# 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.

# 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> ::= <B> indicates that <A> can be replaced by <B> in any statement containing <A>.

| "or." Indicates a choice of one element from a list. For example, <A> | <B> indicates <A> or <B> 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*.

## *CLS

| | |
|---|---|
| Command | `*CLS` |

## *DMC

| | |
|---|---|
| Command | `*DMC <ascii_string>,<block_data>` |
| Where: | `<ascii_string> ::= a quoted ascii string`<br>`<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`<br>`<block_data> ::= definite block data in # format` |

## *IDN?

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

## *LMC

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

## *LRN?

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

## *OPC

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

## *OPT ?

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

## *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`<br>`<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` |

## *TST?

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

## *WAI

| | |
|---|---|
| Command | `*WAI` |

## :ACQuire:COMPlete

| | |
|---|---|
| Command | `:ACQuire:COMPlete <complete_argument>` |
| Query | `:ACQuire:COMPlete?` |
| Returned Format | `[:ACQuire:COMPlete] <complete_argument><NL>` |
| Where: | `<complete_argument> ::= integer, 0 to 100 percent` |

## :ACQuire:COUNt

| | |
|---|---|
| | `:ACQuire:COUNt <count_argument>` |
| Query | `:ACQuire:COUNt?` |
| Returned Format | `[:ACQuire:COUNt] <count_argument><NL>` |
| Where: | `<count_argument> ::= integer, 1 to 2048 (depending on the acquisition type)` |

## :ACQuire:POINts

| | |
|---|---|
| Command | `:ACQuire:POINts <points_argument>` |
| Query | `:ACQuire:POINts?` |
| Returned Format | `[:ACQuire:POINts] <points_argument><NL>` |
| Where: | `<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)` |

## :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>
```

## :BEEPer

```
Command          :BEEPer [{{OFF | 0} | {ON | 1}}]
Query            :BEEPer?
Returned Format  [:BEEPer] {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>
```

## :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]
                  <null_value_n>,<null_value_n>,<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) 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)`<br>`<offset_argument> ::= exponential, offset value in volts` |

## :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> ::= DDMMMYY
<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}
```

## :DISK:LOAD

Command<br>
Where:

```
:DISK:LOAD <file_name>,<destination> [,<format>]
<filename> ::= quoted ascii string  DOS compatible filename.
  1 to 8 character ASCCI 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> ::= {WMEMory {1 | 2 | 3 | 4 } |
  SETup{0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9} |
  PMEMory{ 1 | 2}}
<format>::= {TEXT | INTernal}
```

## :DISK:MDIRectory

Command<br>
Where:

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

## :DISK:PWD?

Query<br>
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          :DISK:STORe <source>,<file_name>[,<format>]
Where:           <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          :DISPlay:COLumn <column_number>
Query            :DISPlay:COLumn?
Returned Format  [:DISPlay:COLumn] <column_number><NL>
Where:           <column_number> ::= integer, 0 through 78
```

## :DISPlay:CONNect

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

## :DISPlay:DATA

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

## :DISPlay:FORMat

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

## :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:PERSistence

| | |
|---|---|
| Command | :DISPlay:PERSistence {SINGle \| INFinite \| 0 \| {0.5 to 10} \| 11}} |
| Query | :DISPlay:PERSistence? |
| Returned Format | [:DISPlay:PERSistence] <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>
```

## :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>
```

## :EXTernal:LFReject

| | |
|---|---|
| Command | `:EXTernal:LFReject {{OFF \| 0} \| {ON \| 1}}` |
| Query | `:EXTernal:LFReject?` |
| Returned Format | `[:EXTernalLFReject] {0 \| 1}<NL>` |

## :EXTernal:PROBe

| | |
|---|---|
| Command | `:EXternal:PROBe <probe_argument>` |
| Query | `:EXTernal:PROBe?` |
| Returned Format | `[:EXTernal:PROBe] <probe_argument><NL>`<br>`<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>`<br>`<range_argument> ::= exponential, full-scale range value` |

## :EXTernal:SETup

| | |
|---|---|
| Query | `:EXTernal:SETup?` |
| Returned Format | `:EXT:COUP {AC\|DC\|DCF};`<br>`HFR {0 \| 1};`<br>`LFR {0 \| 1};`<br>`PROB <probe_argument>;`<br>`RANG<range_argument<NL>` |
| Where: | `<range_argument> ::= exponential, full-scale range value`<br>`<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}}`<br>`<n> ::= integer, 1 or 2 (HP 54520C/54522C) or 1 through 4 (HP 54540C/54542C)` |

## :FUNCtion{1 | 2 | 3 | 4}:DIFF

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

## :FUNCtion{1 | 2 | 3 | 4}:INVert

