
Programmer's Quick Reference Guide

Publication number 54542-97042
First edition, November 1995

For Safety information, Warranties, and Regulatory
information, see the pages at the end of this book.

© Copyright Hewlett-Packard Company 1993-1995
All Rights Reserved

HP 54520C and 54540C Series Oscilloscopes

Introduction

The Quick Reference Guide lists the commands and queries with their corresponding arguments and returned formats. The arguments for each command list the minimum argument required. The part of the command or query listed in uppercase letters refers to the short form of that command or query. The long form is the combination of uppercase and lowercase letters.

Conventions

The following conventions are used in this guide:

- <> Angular brackets enclose words or characters that symbolize a program code parameter or an HP-IB command.
- ::= "is defined as." For example, <A> ::= indicates that <A> can be replaced by in any statement containing <A>.
- | "or." Indicates a choice of one element from a list. For example, <A> | indicates <A> or but not both.
- ... An ellipsis (trailing dots) indicate that the preceding element may be repeated one or more times.
- [] Square brackets indicate that the enclosed items are optional.
- { } When several items are enclosed by braces, one, and only one of these elements may be selected.

Suffix Multipliers

The suffix multipliers available for arguments are:

EX ::= 1E18	M ::= 1E-3
PE ::= 1E15	U ::= 1E-6
T ::= 1E12	N ::= 1E-9
G ::= 1E9	P ::= 1E-12
MA ::= 1E6	F ::= 1E-15
K ::= 1E3	A ::= 1E-18

For more information on specific commands or queries, refer to the *HP 54520C and 54540C Series Oscilloscopes Programmer's Reference*.

Programmer's Quick Reference Guide

	*CLS
Command	*CLS
	*DMC
Command	*DMC <ascii_string>,<block_data>
Where:	<ascii_string> ::= a quoted ascii string <block_data> ::= definite block data in IEEE 488.2 # format
	*EMC
Command	*EMC {{OFF 0} {ON 1}}
Query	*EMC?
Returned Format	{0 1}<NL>
	*ESE
Command	*ESE <mask_argument>
Query	*ESE?
Returned Format	<mask_argument><NL>
Where:	<mask_argument> ::= integer, 0 to 255
	*ESR?
Query	*ESR?
Returned Format	<status><NL>
Where:	<status> ::= integer, 0 to 255
	*GMC?
Query	*GMC? <ascii_string>
Returned Format	<block_data><NL>
Where:	<ascii_string> ::= a quoted string <block_data> ::= definite block data in # format

Programmer's Quick Reference Guide

Query Returned Format Where:	*IDN? *IDN? HEWLETT-PACKARD,545XXC,YYYYYYYYYY,ZZ.ZZ,ZZ.ZZ,ZZ.ZZ,Z.ZZZ,<NL> <XXC> ::= model number is 20C (HP 54520C), 22C (HP 54522C), 40C (HP 54540C), or 42C (HP 54542C) <YYYYYYYYYY> ::= the serial number of the instrument <ZZ.ZZ> ::= the software revision of the software modules (Boot ROM, Flash ROM version of Boot ROM, System, Keyboard ROM). 00.00 = not installed <Z.ZZZ> ::= the telecommunication mask option null = no option, 0.001 = option installed
---	--

Query Returned Format Where:	*LMC *LMC? <ascii_string><NL> <ascii_string> ::= string list separated by commas
---	--

Query Returned Format Where:	*LRN? *LRN? :SYSTEM:SETup <setup><NL> <setup> ::= #800002048<learn_string> <learn_string> ::= 2048 bytes in length
---	---

Command Query Returned Format	*OPC *OPC *OPC? 1<NL>
--	---

Query Returned Format Where:	*OPT ? *OPT? 0<NL> or 000.1<NL> 0 = no option 0.001 = telecom mask option installed
---	--

Programmer's Quick Reference Guide

*PMC	
Command	*PMC
*RCL	
Command	*RCL {0 1 2 3 4 5 6 7 8 9}
*RST	
Command	*RST
*SAV	
Command	*SAV {1 2 3 4 5 6 7 8 9}
*SRE	
Command	*SRE <mask_argument>
Query	*SRE?
Returned Format	<mask><NL>
Where:	<mask_argument> ::= integer, 0 to 255 <mask> ::= sum of all bits that are set - integer, 0 through 255
*STB?	
Query	*STB?
Returned Format	<value><NL>
Where:	<value> ::= integer, 0 through 255
*TRG?	
Command	*TRG

Programmer's Quick Reference Guide

*TST?

Query Returned Format Where:	<pre>*TST? <result><NL> <result> ::= 0 or non-zero value. 0 indicates the test passed. non-zero indicates the test failed</pre>
------------------------------------	---

*WAI

Command	<pre>*WAI</pre>
---------	-----------------

:ACquire:COMplete

Command Query Returned Format Where:	<pre>:ACquire:COMplete <complete_argument> :ACquire:COMplete? [:ACquire:COMplete] <complete_argument><NL> <complete_argument> ::= integer, 0 to 100 percent</pre>
---	---

:ACquire:COUNT

Query Returned Format Where:	<pre>:ACquire:COUNT <count_argument> :ACquire:COUNT? [:ACquire:COUNT] <count_argument><NL> <count_argument> ::= integer, 1 to 2048 (depending on the acquisition type)</pre>
------------------------------------	--

:ACquire:POINTs

Command Query Returned Format Where:	<pre>:ACquire:POINTs <points_argument> :ACquire:POINTs? [:ACquire:POINTs] <points_argument><NL> <points_argument> ::= integer, 500 in repetitive mode, 512, 1024, 2048, 4196, 8192, 16384, or 32768 in the real-time mode (sequential mode off), or 4 through 32768 in the real-time mode (sequential mode on)</pre>
---	--

Programmer's Quick Reference Guide

:ACQUIRE:TYPE

Command :ACQUIRE:TYPE {NORMAL | AVERage | ENVELOpe | PDETECT|
 RAWData[,<length>][,<acquisitions>]][,{NORMAL | AVERage |
 ENVELOpe }]
Query :ACQUIRE:TYPE?
Returned Format [:ACQUIRE:TYPE] {NORMAL | AVERage | ENVELOpe | PDETECT|
 RAWData,<length>,<acquisitions><NL>
Where: <length> ::= integer, 4 to 32768
 <acquisitions> ::= dependent on length of acquisitions and
 buffer size

:AUTOScale

Command :AUTOScale
Query :AUTOScale?
Returned Format [:AUTOScale] {0 | 1}<NL>

:BEEPPer

Command :BEEPPer [{(OFF | 0) | (ON | 1)}]
Query :BEEPPer?
Returned Format [:BEEPPer] {0|1}<NL>

:BLANK

Command :BLANK <display>
Where: <display> ::= {CHANnel<n> | FUNctIon{1|2|3|4} |
 WMEMory{1 | 2 | 3 | 4} | PMEMory{1 | 2}}
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:BNC

Command :BNC {PROBe | TRIGger}
Query :BNC?
Returned Format [:BNC] {PROBe | TRIGger}<NL>

Programmer's Quick Reference Guide

:CALibrate:DATA:ASCIi?

Query: :CALibrate:DATA:ASCIi?
 Returned Format: [:CALibrate:DATA:ASCIi] <data>,<data>,...<NL>
 Where: <data> ::= calibration data

:CALibrate:SETup?

Query: :CALibrate:SETup?
 Returned Format: :CALibrate:TNULl
 Where: <null_value_n>,<null_value_n>,<null_value_n><NL>
 <null_value_n> ::= exponential, channel 1 to channel<n> skew,
 where n = 2 (HP 54520C/54522C) or 2 through 4
 (HP 54540C/54542C) in format

:CALibrate:TNULl

Command: :CALibrate:TNULl <null_value_n>
 Query: :CALibrate:TNULl?
 Returned Format: [:CALibrate:TNULl] <null_value_n><NL>
 Where: <null_value_n> ::= exponential, channel 1 to channel<n> skew,
 where n = 2 (HP 54520C/54522C) or 2 through 4
 (HP 54540C/54542C)

:CHANnel<n>:COUPling

Command: :CHANnel<n>:COUPling {AC | DC | DCFifty}
 Query: :CHANnel<n>:COUPling?
 Returned Format: [:CHANnel<n>:COUPling] {AC | DC | DCFifty}<NL>
 Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:CHANnel<n>:DISPLay

Command: :CHANnel<n>:DISPLay {{OFF | 0} | {ON | 1}}
 Query: :CHANnel<n>:DISPLay?
 Returned Format: [:CHANnel<n>:DISPLay] {0 | 1}<NL>
 Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:CHANnel<n>:ECL**Command**

:CHANnel<n>:ECL

Where:<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
(HP 54540C/54542C)

:CHANnel<n>:HFReject**Command**

:CHANnel<n>:HFReject {(OFF | 0) | {ON | 1}}

Query

:CHANnel<n>:HFReject?

Returned Format

[:CHANnel<n>:HFReject] {0 | 1}<NL>

Where:<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
(HP 54540C/54542C)

:CHANnel<n>:LFReject**Command**

:CHANnel<n>:LFReject {(OFF | 0) | {ON | 1}}

Query

:CHANnel<n>:LFReject?

Returned Format

[:CHANnel<n>:LFReject] {0 | 1}<NL>

Where:<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
(HP 54540C/54542C)

:CHANnel<n>:OFFSet**Command**

:CHANnel<n>:OFFSet <offset_argument>

Query

:CHANnel<n>:OFFSet?

Returned Format

[:CHANnel<n>:OFFSet] <offset_argument><NL>

Where:<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
(HP 54540C/54542C)

<offset_argument> ::= exponential, offset value in volts

Programmer's Quick Reference Guide

:CHANnel<n>:PROBe

Command :CHANnel<n>:PROBe <probe_argument>
Query :CHANnel<n>:PROBe?
Returned Format [:CHANnel<n>:PROBe] <probe_argument><NL>
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)
 <probe_argument> ::= exponential, 0.9 to 1000.0

:CHANnel<n>:RANGe

Command :CHANnel<n>:RANGe <range_argument>
Query :CHANnel<n>:RANGe?
Returned Format [:CHANnel<n>:RANGe] <range_argument><NL>
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)
 <range_argument> ::= exponential, full-scale range value

:CHANnel<n>:SETup

Query :CHANnel<n>:SETup?
Returned Format :CHAN<n>:COUP {AC|DC|DCF};
 DISP {0 | 1};
 HFR {0 | 1};
 LFR {0 | 1};
 OFFS <offset_argument>;
 PROB <probe_argument>;
 RANG<range_argument><NL>
Where: <range_argument> ::= exponential, full-scale range value
 <offset_argument> ::= exponential, offset value in volts
 <probe_argument> ::= exponential, 0.9 to 1000.0
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:CHANnel<n>:TTL

Command :CHANnel<n>:TTL
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:DIGitize

Command :DIGitize CHANNEL<n>[,CHANNEL<n>[,CHANNEL<n>[,CHANNEL<n>]]]
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:DISK:CDIRectory

Command :DISK:CDirectory <directory_name>
Where: <directory_name> ::= 1 to 65 character quoted ASCII string

:DISK:DELeTe

Command :DISK:DELeTe <file_name>
Where: <file_name> ::= 1 to 8 character quoted ASCII string, if DOS,
 can have a 0 to 3 character extension

:DISK:DIRectory?

