



Agilent Technologies E1538A Enhanced Frequency/Totalize/PWM Signal Conditioning Plug-on

User's and SCPI Programming Manual

Where to Find it - Online and Printed Information:

System installation (hardware/software) VXIbus Configuration Guide*
Agilent VIC (VXI installation software)*

Module configuration and wiring This Manual
SCPI programming This Manual

VXI*plug&play* programming VXI*plug&play* Online Help
VXI*plug&play* example programs VXI*plug&play* Online Help
VXI*plug&play* function reference VXI*plug&play* Online Help
Soft Front Panel information VXI*plug&play* Online Help



VISA language information Agilent VISA User's Guide

Agilent VEE programming information Agilent VEE User's Manual

**Supplied with Agilent Command Modules , Embedded Controllers, and VXLink.*



Agilent E1538A Enhanced Frequency/Totalize/PWM SCP

About this Manual

This manual describes how to configure the Signal Conditioning Plug-on (SCP) using SCPI commands and explains the capabilities of this SCP. The contents of this manual are:

- Introduction page 4
- Identifying the Plug-on (IMPORTANT) page 5
- Setting Configuration Switches page 6
- Installation page 7
- Connecting To The Terminal Module page 7
- Recommended Signal Connections page 8
- Input and Output Characteristics page 9
- Programming With SCPI Commands page 11
 - Configuring I/O Direction page 12
- Programming Input Channels page 12
 - Setting the Input Threshold Level page 12
 - Set Input Logic Sense page 13
 - Reading Static Digital State page 14
 - Totalize Positive or Negative Edge State Changes page 15
 - About Period and Frequency Measurements page 16
 - Measure Frequency page 17
 - Measure Period page 18
 - Measure Pulse Width page 20
 - Sense Quadrature Position page 21
 - Sense Rotational Velocity page 22
- Programming Output Channels page 24
 - Controlling Output Polarity page 24
 - Output Static Digital State page 24
 - Variable Width Pulse Per Trigger page 25
 - Variable Width Pulse Train (PWM) page 26
 - Variable Frequency Fixed Width Pulse Train (FM) page 27
 - Variable Frequency Square-Wave Pulse Train (FM) page 28
 - Rotationally Positioned Pulse Output page 29
 - Rotational Pulse Command Usage page 30
 - Stepper Motor Control page 37
- *RST and *TST (important!) page 41
- SCPI Command Reference page 42
- Specifications page 77

Introduction

The Agilent E1538A provides eight TTL compatible channels of digital I/O. Channels can be individually configured to perform any one of the following functions:

- Input:
 - Static digital state
 - Frequency measurement
 - Period measurement
 - Totalize positive or negative signal transitions
 - Pulse width measurement
 - Rotational velocity (senses added or missing cogwheel teeth)
 - Quadrature position. (requires 2 channels)
- Output (configurable as Open Drain or passive pull-up):
 - Static digital state
 - Single pulse-per-trigger: Generates a pulse at each algorithm execution. The pulse width is controlled by the algorithm.
 - Pulse Width Modulation: A free-running pulse train where a SCPI command pre-configures the frequency and the algorithm controls the pulse width.
 - Frequency Modulation: A free-running pulse train where a SCPI command pre-configures the pulse width and the algorithm controls the frequency. In this FM mode the duty cycle varies with frequency.
 - Frequency Modulation: A free-running pulse train where the duty cycle remains constant at 50% while the algorithm controls the frequency.
 - Rotationally positioned pulse: The algorithm controls the angular pulse position (relative to an input sensing rotational velocity). The pulse width is fixed by a SCPI command. (requires a reference channel in addition to any rotational pulse output channels)
 - Rotationally positioned pulse: The algorithm controls the width of the pulse. The angular pulse position (relative to an input sensing rotational velocity) is fixed by a SCPI command.(requires a reference channel)
 - Stepper Motor Control: Controls 2-phase and 4-phase motors in

both full and half step modes.(requires 2 or 4 channels)

The logical sense of input and output channels can be configured as inverted or normal.

Input-configured channels have individually programmable threshold levels that can range from -46V to +46V.

Identifying the Plug-on (IMPORTANT)

There are two versions of the E1538A. The early version does not support a PERiod measurement command set. Early versions have ROM revision February 1998 and earlier. The later version adds period measurement, and an improved frequency measurement function. The later versions have ROM revision after February 1998.

In order to access the additional functions of the later E1538A, you must use one of the following drivers:

- The Plug & Play driver with revision A.02.07 or later
- The Command Module driver with revision A.05.11 or later

To determine the driver revision, execute the *IDN? command.

