.

PROGram[:SELected]:TCOUnt?

returns the number of total tests executed for the last program sequence that was run.

Response: <numeric integer>

Example: PROG:TCOU?
512
indicates that the last program sequence that was executed performed 512 measurement and/or call processing tests.

READ Subsystem

The *Read* subsystem commands obtain a previously made measurement value without performing another measurement.

The commands are in the same format as their *Measure* subsystem equivalents.

`PRINt:ITEM DATA, MDATa, and MUNit` reflect the `READ` command measurement values which allow a measurement to be made once, and the data to be read or printed in different units without taking another measurement.

The `READ` commands do not require the test equipment to be in any particular state in order to execute, since they are simply returning a previously measured parameter.

READ:BER:ANALog?

returns the last analog BER measurement performed by `MEAS:BER?` while on an AMPS call.

If the unit is not on an AMPS call when `MEAS:BER?` is performed, the analog BER measurement value is not updated.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Requirement: Unit must be on an AMPS call.

Response: <numeric integer>

Resolution: 0.01

Units: percentage

Measured by: MEAS:BER

READ:CDOMain:FREQuency:OFFSet?

returns the frequency offset value of the signal relative to the ideal signal.

Response: floating point value -1000.0 to 1000.0 in 1.0 increments

Units: Hz

Measured by: MEAS:CDOM:IMM

READ:CDOMain:PHASe:DELTa? <channel>
<channel>

returns the code channel phase offset for the specified code channel relative to the pilot.

Response: floating point value in 0.001 increments

Units: radians

Measured by: MEAS:CDOM:IMM

READ:CDOMain:PHASe:OFFSet?

returns the phase offset value of the signal relative to the ideal signal.

Response: floating point value -0.999 to 0.999 in 0.001 increments

Units: radians

Measured by: MEAS:CDOM:IMM

READ:CDOMain:POWer:DELTa? <channel>
<channel>

returns the code channel power for the specified code channel relative to the pilot.

Response: floating point value in 0.1 increments

Units: dB

Measured by: CALC:LIM:CDOM

READ:CDOMain:POWer:ICHannel? <walsh code>
<walsh code> current channel walsh code

returns the code channel power for the I channel having the specified Walsh code relative to the total received mobile power.

Response: floating point value in 0.1 increments

Units: dB

Measured by: MEAS:CDOM:IMM

READ:CDOMain:POWer:QCHannel? <walsh code>
<walsh code> current channel walsh code

returns the code channel power for the Q channel having the specified Walsh code relative to the total received mobile power.

Response: floating point value in 0.1 increments

Units: dB

Measured by: MEAS:CDOM:IMM

READ:CDOMain:RHO:COMPosite? <channel>
<channel> 0.9 - 1.0 with 0.0001 resolution

returns the composite Rho value derived from all of the code channels.

Response: <floating point value>

Units: none

Measured by: MEAS:CDOM:IMM

READ:CDOMain:RHO:IMAGinary? <channel>
<channel> 0.9 - 1.0 with 0.0001 resolution

returns the imaginary portion of the Rho value for the specified code channel.

Response: <floating point value>

Units: none

Measured by: MEAS:CDOM:IMM

READ:CDOMain:RHO:REAL? <channel>
<channel> 0.9 - 1.0 with 0.0001 resolution

returns the real portion of the Rho value for the specified code channel.

Response: <floating point value>

Units: none

Measured by: MEAS:CDOM:IMM

READ:CDOMain[:STATus]?

returns the pass/fail status of the last code domain measurement performed by `MEAS:CDOM:IMM?`.

Requirement:

Response: PASS, FAIL, NONE

Units: CALC:LIM:CDOM

READ:CDOMain:TIME:DELTa? <channel>
<channel>

returns the code channel time offset for the specified code channel relative to the pilot.

Response: <floating point value> in 1.0 increments

Units: ns

Measured by: MEAS:CDOM:IMM

READ:CDOMain:TIME:OFFSet?

returns the time offset value of the signal relative to the ideal signal.

Response: <floating point value> -0.999 to 0.999 in 0.001 increments

Units: ns

Measured by: MEAS:CDOM:IMM

READ:CDOMain:VALid?

returns code domain measurements performed by the last `MEAS:CDOM:IMM` command.

Response: numeric integer bit flag

READ:CDOMain:WQUality? <channel>
<channel> 0.9 - 1.0 with 0.0001 resolution

returns the waveform quality derived from all of the code channels.

Response: floating point value

Units: none

Measured by: MEAS:CDOM:IMM

READ:DEViation:PAUDio?

returns the last peak audio deviation measurement value.

The peak audio deviation value is reset to 0 at the beginning of `MEAS:DEV:AUDio?` and `MEAS:DEV:PAUD <value>`.

`MEAS:DEV:AUDio?` takes a single reading and sets the peak audio value to this value.

`MEAS:DEV:PAUDio` makes audio deviation measurements at approximately 300 msec intervals for the specified amount of time, updating the peak reading each time during this interval.

`READ:DEV:PAUDio?` can be issued at any time (including prior to completion) to retrieve this information.

To postpone reading the measurement until after the specified amount of time following `MEAS:DEV:PAUD <value>`, send `*WAI` or `*OPC?` prior to `MEAS:DEV:PAUD?` with the appropriate bit (`PEAK DEVIATION` value = bit 7 = 80 hexidecimal = 128 decimal) set in `STATus:OPERation:COMPlete:ENABle`.

Response: <numeric integer>

Units: Hz

READ:FER:FRAMes:ERRor?

returns the number of error frames measured in the last `MEAS:FER:IMM?` query.

Response**:** <numeric value>

Measured by: MEAS:FER:IMM?

READ:FER:FRAMes:TRANsmitted?

returns the number of frames transmitted in the last `MEAS:FER:IMM?` query.

Response: <numeric value>

Measured by: MEAS:FER:IMM?

READ:FER[:IMMediate]?

returns the frame error rate of the last FER test performed.

The minimum value is 0.0% (indicating no frame errors) and the maximum value is 100.0%.

Response: <numeric floating point>

Resolution: 0.0001

Units: percentage

Measured by: MEASure:FER[:IMMediate]?

Example: MEAS:FER:LIM 0.3; CONF 100.0;
MEAS:FER:FRAM:MAX 5000; IMM?
0.0230
READ:FER?
0.0230

READ:FER:STATus:CONFidence:FAILed?

returns the failed with confidence status of the last FER test performed.

A value of 0 indicates that the FER failed with confidence and a value of 1 indicates that the FER passed or failed.

"Failed with confidence" refers to a confidence factor percentage calculated using the setup commands that allow the test to complete prior to reaching the maximum number of frames set.

Response: <boolean> = 0 or 1
where: 0 = passed with confidence

Example: MEAS:FER:LIM 0.5; CONF 95.0; FRAM:MAX 5000; IMM?
0.4200
READ:FER:STAT:CONF:FAIL?
1
READ:FER:FRAM:TRAN?
5000
indicates that the test did not fail with confidence. It either failed without confidence or passed. Failed with confidence is a more restrictive test for smaller number of frames tested.

Measured by: MEASure:FER[IMMediate]?

READ:FER:STATus:CONFidence[:PASSed]?

returns the pass with confidence status of the last FER test performed.

A value of 1 indicates that the FER passed with confidence and a value of 0 indicates that the FER either failed or passed without confidence.

"Passed with confidence" refers to a confidence factor percentage calculated using the setup commands that allows the test to complete prior to reaching the maximum number of frames set if the number of frame errors is small.

It is more restrictive on the FER limit for a specified number of frames tested and is the method of measurement specified by the IS-98 standard.

Use this command if you need to know if the FER passed with confidence only, since it treats a passed without confidence the same as a failure.

Response: <boolean> = 0 or 1 (1 = passed with confidence)

Example 1: MEAS:FER:LIM 0.5; CONF 95.0;
MEAS:FER:FRAM:MAX 5000;IMM?
0.4200
READ:FER:FRAM:TRAN?
5000
READ:FER:FRAM:ERR?
21
READ:FER:STAT?
1
READ:FER:STAT:CONF?
0
indicates that the test passed either with or without confidence.

The return code from the passed with confidence query is 0, indicating that the test did not pass with confidence.

NOTE: Although this test did not exceed the 0.5 maximum FER, it still did not pass with confidence.

Example 2: MEAS:FER?
0.1055
READ:FER:FRAM:TRAN?
948
READ:FER:FRAM:ERR?
1
READ:FER:STAT?
1
READ:FER:STAT:CONF?
1
indicates that it passed with confidence.

NOTE: If the passed with confidence flag is 1, the generic pass/fail status is 1, whereas the reverse is not true.

NOTE: Due to the smaller number of errors that occurred, the test was able to complete after only 948 frames to meet the 95% confidence factor.

Measured by: MEASure:FER[:IMMediate]?

READ:FER:STATus[:PASSed]?

returns the pass/fail status of the last FER test performed.

A value of 1 indicates that the FER either passed or passed with confidence; a value of 0 indicates that the FER failed.

Passed with confidence refers to a confidence factor percentage, calculated using setup commands that allow the test to complete prior to reaching the maximum number of frames set.

Use this command if you are not concerned with the confidence factor, and are only interested in whether it passed or failed.

Response: <boolean>= 0 or 1
where: 1= passed with or without confidence

Example: MEAS:FER:LIM 0.5; CONF 95.0; FRAM:MAX 5000; IMM?
0.4200
READ:FER:FRAM:TRAN?
5000
READ:FER:STAT?
1
indicates that the test passed, either with or without confidence.

Since the number of frames transmitted was reported as 5000, the same as the maximum frames to test, the test passed without confidence. The pass/fail status for this case is determined by checking if the frame errors divided by the frames transmitted exceeds the maximum FER limit, which was specified as 0.5.

It is possible for a value that exactly equals the maximum FER limit to return either a PASS or FAIL value because the status is determined prior to limiting the resolution of the percent FER returned.

If the test passed with confidence, it indicates that the test was able to terminate early due to statistical probability (percentage confidence).

Measured by: MEASure:FER[:IMMediate]?

READ:FREQuency:ERRor:RF?

returns the last mobile transmitter RF frequency error measurement value set by either `MEAS:FREQ:RF?` or `MEAS:FREQ:ERR:RF?`, in units selected by `MEAS:FREQ:UNIT`.

See MEAS:FREQ:UNIT.

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:RF
MEAS:FREQ:ERR:RF

READ:FREQuency:ERRor:SAT?

returns the last SAT frequency error measurement value set by either `MEAS:FREQ:SAT?` or `MEAS:FREQ:ERR:SAT?`, in units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:SAT
MEAS:FREQ:ERR:SAT

READ:FREQuency:ERRor:ST?

returns the last ST frequency error measurement value set by either `MEAS:FREQ:ST?` or `MEAS:FREQ:ERR:ST?`, in units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:ST
MEAS:FREQ:ERR:ST

READ:FREQuency:RF?

returns the last mobile transmitter RF frequency measurement value set by either `MEAS:FREQ:RF?` or `MEAS:FREQ:ERR:RF?`, in units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT`.

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:RF
MEAS:FREQ:ERR:RF

READ:FREQuency:SAT?

returns the last SAT frequency measurement value set by either `MEAS:FREQ:SAT?` or `MEAS:FREQ:ERR:SAT?`, in units selected by `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT.`

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:SAT
MEAS:FREQ:ERR:SAT

READ:FREQuency:ST?

returns the last ST frequency measurement value set by either `MEAS:FREQ:ST?` or `MEAS:FREQ:ERR:ST?`, in units selected by the `MEAS:FREQ:UNIT`.

See `MEAS:FREQ:UNIT.`

Response: <numeric integer>

Units: Hz, kHz, MHz

Measured by: MEAS:FREQ:ST
MEAS:FREQ:ERR:ST

READ:LAST[:DATA]?

returns the measured value of the last measurement performed.

All of the `MEASure` and `READ` commands update this value except for the following:

READ:LAST:*XXXX* (any of these commands)
MEASure:DTMF[:FULL]? (query form only)
MEASure:DTMF?(query form only)
MEASure:DTMF:KEY? (query form only)
any of the MEASure subsystem SETUP commands.
(e.g. MEAS:FREQ:UNIT)

Some of the measurement commands return a non-numeric result, or multiple numeric values. The last measurement data value for these commands are as follows:

MEASure:STATus:SAT?
MEASure:STATus:ST:CONDition?
returns 1.0 if SAT (or ST) detected, 0.0 if not.