Query :DISK:DIRectory?
Returned Format [:DISK:DIRectory] <number_of_files><cr><lf><directory>
Where: <number_of_files> ::= integer, number of files (that follow)
 in the root directory
 <directory> ::= {<filename>,<ext>,<date>,<time>,<size>,
 <description><cr><lf>...}
 <ext> ::= {SETup | WAVEform | TEXT | PIXel}
 <date> ::= DDMMYY
 <time> ::= HH:MM:SS
 <size> ::= an integer
 <description> ::= LIF format: Model + C1 + HP545XX + Source,
 DOS format: Model = DOS file (no description)

:DISK:FORMat

Command :DISK:FORMat <format_type>
Where: <format_type> ::= {DOS | LIF}

Programmer's Quick Reference Guide

:DISK:LOAD

Command
Where:

```
:DISK:LOAD <file_name>,<destination> [,<format>]
<filename> ::= quoted ascii string DOS compatible filename.
    1 to 8 character ASCII string, if DOS, may have a 0 to 3
    character extension
    Either .wav or .txt may be used as a suffix after the
    filename. If no file suffix is specified, the default is
    .wav.
<destination> ::= {WMEemory {1 | 2 | 3 | 4 } |
    SETup{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9} |
    PMEMory{ 1 | 2}}
<format>::= {TEXT | INTernal}
```

:DISK:MDIRectory

Command
Where:

```
:DISK:MDirectory <directory_name>
<directory_name> ::= 1 to 65 character quoted ASCII string
```

:DISK:PWD?

Query
Returned Format:

```
:DISK:PWD?
[:DISK:PWD] <present_working_directory>
```

:DISK:SIMage

Command
Where

```
:DISK:SIMage <file_name> [|,<format>| [|,<compression>|
|,<rendering>|]]
<filename>::= 1 to 8 character quoted ASCII string, if DOS,
    can have a 0 to 3 character extension
<format>::={TIFF | PCX | EPS}
<compression>::={ON | OFF}
<rendering>::={BW | COLOR [ GREen]}
```

:DISK:STORe**Command****Where:**

```
:DISK:STORe <source>,<file_name>[,<format>]
<source> ::= {CHANnel<n> | WMEMory{1 | 2 | 3 | 4} |
  SETupN {0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9} |
  PMEMory {1 | 2}
<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
  (HP 54540C/54542C)
<filename> ::= a descriptive name of the file up to 8
  characters long
<format> ::= {INTernal | TEXT [, {<XYPairs> | <YVALues> |
  <VERBoSe>}]}
```

:DISPlay:COLumn**Command****Query****Returned Format****Where:**

```
:DISPlay:COLumn <column_number>
:DISPlay:COLumn?
[:DISPlay:COLumn] <column_number><NL>
<column_number> ::= integer, 0 through 78
```

:DISPlay:CONNect**Command****Query****Returned Format**

```
:DISPlay:CONNect {{OFF |0} | {ON | 1}}
:DISPlay:CONNect?
[:DISPlay:CONNect] {0 | 1}<NL>
```

:DISPlay:DATA**Command****Query****Returned Format****Where:**

```
:DISPlay:DATA <binary_block>
:DISPlay:DATA?
[:DISPlay:DATA] #800016576<16576 bytes of binary data><NL>
<binary_block> ::= definite block data in # format
```

:DISPlay:FORMat**Command****Query****Returned Format**

```
:DISPlay:FORMat {1 | 2 | 4}
:DISPlay:FORMat?
[:DISPlay:FORMat] {1 | 2 | 4}<NL>
```

 Programmer's Quick Reference Guide

:DISPlay:GRATicule

Command	:DISPlay:GRATicule {AXES GRID FRAME OFF}
Query	:DISPlay:GRATicule?
Returned Format	[:DISPlay:GRATicule] {AXES GRID FRAME OFF}<NL>

:DISPlay:INVerse

Command	:DISPlay:INVerse {{OFF 0} {ON 1}}
Query	:DISPlay:INVerse?
Returned Format	[:DISPlay:INVerse] {0 1}<NL>

:DISPlay:LINE

Command	:DISPlay:LINE <ascii_string>
Where:	<ascii_string> ::= any series of ascii characters enclosed in quotes

:DISPlay:MASK

Command	:DISPlay:MASK <mask_argument>
Query	:DISPlay:MASK?
Returned Format	[:DISPlay:MASK] <mask_argument><NL>
Where:	<mask_argument> ::= integer, 0 through 255

:DISPlay:PERStistence

Command	:DISPlay:PERStistence {SINGle INFinite 0 {0.5 to 10} 11}}
Query	:DISPlay:PERStistence?
Returned Format	[:DISPlay:PERStistence] <value><NL>
Where:	<value> ::= exponential, {0 (minimum) {0.5 to 10} 11 (infinite)} in the repetitive mode, {SINGle INFinite} in real-time mode

:DISPlay:ROW

Command :DISPlay:ROW <row_number>
Query :DISPlay:ROW?
Returned Format [:DISPlay:ROW] <row_number><NL>
Where: <row_number> ::= integer, 0 through 24

:DISPlay:SCReen

Command :DISPlay:SCReen {{OFF | 0} | {ON | 1}}
Query :DISPlay:SCReen?
Returned Format [:DISPlay:SCReen] {0 | 1}<NL>

:DISPlay:SETup?

Query :DISPlay:SETup?
Returned Format :DISP:COL <column_number>;
 CONN {0 | 1};
 FORM {1 | 2 | 4};
 GRAT {AXES | FRAM | GRID | OFF};
 INV {0 | 1};
 MASK <mask_argument>;
 PERS <pers_argument>;
 ROW <row_number>;
 SCR {0 | 1};
 SOUR PMEM {0 | 1 | 2};
 MARK {0 | 1}<NL>

Where: <column_number> ::= integer, 0 through 78
 <mask_argument> ::= integer, 0 to 255
 <pers_argument> ::= exponential, {0 | .5 to 10 | 11} in the
 repetitive mode, {SINGLE | INFinite} in the real-time mode
 <row_number> ::= integer, 0 to 24

:DISPlay:SOURce

Command :DISPlay:SOURce PMEMory{0 | 1 | 2 }
Query :DISPlay:SOURce?
Returned Format [:DISPlay:SOURce] PMEMory{0 | 1 | 2 }<NL>

Programmer's Quick Reference Guide

:DISPlay:STRing

Command :DISPlay:STRing <string_argument>
Where: <string_argument> ::= text string up to 90 characters

:DISPlay:TEXT

Command :DISPlay:TEXT BLANK

:DISPlay:{MARKer|TMARker|VMARker}

Command :DISPlay:{MARKer|TMARker|VMARker} {{OFF | 0} | {ON | 1}}
Query :DISPlay:MARKer|TMARker|VMARker?
Returned Format [:DISPlay:MARKer|TMARker|VMARker] {0 | 1}<NL>

:ERASe

Command :ERASe {PMEMory0<pmemory_num>}
Where: <pmemory_num> ::= integer, 1 or 2 @MODULE RULE =

:EXTernal:COUPling

Command :EXTernal:COUPling {AC | DC | DCFifty}
Query :EXTernal:COUPling?
Returned Format [:EXTernal:COUPling] {AC | DC | DCFifty}<NL>

:EXTernal:HFRejeCt

Command :EXTernal:HFRejeCt {{OFF | 0} | {ON | 1}}
Query :EXTernal:HFRejeCt?
Returned Format [:EXTernal:HFRejeCt] {0 | 1}<NL>

Programmer's Quick Reference Guide

:EXTErnal:LFRejEct

Command :EXTErnal:LFRejEct {{OFF | 0} | {ON | 1}}
 Query :EXTErnal:LFRejEct?
 Returned Format [:EXTErnal:LFRejEct] {0 | 1}<NL>

:EXTErnal:PROBE

Command :EXTErnal:PROBE <probe_argument>
 Query :EXTErnal:PROBE?
 Returned Format [:EXTErnal:PROBE] <probe_argument><NL>
 <probe_argument> ::= exponential, 0.9 to 1000.0

:EXTErnal:RANGe

Command :EXTErnal:RANGe <range_argument>
 Query :EXTErnal:RANGe?
 Returned Format [:EXTErnal:RANGe] <range_argument><NL>
 <range_argument> ::= exponential, full-scale range value

:EXTErnal:SETup

Query :EXTErnal:SETup?
 Returned Format :EXT:COUP {AC|DC|DCF};
 HFR {0 | 1};
 LFR {0 | 1};
 PROB <probe_argument>;
 RANG<range_argument><NL>
 Where: <range_argument> ::= exponential, full-scale range value
 <probe_argument> ::= exponential, 0.9 to 1000.0

:FUNCTion{1 | 2 | 3 | 4}:ADD

Command :FUNCTion{1 | 2 | 3 | 4}:ADD <operand>,<operand>
 Where: <operand> ::= {CHANnel<n> | WMEMory{1 | 2 | 3 | 4}}
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

Programmer's Quick Reference Guide

:FUNCTION{1 | 2 | 3 | 4}:DIFF

Command :FUNCTION{1 | 2 | 3 | 4}:DIFF <operand>
Where: <operand> ::= {CHANnel<n> | WMemory{1 | 2 | 3 | 4}}
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:FUNCTION{1 | 2 | 3 | 4}:DISPLay

Command :FUNCTION{1 | 2 | 3 | 4}:DISPLay {{OFF | 0} | {ON | 1}}
Query :FUNCTION{1 | 2 | 3 | 4}:DISPLay?
Returned Format [:FUNCTION{1 | 2 | 3 | 4}[:DISPLay] {0 | 1}<NL>

:FUNCTION{1 | 2 | 3 | 4}:FFT

Command :FUNCTION{1 | 2 | 3 | 4}:FFT <operand>
Where: <operand> ::= {CHANnel<n> | WMemory{1 | 2 | 3 | 4}}
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:FUNCTION{1 | 2 | 3 | 4}:FREQuency

Command :FUNCTION{1 | 2 | 3 | 4}:FREQuency <frequency_argument>
Query :FUNCTION{1 | 2 | 3 | 4}:FREQuency?
Returned Format [:FUNCTION{1 | 2 | 3 | 4}:FREQuency] <frequency_argument><NL>
Where: <frequency_argument> ::= exponential, center frequency from
 0 Hz to 1.5X of frequency span

:FUNCTION{1 | 2 | 3 | 4}:INTegrate

Command :FUNCTION{1 | 2 | 3 | 4}:INTegrate <operand>
Where: <operand> ::= {CHANnel<n> | WMemory{1 | 2 | 3 | 4}}
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:FUNCTION{1 | 2 | 3 | 4}:INVert **Command****Where:**

```
:FUNCTION{1 | 2 | 3 | 4}:INVert <operand>
<operand> ::= {CHANnel<n> | WMemory{1 | 2 | 3 | 4}}
<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
      (HP 54540C/54542C)
```

:FUNCTION{1 | 2 | 3 | 4}:LEVel**Command****Query****Returned Format****Where:**

```
:FUNCTION{1 | 2 | 3 | 4}:LEVel <level_argument>
:FUNCTION{1 | 2 | 3 | 4}:LEVel?
[:FUNCTION{1 | 2 | 3 | 4}:LEVel] <level_argument><NL>
<level_argument> ::= exponential, level value 0 to + or -
      600 dbm
```

:FUNCTION{1 | 2 | 3 | 4}:MAGNify **Command****Query****Returned Format**

```
:FUNCTION{1 | 2 | 3 | 4}:MAGNify {{OFF | 0} | {ON | 1}}
:FUNCTION{1 | 2 | 3 | 4}:MAGNify?
[:FUNCTION{1 | 2 | 3 | 4}:MAGNify] {0 | 1}<NL>
```

:FUNCTION{1|2|3|4}:MODE?**Query****Returned Format****Where:**

```
:FUNCTION{1 | 2 | 3 | 4}:MODE?
[:FUNCTION{1 | 2 | 3 | 4}:MODE]
<operation>,<operand>[,<operand>]
<operation> ::= {ADD | SUBtract | MULTiply | VERSus | ONLY |
      INVert | INTEgrate | DIFF | FFT}
<operand> ::= {CHANnel<n> | WMemory{1 | 2 | 3 | 4}}
<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
      (HP 54540C/54542C)
```