```
Command          :FUNCtion{1 | 2 | 3 | 4}:INVert <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}:LEVel

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

## :FUNCtion{1 | 2 | 3  | 4}:MAGNify

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

## :FUNCtion{1|2|3|4}:MODE?

```
Query            :FUNCtion{1 | 2 | 3 | 4}:MODE?
Returned Format  [:FUNCtion{1 | 2 | 3 | 4}:MODE]
                 <operation>,<operand>[,<operand>]
Where:           <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          :FUNCtion{1 | 2 | 3 | 4}:MULTiply <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)
```

## :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 ±voltage full scale, or 0 to ±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}}`<br>`<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`<br>`<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

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

Returned Format

```
: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 freqency 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

```
:FUNCtion{1 | 2 | 3 | 4}:SPAN <span_argument>
```

Query

```
:FUNCtion{1 | 2 | 3 | 4}:SPAN?
```

Returned Format

```
[:FUNCtion{1 | 2 | 3 | 4}:SPAN] <span_argument><NL>
```

Where:

```
<span_argument>::= exponential number in hertz
```

## :FUNCtion{1 | 2 | 3 | 4}:SUBTract

Command

```
:FUNCtion{1 | 2|3|4}:SUBTract <operand1>,<operand2>
```

Where:

```
<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

```
:FUNCtion{1 | 2 | 3 | 4}:VERSus <Y_operand>,<X_operand>
```

Where:

```
<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>
```

## :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>
```

## :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

```
:MARKer:SETup?
```

Returned Format

```
: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

```
:MARKer:X1Position <xposition_argument>
```

Query

```
:MARKer:X1Position?
```

Returned Format

```
[:MARKer:X1Position] <xposition_argument><NL>
```

Where:

```
<xposition_argument>::= exponential, xmarker time in seconds,
  or frequency in hertz
```

## :MARKer:X2Position

Command

```
:MARKer:X2Position <xposition_argument>
```

Query

```
:MARKer:X2Position?
```

Returned Format

```
[:MARKer:X2Position] <xposition_argument><NL>
```

Where:

```
<xposition_argument> ::= exponential, xmarker time in
seconds,
  or frequency in hertz
```

## :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
```

## :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] <ressult;[: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]<br><measurement>,<upper_limit>,<lower_limit><NL> |
| Where: | <measurement> ::= {DELay \| DUTycycle \| FALLtime \| FREQuency \| NWIDth \| OVERshoot \| PERiod \| PREShoot \| PWIDth \| RISetime \| VACRms \| VAMPlitude \| VAVerage \| VBASe \| VDCRms \| VMAX \| VMIN \| VPP \| VTOP}<br><upper_limit> ::= exponential, high limit value<br><lower_limit> ::=exponential, low limit value |

## :MEASure:CURSor?

Query

```
:MEASure:CURSor? {DELTa | STARt | STOP}
```

Returned Format

```
[:MEASure:CURSor] <time>,<voltage><NL>
```

Where:

```
<time> ::= exponential, delta time, y1marker time, or y2marker
  time
<voltage> ::= exponential, delta voltage, x1marker voltage, or
  x2marker voltage
```

## :MEASure:DEFine

Command

```
:MEASure:DEFine <define_argument>
```

Query

```
:MEASure:DEFine? {DELay | PWIDth | NWIDth}
```

Returned Format

```
[:MEASure:DEFine] <define_argument><NL>
```

Where:

```
<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

```
:MEASure:DELay
```

Query

```
:MEASure:DELay?
```

Returned Format

```
[:MEASure:DELay] <value><NL>
```

Where:

```
<value> ::= exponential, time value in seconds
```

## :MEASure:DESTination

Command :MEASure:DESTination <source_argument>,<destination_argument>

Query :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:DUTycycle

Command :MEASure:DUTycycle

Query :MEASure:DUTycycle?

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

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

## :MEASure:ESTArt

Command :MEASure:ESTArt <slope_and_occurrence>

Query :MEASure:ESTArt?

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

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

## :MEASure:ESTOp

Command :MEASure:ESTOp <slope_and_occurrence>

Query :MEASure:ESTOp?