MEASure:DTMF[:FULL](the command form only)
MEASure:DTMF:KEY(the command form only)
returns the value 0.0.
MEASure:SENSitivity?
for AMPS and NAMPs, returns the base power level in dBm at which the test was performed.
for CDMA, returns the results of the FER measurement that was made during the base power reduction.
MEASure:BER?
returns the BER measurement value without the type (ANAL).
MEASure:VERRor:IQ?
MEASure:POW:TRAN:GATED[:IMM]
MEASure:POW:TRAN:OPEN[:IMM]
returns the value 0.0.
READ:VERRor:IQ[:DATA]:SAMPle?
returns the value of I for the specified sample.
READ:POW:TRAN:GATed[:DATA]:SAMPle?
READ:POW:TRAN:OPEN[:DATA]:SAMPle?
returns the measurement value for the specified sample.

Response: <numeric floating point>

READ:LAST:MAXimum?

returns the maximum limit value associated with the last measurement performed, and whether the maximum limit was enabled or disabled.

If the maximum limit was disabled for the measurement, or if no limit or measurement value is defined, the value returned is 0.0 and the status is `OFF`.

Response: <numeric floating point>, <character data>
where: <character data> OFF or ON

READ:LAST:MINimum?

returns the minimum limit value associated with the last measurement performed, and whether the minimum limit was enabled or disabled.

If the minimum limit was disabled for the measurement, or if no limit or measurement value is defined, the value returned is 0.0 and the status is `OFF`.

Response: <numeric floating point>, <character data>
where: <character data> OFF or ON

Example: MEAS:DEV:SAT?
1734
READ:LAST:MIN?

1800,ON
indicates that the minimum SAT deviation is 1800 Hz, the minimum limit was enabled, therefore the measurement failed.

READ:LAST:NAME?

returns the name associated with the last measurement performed.

Response: <ASCII string up to 32 chars>

Example: MEAS:DEV:SAT?
READ:LAST:NAME?
"SAT Deviation"

READ:LAST:NOMinal?

returns the nominal limit value associated with the last measurement performed.

If no limit or measurement value is defined, the value returned is 0.0.

Response: <numeric floating point>

READ:LAST:RESolution?

returns the resolution of the measured value of the last measurement performed.

The measured value is read using `READ:LAST[:DATA]?`.

Response: <numeric floating point>

Example: MEAS:VOLT:DC?
11.8
READ:LAST:RES?
0.01
indicates a DC voltage measurement of 11.8 volts with a resolution of 0.01 volts.

READ:LAST:STATus?

returns the pass/fail status of the last measurement performed.

Response: <numeric integer>
where:0 = passed
1 = failed
2 = measurement aborted by the user
3 = not tested (limits were not enabled)

READ:LAST:UNITs?

returns the units associated with the value of the last measurement performed.

Read the measured value using `READ:LAST[:DATA]?`.

If no units were associated with the measurement (and when there is no measurement value), `NONE` is returned.

A list of the unit identifiers returned along with the type of measurements each is associated with includes:

"AMP"	current
"DB"	SINAD, power droop
"DBC"	vector error origin offset
"DBM"	power, sensitivity
"DBW"	power
"DEG"	vector error phase
"HZ"	frequency, deviation
"KHZ"	frequency
"MHZ"	frequency
"MS"	time, duration
"MW"	power
"PCT"	vector error magnitude, BER, FER
"VOLT"	voltage
"WATT"	power

Response: <ASCII string up to 12 characters>

READ:LAST:VALid?

returns a flag to indicate if the last measurement performed was valid.

When a measurement begins, this value is set to 0 (indicating that the measurement value is not valid).

When a measurement completes, conditions are checked to determine if the measurement value is accurate.

If there are no questionable conditions, this value is set to a 1 (indicating that the data is ready to be read).

If a questionable condition exists, use `STATus:QUEStionable` to identify the cause of the measurement inaccuracy.

Response: <numeric integer, 0 or 1>
where:0 = invalid
1 = valid

READ:POWer:TRANsmitter:APRobe[:DATA][:IMMediate]?

returns the power measurements of the probes in the last CDMA access probe power tests.

When run remotely, this command displays all of the access probe power readings collected in the last test.

Only the samples defined by `READ:POW:TRAN:APRPRObes?` are valid.

Samples are listed in chronological order.

Response: 240 floating point entries, comma-separated

Measured by: MEAS:POW:TRAN:APR

READ:POWer:TRANsmitter:APRobe[:DATA]:TIMe?

returns the relative times of the probes collected during the last CDMA access probe power test.

Response: <numeric floating point>

Resolution: 1

Units: milliseconds

Measured by: MEAS:POW:TRAN:APR

READ:POWer:TRANsmitter:APRobe:INITial

reads the absolute power of the initial probe in order to determine if the last access probe power test failed.

See `MEAS:POW:TRAN:UNIT`.

This probe is collected before the actual test is run and is not part of the array of access probe powers.

Response: <numeric floating point>

Resolution: 0.1 dBm, 0.1 mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN:APR

READ:POWer:TRANsmitter:APRobe:LAST

reads the absolute power of the last access probe power test and is also set whenever an access probe is received from the mobile.

See `MEAS:POW:TRAN:UNIT`.

Response: <numeric floating point>

Resolution **:** 0.1 dBW/dBm, 0.1 mW

Units: dBW, dBm, Watt, or mW

READ:POWer:TRANsmitter:APRobe:PROBes?

returns the number of probes received in the last access probe power test.

Response: <numeric integer>

Resolution **:** 1

Units: none

Measured by: MEAS:POW:TRAN:APR

READ:POWer:TRANsmitter:APRobe:RESult?

returns the last CDMA access probe power pass/fail result set by `MEAS:POW:TRAN:APR`.

Response: PASS or FAIL

Measured by: MEAS:POW:TRAN:APR

READ:POWer:TRANsmitter:APRobe:SAMPle[:IMMediate] <index>
<index> numeric integer from 1 - 240.

sets the sample index if no sample index value is supplied when `READ:POWer:TRANsmitter:APRobe:SAMPle:TIMe?` or `READ:POWer:TRANsmitter:APRobe:SAMPle:POWer?` is run.

The sample values are listed in chronological order, starting with 1 and ending with the value returned by `READ:POW:TRAN:PROBes?`.

Response: <numeric integer>

Resolution: 1

Units: none

Measured by: none

READ:POWer:TRANsmitter:APRobe:SAMPle:POWer [sample index]
<sample index> (optional) numeric integer from 1 - 240.

reads the relative power level of a specific access probe.

If no sample index is given, the value set in the last `READ:POWer:TRANsmitter:APRobe:SAMPle[:IMMediate]` command is used.

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Measured by: MEAS:POW:TRAN:APR

READ:POWer:TRANsmitter:APRobe:SAMPle:TIMe [sample index]

<sample index> numeric integer from 1 - 240.

reads the relative time of a specific access probe.

If no sample index is given, the value set in the last `READ:POWer:TRANsmitter:APRobe:SAMPle[:IMMediate]` command is used.

Response: <numeric floating point>

Resolution: 1

Units: milliseconds

Measured by: MEAS:POW:TRAN:APR

READ:POWer:TRANsmitter:CLOSed[:DATA][:IMMediate]?

returns the last CDMA open loop transmitter power measurements taken using `MEAS:POW:TRAN:CLOS[:IMM]?`.

When run remotely, this command displays all of the closed loop power readings collected in the last test.

Response: 200 numeric floating point entries, comma-separated.

Resolution: 0.1

Units: dB

READ:POWer:TRANsmitter:CLOSed[:DATA]:SAMPle? <sample index>

<sample index> numeric integer value from 1 - 200

reads the relative power level at the specific interval in a closed loop power timing measurement.

The sample time depends on the current rate being used

full rate	1.25 msec
half rate	2.5 msec
quarter rate	5 msec
eighth rate	10 msec

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Measured by: MEAS:POW:TRAN:CLOS:INIT?
MEAS:POW:TRAN:CLOS[:IMM]?

READ:POWer:TRANsmitter:CLOSed:INITial?

reads the last CDMA initial closed loop transmitter power value set by `MEAS:POW:TRAN:CLOS:INIT?`, in units selected by `MEAS:POW:TRAN:UNIT`.

See `MEAS:POW:TRAN:UNIT`.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN:CLOS:INIT?
MEAS:POW:TRAN:CLOS[:IMM]?
MEAS:POW:TRAN:CLOS:MAX
MEAS:POW:TRAN:CLOS:MIN
MEAS:POW:TRAN:CLOS:RATE:FALL
MEAS:POW:TRAN:CLOS:RATE:RISE

READ:POWer:TRANsmitter:CLOSed:MAXimum?

reads the last CDMA maximum closed loop transmitter power value set by `MEAS:POW:TRAN:CLOS:MAX?`.

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Measured by: MEAS:POW:TRAN:CLOS:INIT?
MEAS:POW:TRAN:CLOS[:IMM]?
MEAS:POW:TRAN:CLOS:MAX
MEAS:POW:TRAN:CLOS:MIN
MEAS:POW:TRAN:CLOS:RATE:FALL
MEAS:POW:TRAN:CLOS:RATE:RISE

READ:POWer:TRANsmitter:CLOSed:MINimum?

reads the last CDMA minimum closed loop transmitter power value set by `MEAS:POW:TRAN:CLOS:MIN?`.

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Measured by: MEAS:POW:TRAN:CLOS:INIT?
MEAS:POW:TRAN:CLOS[:IMM]?

MEAS:POW:TRAN:CLOS:MAX
MEAS:POW:TRAN:CLOS:MIN
MEAS:POW:TRAN:CLOS:RATE:FALL
MEAS:POW:TRAN:CLOS:RATE:RISE

READ:POWer:TRANsmitter:CLOSed:RATE:FALL?

reads the last CDMA rate of fall value set by `MEAS:POW:TRAN:CLOS:RATE:FALL?`.

Response: <numeric floating point>

Resolution: 0.1

Units: dB/20msec

Measured by: MEAS:POW:TRAN:CLOS:INIT?
MEAS:POW:TRAN:CLOS[:IMM]?
MEAS:POW:TRAN:CLOS:MAX
MEAS:POW:TRAN:CLOS:MIN
MEAS:POW:TRAN:CLOS:RATE:FALL
MEAS:POW:TRAN:CLOS:RATE:RISE

READ:POWer:TRANsmitter:CLOSed:RATE:RISE?

reads the last CDMA rate of fall value set by `MEAS:POW:TRAN:CLOS:RATE:RISE?`.

Response: <numeric floating point>

Resolution: 0.1

Units: dB/20msec

Measured by: MEAS:POW:TRAN:CLOS:INIT?
MEAS:POW:TRAN:CLOS[:IMM]?
MEAS:POW:TRAN:CLOS:MAX
MEAS:POW:TRAN:CLOS:MIN
MEAS:POW:TRAN:CLOS:RATE:FALL
MEAS:POW:TRAN:CLOS:RATE:RISE

READ:POWer:TRANsmitter:EXPected:ERRor?

returns the difference in dB between the last CDMA average power measurement and the expected mobile power, and assumes that an average power measurement was made while on a CDMA traffic channel.

The expected power is the mobile station's estimated open loop output power (based on the input power that the mobile receives).

The difference between the measured and expected mobile power indicates an offset in the power control of the mobile when the power control mode of the 4300 is set to alternating or active.

See `CALL:PCONtrol:SELect`.

When the power measurement is performed, the expected power at the time of the measurement is saved.

Response: <numeric floating point>

Resolution: 0.1

Units: dB

Example: SOUR:POW:LEV –70
MEAS:POW:TRAN:UNIT DBM; IMM?
9.9
READ:POW:TRAN:EXP:ERR?
12.9
indicates a mobile power measurement on a CDMA traffic channel with the base power set to –70 dBm.

The mobile power measurement returned a value of 9.9 dBm and the power offset of this measurement from the expected power is 12.9 dB.

The expected power for this measurement was 9.9 – 12.9 = -3.0 dBm and the mobile power is greater than 8 dB from expected.

Correct this by issuing CALL:PCONtrol:RESet, if the current power control mode is set to alternating or active.

READ:POWer:TRANsmitter:GATed:AVERage?

returns the last CDMA average gated transmitter power measurement value set by `MEAS:POW:TRAN:GAT[:IMM]?` or `MEAS:POW:TRAN:GAT:AVER?`, in units selected by `MEAS:POW:TRAN:UNIT`.

See `MEAS:POW:TRAN:UNIT`.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN:GAT:AVER:DROOp?
MEAS:POW:TRAN:GAT[:IMM]?
MEAS:POW:TRAN:GAT:PEAK

READ:POWer:TRANsmitter:GATed[:DATA][:IMMediate]?

returns the last CDMA gated transmitter power measurements taken by `MEAS:POW:TRAN:GAT[:IMM]?`.

Response: 26640 floating point entries, comma-separated; samples taken at 0.4069 microsecond intervals.