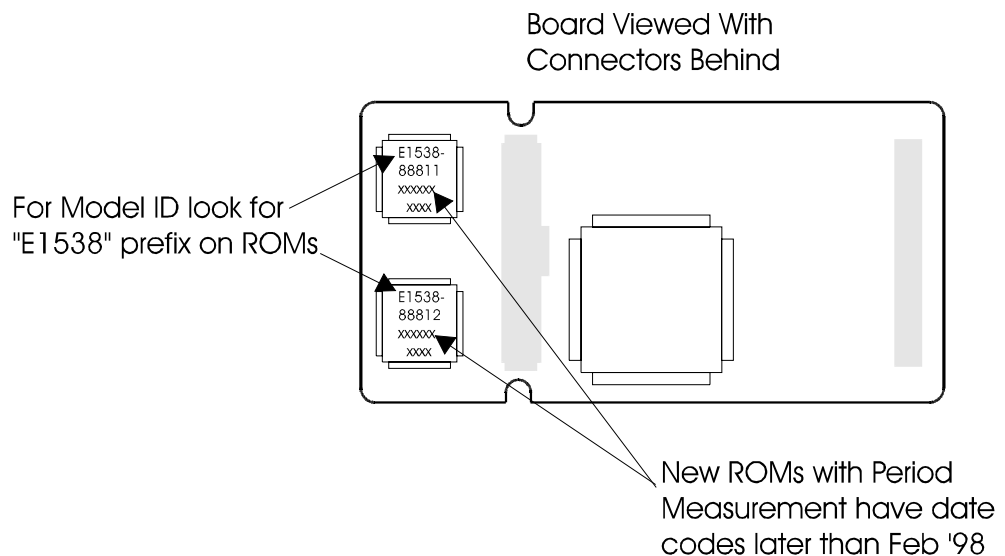


Figure 1. Identifying the SCP and its ROM Revision

Setting Configuration Switches

The SCP has three packages of eight switches each. The package labeled OE (Output Enable) determines a channel's I/O direction. The package labeled PU (pull-up) controls whether or not a channel is floating or pulled up to an internal 5V supply. The package labeled VRS (for channels 0 and 1 only) can enable special input signal conditioning compatible with variable reluctance sensors. For a discussion on using the VRS mode, see "VRS Mode Input Operation" on page 10.

Locating switches Figure 2 shows the location of each channel's configuration switches.

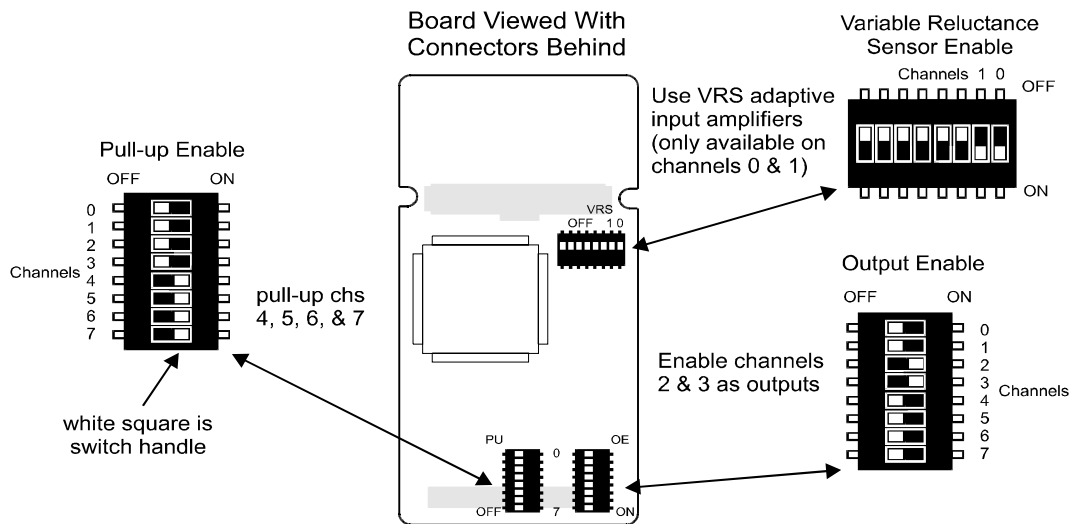


Figure 2. Switch Location and Example Settings

Configuring Input-Output direction

Refer to Figure 2 for the location of the eight Output Enable (OE) switches. Move the channel's switch handle to the ON position for output, and to the OFF position for input.

Configuring Channel Pull-up Resistor

Refer to Figure 2 for the location of the eight Pull-up Enable (PU) switches. Move the switch handle to the ON position to connect the pull-up resistor (connected from channel terminal to an internal +5V), and to the OFF position to disconnect the pull-up resistor (high impedance input/open drain output).

Note

Pull-Up enable ON is not allowed for channels that have their VRS enable ON (VRS is only available on channels 0 and 1).

Installation

Installation for this Plug-on is identical to other SCPs and is covered in Chapter 1 of your Agilent E1415 or E1419 User's Manual.

Connecting To The Terminal Module

The SCP connections for the Terminal Modules are shown on the self-adhesive labels that come with the SCP. Use these to label terminal definitions on your terminal module. The connections are shown in Figure 3.

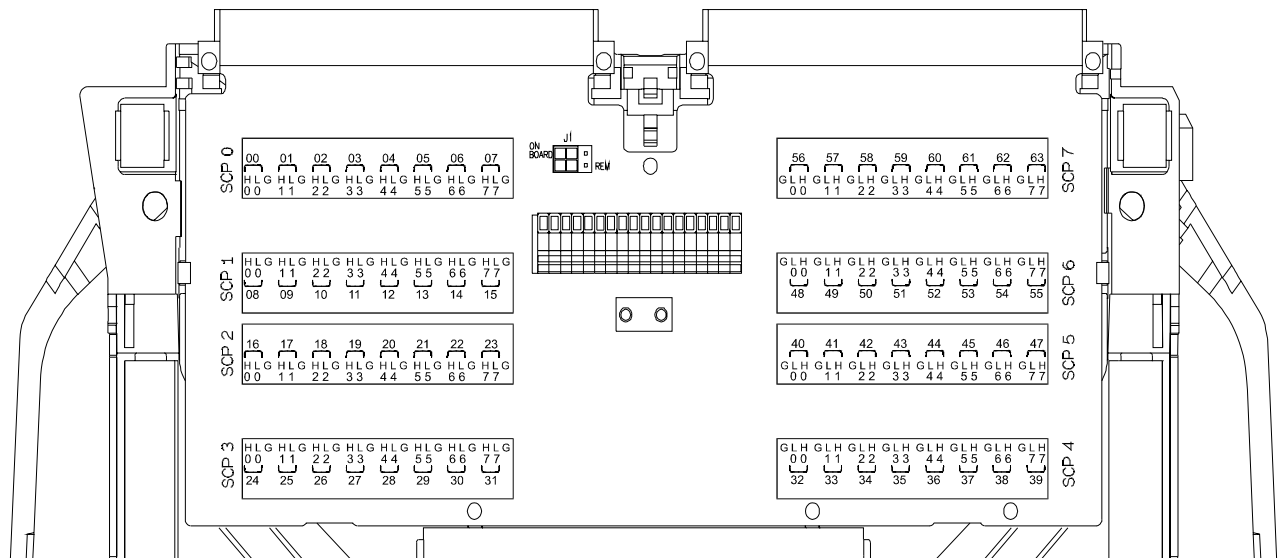


Figure 3. E1538A Terminal Module Connections

Figure 4 shows the screw terminal Option 11 for the E1419A.