:FUNCTION{1 | 2 | 3 | 4}:MULTiply **Command****Where:**

```
:FUNCTION{1 | 2 | 3 | 4}:MULTiply <operand>,<operand>
<operand> ::= {CHANnel<n> | WMemory{1 | 2 | 3 | 4}}
<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
      (HP 54540C/54542C)
```

Programmer's Quick Reference Guide

:FUNCTION{1 | 2 | 3 | 4}:OFFSet

Command :FUNCTION{1 | 2 | 3 | 4}:OFFSet <offset_argument>
Query :FUNCTION{1 | 2 | 3 | 4}:OFFSet?
Returned Format [:FUNCTION{1 | 2 | 3 | 4}:OFFSet] <offset_argument><NL>
Where: <offset_argument> ::= exponential, offset value of 0 to
 $\pm$ voltage full scale, or 0 to $\pm$ 200 dbm for FFT

:FUNCTION{1 | 2 | 3 | 4}:ONLY

Command :FUNCTION{1 | 2 | 3 | 4}:ONLY <operand>
Where: <operand> ::= {CHANnel<n> | WMEMory{1 | 2 | 3 | 4}}
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
 (HP 54540C/54542C)

:FUNCTION{1 | 2 | 3 | 4}:PEAK

Command :FUNCTION{1 | 2 | 3 | 4}:PEAK <peak1_number>,<peak2_number>
Query :FUNCTION{1 | 2 | 3 | 4}:PEAK?
Returned Format [:FUNCTION{1 | 2 | 3 | 4}:PEAK]
 <peak1_number>,<peak2_number><NL>
Where: <peak1_number> ::= integer, 1 through 99
 <peak2_number> ::= integer, 1 through 99

:FUNCTION{1 | 2 | 3 | 4}:POINTs

Command :FUNCTION{1 | 2 | 3 | 4}:POINTs <points_argument>
Query :FUNCTION{1 | 2 | 3 | 4}:POINTs?
Returned Format [:FUNCTION{1 | 2 | 3 | 4}:POINTs] <points_argument><NL>
Where: <points_argument> ::= integer, 512, 1024, 2048, 4096, 8192,
 16384, or 32768

:FUNCTION{1 | 2 | 3 | 4}:RANGe

Command :FUNCTION{1 | 2 | 3 | 4}:RANGe <range_argument>
Query :FUNCTION{1 | 2 | 3 | 4}:RANGe?
Returned Format [:FUNCTION{1 | 2 | 3 | 4}:RANGe] <range_argument><NL>
Where: <range_argument> ::= exponential full scale vertical range in
 volts, dB, or dBm

:FUNCTION{1 | 2 | 3 | 4}:SETup?

Query
Returned Format

```
:FUNCTION{1 | 2 | 3 | 4}:SETup?
:FUNCTION{1 | 2 | 3 | 4}:DISP {0|1};
  {ADD | SUBT | MULT | VERS | ONLY | INV | INT | DIFF |
   FFT} {CHAN<n> | WMEM {1 | 2 | 3 | 4} [, {CHAN<n> |
   WMEM {1 | 2 | 3 | 4}}];
  OFF <offset_argument>;
  RANG <range_argument>;
  FREQ <frequency_argument>; (FFT only)
  LEV <level_argument>; (FFT only)
  MAGN {0 | 1}; (FFT only)
  PEAK <peak1_number>, <peak2_number>; (FFT only)
  POIN <points_argument>; (FFT only)
  SPAN <span_argument>; (FFT only)
  WIND {RECT | HANN | FLAT} (FFT only) <NL>
```

Where:

```
<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
      (HP 54540C/54542C)
<offset_argument> ::= exponential offset value of 0 to ± 200
                    dBm for FFT function
<range_argument> ::= exponential full scale vertical range in
                    volts, dB, or dBm
<frequency_argument> ::= exponential center frequency from
                       0 Hz to 1.5X of frequency span
<level_argument> ::= exponential level value 0 to ± 600 dBm
<peak1_number> ::= integer, 1 through 99
<peak2_number> ::= integer, 1 through 99
<points_argument> ::= integer, 512, 1024, 2048, 4096, 8192,
                    16384, or 32768
<span_argument> ::= exponential number in hertz
```

:FUNCTION{1 | 2 | 3 | 4}:SPAN

Command
Query
Returned Format
Where:

```
:FUNCTION{1 | 2 | 3 | 4}:SPAN <span_argument>
:FUNCTION{1 | 2 | 3 | 4}:SPAN?
[:FUNCTION{1 | 2 | 3 | 4}:SPAN] <span_argument> <NL>
<span_argument> ::= exponential number in hertz
```

Programmer's Quick Reference Guide

:FUNCTION{1 | 2 | 3 | 4}:SUBTRACT

Command
Where:

```
:FUNCTION{1 | 2 | 3 | 4}:SUBTRACT <operand1>,<operand2>
<operand1> ::= {CHANNEL<n> | WMemory{1 | 2 | 3 | 4}}
<operand2> ::= {CHANNEL<n> | WMemory{1 | 2 | 3 | 4}}
<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
(HP 54540C/54542C)
```

:FUNCTION{1 | 2 | 3 | 4}:VERSUS

Command
Where:

```
:FUNCTION{1 | 2 | 3 | 4}:VERSUS <Y_operand>,<X_operand>
<Y_operand1> ::= {CHANNEL<n> | WMemory{1 | 2 | 3 | 4}}
<X_operand2> ::= {CHANNEL<n> | WMemory{1 | 2 | 3 | 4}}
<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
(HP 54540C/54542C)
```

:FUNCTION{1 | 2 | 3 | 4}:WINDOW

Command

```
:FUNCTION{1 | 2 | 3 | 4}:WINDOW {RECTangular | HANNing |
FLATtop}
```

Query

```
:FUNCTION:WINDOW?
```

Returned Format

```
[ :FUNCTION{1|2|3|4:WINDOW] {RECTangular | HANNing |
FLATtop}<NL>
```

:HARDcopy:LENGTH

Command

```
:HARDcopy:LENGTH {11 | 11.6 | LETTER | A4}
```

Query

```
:HARDcopy:LENGTH?
```

Returned Format

```
[ :HARDcopy:LENGTH] {11 | 11.6 | LETTER | A4}<NL>
```

Programmer's Quick Reference Guide

:HARDcopy:MODE**Command**

```
:HARDcopy:MODE {THINKJET | DJ5XBW75DPI | DJ5XBW100DPI |
DJ5XBW150DPI | DJ5XBW300DPI | DJET75DPI | DJET100DPI |
DJET150DPI | DJET300DPI | LASERJET_II | LASERJET_IIP |
PJETXL300 | PJETXL | PAINTJET | COLORPRO | HP7475A |
HP7470A | HP7550A | DJET310C | DJET320C | DJET500C |
DJET540C | DJET560C | DJET600C | DJET660C | DJET850C |
DJET855C | P1200C | QUIETJETALT | EPSON5000}
```

Query

```
:HARDcopy:MODE?
```

Returned Format

```
[ :HARDcopy:MODE ] {THINKJET | DJ5XBW75DPI | DJ5XBW100DPI |
DJ5XBW150DPI | DJ5XBW300DPI | DJET75DPI | DJET100DPI |
DJET150DPI | DJET300DPI | LASERJET_II | LASERJET_IIP |
PJETXL300 | PJETXL | PAINTJET | COLORPRO | HP7475A |
HP7470A | HP7550A | DJET310C | DJET320C | DJET500C |
DJET540C | DJET560C | DJET600C | DJET660C | DJET850C |
DJET855C | P1200C | QUIETJETALT | EPSON5000}<NL>
```

:HARDcopy:PAGE**Command**

```
:HARDcopy:PAGE {MANual | AUTomatic}
```

Query

```
:HARDcopy:PAGE?
```

Returned Format

```
[ :HARDcopy:PAGE ] {MANual | AUTomatic}<NL>
```

:HARDcopy:PLOT:AREA**Command**

```
:HARDcopy:PLOT:AREA {ALL | DISPlay | FACTors | GRATICule |
LABeled}
```

Query

```
:HARDcopy:PLOT:AREA?
```

Returned Format

```
[ :HARDcopy:PLOT:AREA ] {ALL | DISPlay | FACTors | GRATICule |
LABeled}<NL>
```

:HARDcopy:PLOT:INITialize**Command**

```
:HARDcopy:PLOT:INITialize {{OFF | 0} | {ON | 1}}
```

Query

```
:HARDcopy:PLOT:INITialize?
```

Returned Format

```
[ :HARDcopy:PLOT:INITialize ] {0 | 1}<NL>
```

Programmer's Quick Reference Guide

:HARDcopy:PLOT:{PEN|COLor}

Command	:HARDcopy:PLOT:{PEN COLor } <item>,<pen_number>
Query	:HARDcopy:PLOT:{PEN COLor}? <item>
Returned Format	[:HARDcopy:PLOT:{PEN COLor}] <pen_number><NL>
Where:	<item> ::= {CHANnel<n> WMEMory{1 2 3 4} FUNCTION{1 2 3 4} PMEMory{1 2} Y{1 2}Marker X{1 2}Marker GRATicule TRIGger TIMEbase MEASure TITLes} <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4 (HP 54540C/54542C) <pen_number> ::= integer, 0 through 8

:LER?

Query	:LER?
Returned Format	[:LER] {0 1}<NL>

:LTER?

Query	:LTER?
Returned Format	[:LTER] {0 1}<NL>

:MARKer:DISPlay

Command	:MARKer:DISPlay {{OFF 0} {ON 1}}
Query	:MARKer:DISPlay?
Returned Format	[:MARKer:DISPlay] {0 1}<NL>

:MARKer:MODE

Command	:MARKer:MODE {MANual WAVeform}
Query	:MARKer:MODE?
Returned Format	[:MARKer:MODE] {MANual WAVeform}<NL>

:MARKer:SETup?

Query
Returned Format

```

:MARKer:SETup?
:MARK:MODE {MAN| WAV};DISP {0 | 1};
XDEL <xdelta>;
X1P <xposition_argument>;
X2P <xposition_argument>;
X1Y1 {CHAN<n> | FUNC{1 | 2 | 3 | 4} | WMEM{1 | 2 | 3 | 4}};
X2Y2 {CHAN<n> | FUNC{1 | 2 | 3 | 4} | WMEM{1 | 2 | 3 | 4}};
Y1P <yposition_argument>;      (MAN mode only)
Y2P <yposition_argument>;      (MAN mode only)
YDEL <ydelta>;                (MAN mode only)
Y1P <yposition_argument>;      (MAN mode only)
Y2P <yposition_argument><NL>    (MAN mode only)

```

Where:

```

<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4
      (HP 54540C/54542C)
<xdelta> ::= exponential, difference in seconds between x1 and
           x2 markers
<xposition_argument>::= exponential, xmarker position in
           seconds or hertz
<ydelta>::= exponential, difference between y1 and y2 markers
<yposition_argument>::= exponential, ymarker position in volts
           or power

```

:MARKer:X1Position

Command
Query
Returned Format
Where:

```

:MARKer:X1Position <xposition_argument>
:MARKer:X1Position?
[:MARKer:X1Position] <xposition_argument><NL>
<xposition_argument>::= exponential, xmarker time in seconds,
           or frequency in hertz

```

:MARKer:X2Position

Command
Query
Returned Format
Where:

```

:MARKer:X2Position <xposition_argument>
:MARKer:X2Position?
[:MARKer:X2Position] <xposition_argument><NL>
<xposition_argument> ::= exponential, xmarker time in
           seconds,
           or frequency in hertz

```

Programmer's Quick Reference Guide

:MARKer:X1Y1source

Command :MARKer:X1Y1source {CHANnel<n> | {FUNctIon{1 | 2 | 3 | 4} | WMEMory{1 | 2 | 3 | 4}}
Query :MARKer:X1Y1source?
Returned Format [:MARKer:X1Y1source] {CHANnel<n>| {FUNctIon{1 | 2 | 3 | 4} | WMEMory{1 | 2 | 3 | 4}}<NL>
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4 (HP 54540C/54542C)

:MARKer:X2Y2source

Command :MARKer:X2Y2source {CHANnel<n> | {FUNctIon{1 | 2 | 3 | 4} | WMEMory{1 | 2 | 3 | 4}}
Query :MARKer:X2Y2source?
Returned Format [:MARKer:X2Y2source] {CHANnel<n>| {FUNctIon{1 | 2 | 3 | 4} | WMEMory{1 | 2 | 3 | 4}}<NL>
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4 (HP 54540C/54542C)

:MARKer:XDELta?