Resolution: 0.1

Units: dBm

Measured by: MEAS:POW:TRAN:GAT:AVER:
MEAS:POW:TRAN:GAT:DROOp?
MEAS:POW:TRAN:GAT[:IMM]?
MEAS:POW:TRAN:GAT:PEAK

READ:POWer:TRANsmitter:GATed[:DATA]:SAMPle? <sample index>
<sample index> numeric integer value from 1 - 6660

reads the power level at the specific interval in a gated power measurement with intervals sampled every 0.4069 msec.

For RC 3-5, the maximum sample index = 6660; for all other RC values, the maximum sample index = 3330.

See `MEAS:POW:TRAN:UNIT`.

Response: <numeric floating point>

Resolution: 0.1 dBW/dBm or 0.1mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN:GAT[:IMM]?
MEAS:POW:TRAN:GAT:AVER?
MEAS:POW:TRAN:GAT:PEAK?
MEAS:POW:TRAN:GAT:DROO?

READ:POWer:TRANsmitter:GATed:PEAK?

returns the last CDMA peak gated transmitter power measurements taken by `MEAS:POW:TRAN:GAT[:IMM]?` or `MEAS:POW:TRAN:GAT:PEAK?`, in units selected by `MEAS:POW:TRAN:UNIT`.

See `MEAS:POW:TRAN:UNIT.`

Response: <numeric floating point>

Resolution: 0.1 dBW/dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN:GAT[:IMM]?
MEAS:POW:TRAN:GAT:AVER?
MEAS:POW:TRAN:GAT:PEAK?
MEAS:POW:TRAN:GAT:DROO?

READ:POWer:TRANsmitter[:IMMediate]?

returns the last mobile transmitter power measurement value set by `MEAS:POW:TRAN`, in units selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

Response <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN[:IMM]

READ:POWer:TRANsmitter:MAXimum?

returns the last CDMA maximum transmitter power measurement value set by `MEAS:POW:TRAN:MAX?`, in units selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN:MAX?

READ:POWer:TRANsmitter:MINimum?

returns the last CDMA minimum transmitter power measurement value set by `MEAS:POW:TRAN:MIN?`, in units selected by `MEAS:POW:TRAN:UNIT`.

See MEAS:POW:TRAN:UNIT.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, mW

Measured by: MEAS:POW:TRAN:MIN?

READ:POWer:TRANsmitter:OPEN[:DATA][:IMMediate]?

returns the last CDMA open loop transmitter power measurements taken by `MEAS:POW:TRAN:OPEN[:IMM]?`.

Response: 500 floating point entries, comma-separated; samples taken at 0.2 ms intervals.

Resolution: .1

Units: dBm

Measured by: MEAS:POW:TRAN:OPEN[:IMM]?
MEAS:POW:TRAN:OPEN:INITial?

READ:POWer:TRANsmitter:OPEN[:DATA]:SAMPle? <sample index>
<sample index> numeric integer value from 1 to 500

reads the power level at the specific interval in an open loop power measurement with intervals sampled at 0.2 ms.

See `MEAS:POW:TRAN:UNIT`.

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, mW

Measured by: MEAS:POW:TRAN:OPEN[:IMM]?
MEAS:POW:TRAN:OPEN:INITial?

READ:POWer:TRANsmitter:OPEN:INItial?

returns the last CDMA initial open loop transmitter power value in units selected by `MEAS:POW:TRAN:UNIT`.

See `MEAS:POW:TRAN:UNIT`.

Response: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN:OPEN:INI?
MEAS:POW:TRAN:OPEN[:IMM]?

READ:POWer:TRANsmitter:STANdby?

returns the last CDMA standby transmitter power measurement value set by `MEAS:POW:TRAN:STAN?`, in units selected by `MEAS:POW:TRAN:UNIT`.

See `MEAS:POW:TRAN:UNIT` .

Response: <numeric floating point>

Resolution: 0.1 dBm or 0.1 mW

Units: dBW, dBm, Watt, or mW

Measured by: MEAS:POW:TRAN:STAN?

READ:TIME:ERRor?

returns the last time error estimate value set by `MEAS:TIME:ERR?`.

Response: <numeric floating point>

Resolution: 0.001

Units: milliseconds

Measured by: MEAS:TIME:ERR?

READ:VERRor:IQ[:DATA][:IMMEDIATE]?

responds with a string of 1280 comma-separated entries.

<I for symbol 1>, <I for symbol 2>, <I for symbol 3>,... <I for symbol 640>, <Q for symbol 1>, <Q for symbol 2>,... <Q for symbol 640>

Resolution: 0.001

Units: percentage

Limit checking: none

Example: 1.0, 0.0, 0.707,...
I for symbol 1 = 1.0
I for symbol 2 = 0.0
I for symbol 3 = 0.707

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:IQ[:DATA]:SAMPle?<sample index>
<symbol> numeric integer value from 1 - 640

returns the I/Q position of a selected symbol for the last vector error measurement performed.

Printer access to the I and Q values and name of this measurement for the selected symbol are available following this command.

Limit checking is not performed on this parameter.

Response: <numeric floating point>, <numeric floating point>

Resolution: 0.001

Units: none

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:IQIMbalance?

returns the last I/Q imbalance value set by `MEAS:VERR:IQIM?`.

Response: <numeric floating point>

Resolution: 0.1

Units: dBc

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:MAGNitude:PEAK?

returns the peak magnitude component of vector errors for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:MAGNitude:RMS?

returns the RMS of the magnitude component of vector errors for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:OOFFset?

returns the origin offset for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: dBc

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:PEAK?

returns the peak error vector for the last single-burst vector error measurement performed.

Peak error vector is also known as the peak EVM (Error Vector Magnitude).

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:IQIM?
MEAS:VERR:OOFF?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:PHASe:PEAK?

returns the peak phase component of vector errors for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.01

Units: degrees

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:PHASe:RMS?

returns the RMS of the phase component of vector errors for the last single-burst vector error measurement performed.

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.01

Units: degrees

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:VERRor:RMS?

returns the RMS vector error for the last single-burst vector error measurement performed.

RMS vector error is also known as the RMS EVM (Error Vector Magnitude).

Symbols 6 - 162 are measured in the burst.

Sequencer and printer access to the pass/fail status, limit values, units, and name of measurement of this parameter are available in the same way as when the measurement was performed.

Response: <numeric floating point>

Resolution: 0.1

Units: percentage

Measured by: MEAS:VERR:RMS?
MEAS:VERR:PEAK?
MEAS:VERR:OOFF?
MEAS:VERR:IQIM?
MEAS:VERR:MAGN:RMS?
MEAS:VERR:MAGN:PEAK?
MEAS:VERR:PHAS:RMS?
MEAS:VERR:PHAS:PEAK?
MEAS:VERR:IQ?

READ:WQUality?

returns the last Waveform QUality (Rho) value set by `MEAS:WQU?`.

Response: <numeric floating point>

Resolution: 0.0001

Units: none

Measured by: MEAS:WQU?

SOURce Subsystem

The *Source* subsystem commands affect the signal generation sources of the test equipment.

SOURce:AUDio:DEViation <deviation value>
<deviation value> numeric value from 0.0 - 25575.0 Hz (resolution of 25)

sets the amount of deviation in the output to the mobile due to the audio generator source.

This parameter is also expressed as the amplitude (level) of the audio generator level.

Setting this deviation value causes a corresponding change in the `SOUR:AUD:LEVel` parameter.

See `SOUR: AUD:LEVel`.

Units: Hz, KHz, MHz (default = Hz)

Example 1: SOUR:AUD:DEV 1.27 KHZ
sets the audio deviation to 1270 Hz.

Example **2:** SOUR:AUD:DEV 15000
sets the audio deviation to 15000 Hz.

SOURce:AUDio:FREQuency <frequency value>
<frequency value> numeric integer value from 1 - 100000 Hz

sets the frequency of the internal audio generator.

Units: Hz

Example: SOUR:AUD:FREQ 1000
sets the internal audio frequency to 1000 Hz.

SOURce:AUDio:LEVel <amplitude>
<amplitude> numeric value from 0.0 - 7989.0 mV (resolution of 7.81)

sets the amplitude level of the audio generator output to the Audio Out jack on the front panel.

This parameter can also be expressed as the amount of deviation in the output to the mobile due to the audio generator source.

Setting this amplitude value causes a corresponding change in the `SOUR:AUD:DEViation` parameter.

The relationship is expressed by: `DEV` = `LEV` x 3.201.

See `SOUR:AUD:DEViation.`

Units: mV

Example: SOUR:AUD:LEV 1600
sets the internal audio generator to an amplitude of 100mV.

SOURce:AUDio[:SOURce]:EXTernal <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the external audio source.

SOURce:AUDio[:SOURce]:INTernal <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the internal audio source.

SOURce:AWGN:LEVel <delta amplitude>
<delta amplitude> numeric value from -25.0 to 5.0 dB (resolution of 0.1)

sets the power level of the AWGN present on the RF Channel.

The AWGN level is relative to the CDMA sector 1 RF power level.

This command is valid only if the CCM option is installed.

SOURce:AWGN:STATe <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the AWGN present on the RF Channel.

This command is valid only if the CCM option is installed.

SOURce:CDMA:OCNS1:ECIO?
SOURce:CDMA:OCNS2:ECIO?

returns the ratio of Orthogonal Channel Noise Source (OCNS) channel energy to total power (Ec/Io) expressed as dB.

Response: <numeric float>

Resolution: .01

Units: dB

SOURce:CDMA:OCNS1[:LEVel]?
SOURce:CDMA:OCNS2[:LEVel]?

reads the power level of the OCNS (additional traffic channels in the sector) in dB relative to sector power.

This query command returns the ratio of traffic channel energy to total power (Ec/Io) expressed as dB.

These commands are valid only if the CCM option is installed.

Response: <numeric value>

Resolution: .1

Units: dB

SOURce:CDMA:PAGing:ECIO?

returns the ratio of paging channel energy to total power (Ec/Io) expressed as dB.

Response: <numeric value>

Resolution: .1

Units: dB

SOURce:CDMA:PAGing:LEVel <delta amplitude>
<delta amplitude> numeric value from -32.0 to -5.0 dB (resolution of 0.1)

sets the power level of the CDMA paging channel (sector 1 only) relative to the CDMA sector 1 RF power level.

The paging channel power level range is 5.0 to 32.0 dB below the CDMA sector RF power level.

This command is only valid if the CCM option is installed.

The query form of this command returns the ratio of sync channel energy to total power (Ec/Io) expressed as dB.

SOURce:CDMA:PILot1:ECIO?
SOURce:CDMA:PILot2:ECIO?

returns the ratio of pilot channel energy to total power (Ec/Io) expressed as dB.

Response: <numeric value>

Resolution: .1

Units: dB

SOURce:CDMA:PILot1:LEVel <delta amplitude>
SOURce:CDMA:PILot2:LEVel <delta amplitude>
<delta amplitude> numeric value from -32.0 to -5.0 dB (resolution of 0.1)

sets the power level of the first and second CDMA sector pilot channels.

The levels are relative to the CDMA sector 1 and sector 2 RF power levels for pilot 1 and pilot 2, respectively.

The pilot channel level can not exceed the RF power level of its sector. The pilot channel level can be no more than 32.0 dB less than the RF power level of its sector.

This command is valid only if the CCM option is installed.

SOURce:CDMA:QPCHannel:ECIO?

specifies the Ec/Io of the quick paging channel when the protocol revision is >6.

Response: <floating point numeric value>

Units: dB in 0.1 steps

Example: SOUR:CDMA:ECIO?
17.2

This command is valid only if the CDMA2000 option is installed.

SOURce:CDMA:QPCHannel[:LEVel] <dBLevel>
<dBLevel> -32.0 to -5.0dB in 0.1 steps

specifies the relative power of the quick paging channel when the protocol revision is >6.

Response: <floating point numeric value>

Units: dB

Example: SOUR:CDM;LQPCH -15.0

This command is valid only if the CDMA2000 option is installed.

SOURce:CDMA:ROSCillator:ESEC and SOURce <direction>
<direction> INTernal or EXTernal

selects the direction of the even second clock reference signal at the rear panel connector.

If `INTernal` is selected, the internally generated even second clock is used for synchronization and output to the rear panel connector for synchronizing other equipment.

If `EXTernal` is selected, connect an external even second source to the rear panel connector; the 4300 uses the signal for synchronization.

The even second clock generated produces a pulse bandwidth of 25 and 75 nanoseconds every 2 seconds.

The externally-supplied even second clock pulse width range is 25 nanoseconds to >100 msec.