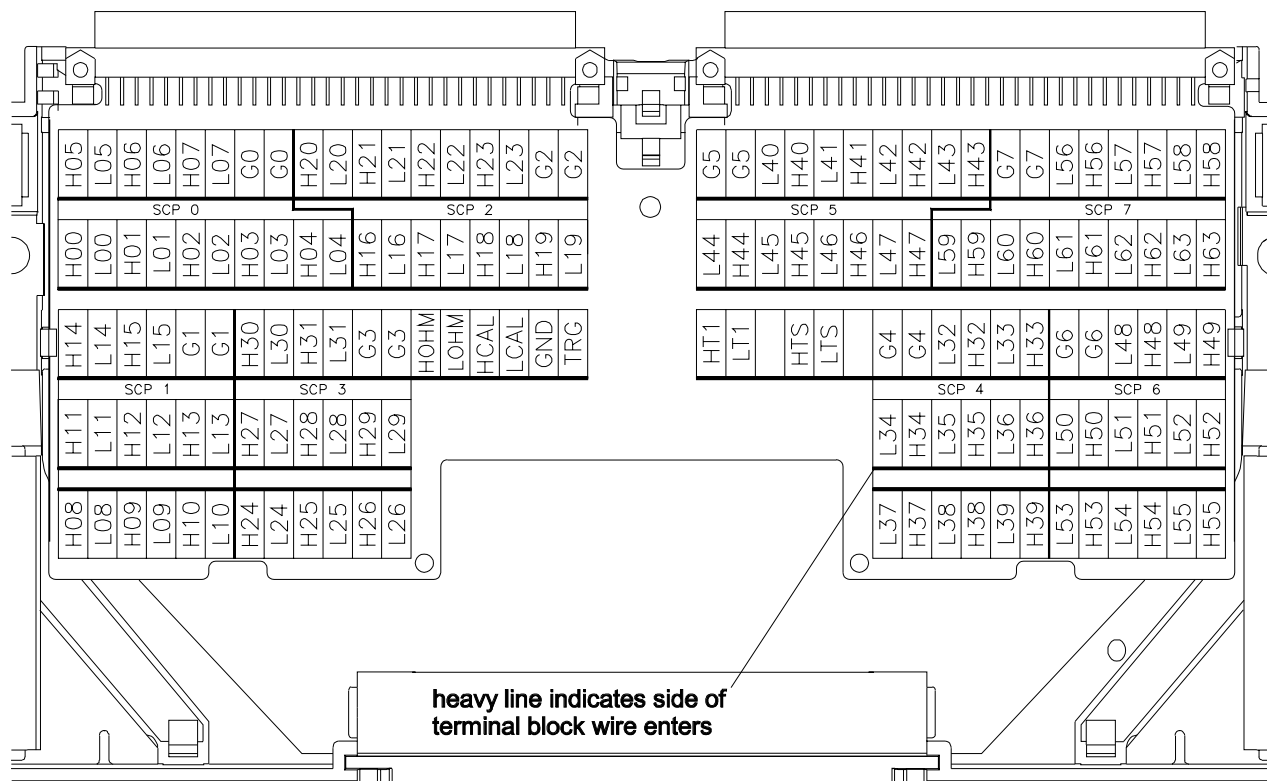


Figure 4. E1419A Option 11 Terminal Module Connections

Recommended Signal Connections

Figure 5 shows the recommended method of wiring digital I/O channels, as well as the maximum voltage limitations for the E1538A.

Figure 5 shows the shields connected directly to the E1415 ground. This is to limit potential noise on the digital wiring from affecting low-level analog channel wiring within the Terminal Module.

Note The G (analog guard) terminals are connected through 10K Ohm resistors to chassis ground. To connect the shields directly to chassis ground on the E1415 and the E1419 Option 12 Terminal Module, install the guard-to-ground jumpers for the E1538 channels