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

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

## :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)
```

## :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>` |

## :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: | see below |

```
<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
```

## :MEASure:TMIN?

| | |
|---|---|
| Query | `:MEASure:TMIN?` |
| Returned Format | `[:MEASure:TMIN] <time><NL>` |
| Where: | `<time> ::= exponential, time at minimum voltage` |

## :MEASure:TSTArt

| | |
|---|---|
| Command | `:MEASure:TSTArt <tstart _argument>` |
| Query | `:MEASure:TSTArt?` |
| Returned Format | `[:MEASure:TSTArt] <tstart _argument><NL>` |
| Where: | `<tstart_argument> ::= exponential, time at x1marker in seconds` |

## :MEASure:TSTOp

| | |
|---|---|
| Command | `:MEASure:TSTOp <tstop_argument>` |
| Query | `:MEASure:TSTOp?` |
| Returned Format | `[:MEASure:TSTOp] <tstop _argument><NL>` |
| Where: | `<tstop_argument> ::= exponential, time at x2marker in seconds` |

## :MEASure:TVOLt?

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

## :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
```

## :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` |

## :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`<br>`<voltage> ::= exponential, voltage at specified time` |

## :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)
```

## :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) |

## :MERGe

```
Command            :MERGe <pmemory_num>
Where:             <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 {WMEMory{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
```

## :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}
```

## :PRINt?

Query

```
:PRINt?
```

## :RUN

Command

```
:RUN
```

## :SEQuential:DISPlay

Command
Query
Returned Format

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

## :SEQuential:EXCLude

Command
Query
Returned Format
Where:

```
:SEQuential:EXCLude <from_argument>,<to_argument>
:SEQuential:EXCLude?
[:SEQuential:EXCLude] <exclude_list><NL>
<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
Query
Returned Format
Where:

```
:SEQuential:INCLude <from_argument>,<to_argument>
:SEQuential:INCLude?
[:SEQuential:INCLude] <include_list><NL>
<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};`<br>`EXCL <exclude_list>;`<br>`INCL <include_list>;`<br>`NPO <points_argument>;`<br>`NSEG <segment_argument>;`<br>`SNUM <segments_argument>;` (Sequential Single Shot Mode Only)<br>`SOURce CHANnel<n><NL>` (Sequential Single Shot Mode Only) |
| Where: | `<exclude_list> ::= integer, a list of previously captured segment numbers separated by commas`<br>`<include_list> ::= integer, a list of previously captured segment numbers separated by commas`<br>`<points_argument>::= integer, 4 to 32768`<br>`<segments_argument>::= integer, 1 to 8888 dependent on the points selected`<br>`<n> ::= integer, 1 or 2 (HP 54520C/54522C), or 1 through 4 (HP 54540C/54542C)` |

## :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?`<br>`<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             :STATus? <display>
Returned Format   [:STATus] {0 | 1}<NL>
Where:            <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           :STORe <source>,<destination>
Where:            <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           :SYSTem:DATE <year>,<month>,<day>
Query             :SYSTem:DATE?
Returned Format   [:SYSTem:DATE] "DDMMMYYYY"<NL>
Where:            <year>/<YYYY> ::= integer, 1990 through 2059
                  <month> ::= integer, 1 through 12
                  <MMM> ::= three digit alphabetic month
                  <day>/<DD> ::= integer, 1 through 31
```

## :SYSTem:DSP

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

## :SYSTem:ERRor?

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

## :SYSTem:HEADer

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

## :SYSTem:KEY

```
Command           :SYSTem:KEY <key_code>
Query             :SYSTem:KEY?
Returned Format   [:SYSTem:KEY] <key_code><NL>
Where:            <key_code> ::= integer, 0 through 63
```

## :SYSTem:LONGform

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

## :SYSTem:PIMacro

```
Command           :SYSTem:PIMacro <ascii_string>
Where:            <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>
```

## :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
```

## :TRIGger:CONDition

Command

```
:TRIGger:CONDition {ENTer | EXIT | TRUE | FALSe |
  GT,<gt_argument>| LT,<lt_argument> |
  RANGe,<range_gt>,<range_lt>}
```

Query

```
:TRIGger:CONDition?
```

Returned Format

```
[:TRIGger:CONDition] <argument><NL>
```

Where:

```
<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>)
```

## :TRIGger:COUPling

Command

```
:TRIGger:COUPling {AC | DC | LFReject}
```

Query

```
:TRIGger:COUPling?
```

Returned Format

```
[:TRIGger:COUPling] {AC | DC | LFReject}<NL>
```

## :TRIGger:DELay

Command

```
:TRIGger:DELay {TIME,<time_argument> | EVENt,<event_argument>}
```

Query

```
:TRIGger:DELay?
```

Returned Format

```
[:TRIGger:DELay] {TIME,<time_argument> |
  EVENt,<event_argument>}<NL>
```

Where:

```
<time_argument> ::= exponential, amount of delay from 30 ns
to
  160 ms
<event_argument> ::= integer, number of events from 1 to
  16000000
```

## :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` |

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

## :TRIGger:LEVel

| | |
|---|---|
| Command | :TRIGger:LEVel <level_argument> |
| Query | :TRIGger:LEVel? |
| Returned Format | [:TRIGger:LEVel] <level_argument><NL> |
| Where: | <level_argument> ::= for internal triggers, ±1.5 x 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 \| GILTch}<NL> |