SOURce:CDMA:ROSCillator:OUTPUT1 <select>
SOURce:CDMA:ROSCillator:OUTPUT2 <select>
<select> CHIPx1, CHIPx4, CHIPx16, PCONtrol1,PNCLock, FRAMe, or SFRame

sets the reference signal selection that is output to the rear panel connector.

Two signals (out of the eight selections) can be simultaneously output.

The `CHIPx1 - CHIPx16` selections are based on the chip clock frequency of 1.2288 MHz.

`PCONtrol` is a 1.25 msec pulse for every power control group.

`FRAMe` is output at the start of every frame (20 msec).

`SFRame` is output every superframe (80 msec).

The `PNCLock` cycle time is 26.67 msec.

SOURce:CDMA:SSECtor:LEVel <delta amplitude>
<delta amplitude> numeric value from -12.0 to 3.0 dB (resolution of 0.1)

sets the power level of the second CDMA sector on the RF output to a given value relative to the first CDMA sector power.

The second sector power can exceed the first sector level by 3.0 dB, and can be no more than 12 dB less than it.

This command is valid only if the CCM option is installed.

SOURce:CDMA:SSECtor:STATe <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

enables or disables the second CDMA sector.

This command is only valid if the CCM option is installed.

The PN offset for the second sector is a fixed offset of 12 from the primary CDMA control channel. The second sector consists of a pilot channel, a traffic channel, and two orthogonal channel noise source (OCNS) channels.

SOURce:CDMA:SSECtor:TCODE <channel code>
<channel code> numeric value from 2 - 31 or 33 - 63.

sets the traffic channel code selection of the second CDMA channel.

This command is valid only if the CCM option is installed.

SOURce:CDMA:SYNC:ECIO?

returns the ratio of sync channel energy to total power (Ec/Io) expressed as dB.

Response: <numeric value>

Resolution: .1

Units: dB

SOURce:CDMA:SYNC:LEVel <delta amplitude>
<delta amplitude> numeric value from -32.0 to -5.0 dB with a resolution of 0.1

sets the power level of the CDMA sync channel (sector 1 only) relative to the CDMA sector 1 RF power level.

The sync channel power level range is 5.0 - 32.0 dB below the CDMA sector RF power level.

This command is valid only if the CCM option is installed.

The query form of this command returns the ratio of pilot channel energy to total power (Ec/Io) expressed as dB.

SOURce:CDMA:TRAFfic1:EBNT?
SOURce:CDMA:TRAFfic2:EBNT?

queries the Eb/Nt calculation results for sector 1 and sector 2 traffic channels.

The Eb/Nt calculation results indicate the signal-to-noise ratio for the traffic channel as perceived by the mobile, and is used in the FER measurements, defined in IS-98.

This is equivalent to the traffic Ec/Io value in dB (`SOURCe:CDMA:TRAFfic1:LEVel)` minus the AWGN Ioc/Ior level in dB (`SOURce:AWGN:LEVel`) plus the code gain in dB.

The code gain is the traffic channel PN chips per bit expressed as dB and is both rate set and data rate dependant. See Table 35.

The Eb/Nt value returned is only valid when on a CDMA call in loopback service option with AWGN turned on.

The AWGN on/off status is ignored by this calculation, it assumes the AWG is on.

When on a call in voice service option, full rate is assumed.

If not on a call, full rate and rate set 1 (9600) are assumed. See Table 35. .

TABLE 35. Code Gain Values for Data Entry Setting

Data Rate	Rate set 1	Rate set 2
Full	21.07 dB	19.31 dB
Half	24.08 dB	22.32 dB

TABLE 35. Code Gain Values for Data Entry Setting

Data Rate	Rate set 1	Rate set 2
Quarter	27.09 dB	25.33 dB
Eighth	30.10 dB	28.34 dB

SOURce:CDMA:TRAFfic1:ECIO?
SOURce:CDMA:TRAFfic2:ECIO?

queries the traffic Ec/Io values for sectors 1 and 2.

Ec/Io is the ratio of the average transmit energy per PN chip for the forward traffic channel to the total received power spectral density at the mobile station antenna.

This is similar to the Ec/Ior value, except the ratio is the percentage of the traffic channel to the total power transmitted by the 4300.

See `SOURce:CDMA:TRAFfic1:LEVel`.

This consists of the sector 1 and sector 2 powers plus the Additive White Gaussian Noise (AWGN) power. Ec/Io equals Ec/Ior for sector 1 if both sector 2 and AWGN are turned off. Since the traffic channel power level is dependant on the data rate when on a CDMA call in loopback service option, the Ec/Io value returned reflects the current data rate.

When not on a call or on a CDMA call in a voice service option, the 4300 assumes the traffic channel is at full rate.

Response: <numeric value>

Resolution: .1

Units: dB

SOURce:CDMA:TRAFfic1[:LEVel]<delta amplitude>
SOURce:CDMA:TRAFfic2[:LEVel]<delta amplitude>

<delta amplitude>numeric value -41.0 to –5.0 (data rate dependant) in 0.1 dB steps

sets the relative power level of the traffic channel for sector 1 and sector 2.

The levels define the Ec/Ior value, which is defined as the ratio of the average transmit energy per PN chip for the forward traffic channel to the total transmit power spectral density for the sector.

This is equivalent to the percentage of the total sector 1 or sector 2 power that is dedicated to the traffic channel, expressed in dB.

The transmitted power level is changed automatically by the hardware when the traffic channel data rate changes in order to keep the effective power to the mobile constant. (Effective power includes gain due to code repetition). This causes the actual transmitted power for the traffic channel to be dependant on its data rate, set by `CALL:SERVice:RATE`.

This data rate selection is only meaningful when on a CDMA call in loopback service option, when the 4300 is controlling the transmitted data rate. When not on a call or on a CDMA call in a voice service option, the 4300 assumes the traffic channel is at full rate. The actual rate is determined by the mobile's vocoder. See Table 35.

This command is valid only if the CCM option is installed.

See Table 36 for loopback service option calls range limits at each data rate selection.

TABLE 36. Parameter range limits for SOURce:CDMA:TRAFfic*n*[:LEVel]

Data Rate	Minimum level	Maximum level
Full	-5.0	-32.0
Half	-8.0	-35.0
Quarter	-11.0	-38.0
Eighth	-14.0	-41.0

SOURce:CDMA:TRAFfic1:RATE?
SOURce:CDMA:TRAFfic2:RATE?

queries the data rate specified for traffic channel levels.

This value is the same as the data rate specified by `CALL:SERVice:RATE`, if mobile is on a CDMA call in a loopback service option, otherwise, the value is full.

SOURce:POWer:LEVel[:IMMediate] <amplitude>
<amplitude> numeric value from -125.0 to 10.0 dBm (resolution of 0.1)

sets the power level of the RF output carrier to a given value in dBm.

This can also be accessed through the front panel base power level increment and decrement hard keys.

This command allows the user to set the output power to a higher level than the maximum output of -23.00 dBm, and then adjusts the output to the highest available power.

Units: dBm

Example: SOUR:POW:LEV:IMM -72.5
sets the output power level to -72.5 dBm.

SOURce:POWer:LEVel:MAXimum?

query-only command which reports the maximum allowable base power.

SOURce:POWer:LEVel:MINimum?

query-only command which reports the minimum allowable base power.

SOURce:POWer:STATe <boolean>
<boolean> OFF or ON (0 = OFF, 1 = ON)

turns the RF output to the front panel jack on or off.

To turn off both carrier and modulation, either set the RF output to `OFF` or, toggle the base power switch on the front panel on and off.

STATus Subsystem

The *Status* subsystem commands control the reporting of test equipment status conditions.

Several layers of status registers can be accessed. Some are at the lowest level and detect specific conditions, others are more general, grouping many conditions into a single status event by allowing all event registers to report a summary condition (single event) to the next level up register.

The summary condition detects only changes in register conditions and selects whether a positive-going or negative-going change (or neither) is of interest.

All status registers report a summary bit to a parent register, resulting in a final status byte register (accessed by the `*STB?` query as well as a serial poll).

For GPIB remote operation, the status byte register issues a service request to the controller, indicating when a selected event(s) occurs.

For each grouping of status events, five registers exist which conform to IEEE-488.2 specifications and include: a condition register, an event register, an enable register, and positive and negative transition filters.

These groups report to each other in a hierarchical fashion; the upper level registers indicate general conditions, and the register groups that report to them indicate specific conditions.

There are three basic groups of status registers:

- event status which are defined by IEEE 488.2 (includes overlapped command complete status)
- operational status which report normal operating conditions of the instrument
- questionable status which report possible error conditions that have occurred in the instrument.

See figure 5-3, “Reporting Structure for each Register Group" and table 37, Bit Definitions for Register Groups* .

FIGURE 5-3. Reporting Structure for each Register Group

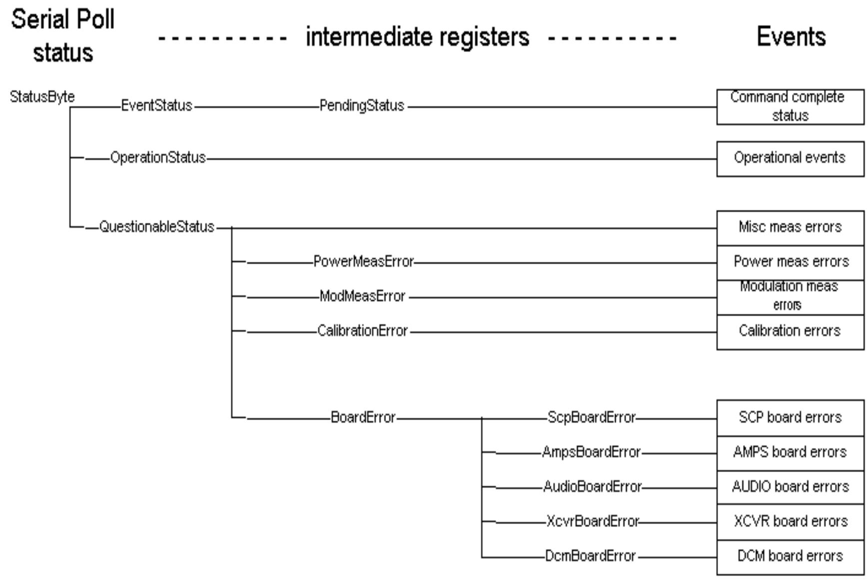


TABLE 37. Bit Definitions for Register Groups*

Register Name	Bit Pos	(Value)	Bit Description
Status Byte	bit 0	(dec 1)	(not used)
	bit 1	(dec 2)	(not used)
	bit 2	(dec 4)	Error queue contains error messages
	bit 3	(dec 8)	summary: **QuestionableStatus**
	bit 4	(dec 16)	Response message available for reading
	bit 5	(dec 32)	summary: **EventStatus**
	bit 6	(dec 64)	summary: **StatusByte** (for SRQ output)
	bit 7	(dec 128)	summary: **OperationStatus**
EventStatus	bit 0	(dec 1)	OPC (from **PendingStatus** when *OPC received)
	bit 1	(dec 2)	Request control (not used)
	bit 2	(dec 4)	Query error (error codes -400 to -499)
	bit 3	(dec 8)	Device error (error codes -300 to -399)
	bit 4	(dec 16)	Execution error (error codes -200 to -299)
	bit 5	(dec 32)	Command error (error codes -100 to -199)
	bit 6	(dec 64)	User request (front panel exit remote key)

TABLE 37. Bit Definitions for Register Groups*

Register Name	Bit Pos	(Value)	Bit Description
	bit 7	(dec 128)	Power on
PendingStatus	bit 0	(dec 1)	Special Test command pending MEAS:DTMF MEAS:HFLash[REC][:IMM] PROgram:GRaph[:STATe]:RUN
PendingStatus (cont.)	bit 1	(dec 2)	Call Task command pending CALL:REG CALL:ORIG CALL:PAGE CALL:BREL CALL:MREL MEAS:POW:TRAN:APRobe[:IMM]
	bit 2	(dec 4)	Digital Task thread 1 command pending (not used)
	bit 3	(dec 8)	Digital Task thread 2 command pending MEAS:FER[:IMM]
	bit 4	(dec 16)	ADC Task command pending MEAS:DEV:PAUDio
	bit 5	(dec 32)	(not used)
	bit 6	(dec 64)	(not used)
	bit 7	(dec 128)	(not used)
	bit 8	(dec 256)	Program Sequence pending
	bit 9	(dec 512)	Printer command pending
	bit 10	(dec 1024)	Disk command pending
	bit 11- 15		(not used)
OperationStatus	bit 0	(dec 1)	(not used)
	bit 1	(dec 2)	(not used)
	bit 2	(dec 4)	(not used)
	bit 3	(dec 8)	(not used)
	bit 4	(dec 16)	Measurement command in progress
	bit 5	(dec32)	(not used)
	bit 6	(dec 64)	power control reset completed
	bit 7	(dec 128)	power report received from mobile
	bit 8	(dec 256)	pilot strength report received from mobile
OperationStatus (cont.)	bit 9	(dec 512)	DTMF message received from mobile
	bit 10	(dec 1024)	Call processing command in progress
	bit 11- 15		(not used)