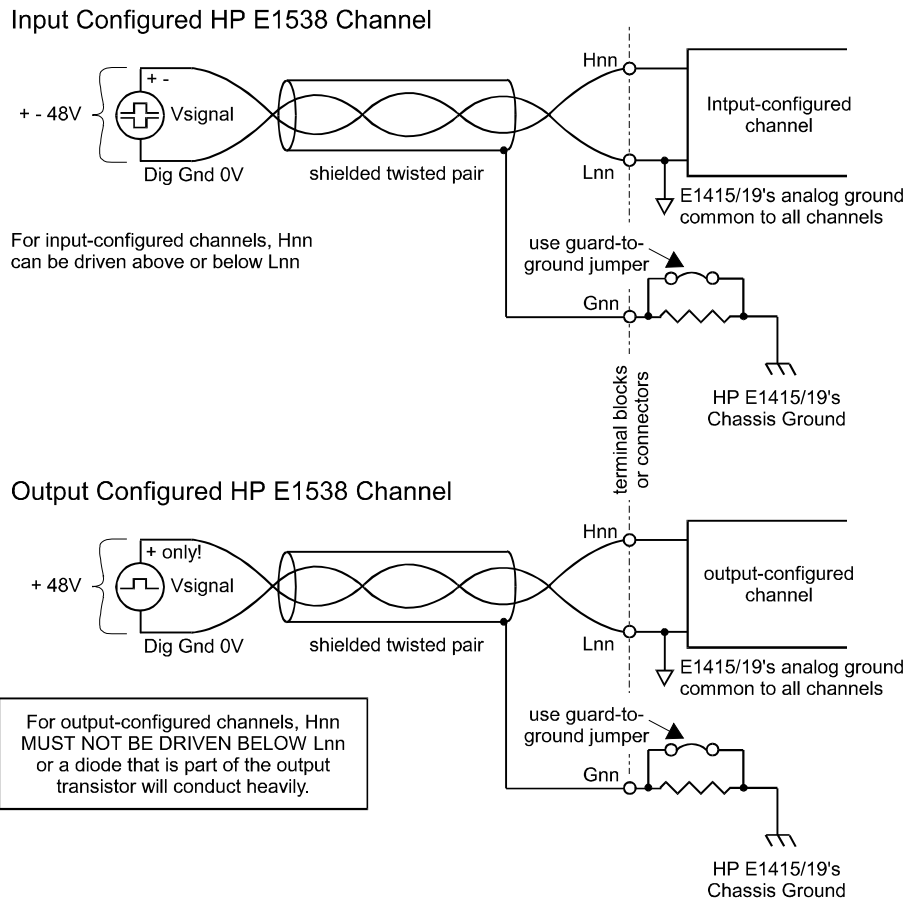


Figure 5. Recommended Connections and Voltage Limits

Input and Output Characteristics

This section describes the E1538's channel input and output electrical characteristics. Refer to Figure 6 for the following discussions.

Input Characteristics

When configured for input, E1538 channels provide digital input through the threshold comparator. The digital input threshold level is programmable with a SCPI command from -48 to +47.625 VDC in .375V steps (relative to the Lnn terminal). The threshold amplifier also provides typically 0.5 volts of hysteresis regardless of the threshold level setting. The input impedance in this configuration is greater than 100K Ω (as long as the 10K Ω pull-up resistor is OFF).

Channels 0 and 1 also provide the capability (when the VRS switch is ON) to read the output of variable reluctance sensors. Because the output of a VRS varies in relation to the velocity of the toothed wheel it is reading, the E1538A provides adaptive amplifiers for these channels. The function of the amplifier

is to maintain a constant-level digital output while the input varies from millivolts to several tens of volts.

For simple sensing of switches and open collector logic devices, a channel's pull-up resistor can be connected by closing its PU switch.

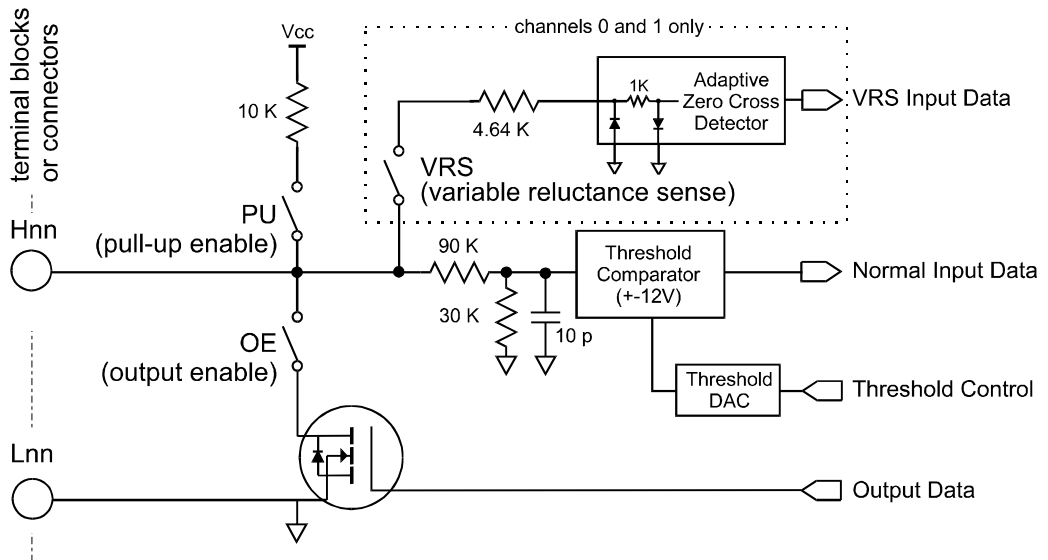


Figure 6. The E1538A Input/Output Characteristics

VRS Mode Input Operation (SCP channels 0 & 1 only)

When the VRS configuration switch is set to on, the input signal conditioning for that channel is changed to make it compatible with a typical variable reluctance sensor. The variable reluctance sensor is commonly used to detect rotational shaft position and/or velocity. Because the voltage output of a VRS is proportional to the rate of change of a magnetic field, different rotational velocities generate different signal amplitudes. The VRS-configured channel detects the negative going zero-crossing point of the signal. To minimize the effects of input noise, the zero-crossing detector can only be triggered if the positive-going portion of the signal exceeded an "arming" threshold. The arming circuit is reset when zero-crossing detector is triggered so it can't re-trigger until after the signal exceeds the arming threshold again. The arming threshold tracks the positive peak input level and is 80% of this peak value. By sensing the "zero-crossing" point of the input signal, the VRS mode isolates signal amplitude changes from affecting signal timing.

Note VRS enable ON is not allowed if PU enable is ON.

At high rotational speeds, variable reluctance sensors can generate voltage levels over 100VAC. The VRS inputs must be protected against signal levels over 17.5 Volts. If your VRS will generate voltages over 17.5, you must provide a resistor in series with the VRS input. The user-supplied resistor, together with the VRS input's 5.38K input impedance form a voltage divider that attenuates the input signal at the channel's Hi input terminal. Use the

formula $R_{external} = \frac{(V_{sensor} - 17.5)}{0.0032}$ to calculate the protection resistor's value.

Figure 7 shows the VRS mode input characteristics.