Query :MARKer:XDELta?
Returned Format [:MARKer:XDELta] <xdelta><NL>
Where: <xdelta> ::= exponential, difference between x1 and x2 markers

:MARKer:Y1Position (Command ignored in waveform mode)

Command :MARKer:Y1Position <yposition_argument>
Query :MARKer:Y1Position?
Returned Format [:MARKer:Y1Position] <yposition_argument><NL>
Where: <yposition_argument> ::= exponential, ymarker level in volts, or power in dBm

Programmer's Quick Reference Guide

:MARKer:Y2Position (Command ignored in waveform mode)

Command :MARKer:Y2Position <yposition_argument>
Query :MARKer:Y2Position?
Returned Format [:MARKer:Y2Position] <yposition_argument><NL>
Where: <yposition_argument> ::= exponential, ymarker level in volts,
or power in dBm

:MARKer:YDELta?

Query :MARKer:YDELta?
Returned Format [:MARKer:YDELta] <ydelta><NL>
Where: <ydelta> ::= exponential, difference between y1 and y2 markers

:MEASure:ALL?

Query :MEASure:ALL?
Returned Format [:MEASure] [:DELay] <result>; [:DUTYcycle] <result>;
[:FALLtime] <result>; [:FREQuency] <result>; [:NWIDth]
<result>; [:OVERshoot] <result>; [:PERiod] <result>;
[:PREShoot] <result>; [:PWIDth] <result>; [:RISetime]
<result>; [:VACRms] <result>; [:VAMplitude] <result>;
[:VAverage] <result>; [:VBASe] <result>; [:VDCRms] <result>;
[:VMAX] <result>; [:VMIN] <result>; [:VPP] <result>; [:VTOP]
<result>;<NL>
Where: <result> ::=exponential, individual measurement results

:MEASure:COMPare

Command :MEASure:COMPare <measurement>,<upper_limit>,<lower_limit>
Query :MEASure:COMPare? <measurement>
Returned Format [:MEASure:COMPare]
<measurement>,<upper_limit>,<lower_limit><NL>
Where: <measurement> ::= {DELay | DUTYcycle | FALLtime | FREQuency |
NWIDth | OVERshoot | PERiod | PRESShoot | PWIDth | RISetime |
VACRms | VAMplitude | VAverage | VBASe | VDCRms | VMAX |
VMIN | VPP | VTOP}
<upper_limit> ::= exponential, high limit value
<lower_limit> ::=exponential, low limit value

Programmer's Quick Reference Guide

:MEASure:CURSor?

Query
Returned Format
Where:

```
:MEASure:CURSor? {DELTA | START | STOP}
[:MEASure:CURSor] <time>,<voltage><NL>
<time> ::= exponential, delta time, y1marker time, or y2marker
time
<voltage> ::= exponential, delta voltage, x1marker voltage, or
x2marker voltage
```

:MEASure:DEFine

Command
Query
Returned Format
Where:

```
:MEASure:DEFine <define_argument>
:MEASure:DEFine? {DELay | PWIDTH | NWIDTH}
[:MEASure:DEFine] <define_argument><NL>
<define_argument> ::= {DELay,<polarity>,<edge_number>,<level>,<
polarity>,<edge_number>,<level> | PWIDTH,<level> |
NWIDTH,<level>}
<polarity> ::= {POSitive | NEGative}
<edge_number> ::= integer, 1 to 4000 specifying a displayed
edge
<level> ::= {MIDDLE | UPPER | LOWER}
```

:MEASure:DELay

Command
Query
Returned Format
Where:

```
:MEASure:DELay
:MEASure:DELay?
[:MEASure:DELay] <value><NL>
<value> ::= exponential, time value in seconds
```

Programmer's Quick Reference Guide

:MEASure:DESTination

Command Query :MEASure:DESTination <source_argument>,<destination_argument>
 :MEASure:DESTination? {CHANnel<n> | FUNction{1 | 2 | 3 | 4} | SCReen}?

Returned Format [:MEASure:DESTination] {{CHANnel<n> | {FUNction{1 | 2 | 3 | 4} | SCReen}} {WMEMory{1 | 2 | 3 | 4} | PMEMory{1 | 2} | MULTiple | OFF}<NL>

Where: <source_argument> ::= {{SCReen,{OFF | PMEMory{1 | 2}}}| {{CHANnel<n> | FUNction{1 | 2 | 3 | 4}}}}
 <distination_argument> ::= {WMEMory{1 | 2 | 3 | 4} | MULTiple | OFF}
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4 (HP 54540C/54542C)

:MEASure:DUTcycle

Command Query :MEASure:DUTcycle
 :MEASure:DUTcycle?

Returned Format [:MEASure:DUTcycle] <value><NL>

Where: <value> ::= exponential, ratio of positive pulse width to period

:MEASure:ESTArt

Command Query :MEASure:ESTArt <slope_and_occurrence>
 :MEASure:ESTArt?

Returned Format [:MEASure:ESTArt] <slope_and_occurrence><NL>

Where: <slope_and_occurrence> ::= integer, -4000 to 4000 (excluding 0)

:MEASure:ESTOp

Command Query :MEASure:ESTOp <slope_and_occurrence>
 :MEASure:ESTOp?

Returned Format [:MEASure:ESTOp] <slope_and_occurrence><NL>

Where: <slope_and_occurrence> ::= integer, -4000 to 4000 (excluding 0)

Programmer's Quick Reference Guide

:MEASure:EANalysis

Command :MEASure:EANalysis {{OFF | 0} | {ON | 1}}
 Query :MEASure:EANalysis?
 Returned Format [:MEASure:EANalysis] {0 | 1}<NL>

:MEASure:FALLtime

Command :MEASure:FALLtime
 Query :MEASure:FALLtime?
 Returned Format [:MEASure:FALLtime] <value><NL>
 Where: <value> ::= exponential, time value in seconds between lower
 threshold and upper threshold voltage points

:MEASure:FREQuency

Command :MEASure:FREQuency
 Query :MEASure:FREQuency?
 Returned Format [:MEASure:FREQuency] <value><NL>
 Where: <value> ::= exponential, frequency in hertz

:MEASure:LIMittest

Command :MEASure:LIMittest {MEASure | OFF}

:MEASure:LOWer

Command :MEASure:LOWer [<source>,<lower_threshold>
 Query :MEASure:LOWer? [<source>]
 Returned Format [:MEASure:LOWer] <source>,<lower_threshold><NL>
 Where: <source> ::= {CHANnel<n> | FUNction{1 | 2 | 3 | 4} |
 WMemory{1 | 2 | 3 | 4}}
 <lower_threshold> ::= integer, user defined lower threshold
 in percent or volts (selected by :MEASure:UNITs)
 <n> ::= integer 1 or 2 (HP 54520C/54522C), or 1 through 4
 (HP 54540C/54542C)

Programmer's Quick Reference Guide

:MEASure:MODE

Command	:MEASure:MODE {STANDARD USER}
Query	:MEASure:MODE?
Returned Format	[[:MEASure:MODE] {STANDARD USER}<NL>

:MEASure:MWINDow

Command	:MEASure:MWINDow {MARKers SCReen}
Query	:MEASure:MWINDow?
Returned Format	[[:MEASure:MWINDows] {MARKers SCReen}<NL>

:MEASure:NWIDth

Command	:MEASure:NWIDth
Query	:MEASure:NWIDth?
Returned Format	[[:MEASure:NWIDth] <value><NL>
Where:	<value> ::= exponential, negative pulse width in seconds

:MEASure:OVERshoot

Command	:MEASure:OVERshoot
Query	:MEASure:OVERshoot?
Returned Format	[[:MEASure:OVERshoot] <value><NL>
Where:	<value> ::= exponential, ratio of overshoot to Vamplitude

:MEASure:PERiod

Command	:MEASure:PERiod
Query	:MEASure:PERiod?
Returned Format	[[:MEASure:PERiod] <value><NL>
Where:	<value> ::= exponential, waveform period in seconds

:MEASure:POSTfailure

Command	:MEASure:POSTfailure {CONTInue STOP}
Query	:MEASure:POSTfailure?
Returned Format	[[:MEASure:POSTfailure] {CONTInue STOP}<NL>

Programmer's Quick Reference Guide

:MEASure:PREShoot

Command	:MEASure:PREShoot
Query	:MEASure:PREShoot?
Returned Format	[:MEASure:PREShoot] <value><NL>
Where:	<value> ::= exponential, ratio of preshoot to Vamplitude

:MEASure:PWIDth

Command	:MEASure:PWIDth
Query	:MEASure:PWIDth?
Returned Format	[:MEASure:PWIDth] <value><NL>
Where:	<value> ::= exponential, width of positive pulse in sseconds

:MEASure:RESults?

Query	:MEASure:RESults?
Returned Format	[:MEASure:RESults] <number_of_meas>[;<measurement>]...<NL>
Where:	<number_of_meas> ::= integer, number of measurements displayed on the screen, 0 through 23 <measurement> ::= { DELay <result> DUTycycle <result> FALLtime <result> FREQuency <result> NWIDth <result> OVERshoot <result> PERiod <result> PREShoot <result> PWIDth <result> RISetime <result> TMAX <result> TMIN <result> TVOLt <result> VACRms <result> VAMplitude <result> VAVerage <result> VBASe <result> VDCRms <result> VMAX <result> VMIN <result> VPP <result> VTIME <result> VTOP <result> } <result> ::=exponential, individual measurement results

:MEASure:RISetime

Command	:MEASure:RISetime
Query	:MEASure:RISetime?
Returned Format	[:MEASure:RISetime] <value><NL>
Where:	<value> ::= exponential, rise time in seconds

:MEASure:SCRatch

Command :MEASure:SCRatch

:MEASure:SOURce

Command :MEASure:SOURce <source>[,<source>]
Query :MEASure:SOURce?
Returned Format [:MEASure:SOURce] <source>[,<source>]<NL>
Where: <source> ::= {CHANnel<n> | FUNCTION{1 | 2 | 3 | 4} |
WMEMory{1 | 2 | 3 | 4}}
<n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
(HP 54540C/54542C)

:MEASure:STATistics

Command :MEASure:STATistics {{OFF | 0} | {ON | 1}}
Query :MEASure:STATistics?
Returned Format [:MEASure:STATistics] {0 | 1}<NL>

MEASure:STATistics:MODE

Command :MEASure:STATistics:MODE {NORMAL | SDEVIation}
Query :MEASure:STATistics:MODE?
Returned Format [:MEASure:STATistics:MODE] {NORMAL | SDEVIation}<NL>

:MEASure:TDELta

Query :MEASure:TDELta?
Returned Format [:MEASure:TDELta] <value><NL>
Where: <value> ::= exponential, difference between x2 and x1 markers

:MEASure:TMAX?

Query :MEASure:TMAX?
Returned Format [:MEASure:TMAX] <time><NL>
Where: <time> ::= exponential, time at maximum voltage

Programmer's Quick Reference Guide

:MEASure:TMIN?