TABLE 37. Bit Definitions for Register Groups*

Register Name	Bit Pos	(Value)	Bit Description
QuestionableStatus	bit 0	(dec 1)	(not used)
	bit 1	(dec 2)	(not used)
	bit 2	(dec 4)	(not used)
	bit 3	(dec 8)	summary: **PowerMeasError**
	bit 4	(dec 16)	(not used)
	bit 5	(dec 32)	(not used)
	bit 6	(dec 64)	(not used)
	bit 7	(dec 128)	summary: **ModMeasError**
	bit 8	(dec 256)	summary: **CalibrationError**
	bit 9	(dec 512)	summary: **BoardError**
	bit 10	(dec 1024)	BER measurement accuracy questionable
	bit 11	(dec 2048)	SINAD measurement accuracy questionable
	bit 12 - 15		(not used)
CalibrationError	bit 0	(dec 1)	Calibration table checksum failure
		bit 1-15	(not used)
ModMeasError	bit 0	(dec 1)	Mobile ST ordered ON during manual mode
	bit 1	(dec 2)	(not used)
	bit 2		AWGN enabled in base station modulation
	bit 3		Sector 2 enabled in base station modulation
	bit 4-15		(not used)
Power Meas Error	bit 0	(dec 1)	Calibration table bad
	bit 1	(dec 2)	Tx power meas low w/ base power high
	bit 2		base power includes AWGN
	bit 3		closed loop power control applied
	bit 4-15		(not used)
BoardError	bit 0	(dec 1)	summary: **ScpBoardError**
	bit 1	(dec 2)	summary: **AmpsBoardError**
	bit 2	(dec 4)	summary: **CcmBoardError**
	bit 3	(dec 8)	summary: **AudioBoardError**
	bit 4	(dec 16)	summary: **XcvrBoardError**
	bit 5-15	(dec 1)	(not used)
ScpBoardError	bit 0		Generic failure
	bit 1	(dec 2)	Static RAM memory failure

TABLE 37. Bit Definitions for Register Groups*

Register Name	Bit Pos	(Value)	Bit Description
	bit 2	(dec 4)	Printer failure
	bit 3	(dec 8)	Disk drive failure
	bit 4	(dec 16)	Remote port failure
	bit 5-15		(not used)
AmpsBoardError	bit 0	(dec 1)	Generic failure
	bit 1- 15		(not used)
AudioBoardError	bit 0	(dec 1)	Generic failure
	bit 1- 15		(not used)
XcvrBoardError	bit 0	(dec 1)	Generic failure
	bit 1- 15		(not used)
CcmBoardError	bit 0	(dec 1)	Generic failure
	bit 1	(dec 2)	Restart failure
	bit 2	(dec 4)	Version ID failure
CcmBoardError (cont.)	bit 3	(dec 8)	Dual port RAM failure on SCP
	bit 4	(dec 16)	Dual port RAM failure on CCM
	bit 5	(dec 32)	Loopback diagnostics failure
	bit 6	(dec 64)	CCM is dead
	bit 7	(dec 128)	MIC checksum failure
	bit 8	(dec 256)	DSP checksum failure
	bit 9	(dec 512)	SW/HW mismatch (software does not support hardware)
	bit 10	(dec 512)	FPGA checksum failure
	bit 11	(dec 1024)	MIC RAM failure
	bit 12	(dec 4096)	DSP RAM failure
	bit 13	(dec 8192)	DSP health check failure (DSP is dead)
	bit 14–15		(not used)

* Multiple events can occur in any register, and their values are cumulative.

TABLE 37. Bit Definitions for Register Groups*

Register Name	Bit Pos	(Value)	Bit Description

Each of the status register groups defined in Table 37, with the exception of **StatusByte, EventStatus,** and **PendingStatus,** consist of 2 distinct read-only registers and 3 read-write mask registers (filters).

The status events presented to each register (either a specific event or the summary status from a child register set) are reported directly to a condition register which reflects the actual status of events at any given time.

The register status may be read at any time without affecting its own value, or the value of any other register.

The purpose of the event register is to store the occurrence of certain events, in order to guarantee that the event is captured.

When the condition register value changes, its change (not its value) is passed through the transition filters, capturing the specified event changes, to the event register.

The **StatusByte**, **EventStatus**, and **PendingStatus** register groups consist of a condition register and an enable register.

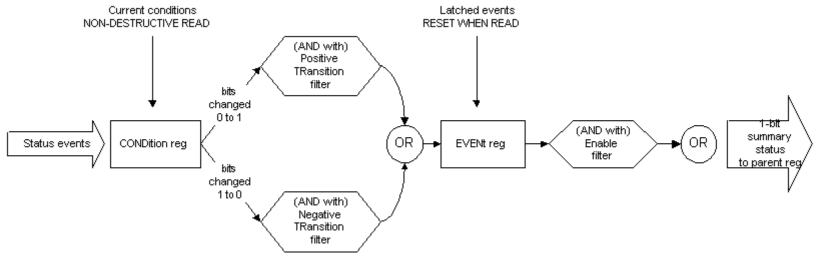
The enable register on the **PendingStatus** register group specifies which bits will set or clear an internal 1-bit flag **OpcStatus**. This result is passed on to the OPC bit in the **EventStatus** condition register if *OPC has been received. The **EventStatus** group is 8-bits instead of 16, and reports its summary bit to the **StatusByte** condition register if a binary ANDing of its condition and enable registers is non-zero.

The **StatusByte** group is also 8-bits. Bit 7 and bits 0 - 5 function like the other condition and enable register bits. Bit 6 is the summary bit for this register. It is set if the binary ANDing of the other 7 bits in the condition and enable registers is non-zero, otherwise it is cleared. Bit 6 of the enable register is used to enable/disable sending a service request (SRQ) to the controller based on the **StatusByte** summary bit value.

This function is implemented only for GPIB remote operation.

See figure 5-4, "GPIB Remote Operation".

FIGURE 5-4. GPIB Remote Operation



STATus:OPERation:COMPlete:CONDition?

returns a value that represents all overlapped processes that have been started, but have not yet completed.

Response: <numeric integer value> (range 0x0000 - 0x7FFF).

Example: STAT:OPER:COMP:COND?
48
indicates that both the DTMF and hook flash tests are in progress.

STATus:OPERation:COMPlete:ENABle <numeric value>

sets the mask to determine which overlapped processes are evaluated in determining an **PendingStatus** bit value.

Only those processes with corresponding bits enabled in this register, cause a change in the **PendingStatus** value.

When any enabled overlapped process starts, **PendingStatus** is set to 1 where it remains until all enabled overlapped processes have completed, at which point it is set to 0.

This value is used by three commands to pause processing of any more commands until pending commands have completed:

- the `*WAI` command waits until PendingStatus is 0 before executing the next command.
- the `*OPC` command causes the OPC bit in the EventStatus register to be set to 1 when the PendingStatus value is set to 0. The controller monitors this value to determine when the command has completed.
- the `*OPC?` command waits for the PendingStatus value to become 0, and will then place a "1" response message in the output queue. This causes the Message Available bit to become set in the StatusByte register. The controller monitors the status of this bit to determine when the command has completed.

`*OPC` and `*OPC?` are improved by performing a serial poll to query for status, or by issuing a service request to notify the controller when the command(s) have completed.

This requires GPIB, and cannot be accomplished using RS-232.

STATus:OPERation:COMPlete:ERRor?

returns the bit value of the sum of all of the overlapped commands that completed with an error.

These bits are initially 0 when the overlapped command begins, and are set when the command completes with an error.

*CLS and a device clear (GPIB only) also resets this register to 0.

Response: <numeric integer (decimal value)>

Example: STAT:OPER:COMP:ERR?
2
indicates that a CALL:ORIGination command failed.

STATus:OPERation:CONDition?

returns the value of the condition register associated with the 16-bit **OperationStatus** register group.

The condition register reflects the current status condition of the test equipment. The returned value is a decimal value that represents the sum of all current conditions.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus: OPERation:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **OperationStatus** summary bit should be set or cleared in the **StatusByte** register group's condition register.

STATus:OPERation[:EVENt]?

returns the value of the event register associated with the 16-bit **OperationStatus** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transitional filters determine the importance of events occurring (bit value changing from 0 to 1) or being cleared out (bit value changing from 1 to 0).

The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:OPERation:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **OperationStatus** register group to determines which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:OPERation:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **OperationStatus** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

See `STATus: OPERation:PTRansition` in the TDMA chapter for examples.

STATus:PRESet

sets all of the `STATus` subsystem enable and transition registers to a known state.

The values for STATus:OPERation and STATus:QUEStionable register groups are set as follows:

ENABle registers > 0 (off)
Positive TRanstion regs> 1 (on)
Negative TRanstion regs> 0 (off)

For all other register groups (those that report to STATus:QUEStionable) the values are set as follows:

ENABle registers > 1 (on)
Positive TRanstion regs > 1 (on)
Negative TRanstion regs> 0 (off)

STATus:QUEStionable:BOARd:AMPS:CONDition?

returns the value of the condition register associated with the 16-bit **AmpsBoardError** register group.

The condition register always reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:AMPS:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **AmpsBoardError** summary bit should be set or cleared in the **BoardError** register group's condition register.

STATus:QUEStionable:BOARd:AMPS [:EVENt]?

returns the value of the event register associated with the 16-bit **AmpsBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:AMPS:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **AmpsBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:AMPS:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **AmpsBoardError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:AUDio:CONDition?

returns the value of the condition register associated with the 16-bit **AudioBoardError** register group.

The condition register reflects the current status condition of the audio measurement board. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers is affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:AUDio:ENABle <enable>
<enable> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **AudioBoardError** summary bit should be set or cleared in the **BoardError** register group's condition register.

STATus:QUEStionable:BOARd:AUDio [:EVENt]?

returns the value of the event register associated with the 16-bit **AudioBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:AUDio:NTRansition <enable>
<enable> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **AudioBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:AUDio:PTRansition <enable>
<enable> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **AudioBoardError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:CCM:CONDition?

returns the value of the condition register associated with the 16-bit **CcmBoardError** register group.

The condition register reflects the current status condition of the CCM board. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:CCM:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **CcmBoardError** summary bit should be set or cleared in the **BoardError** register group's condition register.

STATus:QUEStionable:BOARd:CCM [:EVENt]?

returns the value of the event register associated with the 16-bit **CcmBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:CCM:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **CcmBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:CCM:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **CcmBoardError** register group to determines which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:CONDition?

returns the value of the condition register associated with the 16-bit **BoardError** register group.

The condition register reflects the current status condition of all the boards. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **BoardError** summary bit should be set or cleared in the **QuestionableStatus** register group's condition register.

STATus:QUEStionable:BOARd [:EVENt]?

returns the value of the event register associated with the 16-bit **BoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **BoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **BoardError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:SCP:CONDition?

returns the value of the condition register associated with the 16-bit **ScpBoardError** register group.

The condition register reflects the current status condition of the SCP board. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus: QUEStionable:BOARd:SCP:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **ScpBoardError** summary bit should be set or cleared in the **BoardError** register group's condition register.

STATus:QUEStionable:BOARd:SCP [:EVENt]?

returns the value of the event register associated with the 16-bit **ScpBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:SCP:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **ScpBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:SCP:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **ScpBoardError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:XCVR:CONDition?

returns the value of the condition register associated with the 16-bit **XcvrBoardError** register group.

The condition register reflects the current status condition of the transceiver board. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:BOARd:XCVR:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **BoardError** summary bit should be set or cleared in the **XcvrBoardError** register group's condition register.

STATus:QUEStionable:BOARd:XCVR [:EVENt]?

returns the value of the event register associated with the 16-bit **XcvrBoardError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:BOARd:XCVR:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **XcvrBoardError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:BOARd:XCVR:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **XcvrBoardError** register group. It determines which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:CALibration:CONDition?

returns the value of the condition register associated with the 16-bit **CalibrationError** register group.

The condition register reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus: QUEStionable:CALibration:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **CalibrationError** summary bit should be set or cleared in the **QuestionableStatus** register group's condition register.

STATus:QUEStionable:CALibration [:EVENt]?

returns the value of the event register associated with the 16-bit **CalibrationError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

STATus:QUEStionable:CALibration:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **CalibrationError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus: QUEStionable:CALibration:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **CalibrationError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:CONDition?

returns the value of the condition register associated with the 16-bit **QuestionableStatus** register group.