## :TRIGger:NREJect

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

## :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>
```

## :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,<holdff_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)
```

```
: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>
```

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)
```

## :VIEW

Command

```
:VIEW <display>
```

Where:

```
<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

```
:WAVeform:DATA <block_data>
```

Query

```
:WAVeform:DATA?
```

Returned Format

```
[:WAVeform:DATA] <block_data><NL>
```

Where:

```
<block_data> ::= definite block data in IEEE 488.2 # format
```

## :WAVeform:FORMat

Command

```
:WAVeform:FORMat {ASCii | WORD | BYTE | COMPressed}
```

Query

```
:WAVeform:FORMat?
```

Returned Format

```
[:WAVeform:FORMat] {ASCii | WORD | BYTE | COMPressed}<NL>
```

## :WAVeform:POINts

Query

```
:WAVeform:POINts?
```

Returned Format

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

## :WAVeform:PREamble

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

## :WAVeform:TYPE?

```
Query             :WAVeform:TYPE?
Returned Format   [:WAVeform:TYPE] {INValid | AVERage | ENVelope | NORMal |
                    PDETect | RAWData}<NL>
```

## :WAVeform:XINCrement?

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

## :WAVeform:XORigin?

| | |
|---|---|
| Query | :WAVeform:XORigin? |
| Returned Format | [:WAVeform:XORigin] <value>[,<value>]...<NL> |
| Where: | <value> ::= exponential, x-origin value in the current preamble |

## :WAVeform:XREFerence?

| | |
|---|---|
| Query | :WAVeform:XREFerence? |
| Returned Format | [:WAVeform:XREFerence] <value><NL> |
| Where: | <value> ::= integer, x-reference value in the current preamble |

## :WAVeform:YINCrement?

| | |
|---|---|
| Query Syntax: | :WAVeform:YINCrement? |
| Returned Format | [:WAVeform:YINCrement] <value><NL> |
| Where: | <value>::= exponential, y-increment value in the current preamble |

## :WAVeform:YORigin?

| | |
|---|---|
| Query | :WAVeform:YORigin? |
| Returned Format | [:WAVeform:YORigin] < value><NL> |
| Where: | <value> ::= exponential, y-origin in the current preamble |

## :WAVeform:YREFerence?

| | |
|---|---|
| Query | :WAVeform:YREFerence? |
| Returned Format | [:WAVeform:YREFerence] <value><NL> |
| Where: | <value> ::= integer, y-reference value in the current preamble |

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

| | |
|---|---|
| Command | :WMEMory{1 \| 2 \| 3 \| 4}:DISPlay {{OFF \| 0} \| {ON \| 1}} |
| Query | :WMEMory{1 \| 2 \| 3 \| 4}:DISPlay? |
| Returned Format | [:WMEMory{1 \| 2 \| 3 \| 4}:DISPlay] {0 \| 1}<NL> |

## :WMEMory{1 | 2 | 3 | 4}:GET

Command

```
:WMEMory{1 | 2 | 3 | 4}:GET {CHANnel<n> |
  WMEMory{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)
```

## :WMEMory{1 | 2 | 3 | 4}:PROTect

Command

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

Query

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

Returned Format

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

## :WMEMory{1 | 2 | 3 | 4}:SETup?

Query

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

Returned Format

```
:WMEM{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
```

## :WMEMory{1 | 2 | 3 | 4}:XOFFset

Command

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

Query

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

Returned Format

```
[:WMEMory{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 :WMEMory:XRANge setting
```

## :WMEMory{1 | 2 | 3 | 4}:XRANge

Command `:WMEMory{1 | 2 | 3 | 4}:XRANge <range_argument>`

Query `:WMEMory{1 | 2 | 3 | 4}:XRANge?`

Returned Format `[:WMEMory{1 | 2 | 3 | 4}:XRANge] <range_argument><NL>`

Where: `<range_argument> ::= exponential, 5 ns to 50 s in a 1,2,5 sequence`

## :WMEMory{1 | 2 | 3 | 4}:YOFFset?

Query `:WMEMory{1 | 2 | 3 | 4}:YOFFset?`

Returned Format `[:WMEMory{1 | 2 | 3 | 4}:YOFFset] <yoffset><NL>`

Where: `<yoffset> ::= exponential, offset value in volts`

## :WMEMory{1 | 2 | 3 | 4}:YRANge?

Query `:WMEMory{1 | 2 | 3 | 4}:YRANge?`

Returned Format `[:WMEMory{1 | 2 | 3 | 4}:YRANge] <yrange><NL>`

Where: `<yrange> ::= exponential, full-scale range value`

**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 of 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, Novemer 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