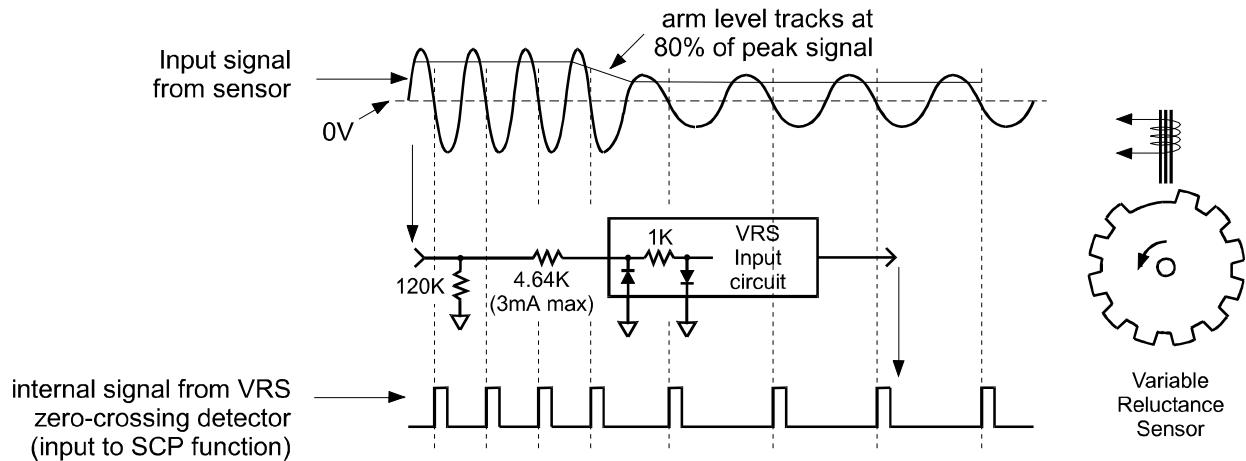


Figure 7. VRS Mode Input Characteristics

Output Characteristics

The output stage of the E1538A is simply a MOS FET transistor that is configured as "open-drain" when the pull-up resistor is not connected (PU switch is OFF). For simple interfacing to logic devices, the pull-up resistor can be connected by turning the PU switch ON. Operating voltages (output transistor off) at an output-configured channel can range from 0 to 48 volts. The output can sink up to 100mA of current (output transistor on).

Caution If the Hnn terminal is driven below the Lnn terminal while a channel is output-configured, an "inherent diode" in the output transistor will conduct heavily. This reverse current must be limited to 100mA, or damage to the SCP could result.

Note The *RST and power-on condition (true also after *TST) for output-configured channels will output a logical one (open-drain output off). You should keep this behavior in mind when applying the E1415 to your system. It is best to have your system's digital inputs use a high (one) as their safe state.

Programming With SCPI Commands

The SCPI commands shown here configure E1538 functions. The E1415 and E1419 don't provide SCPI commands to read an input channel or control an output channel. This communication with the SCP is provided by the Algorithm Language. Examples will show communication with algorithms.

Checking the ID of the SCP

To verify the SCP type(s) installed on your VXI module, use the `SYSTem:CTYPE? (@<channel>)` command.

- The *channel* parameter specifies a single channel in the channel range covered by the SCP of interest. The first channel number for each of the eight SCP positions are; 0,8,16,24,32,40,48, and 56.

The value returned for the E1538A SCP is:

HEWLETT-PACKARD,E1538A Enhanced Frequency/Totalize/PWM SCP,0,0

To determine the type of SCP installed on channels 0 through 7 send

```
SYST:CTYPE? (@100)           query SCP type @ ch 0
enter statement here         enter response string
```

Configuring the Channels

The E1538A has eight digital channels. The Power-on and *RST state is that all input-configured channels sense static digital state (`SENS:FUNC:COND`), and all output-configured channels output static digital state (`SOUR:FUNC:COND`). Logical sense is normal (`INP:POL NORM` and `OUTP:POL NORM`).

Configuring I/O Direction

Channels are configured for input or output with the I/O direction switches (see "Setting Configuration Switches" on page 6).

Programming Input Channels

This section deals with all aspects of programming input channel functions. Channels are configured for input with the I/O direction switches (see "Configuring Input-Output direction" on page 6). A related error message: 3123,"E1538 OE switch ON conflicts with this command."

Setting the Input Threshold Level

The E1538 allows programmatically setting the input threshold level for each input configured channel. The input threshold can be set from -46VDC to +46VDC with .375V resolution. While input polarity is set to `NORMAL`, an input level higher than the threshold level is considered a logic one, and an input level lower than the threshold level is considered a logic zero. If input polarity is set to `INVERTed`, an input level higher than the threshold level is considered a logic zero and an input level lower than the threshold level is considered a logic one. To set input threshold level use the command

```
INPut:THReshold:LEVel <level>,(@<ch_list>)
```

- *<level>* is a value between -46 and +46 inclusive. The resolution for *<level>* is 0.375 Volts. The *RST and power-on default for *<level>* is 1.78 volts.

Note

The value sent for *<level>* will be rounded to the nearest multiple of 0.375 Volts. For instance, 5 would be 4.875, 10 would be 10.125, 9.5 would be 9.375, and 15 would be 15. The `INP:THR:LEV?` query will return the actual setting.

- Channels in `<ch_list>` must be input configured channels

Determining the Input Threshold Level

To determine a channel's input threshold level, use the command:
 INPut:THReshold:LEVel? (@<channel>)

Note

Because the E1538 rounds `<level>` to the nearest multiple of 0.375, the returned value can be different from the value sent.

- `<channel>` must specify a single input-configured channel.
- INP:THR:LEV? returns a numeric value between -46 and +46. The C-SCPI type is **int32**.

To query the threshold level on the second channel at SCP position 4 send:

INP:THR:LEV? (@133)	<i>query 2nd chan on SCP pos. 4</i>
enter statement here	<i>returns threshold value</i>

Set Input Logic Sense

Use INPut:POLarity NORMal | INVerted,(@<ch_list>) to configure input channel logic sense. The operation is as follows:

INP:POL NORM input voltage greater than the threshold level sends a value of 1 (one) to the algorithm channel specifier.

INP:POL INV input voltage greater than the threshold level sends a value of 0 (zero) to the algorithm channel specifier.