The condition register reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **QuestionableStatus** summary bit should be set or cleared in the **StatusByte** register group's condition register.

STATus:QUEStionable[:EVENt]?

returns the value of the event register associated with the 16-bit **QuestionableStatus** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:MODulation:CONDition?

returns the value of the condition register associated with the 16-bit **ModMeasError** register group.

The condition register reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus:QUEStionable:MODulation:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **ModMeasError** summary bit should be set or cleared in the **QuestionableStatus** register group's condition register.

STATus:QUEStionable:MODulation [:EVENt]?

returns the value of the event register associated with the 16-bit **ModMeasError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:MODulation:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **ModMeasError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:MODulation:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **ModMeasError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **QuestionableStatus** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus:QUEStionable:POWer:CONDition?

returns the value of the condition register associated with the 16-bit **PowerMeasError** register group.

The condition register reflects the current status condition of the test equipment. The value returned is a decimal value that represents the sum of all conditions that currently exist.

This query is non-destructive. None of the registers are affected as a result of this query.

Response: <numeric integer>

STATus: QUEStionable:POWer:ENABle <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the mask that is used on the event register to determine if the **PowerMeasError** summary bit should be set or cleared in the **QuestionableStatus** register group's condition register.

STATus:QUEStionable:POWer [:EVENt]?

returns the value of the event register associated with the 16-bit **PowerMeasError** register group.

The event register accumulates changes in the condition register as specified by the positive and negative transition filter registers.

When the condition register value changes, it is passed through the transition filters to determine if a modification to the event register is required.

The transition filters determine whether there is an interest in events occurring (bit value changing from 0 to 1) or events being cleared out (bit value changing from 1 to 0). The value returned is a decimal value that represents the sum of all events since the register was last read.

This is a self-destructive query command. The event register value is set to 0 after it has been read.

Response: <numeric integer>

STATus:QUEStionable:POWer:NTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the negative transition filter value for the **PowerMeasError** register group to determine which of the bits in the condition register, that change from 1 to 0, set the corresponding bit in the event register.

STATus: QUEStionable:POWer:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **PowerMeasError** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

STATus:QUEStionable:PTRansition <enable pattern>
<enable pattern> numeric integer value from 0 - 32767 (hexidecimal 7FFF)

sets the positive transition filter value for the **QuestionableStatus** register group to determine which of the bits in the condition register, that change from 0 to 1, set the corresponding bit in the event register.

Response: <numeric integer>

SYSTem Subsystem

The *System* subsystem commands, though not related to instrument performance, include general housekeeping functions and setting up global configurations (remote communication, time, date, etc.).

SYSTem:COMMunicate:GPIB:ADDRess <primary> [secondary]
<primary> numeric value 0 - 30
[secondary] numeric value 0 - 30 (optional)

sets up the GPIB address to be used.

The primary address, which selects both the talk and listen address of the instrument, is required.

The secondary address which is used only when the controller sends secondary addressing with the commands, is optional.

The secondary addressing is disabled on powerup, and whenever this command is received without the secondary address parameter, and enabled when this command is received with the secondary address specified.

Example 1: SYST:COMM:GPIB:ADDR 9
sets the GPIB address to 09, and disables secondary addressing.

Example 2: SYST:COMM:GPIB:ADDR 14, 23
sets the GPIB primary address to 14, the secondary address to 23, and enables secondary addressing.

SYSTem:COMMunicate:GPIB:TERMinator <terminator selection>
<terminator selection> CR, LF, or CRLF

sets the output terminator character selection to be used for the remote GPIB port.

This does not affect the remote serial terminator selection or the input termination characters for the GPIB port.

The input command terminates when any of the following is received: a carriage return character, a linefeed character, or an END signal (EOI asserted with the data).

The output terminator selection determines how response data (sent by the 4300 to the controller) is terminated.

An END bus signal accompanies the last character sent, regardless of the selection.

Example: SYST:COMM:GPIB:TERM CRLF
terminates all GPIB response messages with a carriage return followed by a linefeed which asserts the EOI.

SYSTem:COMMunicate:SERial:BAUD <baud rate>
<baud rate> numeric value (300, 600, 1200, 2400, 4800, 9600, or 19200)

sets the baud rate for the remote serial port to one of seven standard rates.

This selection is used for both receiving commands and transmitting responses.

SYSTem:COMMunicate:SERial:BITS <bits per character>
<bits per character> numeric value 7 or 8

sets the number of bits per character to be used for the remote serial port.

This selection is used for both receiving commands and transmitting responses.

SYSTem:COMMunicate:SERial:PACE <pace selection>
<pace selection> NONE or XON

enables or disables the `XON/XOFF` data flow control used for the remote serial port.

This selection is used for both receiving commands and transmitting responses.

When `NONE` is selected, the unit ignores the `XON` and `XOFF` characters if they are received, and does not issue either character to control data flow. If this mode is selected, the controller must implement some other way of preventing data loss by enabling echo and verifying the data character by character.

When `XON` is selected, the unit stops sending response characters after the `XOFF` character has been received, and does not continue until the `XON` character has been received.

The unit also sends an `XOFF` character if its input buffer is nearly full (less than 100 bytes available) and during disk drive accesses. This is to indicate to the remote controller device that the unit is unable to accept further input until it is allowed time to process the data.

An `XON` character is sent when more buffer space is available (200 bytes or more), and it is ready to accept more input.

The input buffer character length is 8192.

SYSTem:COMMunicate:SERial:PARity[:TYPE] <parity selection>
<parity selection> NONE, EVEN, ODD, MARK, or SPACe

sets the parity selection used for the remote serial port.

This selection is used for both receiving commands and transmitting responses.

SYSTem:COMMunicate:SERial:SBITs <stop bits>
<stop bits> numeric value 1 or 2

sets the number of stop bits to be used for the remote serial port.

This selection is used for both receiving commands and transmitting responses.

SYSTem:COMMunicate:SERial:TRANsmit:CREPort <status>
<status> OFF or ON (0= OFF, 1= ON)

enables or disables generation of a command report response on the serial port.

When enabled, if the command is a query that generates a response message without executing any errors, only the response message will be output

If the command does not generate a response message without executing without any errors, the command report response is:

`>OK`

If an error occurs in executing the command, the command report response is:

`>ERROR -XXX`

where: *XXX* is the 3-digit SCPI error code

See `SYSTem:ERRor?` for error code descriptions.

This command has no effect on GPIB responses.

SYSTem:COMMunicate:SERial:TRANsmit:ECHO <status>
<status> OFF or ON (0 = OFF, 1 = ON)

enables or disables echo of the input characters received to the serial port.

When enabled, the characters are output to the serial port as they are parsed.

This is different than echoing in that the data is contained in the input buffer waiting to be parsed.

Input terminator characters are converted to the current serial output terminator character selection prior to echoing which allows the user to treat the echo response as any other response on termination.

Echo is used as a data flow control method.

When enabled, the echo response is compared to the issued command to verify that it was correctly received.

This command has no effect on GPIB operation.

SYSTem:COMMunicate:SERial:TRANsmit:TERMinator <selection>
<selection> CR, LF, or CRLF

sets the output terminator character selection used for the remote serial port.

This does not affect the remote GPIB terminator selection or the input termination characters for the serial port.

The input command terminates when either a carriage return or a linefeed character is received. The output terminator selection determines how response data sent by the 4300 to the controller is terminated.

Example: SYST:COMM:SER:TRAN:TERM LF
terminates all serial response messages with a linefeed character.

SYSTem:COMMunicate:SOURce <remote select>
<remote select> SERial, GPIB, or AUTO

selects the source of the remote input.

If `SERial` is selected, the RS-232 port is used for remote communication and the GPIB port is disabled.

If `GPIB` is selected, the IEEE-488.2 connector is used and the RS-232 port is disabled.

If `AUTO` is selected, the first remote port (RS-232 or GPIB) that sends a command to the instrument following power up is used as the remote communication port. Once a port is selected in the `AUTO` mode, the other port is not re-selected if commands are issued to it.

Example: SYST:COMM:SOUR GPIB
selects the GPIB (IEEE488.2) for remote communication.

SYSTem:DATE[:DATA] <year>, <month>, <day>
<year> numeric value 1990 - 2089
<month> numeric value 1 - 12 (representing month of the year)
<day> numeric value 1 - 31 (representing day of month)

sets the current date for the real-time clock in the tester.

It is used when files are saved-to-disk and can be included in printouts.

SYSTem:DATE[:DATA]?

returns the current date read from the real-time clock in the tester.

The response is always year, month, day regardless of the format set by `SYSTem:DATE:FORMat`.

Response: <numeric integer>, <numeric integer>, <numeric integer>

The values represent year (1989- 2089), month (1-12), and day of the month (1-31).

Units: none

Example: 1994,10,23
represents the date October 23, 1994.

SYSTem:DATE:FORMat <format select>
<format select> US or INTer

sets the LCD display format for date information and selects the format for printing `DATE` and `VDATe` selections for `PRINt:ITEM` and `PRINt:SELected:ITEM`.

This has no effect on the format of the remote query command.

The formats are defined as:

USmonth, day, year
INTerday, month, year

Example 1: SYST:DATE:FORMAT US
The date is displayed on the LCD as Aug 02, 1994.

Example **2:** SYST:DATE:FORMAT INT
The date is displayed on the LCD as 02 Aug, 1994.

SYSTem:ERRor?

returns the oldest entry in the error queue.

The error queue is empty on powerup, and is cleared when a device clear or interface clear is received from the GPIB port or the control-C character is received from the serial port.

The error queue holds up to 10 errors and deletes them as they are read using `SYSTem:ERRor?`.

The error information returned from this query consists of a numeric error code, followed by an ASCII string describing the error.

If the error queue is empty when this command is received, the following message is returned:

0,"No error"

If the error queue is full and another error occurs prior to reading an error, the most recent error is discarded and replaced with:

-350,"Queue overflow"

The numeric portion of the response is always a negative value (-100 to -499) or 0 if there are no errors.

The string consists of an English language translation of the error, as defined by SCPI, and contains additional information.

If there is additional information, it is displayed as follows:

"English description; (return code) offending command plus ˆ mark"

A semicolon and a numeric value enclosed in parenthesis are displayed after the English description. The numeric value is a return code that gives additional details about the cause of the error.

If no additional details are available, the numeric value is 0.

Otherwise, it will be a negative value. See Appendix F.

The command on which the error occurred follows the return code. A 'ˆ' character placed after the character in the command at which the error occurred helps pinpoint the cause of command errors (error codes -100 to -199).

Response: <numeric integer>, <ASCII string>

Example 1: CALL:REGI:IMM
SYST:ERR?
-113,"Undefined header;(0)CALL:REGI:ˆIMM"
indicates that the intended command was CALL:REG:IMM.

The 4300 saw that REGI was not a valid subnode of CALL, hence the "undefined header" error.

NOTE: The 'ˆ' was placed after the REGI: indicating this is the node that could not be found.

Example 2: *RCL 7
SYST:ERR?
-314,"Save/recall memory lost;(-744)*RCL 7ˆ"
indicates an error in retrieving stored settings location #7.

Response code -744 indicates that the location had an invalid checksum due to a memory failure or recalling a location that had never been stored.

SYSTem:KEY:DEBounce <debounce time>
<debounce time> numeric value 10 - 200 (default = 20)

sets the debounce time (in ms) for the front panel keys.

Adjust it only if there is a problem with the keypad. Larger values should be used for noisy keypads.

SYSTem:KEY:DELay <delay time>
<delay time> numeric value 10 - 10000 (default = 800)

sets the amount of delay time (in ms) to wait after a front panel keypress before entering the repeat mode.

Any key held down for less than this duration is accepted as a single keypress. If a key is held down longer than this time, the key is repeated multiple times, at a rate given by `SYSTem:KEY:REPeat`.

See `SYSTem:KEY:REPeat`.

SYSTem:KEY:REPeat <repeat time>
<repeat time> numeric value 10 - 2000 (default = 50)

sets the amount of delay time (in ms) to wait after a front panel keypress before entering the repeat mode.

Any key held down for less than this duration is accepted as a single keypress. If a key is held down longer than this time, the key will be repeated multiple times, at a rate given by `SYSTem:KEY:REPeat`.

See `SYSTem:KEY:REPeat`.

SYSTem:PRESet

initializes many of the 4300 parameters to their corresponding default values.