Query
Returned Format
Where:

```
:MEASure:TMIN?
[:MEASure:TMIN] <time><NL>
<time> ::= exponential, time at minimum voltage
```

:MEASure:TSTArt

Command
Query
Returned Format
Where:

```
:MEASure:TSTArt <tstart _argument>
:MEASure:TSTArt?
[:MEASure:TSTArt] <tstart _argument><NL>
<tstart_argument> ::= exponential, time at x1marker in seconds
```

:MEASure:TSTOp

Command
Query
Returned Format
Where:

```
:MEASure:TSTOp <tstop_argument>
:MEASure:TSTOp?
[:MEASure:TSTOp] <tstop _argument><NL>
<tstop_argument> ::= exponential, time at x2marker in seconds
```

:MEASure:TVOLT?

Query
Returned Format
Where:

```
:MEASure:TVOLT? <tvolt_argument>,<slope_and_occurrence>
[:MEASure:TVOLT] <time><NL>
<tvolt_argument> ::= real number representing positive or
negative voltage level that the waveform must cross
<slope_and_occurrence> ::= slope is the direction of the
waveform when <voltage> is crossed - rising (space character
or +) or falling (-)
occurrence is the number of crossings to be reported (if one
- the first crossing is reported, if two - the second
crossing is reported
<time> ::= exponential, time in seconds of specified voltage
crossing
```

Programmer's Quick Reference Guide

:MEASure:UNITs

Command :MEASure:UNITs {PERCent | VOLTs}
Query :MEASure:UNITs?
Returned Format [:MEASure:UNITs] {PERCent | VOLTs}<NL>

:MEASure:UPPer

Command :MEASure:UPPer [<source,>]<upper_threshold>
Query :MEASure:UPPer? [<source>]
Returned Format [:MEASure:UPPer] <source>,<upper_threshold><NL>
Where: <source> ::= {CHANnel<n> | FUNction{1 | 2 | 3 | 4} |
 WMEMory{1 | 2 | 3 | 4}}
 <upper_threshold> ::= integer, user defined upper threshold in
 percent or volts (selected by :MEASure:UNITs)
 <n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
 (HP 54540C/54542C)

:MEASure:VACRms

Command :MEASure:VACRms
Query :MEASure:VACRms?
Returned Format [:MEASure:VACRms] <value><NL>
Where: <value> ::= exponential, calculated ac rms voltage

:MEASure:VAMPlitude

Command :MEASure:VAMPlitude
Query :MEASure:VAMPlitude?
Returned Format [:MEASure:VAMPlitude] <value><NL>
Where: <value> ::= exponential, difference between top and base
 voltages

:MEASure:VAVerage

Command :MEASure:VAVerage
Query :MEASure:VAVerage?
Returned Format [:MEASure:VAVerage] <value><NL>
Where: <value> ::= exponential, calculated average voltage

Programmer's Quick Reference Guide

:MEASure:VBASe

Command	:MEASure:VBASe
Query	:MEASure:VBASe?
Returned Format	[[:MEASure:VBASe] <value><NL>
Where:	<value> ::= exponential, voltage at base of selected waveform

:MEASure:VDCRms

Command	:MEASure:VDCRms
Query	:MEASure:VDCRms?
Returned Format	[[:MEASure:VDCRms] <value><NL>
Where:	<value> ::= exponential, calculated dc rms voltage

:MEASure:VDELta?

Query	:MEASure:VDELta?
Returned Format	[[:MEASure:VDELta] <value><NL>
Where:	<value> ::= exponential, delta V value in volts

:MEASure:VFIFty

Command	:MEASure:VFIFty
---------	-----------------

:MEASure:VMAX

Command	:MEASure:VMAX
Query	:MEASure:VMAX?
Returned Format	[[:MEASure:VMAX] <value><NL>
Where:	<value>::= exponential, maximum voltage of selected waveform

:MEASure:VMIN

Command	:MEASure:VMIN
Query	:MEASure:VMIN?
Returned Format	[[:MEASure:VMIN] <value><NL>
Where:	<value> ::= exponential, minimum voltage value of the selected waveform

Programmer's Quick Reference Guide

:MEASure:VPP

Command :MEASure:VPP
Query :MEASure:VPP?
Returned Format [:MEASure:VPP] <value><NL>
Where: <value> ::= exponential, voltage peak to peak

:MEASure:VRELative

Command :MEASure:VRELative <percent_argument>
Query :MEASure:VRELative?
Returned Format [:MEASure:VRELative] <percent_argument><NL>
Where: <percent_argument> ::= integer, Vmarker2 relative position in percent, from 0 through 100

:MEASure:VSTart

Command :MEASure:VSTart <vstart_argument>
Query :MEASure:VSTart?
Returned Format [:MEASure:VSTart] <vstart_argument><NL>
Where: <vstart_argument> ::= exponential, voltage at y1marker

:MEASure:VSTOp

Command :MEASure:VSTOp <vstop_argument>
Query :MEASure:VSTOp?
Returned Format [:MEASure:VSTOp] <vstop_argument><NL>
Where: <vstop_argument> ::= exponential, voltage at y2marker

:MEASure:VTIME?

Query :MEASure:VTIME? <vtime_argument>
Returned Format [:MEASure:VTIME] <voltage><NL>
Where: <vtime_argument> ::= real number representing the displayed time from the trigger in seconds
 <voltage> ::= exponential, voltage at specified time

Programmer's Quick Reference Guide

:MEASure:VTOP

Command :MEASure:VTOP
Query :MEASure:VTOP?
Returned Format [:MEASure:VTOP] <value><NL>
Where: <value> ::= exponential, voltage at the top of the waveform

:MEASure:WCOMpare:ALLowance

Command :MEASure:WCOMpare:ALLowance <allow_argument>
Query :MEASure:WCOMpare:ALLowance?
Returned Format [:MEASure:WCOMpare:ALLowance] <allow_argument><NL>
Where: <allow_argument> ::= real number representing number of vertical divisions of allowance, from 0.0 to 8.0

:MEASure:WCOMpare:COMpare

Command :MEASure:WCOMpare:COMpare {CHANnel<n> | FUNCTION{1 | 2 | 3 | 4}}, WMEMory{1 | 2 | 3 | 4}}
Query :MEASure:WCOMpare:COMpare?
Returned Format [:MEASure:WCOMpare:COMpare] {CHANnel<n> | FUNCTION{1 | 2 | 3 | 4}}, WMEMory{1 | 2 | 3 | 4}<NL>
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4 (HP 54540C/54542C)

:MEASure:WCOMpare:DESTination

Command :MEASure:WCOMpare:DESTination
 <source_argument>,<destination_argument>
Query :MEASure:WCOMpare:DESTination? {CHANnel<n> | FUNCTION{1 | 2 | 3 | 4} | SCReen}
Returned Format [:MEASure:WCOMpare:DESTination] {CHANnel<n>} | {FUNCTION{1 | 2 | 3 | 4} | SCReen}} {WMEMory{1 | 2 | 3 | 4} | PMEMory{1 | 2} | MULTiple | OFF}<NL>
Where: <source_argument> ::= {{SCReen,{OFF | PMEMory{1 | 2}}}} | {{CHANnel<n> | FUNCTION{1 | 2 | 3 | 4}}}
 <distination_argument> ::= {WMEMory{1 | 2 | 3 | 4} | MULTiple | PMEMory{1 | 2} | OFF}
 <n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4 (HP 54540C/54542C)

Programmer's Quick Reference Guide

:MEASure:WCOMpare:HALLowance

Command :MEASure:WCOMpare:HALLowance <allow_argument>
Query :MEASure:WCOMpare:HALLowance?
Returned Format [:MEASure:WCOMpare:HALLowance] <allow_argument><NL>
Where: <allow_argument> ::= real number representing number of horizontal divisions of allowance, from 0.0 to 8.0

:MEASure:WCOMpare:POSTfailure

Command :MEASure:WCOMpare:POSTfailure {CONTInue | STOP}
Query :MEASure:WCOMpare:POSTfailure?
Returned Format [:MEASure:WCOMpare:POSTfailure] {CONTInue | STOP}<NL>

:MEASure:WCOMpare:VALLowance

Command :MEASure:WCOMpare:VALLowance <allow_argument>
Query :MEASure:WCOMpare:VALLowance?
Returned Format [:MEASure:WCOMpare:VALLowance] <allow_argument><NL>
Where: <allow_argument> ::= real number representing number of vertical divisions of allowance, from 0.0 to 8.0

:MEASure:WCOMpare:WTESt

Command :MEASure:WCOMpare:WTESt {MEASure | OFF}
Query :MEASure:WCOMpare:WTESt?
Returned Format [:MEASure:WCOMpare:WTESt] {MEASure | OFF}<NL>

:MENU

Command :MENU {CHANnel<n>TIMEbase | TRIGger | DISK | DISPlay | DELTa | MATH | SAVE | MEASure | UTILity | SHOW}
Query :MENU?
Returned Format [:MENU] {CHANnel<n>TIMEbase | TRIGger | DISK | DISPlay | DELTa | MATH | SAVE | MEASure | UTILity | SHOW}
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4 (HP 54540C/54542C)

Programmer's Quick Reference Guide

:MERGe

Command
Where: :MERGe <pmemory_num>
 <pmemory_num> ::= integer, 1 or 2

:MMEMory:DISPlay

Command :MMEMory:DISPlay {{OFF | 0} | {ON | 1}}
Query :MMEMory:DISPlay?
Returned Format [:MMEMory:DISPlay] {0 | 1}<NL>

:MMEMory:FNUMber

Command :MMEMory:FNUMber <failure_number>
Query :MMEMory:FNUMber?
Returned Format [:MMEMory:FNUMber] <failure_number><NL>
Where: <failure_number> ::= integer, 0 to 665

:MMEMory:SOURce

Command :MMEMory:SOURce {CHANnel<n> | FUNCTION{1 | 2 | 3 | 4}}
Query :MMEMory:SOURce?
Returned Format [:MMEMory:SOURce] {CHANnel<n> | FUNCTION{1 | 2 | 3 | 4}<NL>
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
 (HP 54540C/54542C)

:MMEMory:STORE

Command :MMEMory:STORE {WMMEMory{1 | 2 | 3 | 4}}

:PCFRequency

Command :PCFRequency <cal_freq>
Query :PCFRequency
Returned Format [:PCFRequency] <cal_freq>
Where: <cal_freq> ::= exponential, probe compensation signal
 frequency in hertz

Programmer's Quick Reference Guide

:PLOT

Query :PLOT?

:PMEMory{1 | 2}:CLEar

Command :PMEMory{1 | 2}:CLEar

:PMEMory{1 | 2}:DISPlay

Command :PMEMory{1 | 2}:DISPlay {{OFF | 0} | {ON | 1}}
Query :PMEMory{1 | 2}:DISPlay?
Returned Format [:PMEMory{1 | 2}:DISPlay] {0 | 1}<NL>

:PMEMory{1 | 2}:MERGe

Command :PMEMory{1 | 2}:MERGe

:PMEMory{1 | 2}:SETup?

Query :PMEMory{1 | 2}:SETup?
Returned Format PMEMory{1 | 2}:DISP{0 | 1}<NL>

:PLOT?

Query :PLOT?

:POWERup{RUNning | STOPped}

Command :POWERup{RUNning | STOPped}
Query :POWERup?
Returned Format [:POWERup] {STOPped | RUNning}

Programmer's Quick Reference Guide

:PRINT?