To configure channels 40 to 43 to sense low input as logic 1

INP:POL INV,(@140:143)

Reading Static Digital State

This means reading a channel's current digital state when an algorithm executes. This is the default function assigned to all digital input channels after *RST and at power-up. To set individual channels to this function use the SCPI command [SENSE:]FUNCTION:CONDition (@<ch_list>). The value returned to an algorithm is a floating point representation of 0 or 1, depending on the state of the input signal and the channel's INP:POL setting.

To set channels 40 through 43 to input digital states

```
*RST
SENS:FUNC:COND (@140:143)           default for all dig inputs
ALG:DEF 'ALG1', writecvt(1140,40); writecvt(1141,41); writecvt(142,42);
writecvt(143,43);
```

```
INIT
do loop
  SENSE:DATA:CVT? (@40:43)
  read 4 CVT values
end loop
```

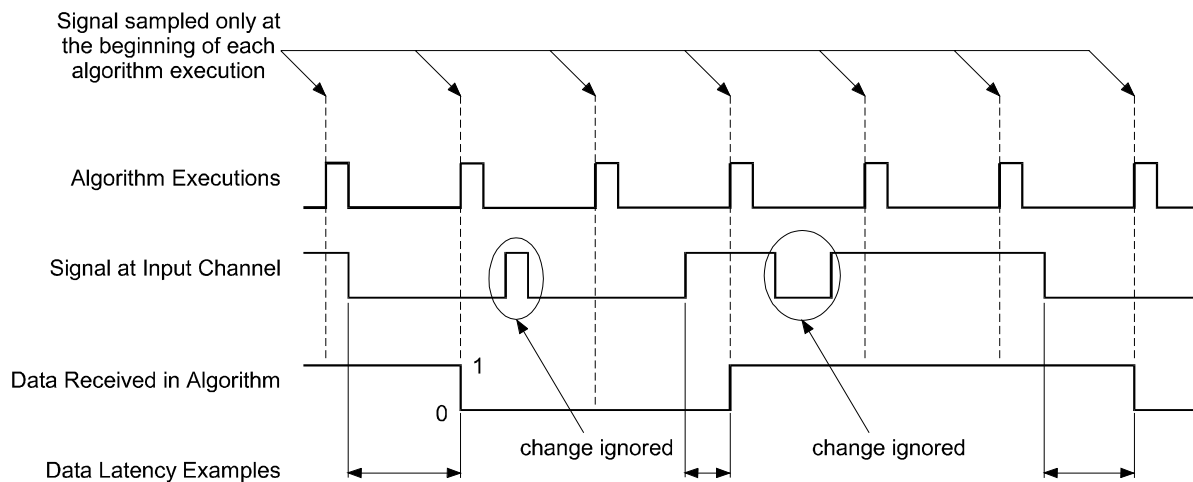


Figure 8. Input Static Digital States

Totalize Positive or Negative Edge State Changes

Use [SENSe:]FUNCTION:TOTalize (@<ch_list>) to configure channels to totalize. Totalize means to simply count state transitions (either positive going, or negative going). Figure 9 A shows totalizing transitions between each algorithm execution. Figure 9 B shows totalizing all transitions starting from the time the module last received an INITiate command.

Use [SENSe:]TOTalize:RESet:MODE INIT | TRIG,(@<ch_list>) to configure the totalize channel to either reset its count once each trigger event, or only when the module is INITiated. Use INP:POL INV to sense negative edges. The count capacity is 16,777,215 (24-bits, unsigned)

To totalize state changes at channel 44 starting from INITiate time

```
*RST
SENS:TOT:RES:MOD INIT,(@144)           ch 44 totalize reset at INIT
SENS:FUNC:TOT (@144)                   ch 44 is totalize input
ALG:DEF 'ALG1','writecvt(144,44);'     alg sends count to CVT
INIT
...
SENS:DATA:CVT? (@44)                   get totalize count from cvt
```