This is more comprehensive than `*RST` in that it performs all of the actions of `*RST` and initializes the following parameters:

SCPI command	Default Value
CALibration[:ALL]:AUTO	OFF
CALibration:CABLe:COMPensation	OFF
CALibration:CABLe:SELection	1
CALL:MDATa:ESN:FORMat	STANDARD2
CALL:PAGE:MIN1	0
CALL:PAGE:MIN2	0
CALL:SSD:AKEY:DATA	"00000000000000000000000000"
CALL:SSD:SSDA	"00000000000000000000"
DISPlay:RATio[:IMMEDIATE]	RECeive
DISPlay:RATio:RECeiver	SNDRATIO
DISPlay:RATio:TRANsmitter	SNDRATIO
DISPlay:UNITs:FREQuency:RF	KHZ
DISPlay:UNITs:PHASe	DEGRee
DISPlay:UNITs:POWer:TRANsmitter	WATT
PROGram:AGRaph:CHANnel:BAND	CELLular
PROGram:AGRaph:CHANnel:STARt	100
PROGram:AGRaph:CHANnel:STEP	10

SCPI command	**Default Value**
PROGram:AGRaph:CHANnel:STOP	650
PROGram:AGRaph:CHANnel:TYPE	AMPS
PROGram:AGRaph:DLOG	OFF
PROGram:AGRaph:MAC0	ON
PROGram:AGRaph:MAC1	ON
PROGram:AGRaph:MAC2	ON
PROGram:AGRaph:MAC3	ON
PROGram:AGRaph:MAC4	ON
PROGram:AGRaph:MAC5	ON
PROGram:AGRaph:MAC6	ON
PROGram:AGRaph:MAC7	ON
PROGram:AGRaph:PAUSe	OFF
PROGram:AGRaph:PLOG	OFF
SYSTem:KEY:DEBounce	20 msec
SYSTem:KEY:DELay	800 msec
SYSTem:KEY:REPeat	50 msec
PROGram[:SELected]:BUFFer:MLISt	MEAS, NONE, NONE, NONE
PROGram[:SELected]:BUFFer:MLOG	OFF

The following commands have no associated accessible parameters:

PRINt:PRESet
PROGram[:SELected]:BUFFer:CLEAr

They have internal parameters or buffers that are accessible and used to initialize them.

The following command places the 4300 (and mobile phone) on the control channel selection defined by `CALL:CONTrol:XXX`:

CALL:FCC ON

NOTE: If the mobile phone was up on a call, the call is dropped.

SYSTem:SNUMber?

returns the serial number of the instrument.

The serial number is established when the calibration table is loaded into the test equipment from disk at the factory and when a software upgrade is performed.

Response: <ASCII string (maximum length 21 characters)>

Units: none

Example: SYST:SNUM?
045769
indicates that the serial number for the test equipment is 045769.

SYSTem:TIME[:DATA] <hour>, <minute>, <second>
<hour> numeric value 0 - 23 (24-hour clock format)
<minute> numeric value 0 - 59
<second> numeric value 0 - 59

sets the current time for the real-time clock in the instrument.

Use this command when files are saved-to-disk and can be included in print-outs.

SYSTem:TIME[:DATA]?

returns the current time read from the real-time clock in the test instrument.

The response is in the 24-hour format regardless of the `SYSTem:DATE:FORMat` setting.

Response: <numeric integer>, <numeric integer>, <numeric integer>

The values represent the hour (0 to 23), the minute (0 to 59), and the second (0 to 59).

Units: none

Example: 17,22,56
represents 17:22:56 or 5:22:56 PM.

SYSTem:TIME:FORMat <format select>
<format select> AMPM or HR

sets the LCD display format for the time information and selects the format for printing the `TIME` selection for `PRINt:ITEM` and `PRINt:SELected:ITEM`.

This has no effect on the format of the remote query command.

The formats are defined as:

AMPMtime in 12-hour format with an AM/PM suffix
HRtime in 24-hour format

Example 1: SYST:TIME:FORMAT HR
The time is displayed on the LCD as 18:20:15.

Example 2: SYST:TIME:FORMAT AMPM
The time is displayed on the LCD as 06:20:15 PM.

SYSTem:TSTamp?

returns a timestamp value that is equivalent to the number of seconds from the beginning of the current calendar year.

The day of the year is multiplied by 86400 (leap year is accounted for), the hour of the day is multiplied by 4300 (24-hr. clock format used), the minutes are multiplied by 60, and all of these are added to the seconds. This results in a range of values from 0.0 to 31622399.0, with 0.0 representing 0:00:00 (midnight) on January 1, and 31622399.0 representing 23:59:59 (1 second before midnight) on December 31 on a leap year. If there is no leap year, this value is 31535999.0. The decimal portion of seconds is truncated, so each reading may have an error by as much as 1.0 second.

Response: <numeric floating point>

Resolution: 1.0

Units: seconds

SYSTem:TSTamp? <reference timestamp>
<reference timestamp> numeric value from 0.0 - 31622399.0

returns the time that has elapsed since the reference timestamp value.

The timestamp value is the value that is returned by `SYSTem:TSTamp?` without any arguments.

Since this timestamp value does not include information on the year, the maximum elapsed time that can be determined is one year, however it does compensate for end-of-year rollover. If the reference timestamp is collected on December 31 and this command is used on January 1 (the following day), the correct elapsed time will be returned.

Due to the timestamp truncation of fractional seconds, the elapsed time may have a 1.0 second error.

Response: <numeric floating point>

Resolution: 1.0

Units: seconds

Example: SYST:TST?
4031183.0
CALL:REG; *WAI
SYST:TST? 4031183.0
14.0

The initial SYST:TST? gets a timestamp value that is used as a reference for obtaining elapsed time.

A registration is initiated and completed before the second SYST:TST? is executed, to get the elapsed time for the command execution.

The final result indicates that the registration took ~ 14 seconds.

SYSTem:VERSion:AMPS?

returns the software version for the AMPS microprocessor.

See `SYSTem:VERSion:SCP?` for a description of the response format.

SYSTem:VERSion:CCM:DSP?

returns the software version for the DSP on the CCM card.

This command produces a -708,"Hardware missing" error, if the CCM card is not installed in the unit.

See `SYSTem:VERSion:SCP?` for a description of the response format.

This command is valid only if the CCM option is installed.

SYSTem:VERSion:CCM:FPGA?

returns the software version for the FPGA on the CCM card.

This command produces a -708,"Hardware missing" error, if the CCM card is not installed in the unit.

See `SYSTem:VERSion:SCP?` for a description of the response format.

This command is valid only if the CCM option is installed.

SYSTem:VERSion:CCM[:MICR]?

returns the software version for the CCM microprocessor.

This command produces a -708,"Hardware missing"error, if the CCM card is not installed in the unit.

See `SYSTem:VERSion:SCP?` for a description of the response format.

This command is valid only if the CCM option is installed.

SYSTem:VERSion:SCP?

returns the software version for the SCP microprocessor.

Response: <string>

Units: none

Example: "2.1b"
The string is composed of a numeric value representing the major release version, a decimal point, another numeric value representing the minor revision for the release, and an optional alpha character representing the maintenance revision.

SYSTem:VERSion[:SCPI]?

returns the SCPI version for which the software is compliant.

Response: <numeric value>

Units: none

Example: 1994.000000
indicates the software is compliant with SCPI version 1994.0.

The value proceeding the decimal point is the year in which the version was released, and the first digit following the decimal point indicates the minor revision made for that year and is 0 for the initial release for the year.

SYSTem:VOLume <level>
<level> numeric value 0 to 1023 (0=softest, 1023=loudest)

sets the audio speaker volume level.

SPCI Command Errors

6

This chapter describes the functionality of the instrument. Topics discussed in this chapter are as follows:

Command Errors

Error	Error Definition
-100, "Command error"	A generic error in finding the command. It is used by the 4300 to report that nodes to the command header that were omitted (e.g. CALL:ORIG is a valid command. If CALL is received and is followed by a terminator, it is reported as a command error).
-102, "Valid sequencer and remote error"	An error which occurs when issuing the command version of a query-only command.
-101, "Invalid character"	An invalid character found in the command header.
-103, "Invalid separator"	A command having more than one parameter was received, and the data was not comma separated.
-104, "Data type error"	The data type (numeric, string, etc.) for the given command not the correct type.
-108, "Parameter not allowed"	Too many parameters sent with the command.
-109, "Missing parameter"	One or more required parameters omitted from the command.
-111, "Header separator error"	During parsing of the command header, an invalid character (not a valid header separator (ASCII space or colon), command separator (;), query annunciator (?), or command terminator) was encountered.
-112, "Program mnemonic too long"	The header contains more than 12 characters.
-113, "Undefined header"	One of the command header rnodes was invalid. (e.g. MEAS:DEV:PEAK? is not valid since there is no PEAK node valid for MEAS:DEV. MEAS:DEV:PAUD? measures the peak audio deviation).
-114, "Header suffix out of range"	Too large of a value for an indexed command. (e.g. CALC:LIM:POW:MAC11:NOM -35.0 is invalid, since only MAC levels 0 to 10 are defined).

-121, "Invalid character in number"	An invalid character was received as part of a numeric parameter. Numeric parameters can be expressed in the following forms: +12.00002 +123.4567E-2 where: '+' and '-' are interchangeable or may be omitted; the decimal point and digits to the right of the decimal point are optional; the 'E' can be upper or lowercase.
-123, "Exponent too large"	The exponent value exceeded +32000 or -32000.
-128, "Numeric data not allowed"	The command received numeric data when a different data type was required. (e.g. INPut:RF 4 generates this error; the correct parameters are DIRect or OFFair).
-131, "Invalid suffix"	An invalid units suffix appended to the command parameter.
-134, "Suffix too long"	The units suffix string exceeds the maximum of 12 characters.
-138, "Suffix not allowed"	The parameter for the specified command does not accept any unit suffixes.
-141, "Invalid character data"	The character program data not a valid selection for the specified command.
-144, "Character data too long"	The character program data exceeded 12 characters. (e.g. INPut:RF DIRECTCONNECTION is invalid because the character data exceeds 12 characters.) The character string is not defined, since all defined character parameters are 12 characters or less.
-158, "String data not allowed"	The command received string data when a different data type was required. (e.g. INPut:AUD:DEV "ten" generates this error; the correct parameter is numeric data type).
-160, "Block data error"	The command received indefinite length arbitrary block data.
-168, "Block data not allowed"	The command received definite length arbitrary block data when a different data type was required.

Execution Errors

Errors	**Error Definition**
-200, "Execution error"*	The last command not able to execute properly. Additional details are contained in the return code value (in parenthesis) following "Execution error".
-221, "Settings conflict"*	An invalid *value-coupled* command combination received. Value-coupled commands have parameters that affect and are affected by other parameters. For example, the start and stop channel settings for the Autograph mode are dependent on each other. The start value must be less than or equal to the stop value. If a command was received that set both the start and the stop values, and the stop value was less than the start value, a "Settings Conflict" error is indicated.
-222, "Data out of range"	The limits of the numeric parameter of a command was exceeded.
-225, "Out of memory"	Memory allocation not sufficient to complete command.
-241, "Hardware missing"	The hardware/software option required to execute the command not installed on the test equipment.
-250, "Mass storage error"	Generic disk error. Additional details are contained in the return code value (in parenthesis) following "Mass storage error".
-252, "Missing media"	No disk inserted when a command attempted to access the disk. Insert disk and try again.
-253, "Corrupt media"*	The disk format could not be read. If the disk was not formatted, format the disk and try again. If the disk is damaged, try a new (formatted disk). **Caution:** Formatting a disk deletes any existing files on the disk.

-254, "Media full"	A write-to- disk command failed due to insufficient disk space. Replace with another disk that is less full, or delete unneeded files on the current disk.
-255, "Directory full"	Command was executed after the maximum number of files (223) that can be stored on the disk has been reached.
-256, "File name not found"	The specified filename not located on the disk.
-257, "File name error"	The specified filename contained illegal characters, or had an invalid extension. The filename must contain only alpha numerics, plus the special characters; "-", "_", ".". The extension specifies a certain type of file. See DISK:DIRectory? for a list of extensions and their corresponding file types.
-258, "Media protected"	A store-to-disk command was executed with the write protect tab set. Switch the write protect tab on the disk to OFF, and try again.
-284,"Program currently running"	A command that modifies the program sent while that program is running. The program sequence issued a command that may modify itself. PROGram:DEFine and DISK:LOAD:PROGram cause this error. DISK:LOAD:PROGram is executable from AUTO, QUICK or CUSTOM programs, but not from a nested program.
-292,"Referenced name does not exist"	The specified label in the command not found.

* See *Non-SCPI Errors* for additional details.