Query

`:PRINT?`**:RUN**

Command

`:RUN`**:SEquential:DISPlay**

Command

`:SEquential:DISPlay {{OFF | 0} | {ON | 1}}`

Query

`:SEquential:DISPlay?`

Returned Format

`[ :SEquential:DISPlay ] {0 | 1} <NL>`**:SEquential:EXCLude**

Command

`:SEquential:EXCLude <from_argument>, <to_argument>`

Query

`:SEquential:EXCLude?`

Returned Format

`[ :SEquential:EXCLude ] <exclude_list> <NL>`

Where:

`<from_argument>` ::= integer, segment number of the lower end of the exclude range
`<to_argument>` ::= integer, segment number of the upper end of the exclude range
 number greater than or equal to the value of `<from_argument>`
`<exclude_list>` ::= integer, a list of previously captured segment numbers separated by commas

:SEquential:INCLude

Command

`:SEquential:INCLude <from_argument>, <to_argument>`

Query

`:SEquential:INCLude?`

Returned Format

`[ :SEquential:INCLude ] <include_list> <NL>`

Where:

`<from_argument>` ::= integer, segment number of the lower end of the include range
`<to_argument>` ::= integer, segment number of the upper end of the include range
`<include_list>` ::= integer, a list of previously captured segment numbers separated by commas

:SEquential:NPOints

Command :SEquential:NPOints <points_argument>
Query :SEquential:NPOints?
Returned Format: [:SEquential:NPOints] <points_argument><NL>
Where: <points_argument> ::= integer, 4 to 32768

:SEquential:NSEGments

Command :SEquential:NSEGments <segments_argument>
Query :SEquential:NSEGments?
Returned Format [:SEquential:NSEGments] <segments_argument><NL>
Where: <segments_argument> ::= integer, 1 to 8888 dependent on the
 SEquential:NPOints selection

:SEquential:SETup?

Query :SEquential:SETup?
Returned Format :SEQ:DISP {0 | 1};
 EXCL <exclude_list>;
 INCL <include_list>;
 NPO <points_argument>;
 NSEG <segment_argument>;
 SNUM <segments_argument>; (Sequential Single Shot Mode Only)
 SOURce CHANnel<n><NL> (Sequential Single Shot Mode Only)
Where: <exclude_list> ::= integer, a list of previously captured
 segment numbers separated by commas
 <include_list> ::= integer, a list of previously captured
 segment numbers separated by commas
 <points_argument> ::= integer, 4 to 32768
 <segments_argument> ::= integer, 1 to 8888 dependent on the
 points selected
 <n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
 (HP 54540C/54542C)

Programmer's Quick Reference Guide

:SEquential:SNUMber

Command	:SEquential:SNUMber <segments_argument>
Query	:SEquential:SNUMber?
Returned Format	[:SEquential:SNUMber] <segments_argument><NL>
Where:	<segments_argument> ::= integer, 1 to 8888 dependent on the number of segments acquired and ACquire:TYPE currently selected

:SEquential:SOURce

Command	:SEquential:SOURce {CHANnel<n>}
Query	:SEquential:SOURce?
Returned Format	[:SEquential:SOURce] {CHANnel<n>}<NL>
Where:	<n> ::= integer 1 or 2 (HP 54520C/54522C), or 1 through 4 (HP 54540C/54542C)

:SEquential:TTAGs?

Query	:SEquential:TTAGs? <segments_argument>
Where:	<segments_argument> ::= integer, 1 to 8888 dependent on the SEquential:NSEgments selection

:SEquential:TTDifference?

Query	:SEquential:TTDifference?
	<segments_argument>, <segments_argument>
Where:	<segments_argument> ::= integer, 1 to 8888 dependent on the SEquential:NSEgments selection

:SERial

Command	:SERial <serial_number>
Where:	<serial_number> ::= 10 character serial number within quotes

:STATus?

Query
Returned Format
Where:

```
:STATus? <display>
[:STATus] {0 | 1}<NL>
<display> ::= {CHANnel<n> | FUNCTION{1 | 2 | 3 | 4} |
              WMemory{1 | 2 | 3 | 4} | PMemory{1 | 2}<NL>
<n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
              (HP 54540C/54542C)
```

:STOP

Command

```
:STOP
```

:STORe

Command
Where:

```
:STORe <source>,<destination>
<source> ::= {CHANnel<n> | FUNCTION{1 | 2 | 3 | 4} |
              WMemory{1 | 2 | 3 | 4}}
<destination> ::= {WMemory{1 | 2 | 3 | 4}}
<n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
              (HP 54540C/54542C)
```

:SYSTem:DATE

Command
Query
Returned Format
Where:

```
:SYSTem:DATE <year>,<month>,<day>
:SYSTem:DATE?
[:SYSTem:DATE] "DDMMYYYY"<NL>
<year>/<YYYY> ::= integer, 1990 through 2059
<month> ::= integer, 1 through 12
<MM> ::= three digit alphabetic month
<day>/<DD> ::= integer, 1 through 31
```

:SYSTem:DSP

Command
Query
Returned Format
Where:

```
:SYSTem:DSP <ascii_string>
:SYSTem:DSP?
[:SYSTem:DSP] <ascii_string><NL>
<ascii_string> ::= string response data containing the last
                  information written on the advisory line
```

Programmer's Quick Reference Guide

:SYSTem:ERRor?

Query
Returned Format
Where:

```
:SYSTem:ERRor [{NUMBer | STRing}]
[:SYSTem:ERRor] <error> [, <ascii_string>] <NL>
<error> ::= an integer error code
<ascii_string> ::= an alpha string specifying the error
condition
```

:SYSTem:HEADer

Command
Query
Returned Format

```
:SYSTem:HEADer [{OFF | 0} | {ON | 1}]
:SYSTem:HEADer?
[:SYSTem:HEADer] {0 | 1} <NL>
```

:SYSTem:KEY

Command
Query
Returned Format
Where:

```
:SYSTem:KEY <key_code>
:SYSTem:KEY?
[:SYSTem:KEY] <key_code> <NL>
<key_code> ::= integer, 0 through 63
```

:SYSTem:LONGform

Command
Query
Returned Format

```
:SYSTem:LONGform [{OFF | 0} | {ON | 1}]
:SYSTem:LONGform?
[:SYSTem:LONGform] {0 | 1} <NL>
```

:SYSTem:PIMacro

Command
Where:

```
:SYSTem:PIMacro <ascii_string>
<ascii_string> ::= name of the defined macro
```

:SYSTem:SETup

Command :SYSTem:SETup <block_data>
Query :SYSTem:SETup?
Returned Format [:SYSTem:SETup] <block_data><NL>
Where: <block_data> ::= #800002048<setup_string>
 <setup_string> ::= block of binary data bytes

:SYSTem:TIME

Command :SYSTem:TIME <hour>,<minute>,<second>
Query :SYSTem:TIME?
Returned Format [:SYSTem:TIME] "HH:MM:SS"<NL>
Where: <hour>/<HH> ::= integer, 0 through 23
 <month>/<MM> ::= integer, 0 through 59
 <second>/<SS> ::= integer, 0 through 59

:SYSTem:UTILity

Command :SYSTem:UTILity:GMArkers {ON | OFF}
 :SYSTem:UTILity:LABels {ON | OFF}
 :SYSTem:UTILity:FACTOR {ON | OFF}
 :SYSTem:UTILity:FPANel {ON | OFF}
 :SYSTem:UTILity:FPANel:TIMEout {<timeout> | INFINITY}
Query :SYSTem:UTILity:GMArkers?
 :SYSTem:UTILity:LABels?
 :SYSTem:UTILity:FACTOR?
 :SYSTem:UTILity:FPANel?
 :SYSTem:UTILity:FPANel:TIMEout?
Returned Format [:SYSTem:UTILity:GMArkers] {ON | OFF}<NL>
Where: <timeout> ::= integer, 0 through 12 (hours)

:TER?

Query :TER?
Returned Format [:TER] {0 | 1}<NL>

Programmer's Quick Reference Guide

:TIMEbase:DELaY

Command	:TIMEbase:DELaY <delay_value>
Query	:TIMEbase:DELaY?
Returned Format	[:TIMEbase:DELaY] <delay_value><NL>
Where:	<delay_value> ::= exponential, time from trigger to display reference in seconds

:TIMEbase:MODE

Command	:TIMEbase:MODE {AUTO TRIGgered SINGLE}
Query	:TIMEbase:MODE?
Returned Format	[:TIMEbase:MODE] {AUTO TRIGgered SINGLE}<NL>

:TIMEbase:RANGe

Command	:TIMEbase:RANGe <range_value>
Query	:TIMEbase:RANGe?
Returned Format	[:TIMEbase:RANGe] <range_value><NL>
Where:	<range_value> ::= exponential, 10 ns to 50 s in a 1,2,5 sequence

:TIMEbase:REFeRence

Command	:TIMEbase:REFeRence {LEFT CENTer RIGHT}
Query	:TIMEbase:REFeRence?
Returned Format	[:TIMEbase:REFeRence] {LEFT CENTer RIGHT}<NL>

:TIMEbase:RLENgth

Command	:TIMEbase:RLENgth <length_argument>
Query	:TIMEbase:RLENgth?
Returned Format	[:TIMEbase:RLENgth] <length_argument><NL>
Where:	<length_argument> ::= integer, 500 in the repetitive mode, 512, 1024, 2048, 4196, 8192, 16384, or 32768 in the real-time mode (sequential off), or 4 to 32768 in the real-time mode (sequential on), depending on the current :SEQuential:NSEgments value

:TIMebase:SAMPle

Command :TIMebase:SAMPle {REALtime | REPetitive}
Query :TIMebase:SAMPle?
Returned Format [:TIMebase:SAMPle] {REALtime | REPetitive}<NL>

:TIMebase:SAMPle:CLOCK

Command :TIMebase:SAMPle:CLOCK {AUTO | <clock_value>}
Query :TIMebase:SAMPle:CLOCK?
Returned Format [:TIMebase:SAMPle:CLOCK] <clock_value><NL>
Where: <clock_value> ::= exponential, 10S/s to 2GS/s

:TIMebase:SETup?

Query :TIMebase:SETup?
Returned Format :TIM:DEL <delay_value>;
 MODE {AUTO|TRIG|SING};
 RANGE <range_value>;
 RLEN <length_argument>;
 REF {LEFT|CENT|RIGH};
 SAMPle {REAL|REP};
 SAMP:CLOC {AUTO | <clock_value>}<NL>

Where: <delay_value> ::= exponential, time from trigger to display
 reference in seconds
 <range_value> ::= exponential, 5ns to 50s
 <length_argument> ::= 500 in the repetitive mode, 512, 1024,
 2048, 4196, 8192, 16384, or 32768 in the real-time mode
 (sequential off), or 4 to 32768 in the real-time mode
 (sequential on), depending on the current
 :SEquential:NSEGments value
 <clock_value> ::= exponential, 10S/s to 2GS/s

:TRIGger:CENTERed

Command :TRIGger:CENTERed

Programmer's Quick Reference Guide

:TRIGger:CONDition

Command	<code>:TRIGger:CONDition {ENTER EXIT TRUE FALSE GT,<gt_argument> LT,<lt_argument> RANGE,<range_gt>,<range_lt>}</code>
Query	<code>:TRIGger:CONDition?</code>
Returned Format	<code>[:TRIGger:CONDition] <argument><NL></code>
Where:	<code><argument> ::= {ENTER EXIT GT,<gt_argument> LT,<lt_argument> RANGE,<range_gt>,<range_lt> } in PATTERN or DELay with QUALify:PATtern selected; {TRUE FALSE} (in STATE or DELay with QUALify:STATE selected; or RANGE, <range_gt>,<range_lt> in TV mode <gt_argument> ::= exponential, 20 ns to 160 ms <lt_argument> ::= exponential, 20 ns to 160 ms <range_gt> ::= exponential, 20 ns to 159.999 ms (must be less than <range_lt>) <range_lt> ::= exponential, 30 ns to 160 ms (must be greater than <range_gt>)</code>

:TRIGger:COUpling

Command	<code>:TRIGger:COUpling {AC DC LFReject}</code>
Query	<code>:TRIGger:COUpling?</code>
Returned Format	<code>[:TRIGger:COUpling] {AC DC LFReject}<NL></code>