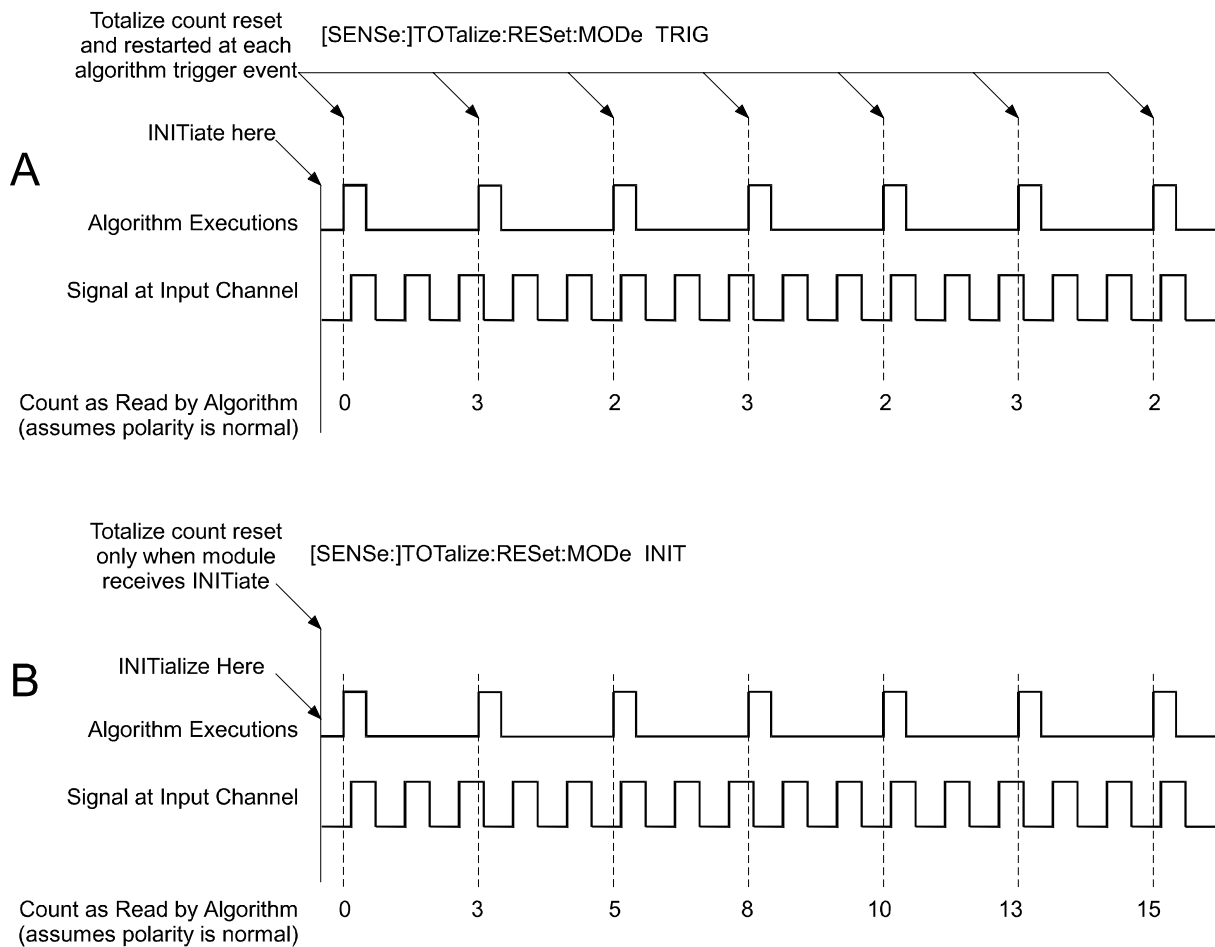


Figure 9. Input Totalize Count

About Period and Frequency Measurements

The E1538A actually measures signal period for both the period and frequency functions. If the measurement function is set to frequency rather than period, the SCP returns the reciprocal of the measured period. The resolution of each period measurement is based on the time processor chip's timer period (238.4nS). To improve resolution on faster input signals, multiple signal periods can be measured and averaged. For period measurements there are two different modes that can be used to control the number of periods to average. For frequency measurements only the APERTure mode is available.

1. The [SENSe:]PERiod:NPERiod mode explicitly sets the number of signal periods to measure and average. The time it takes the SCP to return a reading is dependent on the input signal period (for a given NPERiod setting), longer signal periods take longer to return a reading.

In NPERiod mode the actual measurement resolution (in seconds) is fixed while the relative resolution (as a percentage of the input signal period) is variable. That is, when NPERiods is set to provide an adequate resolution for short period signals, long period signals will have increased resolution.

2. The APERTure mode sets a fixed duration that the SCP will use to measure multiple signal periods. The actual effective APERTure *<time>* will be:

$$INT\left(\frac{\text{<time>}}{\text{signal_period}}\right) \times \text{signal_period} \cdot$$

The minimum aperture will be 1 signal period, and the maximum will be 255 signal periods.

In APERTure mode, the effective resolution (in seconds) varies with the period of the input signal. That is, as the signal period is reduced, the number of measurements averaged increases, thereby improving the effective resolution. However, the relative resolution (as a percentage of the input signal period) is fairly constant with changes in signal period.

Generally, more measurements (greater NPERiod count or longer APERTure time) means a more accurate frequency value. Of course more measurements means that the reading returned contains more latency (is "older" in relation to the signal's current frequency). To track fast changing frequency, you have to trade-off some accuracy with a shorter aperture time.

Measure Frequency

Use [SENSe:]FREQuency:APERture <time>,(@<ch_list>) to configure the frequency counter channels' measurement interval.

Use [SENSe:]FUNction:FREQuency (@<ch_list>) to configure channels to measure signal frequency.

To measure frequency at channel 45 with aperture of 1 second

```
*RST
TRIGGER:TIMER .2
SENS:FUNC:FREQ (@145)
SENS:FREQ:APER 1,(@145)
ALG:DEF 'ALG1','writecvt(1145,45);'
INIT
do loop
  SENS:DATA:CVT? (@45)
  read value from CVT query above
end loop
```

*Alg executes at .2 sec intervals
ch 45 is frequency counter
meas and avg sig periods for 1S
alg puts frequency in CVT
start algorithm execution*

get frequency from CVT

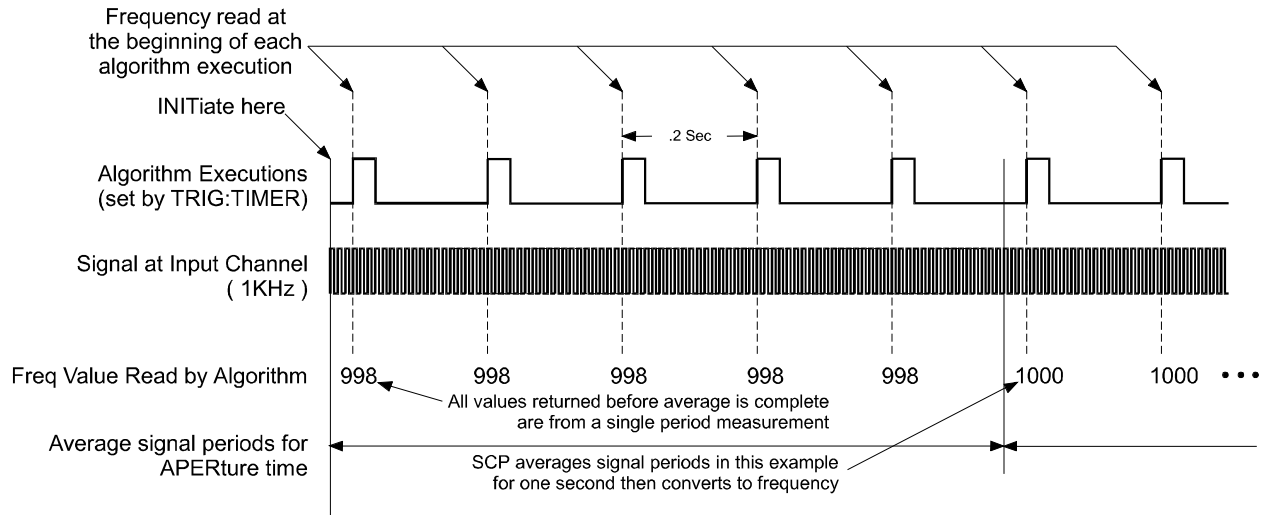


Figure 10. Input Frequency

Measure Period

Use [SENSE:]PERiod:MODE APERture|NPERiods,(@<ch_list>) to select the measurement interval setting mode.