Device-Specific Errors

Error	Error Definition
-310, “System error”*	A fatal error that prevents the system from executing its current process occurred.
-314,“Save/recall memory lost”*	*RCL was attempted on a stored setting location that was either empty, or had invalid data.
-350, “Queue overflow	> 10 errors occurred prior to reading the error queue. The most recent error is replaced with this error, and the new error is not recorded.

* See *Non-SCPI Errors* for additional details.

Query Errors

Error	Error Definition
-410, "Query Interrupted"	An input character sent from the remote port when a response message was pending. A response generated by a query command should be read prior to sending another command. If a command follows the query, the response message is purged. See IEEE 488.2 sect 6.3.2.3 for information of Interrupted Query error conditions.
-430, "Query Deadlocked"	The input and output buffer were filled and execution could not continue due to lack of memory. The output buffer has been deleted in order to break the deadlock.

Non-SCPI Errors

These errors, in conjunction with certain SCPI error codes, provide detailed information on the cause of an error condition.

They are reported in the response section of the SYSTem:ERRor? string.

All Non-SCPI errors are negative values between -500 and -9999.

Each return code value has an associated SCPI error code.

SCPI ERROR -310, System error

These errors indicate either a hardware failure or an undetected bug in the internal operations of the 4300.

Report these errors to the factory and cycle the 4300 power to correct any associated problems.

–1	Operating system generic command failure
–2	Operating system process (task) command failure
–3	Operating system process (task) signal failure
–4	Operating system send message failure
–5	Operating system request message failure
–6	Operating system message exchange command failure
–10	Memory allocation failure
–20	Undefined disk error
–21	Disk task command time-out
–22	Invalid Disk task command
–23	Disk File system missing
–24	Filename is a directory
–25	No file handles left
–26	Invalid R/W mode for file
–27	Invalid mode for disk seek
–28	Invalid offset for disk seek
–29	Invalid file handle
–40	Disk BIOS error (generic)
–41	Disk BIOS system error
–42	Disk BIOS READ command error
–43	Disk BIOS WRITE command error
–44	Disk BIOS FORMAT command error
–60	Invalid command token

–61 Invalid index value for parameter

–62 Invalid parameter (specified parameter does not exist)

–63 Wrong parameter data type

–64 Missing parameter

–65 Too many parameters specified

–70 Test task command failure

–99 The command has not been implemented.

SCPI ERROR -221, Settings conflict

These errors indicate that some required setup condition was not met for the command that was executed.

–600 A call processing or measurement command failed due to invalid setup conditions other than the voice/control channel selection (-603) or service mode with no sync (-602).

(e.g. TDMA measurement performed on a shortened burst, when a normal burst was required; improper handoff conditions selected when a handoff was performed.)

–602 A call processing or measurement command was attempted in service mode when no sync was detected.

Certain commands require sync detection in service mode before they can be executed.

–603 A call processing or measurement command was attempted on an invalid voice or control channel selection.

–605 The base power setting could not be set to the selected value during a sensitivity measurement.

Base power minimum and maximum allowable values depend on whether the 20dB pad on the RF input is in or out and whether fading is enabled in TDMA mode.

–607 A call processing or measurement was attempted on a mobile phone that did not have the required features (e.g. handoff to a TDMA channel when mobile phone is analog only).

–720 The limit table selection is invalid for the specified command (e.g. the print limit table command was sent when the limit table selection was NONE).

SCPI ERROR -200, Execution error

These errors indicate generic execution command errors.

–650 The A-key checksum digits value did not verify.

–651 The mobile never responded to the SSD update message.

–652 The mobile never responded to either a Unique Challenge Order message during a Unique Challenge, or to a Base Station Challenge Confirmation message during an SSD update.

-700 The command was aborted (by a DEVICE CLEAR remote command) prior to completing execution.

-701 A command that is only valid from the remote was received from the sequencer, or a command that was only valid from the sequencer was received from the remote.

-702 Buffer overflow. A command that was either placing data into, or reading the information out of a specified buffer location had exceeded the size of the buffer.

-703 The filename extension is not valid for the file type being stored or recalled.

-721 The limit table checksum is incorrect.

Invalid data in the stored settings table that may be caused by a nonvolatile memory failure.

-722 The limit table version is invalid for this revision of software.

Invalid data in the stored settings table that may be caused by a nonvolatile memory failure, or loading in a limit table with an invalid version from disk.

-723 An out-of-range parameter value in the limit table was loaded.

Invalid data in the stored settings table that may be caused by a nonvolatile memory failure, or loading in a limit table with an invalid parameter data from disk.

-735 One of the stored setting parameters out of range during a stored setting recall.

-736 A stored setting recall execution function error.

-737 The stored setting recall had a command function error.

-740 Calibration table checksum error (in base section) following a calibration download-from-disk or through the remote port.

-741 Same as (-740) except checksum error on PCS section of data.

-742 Invalid data in previously downloaded calibration table.

-780 An attempt was made to run an invalid or NULL program

-790 DCM task time-out.

Indicates a problem with the DCM card interface in the 4300.

SequencerErrors

If an error is found in either sequencer directive or an SCPI command sent through the sequencer, a 2-line message is displayed in inverse video near the bottom of the LCD display.

The first line contains the line number in the sequence where the error occurred, followed by the statement “Program Error:” and the error code.

The second line consists of the command that was in error.

-1 System error. Internal operating system error.

–2	System error. Sequencer nesting info lost when END directive executed.
–3	System error. Invalid sequence checksum.
–4	Sequence has not been loaded (zero length sequence).
–5	The RUNPROG directive was encountered when a sequence has not been loaded into the temporary buffer.
–6	The sequence has completed without finding the END directive.
–7	The specified label was not found.
–8	One or more required parameters was omitted from the directive.
–9	The data for the directive was out of range.
–10	The maximum number of nested loops (40) was exceeded.
–11	The maximum number of nested subroutines (40) was exceeded.
–12	The RUNPROG directive was encountered in a sequence that was already nested by another sequence.
–13	The maximum number of GLOBAL definitions was exceeded (50 allowed).
–14	SCPI command exceeded maximum limit allowable from sequence (200 characters).
–20	The NEXT directive was received when a FOR loop was not running.
–21	The RETURN directive was received when not in a subroutine.
–22	The END directive was received while in a subroutine.
–23	The END directive was received while in a loop.
–24	The RETURN directive was received while inside loop when not in a sub-routine.
–25	The BREAK directive was received when not in a loop.
–26	System error. Loop and subroutine nesting structure incorrect when the NEXT directive was executed.
–30	A BRANCH directive inside the main routine attempted to branch outside the main routine.
–31	A BRANCH directive inside a subroutine attempted to branch outside the subroutine.
–32	A BRANCH directive inside a loop attempted to branch outside the loop.
–40	The CompareLastMeas or AssignGlobalMeas directive was used when the last measurement executed was unable to take a valid reading.
–41	The DefineGlobal directive was used with the "/=" selection to divide the current global value by zero (Division by zero error).
–42	One of the Error Handler command directives was branched to when no SCPI error occurred (The error handler code should never be executed by the main program code. This code is reserved for SCPI errors).

SCPI ERROR -314, Save/recall memory lost

–732	Invalid data in the internal 4300 parameter tables during a stored setting recall.

–733 The stored setting parameter written but not verified.

This may be caused by a nonvolatile memory failure.

–734 The stored setting checksum incorrect.

This may be caused by a nonvolatile memory failure.

SCPI ERROR -250, Mass storage error

These errors indicate general disk-related errors.

–900 Buffer overflow.

A load-from-disk command executed but the receiving buffer not large enough.

Indicates that the file being read was not valid (e.g. The cal table file being read had more data entries than it should have).

–901 A read-only file was opened for writing.

The specified file must be closed prior to writing to it.

–902 A write-only file that was previously opened was opened for writing a second time prior to closing it.

–903 The maximum number of concurrently opened files (2) was exceeded.

Close one of the open files prior to opening another file.

–904 The file to be read or written was not opened first.

Open the file before reading or writing to it.

–905 The specified filename already exists.

Change the filename and try the command again.

SCPI ERROR -253, Corrupt media

These errors indicate that the disk format could not be read.

–910 Disk format unreadable.

Either the disk is not formatted, or there is a problem with the disk BIOS.

Format the disk, or insert another formatted disk and try the command again.

If the same error occurs, either the disk cannot be formatted properly (bad disk) or the 4300 disk BIOS has a failure.

For a BIOS failure, reset the power on the 4300 to correct the problem.

–911 Disk format invalid during read or write operation.

May be caused by a disk format error or a bad disk sector.

Similar to error -910.

Warranty and Repair

A

This chapter describes the customer services available through Willtek. Topics discussed in this chapter include the following:

Warranty information

Willtek warrants that all of its products conform to Willtek's published specifications and are free from defects in materials and workmanship for a period of one year from the date of delivery to the original buyer, when used under normal operating conditions and within the service conditions for which they were designed. This warranty is not transferable and does not apply to used or demonstration products.

In case of a warranty claim, Willtek's obligation shall be limited to repairing, or at its option, replacing without charge, any assembly or component (except batteries) which in Willtek's sole opinion proves to be defective within the scope of the warranty. In the event Willtek is not able to modify, repair or replace nonconforming defective parts or components to a condition as warranted within a reasonable time after receipt thereof, the buyer shall receive credit in the amount of the original invoiced price of the product.

It is the buyer's responsibility to notify Willtek in writing of the defect or nonconformity within the warranty period and to return the affected product to Willtek's factory, designated service provider, or authorized service center within thirty (30) days after discovery of such defect or nonconformity. The buyer shall prepay shipping charges and insurance for products returned to Willtek or its designated service provider for warranty service. Willtek or its designated service provider shall pay costs for return of products to the buyer.

Willtek's obligation and the customer's sole remedy under this hardware warranty is limited to the repair or replacement, at Willtek's option, of the defective product. Willtek shall have no obligation to remedy any such defect if it can be shown: (a) that the product was altered, repaired, or reworked by any party other than Willtek without Willtek's written consent; (b) that such defects were the result of customer's improper storage, mishandling, abuse, or misuse of the product; (c) that such defects were the result of customer's use of the product in conjunction with equipment electronically or mechanically incompatible or of an inferior quality; or (d) that the defect was the result of damage by fire, explosion, power failure, or any act of nature.

The warranty described above is the buyer's sole and exclusive remedy and no other warranty, whether written or oral, expressed or implied by statute or course of dealing shall apply. Willtek specifically disclaims the implied warranties of merchantability and fitness for a particular purpose. No statement, representation, agreement, or understanding, oral or written, made by an agent, distributor, or employee of Willtek, which is not contained in the foregoing warranty will be binding upon Willtek, unless made in writing and executed by an authorized representative of Willtek. Under no circumstances shall Willtek be liable for any direct, indirect, special, incidental, or consequential damages, expenses, or losses, including loss of profits, based on contract, tort, or any other legal theory.

Equipment return instructions

Please contact your local service center for Willtek products via telephone or web site for return or reference authorization to accompany your equipment. For each piece of equipment returned for repair, attach a tag that includes the following information:

– Owner's name, address, and telephone number.
– Serial number, product type, and model.
– Warranty status. (If you are unsure of the warranty status of your instrument, include a copy of the invoice or delivery note.)
– Detailed description of the problem or service requested.
– Name and telephone number of the person to contact regarding questions about the repair.
– Return authorization (RA) number or reference number.

If possible, return the equipment using the original shipping container and material. Additional Willtek shipping containers are available from Willtek on request. If the original container is not available, the unit should be carefully packed so that it will not be damaged in transit. Willtek is not liable for any damage that may occur during shipping. The customer should clearly mark the Willtek-issued RA or reference number on the outside of the package and ship it prepaid and insured to Willtek.

Publication History

Revision	Comment
Rev. C	
0212-100-A	New format.

Worldwide Headquarter and Regional Sales Headquarter

West Europe/Middle East/Africa/Asia Pacific

Willtek Communications GmbH
Gutenbergstr. 2–4
85737 Ismaning
Germany
Tel: +49 (0)89 99641-0
Fax: +49 (0)89 99641 160
info@willtek.com

Regional Sales Headquarters

United Kingdom/Ireland/Benelux

Willtek Communications Ltd.
Roebuck Place
Roebuck Road
Chessington
Surrey KT9 1EU
United Kingdom
Tel: +44 (0)20 8408 5720
Fax: +44 (0)20 8397 6286
willtek.uk@willtek.com

North America/Latin America

Willtek Communications Inc.
7369 Shadeland Station Way, Suite 200
46256 Indianapolis
Indiana
USA
Tel: +1 317 595 2021
Fax: +1 317 595 2023
willtek.us@willtek.com

To find your local service center and sales office go to: www.willtek.com

Manual ident no. M 293 043
Manual version 0212-100-A
English

will’tek