:TRIGger:DELAy

Command	<code>:TRIGger:DELAy {TIME,<time_argument> EVENT,<event_argument>}</code>
Query	<code>:TRIGger:DELAy?</code>
Returned Format	<code>[:TRIGger:DELAy] {TIME,<time_argument> EVENT,<event_argument>}<NL></code>
Where:	<code><time_argument> ::= exponential, amount of delay from 30 ns to 160 ms <event_argument> ::= integer, number of events from 1 to 16000000</code>

Programmer's Quick Reference Guide

:TRIGger:DElay:SLOPe

Command :TRIGger:DElay:SLOPe {POSitive | NEGative}
Query :TRIGger:DElay:SLOPe?
Returned Format [:TRIGger:DElay:SLOPe] {POSitive | NEGative}<NL>

:TRIGger:DElay:SOURce

Command :TRIGger:DElay:SOURce {CHANnel<n> | EXTernal}
Query :TRIGger:DElay:SOURce?
Returned Format [:TRIGger:DElay:SOURce] {CHANnel<n> | EXTernal}
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4 (HP 54540C/54542C)

NOTE: EXTernal available only on HP 54520C/54522C

:TRIGger:FIELD

Command :TRIGger:FIELD {1 | 2}
Query :TRIGger:FIELD?
Returned Format [:TRIGger:FIELD] {1 | 2}<NL>

:TRIGger:GLITch:CENTERed

Command :TRIGger:GLITch:CENTERed

:TRIGger:GLITch:HOLDoff

Command :TRIGger:GLITch:HOLDoff <holdoff_time>
Query :TRIGger:GLITch:HOLDoff?
Returned Format [:TRIGger:GLITch:HOLDoff] <holdoff_time><NL>
Where: <holdoff_time> ::= exponential, 40 ns to 320 ms rounded to nearest 20 ns increment

Programmer's Quick Reference Guide

:TRIGger:GLITch:LEVel

Command :TRIGger:GLITch:LEVel <level_argument>
Query :TRIGger:GLITch:LEVel?
Returned Format [:TRIGger:GLITch:LEVel] <level_argument><NL>
Where: <level_argument> ::= exponential, for internal triggers, ± 1.5
x
full-scale voltage from center screen, for external
triggers
(HP 54520C/54522C) ± 2 volts with probe attenuation at 1:1

:TRIGger:GLITch:SOURce

Command :TRIGger:GLITch:SOURce {CHANnel<n> | EXTErnal},{HIGH | LOW}
Query :TRIGger:GLITch:SOURce?
Returned Format [:TRIGger:GLITch:SOURce] {CHANnel<n> | EXTErnal},{HIGH | LOW}
Where: <n> ::= integer 1 or 2 (HP 54520C/54522C), or 1 through 4
(HP 54540C/54542C)
NOTE: EXTErnal available only on HP 54520C/54522C

:TRIGger:GLITch:WIDth

Command :TRIGger:GLITch:WIDth {GT | LT},<width_argument>
Query :TRIGger:GLITch:WIDth?
Returned Format [:TRIGger:GLITch:WIDth] {GT | LT},<width_argument><NL>
Where: <width_argument> ::= exponential, 5 ns to 160 ms

:TRIGger:HOLDoff

Command :TRIGger:HOLDoff ({TIME,<holdoff_time>} |
{ EVENTt,<event_argument>})
Query :TRIGger:HOLDoff?
Returned Format [:TRIGger:HOLDoff] {TIME,<holdoff_value> |
EVENTt,<event_argument>}<NL>
Where: <holdoff_time> ::= exponential, 40 ns to 320 ms
<event_argument> ::= integer, 1 to 16000000

Programmer's Quick Reference Guide

:TRIGger:LEVel

Command :TRIGger:LEVel <level_argument>
Query :TRIGger:LEVel?
Returned Format [:TRIGger:LEVel] <level_argument><NL>
Where: <level_argument> ::= for internal triggers, $\pm 1.5 \times$ full-scale voltage from center screen; for external triggers, ± 2 volts with probe attenuation at 1:1

:TRIGger:LINE

Command :TRIGger:LINE <line_number>
Query :TRIGger:LINE?
Returned Format [:TRIGger:LINE] <line_number><NL>
Where: <line_number> ::= integer, 1 to 625 (depends on STANDARD and FIELD selection)

:TRIGger:LOGic

Command :TRIGger:LOGic {HIGH | LOW | DONTcare}
Query :TRIGger:LOGic?
Returned Format [:TRIGger:LOGic] {HIGH | LOW | DONTcare}<NL>

:TRIGger:MODE

Command :TRIGger:MODE {EDGE | PATtern | STATE | DELay | TV | GLITCh}
Query :TRIGger:MODE?
Returned Format [:TRIGger:MODE] {EDGE | PATtern | STATE | DELay | TV | GLITCh}<NL>

:TRIGger:NREJect

Command :TRIGger:NREJect {{OFF | 0} | {ON | 1}}
Query :TRIGger:NREJect?
Returned Format [:TRIGger:NREJect] {0 | 1}<NL>

Programmer's Quick Reference Guide

:TRIGger:OCCurrence

Command :TRIGger:OCCurrence <occurrence_argument>
Query :TRIGger:OCCurrence?
Returned Format [:TRIGger:OCCurrence] <occurrence_argument><NL>
Where: <occurrence_argument> ::= integer, 1 to 16000000

:TRIGger:OCCurrence:SLOPe

Command :TRIGger:OCCurrence:SLOPe {POSitive | NEGative}
Query :TRIGger:OCCurrence:SLOPe?
Returned Format [:TRIGger:OCCurrence:SLOPe] {POSitive | NEGative}<NL>

:TRIGger:OCCurrence:SOURce

Command :TRIGger:OCCurrence:SOURce {CHANnel<n> | EXTernal}
Query :TRIGger:OCCurrence:SOURce?
Returned Format [:TRIGger:OCCurrence:SOURce] {CHANnel<n> | EXTernal}<NL>
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
 (HP 54540C/54542C)

NOTE: EXTernal available only on HP 54520C/54522C

:TRIGger:PATH

Command :TRIGger:PATH {CHANnel<n> | EXTernal}
Query :TRIGger:PATH?
Returned Format [:TRIGger:PATH] {CHANnel<n> | EXTernal}<NL>
Where: <n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
 (HP 54540C/54542C)

NOTE: EXTernal available only on HP 54520C/54522C

:TRIGger:POLarity

Command :TRIGger:POLarity {POSitive | NEGative}
Query :TRIGger:POLarity?
Returned Format [:TRIGger:POLarity] {POSitive | NEGative}<NL>

Programmer's Quick Reference Guide

:TRIGger:QUALify

Command
Query
Returned Format

```
:TRIGger:QUALify {{EDGE | PATtern | STATE} | {LOW | HIGH}}
:TRIGger:QUALify?
[:TRIGger:QUALify] {{EDGE | PATtern | STATE} | {LOW |
HIGH}}<NL>
```

:TRIGger:SETup?

Query
Returned Format

```
:TRIGger:SETup?
:TRIG:MODE EDGE;
COUP {AC | DC | LFR};
HOLD {{TIME,<holdoff_time>} | EVENT, <event_argument>}};
LEV <level_argument>;
NREJ {0 | 1};
SLOP {POS | NEG};
SOUR {CHAN<n> | EXT | LINE | AUX}<NL> (EXT is only valid for the
54520/22)
```

```
:TRIG:MODE PATT;
COND {ENT | EXIT | GT,<gt_argument> | LT,<lt_argument> |
RANG,<range_gt>,<range_lt>}};
HOLD {{TIME,<holdoff_time>} | EVEN, <event_argument>}};
LEV <level_argument>;
LOG {HIGH | LOW | DONT};
NREJ {0 | 1};
PATH {CHAN<n> | EXT}<NL> (EXT is only valid for the 54520/22)
```

```
:TRIG:MODE STAT;
COND {TRUE | FALS};
HOLD {{TIME,<holdoff_time>} | {EVEN, <event_argument>}};
LEV <level_argument>;
LOG {HIGH | LOW | DONT};
NREJ {0 | 1};
PATH {CHAN<n> | EXT}; (EXT is only valid for the 54520/22)
SLOP {POS | NEG};
SOUR {CHAN<n> | EXT}<NL> (EXT is only valid for the 54520/22)
```

Programmer's Quick Reference Guide

```

:TRIG:MODE DEL;
COND {ENT | EXIT | TRUE | FALS | GT,<gt_argument> |
    LT,<lt_argument> | RANG,<range_gt>,<range_lt>};
DEL {{TIME,<time_value>} | {EVEN, <event_value>}};
DEL:SLOP {POS | NEG};
DEL:SOUR {CHAN<n> | EXT};    (EXT is only valid for the 54520/22)
LEV <level_argument>;
LOG {HIGH | LOW | DONT};
NREJ {0 | 1};
OCC <occurrence_argument>;
OCC:SLOP {POS | NEG};
OCC:SOUR {CHAN<n> | EXT};    (EXT is only valid for the 54520/22)
PATH {CHAN<n> | EXT};    (EXT is only valid for the 54520/22)
QUAL {EDGE | PATT | STAT};
SLOP {POS | NEG};
SOUR {CHAN<n> | EXT}<NL>    (EXT is only valid for the
                           54520/22)

:TRIG:MODE TV;
COND {RANG,<range_gt>,<range_lt>};
FIEL {1 | 2};
HOLD {{TIME,<holdoff_time>} | {EVEN, <event_argument>}};
LEV <level_argument>;
LINE <line_number>;
NREJ {0 | 1};
OCC <occurrence_argument>;
OCC:SLOP {POS | NEG};
POL {POS | NEG};
QUAL {LOW | HIGH};
SOUR {CHAN<n> | EXT};    (EXT is only valid for the 54520/22)
STAN {525 | 625 | USER}<NL>

:TRIG:MODE GLIT;
GLITCh:HOLD TIME,<holdoff_time>;
GLITCh:LEV <level_argument>;
GLITCh:NREJ {0 | 1};
GLITCh:SOUR {CHAN<n> | EXT}, {LOW | HIGH};    (EXT is only valid
                                                for the 54520/22)
GLITCh:WIDTH {GT | LT}, <width_argument><NL>

```

Programmer's Quick Reference Guide

Where:

```
<n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
      (HP 54540C/54542C)
<gt_argument> ::= exponential, 20 ns to 160 ms
<lt_argument> ::= exponential, 20 ns to 160 ms
<range_gt> ::= exponential, 20 ns to 159.999 ms (must be less
      than <range_lt>)
<range_lt> ::= exponential, 30 ns to 160 ms (must be greater
      than <range_gt>)
<time_value> ::= exponential, amount of delay from 30ns to
      160ms
<event_value> ::= integer, number of events from 1 to 16000000
<holdoff_time> ::= exponential, 40 ns to 320 ms rounded to
      nearest 20 ns increment
<level_argument> ::= exponential, trigger level in volts
<width_argument> ::= exponential, 5 ns to 160 ms
<event_argument> ::= integer, 1 to 16000000
<line_number> ::= integer, 1 to 625 (depends on STANDARD and
      FIELD selection)
```

Programmer's Quick Reference Guide

:VIEW

Command

Where:

```
:VIEW <display>
<display> ::= CHANNEL<n>|FUNCTION{1 | 2 | 3 | 4} |
           PMEMory{1 | 2} | WMEMory{1 | 2 | 3 | 4}}
<n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4
      (HP 54540C/54542C)
<n> ::= integer 1 or 2 (HP 54520C/54522C)
```

:WAVeform:DATA

Command

Query

Returned Format

Where:

```
:WAVeform:DATA <block_data>
:WAVeform:DATA?
[:WAVeform:DATA] <block_data><NL>
<block_data> ::= definite block data in IEEE 488.2 # format
```

:WAVeform:FORMat

Command

Query

Returned Format

```
:WAVeform:FORMat {AScii | WORD | BYTE | COMPRESSED}
:WAVeform:FORMat?
[:WAVeform:FORMat] {AScii | WORD | BYTE | COMPRESSED}<NL>
```

:WAVeform:POINTs

Query

Returned Format

```
:WAVeform:POINTs?
[:WAVeform:POINTs] {512 | 1024 | 2048 | 4096 | 8192 | 16384 |
32768}<NL>
```

:WAVeform:PREamble

Command
Query
Returned Format
Where:

```
:WAVeform:PREamble <preamble_data>
:WAVeform:PREamble?
[:WAVeform:PREamble] <preamble block><NL>
<preamble_data> ::= <format NR1>, <type NR1>, <points NR1>,
    <count NR1>, <xincrement NR3>, <xorigin NR3>, <xreference NR1>,
    <yincrement NR3>, <yorigin NR3>, <yreference NR1>
<format> ::= 0 for ASCII format
    1 for BYTE format
    2 for WORD format
    4 for COMPRESSED format
<type> ::= 0 for INVALID type
    1 for NORMAL type or REALTIME
    2 for AVERAGE type
    3 for ENVELOPE type
    4 for RAWDATA type
    5 for PDETECT type
```

:WAVeform:SOURce

Command
Query
Returned Format
Where:

```
:WAVeform:SOURce {CHANnel<n> | WMEMory{1 | 2 | 3 | 4}}
:WAVeform:SOURce?
[:WAVeform:SOURce] {CHANnel<n> | WMEMory{1 | 2 | 3 | 4}}<NL>
<n> ::= integer 1 or 2 (HP 54520C/54522C), or 1 through 4
    (HP 54540C/54542C)
```

:WAVeform:TYPE?

Query
Returned Format

```
:WAVeform:TYPE?
[:WAVeform:TYPE] {INValid | AVERage | ENVelope | NORMal |
    PDETECT | RAWData}<NL>
```

:WAVeform:XINCrement?

Query
Returned Format
Where:

```
:WAVeform:XINCrement?
[:WAVeform:XINCrement] <value><NL>
<value> ::= exponential, x-increment in the current preamble
```

Programmer's Quick Reference Guide

:WAVeform:XORigin?

Query
Returned Format
Where:

```
:WAVeform:XORigin?
[:WAVeform:XORigin] <value>[,<value>]...<NL>
<value> ::= exponential, x-origin value in the current
preamble
```

:WAVeform:XREference?

Query
Returned Format
Where:

```
:WAVeform:XREference?
[:WAVeform:XREference] <value><NL>
<value> ::= integer, x-reference value in the current preamble
```

:WAVeform:YINcrement?

Query Syntax:
Returned Format
Where:

```
:WAVeform:YINcrement?
[:WAVeform:YINcrement] <value><NL>
<value>::= exponential, y-increment value in the current
preamble
```

:WAVeform:YORigin?

Query
Returned Format
Where:

```
:WAVeform:YORigin?
[:WAVeform:YORigin] < value><NL>
<value> ::= exponential, y-origin in the current preamble
```

:WAVeform:YREference?

Query
Returned Format
Where:

```
:WAVeform:YREference?
[:WAVeform:YREference] <value><NL>
<value> ::= integer, y-reference value in the current preamble
```

:WMEMory{1 | 2 | 3 | 4}:DISPlay

Command
Query
Returned Format

```
:WMEMory{1 | 2 | 3 | 4}:DISPlay {{OFF | 0} | {ON | 1}}
:WMEMory{1 | 2 | 3 | 4}:DISPlay?
[:WMEMory{1 | 2 | 3 | 4}:DISPlay] {0 | 1}<NL>
```

:WMEemory{1 | 2 | 3 | 4}:GET**Command**

```
:WMEemory{1 | 2 | 3 | 4}:GET {CHANnel<n> |  
WMEemory{1 | 2 | 3 | 4} | FUNCTION{1 | 2 | 3 | 4}}
```

Where:

```
<n> ::= integer 1 or 2 (HP 54520C/54522C), or 1 through 4  
(HP 54540C/54542C)
```

:WMEemory{1 | 2 | 3 | 4}:PROTect**Command**

```
:WMEemory{1 | 2 | 3 | 4}:PROTect {{OFF | 0} | {ON | 1}}
```

Query

```
:WMEemory{1 | 2 | 3 | 4}:PROTect?
```

Returned Format

```
[ :WMEemory{1 | 2 | 3 | 4}:PROTect] {0 | 1}<NL>
```

:WMEemory{1 | 2 | 3 | 4}:SETup?**Query**

```
:WMEemory{1 | 2 | 3 | 4}:SETup?
```

Returned Format

```
:WME{1 | 2 | 3 | 4}:DISP {0 | 1};  
PROT {0 | 1};  
XOFF <offset_argument>;  
XRANG <range_argument>;  
YOFF <yoffset>;  
YRANG <yrange><NL>
```

Where:

```
<range_argument> ::= exponential, 5 ns to 50 sec  
<offset_argument> ::= exponential, time from trigger to  
display reference in seconds  
<yoffset> ::= exponential, offset value in volts  
<yrange> ::= exponential, full-scale range value
```

:WMEemory{1 | 2 | 3 | 4}:XOFFset**Command**

```
:WMEemory{1 | 2 | 3 | 4}:XOFFset <xoffset_argument>
```

Query

```
:WMEemory{1 | 2 | 3 | 4}:XOFFset?
```

Returned Format

```
[ :WMEemory{1 | 2 | 3 | 4}:XOFFset] <xoffset_argument><NL>
```

Where:

```
<xoffset_argument> ::= exponential, time from trigger to the  
on screen delay reference point. The maximum value  
depends on the :WMEemory:XRANGe setting
```

Programmer's Quick Reference Guide

:WMEemory{1 | 2 | 3 | 4}:XRANge

Command	<code>:WMEemory{1 2 3 4}:XRANge <range_argument></code>
Query	<code>:WMEemory{1 2 3 4}:XRANge?</code>
Returned Format	<code>[:WMEemory{1 2 3 4}:XRANge] <range_argument><NL></code>
Where:	<code><range_argument> ::= exponential, 5 ns to 50 s in a 1,2,5 sequence</code>

:WMEemory{1 | 2 | 3 | 4}:YOFFset?

Query	<code>:WMEemory{1 2 3 4}:YOFFset?</code>
Returned Format	<code>[:WMEemory{1 2 3 4}:YOFFset] <yoffset><NL></code>
Where:	<code><yoffset> ::= exponential, offset value in volts</code>

:WMEemory{1 | 2 | 3 | 4}:YRANge?

Query	<code>:WMEemory{1 2 3 4}:YRANge?</code>
Returned Format	<code>[:WMEemory{1 2 3 4}:YRANge] <yrange><NL></code>
Where:	<code><yrange> ::= exponential, full-scale range value</code>

© Copyright Hewlett-Packard Company 1993-1995
All Rights Reserved.

Reproduction, adaptation, or translation without prior written permission is prohibited, except as allowed under the copyright laws.

Document Warranty

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

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

Hewlett-Packard shall not be liable for errors contained herein or for damages in connection with the furnishing, performance, or use of this material.

Safety

This apparatus has been designed and tested in accordance with IEC Publication 348, Safety Requirements for Measuring Apparatus, and has been supplied in a safe condition. This is a Safety Class I instrument (provided with terminal for protective earthing). Before applying power, verify that the correct safety precautions are taken (see the following warnings). In addition, note the external markings on the instrument that are described under "Safety Symbols."

Warning

- Before turning on the instrument, you must connect the protective earth terminal of the instrument to the protective conductor of the (mains) power cord. The mains plug shall only be inserted in a socket outlet provided with a protective earth contact. You must not negate the protective action by using an extension cord (power cable) without a protective conductor (grounding). Grounding one conductor of a two-conductor outlet is not sufficient protection.
- Only fuses with the required rated current, voltage, and specified type (normal blow, time delay, etc.) should be used. Do not use repaired fuses or short-circuited fuseholders. To do so could cause a shock or fire hazard.
- Service instructions are for trained service personnel. To avoid dangerous electric shock, do not perform any service unless qualified to do so. Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.
- If you energize this instrument by an auto transformer (for voltage reduction), make sure the common terminal is connected to the earth terminal of the power source.
- Whenever it is likely that the ground protection is impaired, you must make the instrument inoperative and secure it against any unintended operation.
- Do not operate the instrument in the presence of flammable gasses or fumes. Operation of any electrical instrument in such an environment constitutes a definite safety hazard.
- Do not install substitute parts or perform any unauthorized modification to the instrument.
- Capacitors inside the instrument may retain a charge even if the instrument is disconnected from its source of supply.
- Use caution when exposing or handling the flat panel display. Handling or replacing the display shall be done only by qualified maintenance personnel.

Safety Symbols



Instruction manual symbol: the product is marked with this symbol when it is necessary for you to refer to the instruction manual in order to protect against damage to the product.



Hazardous voltage symbol.



Earth terminal symbol: Used to indicate a circuit common connected to grounded chassis.

WARNING

The Warning sign denotes a hazard. It calls attention to a procedure, practice, or the like, which, if not correctly performed or adhered to, could result in personal injury. Do not proceed beyond a Warning sign until the indicated conditions are fully understood and met.

CAUTION

The Caution sign denotes a hazard. It calls attention to an operating procedure, practice, or the like, which, if not correctly performed or adhered to, could result in damage to or destruction of part or all of the product. Do not proceed beyond a Caution symbol until the indicated conditions are fully understood or met.

Hewlett-Packard
P.O. Box 2197
1900 Garden of the Gods Road
Colorado Springs, CO 80901

Product Warranty

This Hewlett-Packard product has a warranty against defects in material and workmanship for a period of three years from date of shipment. During the warranty period, Hewlett-Packard Company will, at its option, either repair or replace products that prove to be defective. For warranty service or repair, this product must be returned to a service facility designated by Hewlett-Packard.

For products returned to Hewlett-Packard for warranty service, the Buyer shall prepay shipping charges to Hewlett-Packard and Hewlett-Packard shall pay shipping charges to return the product to the Buyer. However, the Buyer shall pay all shipping charges, duties, and taxes for products returned to Hewlett-Packard from another country.

Hewlett-Packard warrants that its software and firmware designated by Hewlett-Packard for use with an instrument will execute its programming instructions when properly installed on that instrument.

Hewlett-Packard does not warrant that the operation of the instrument software, or firmware will be uninterrupted or error free.

Limitation of Warranty

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

No other warranty is expressed or implied. Hewlett-Packard specifically disclaims the implied warranties of merchantability or fitness for a particular purpose.

Exclusive Remedies

The remedies provided herein are the buyer's sole and exclusive remedies. Hewlett-Packard shall not be liable for any direct, indirect, special, incidental, or consequential damages, whether based on contract, tort, or any other legal theory.

Assistance

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

For any assistance, contact your nearest Hewlett-Packard Sales Office.

Certification

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

About this edition

This is the first edition of the *HP 54520C and 54540C Series Oscilloscope Programmer's Quick Reference Guide*.

Publication number
54542-97042

Printed in USA.

Edition dates are as follows:

First edition, November 1995

New editions are complete revisions of the manual. Update packages, which are issued between editions, contain additional and replacement pages to be merged into the manual by you. The dates on the title page change only when a new edition is published.

A software or firmware code may be printed before the date. This code indicates the version level of the software or firmware of this product at the time the manual or update was issued. Many product updates do not require manual changes; and, conversely, manual corrections may be done without accompanying product changes. Therefore, do not expect a one-to-one correspondence between product updates and manual updates.

The following list of pages gives the date of the current edition and of any changed pages to that edition.

All pages original edition