Depending on the mode selected above use [SENSE:]PERiod:APERture <time>,(@<ch_list>) or use [SENSE:]PERiod:NPERiods <n_periods>,(@<ch_list>) to set the interval for measuring and averaging signal periods.

For PERiod function, the E1538 supports two distinct measurement ranges:

1. When SENS:PER:RANGE is set to 1sec, the E1538 can measure periods from 10usec - 1sec. The value of SENS:PER:APER can range from 10usec - 1sec.
2. When SENS:PER:RANGE is set to 4sec, the E1538 can measure periods from 40usec - 4sec. The value of SENS:PER:APER can range from 40usec - 4sec. See SENS:PER:RANGE command on page 61

Use [SENSE:]FUNction:PERiod (@<ch_list>) to configure channels to measure signal period.

To measure the signal period at channel 45 with aperture of 01 second

```
*RST
TRIGGER:TIMER .2
SENS:FUNC:PER (@145)
SENS:PER:RANGE 1,(@145)
SENS:PER:MODE APER(@145)
SENS:PER:APER 1,(@145)
ALG:DEF 'ALG1','writecvt(I145,45);'
INIT
do loop
  SENS:DATA:CVT? (@45)
  read value from CVT query above
end loop
```

*Alg executes at .2 sec intervals
ch 45 to measure signal period
set period range 10µsec - 1sec
set meas and avg interval mode
meas and avg sig periods for 1S
alg puts period in CVT
start algorithm execution*

get period from CVT

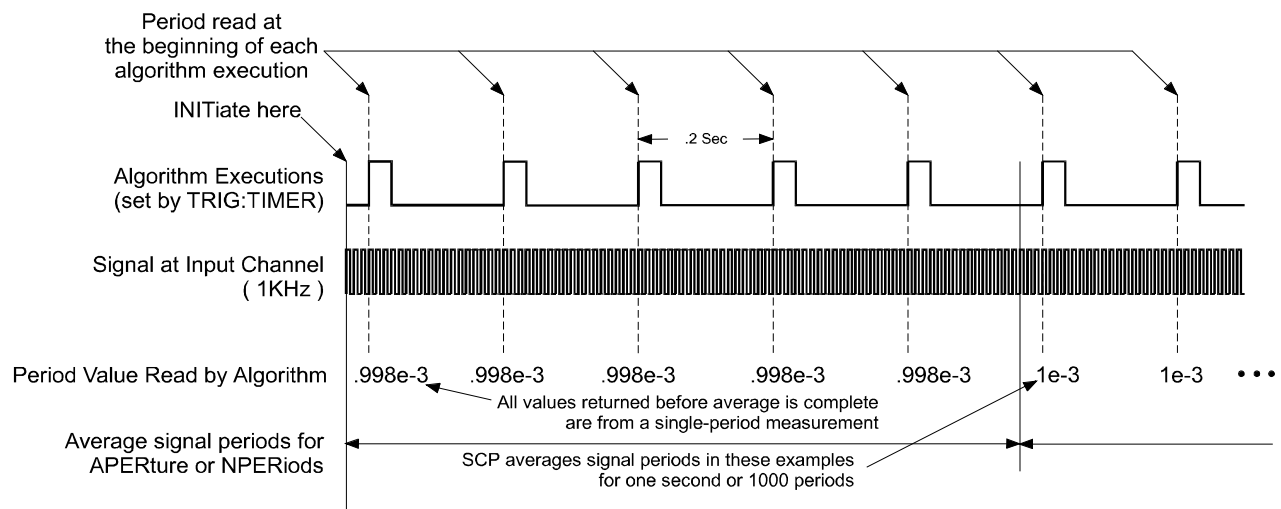


Figure 11. Input Period

To measure period at channel 45 as the average of 1000 signal periods:

```
*RST
TRIGGER:TIMER .2
SENS:FUNC:PER (@145)
SENS:PER:RANGE 1,(@145)
SENS:PER:MODE NPER(@145)
SENS:PER:NPER 1000,(@145)
ALG:DEF 'ALG1','writecvt(1145,45);'
INIT
do loop
  SENS:DATA:CVT? (@45)
  read value from CVT query above
end loop
```

*Alg executes at .2 sec intervals
ch 45 to measure signal period
set period range 10µsec - 1sec
set meas interval by n periods
meas and avg 1000 sig periods
alg puts period in CVT
start algorithm execution*

get period from CVT

Measure Pulse Width

This means that the E1538 will measure the width of the logic 1 portion of a pulse. The pulse width is sent to the algorithm in units of seconds. To measure the high portion of a pulse (positive going edge to negative going edge) set the channel input polarity to `INP:POL NORM,(@<ch_list>)`. To measure the low portion of the pulse (negative going edge to positive going edge) set the channel input polarity to `INP:POL INV,(@<ch_list>)`.

The value returned to an algorithm can be from 5 μ Sec to 1 Second with 59.6nSec resolution.

To configure channels to measure pulse width use the command `[SENSe:]:FUNCtion:PWIDth <avg_count>,(@<ch_list>)`

- `<avg_count>` sets the number of pulses to average when forming the pulse duration value. More counts give more accurate readings, but slower response to changing pulse widths.
- `<ch_list>` specifies the channels that will read pulse widths

To measure pulse width on channels 46&47

```
*RST
SENS:FUNC:PWID 4,(@146,147)           read puls width on chs 46&47
      Algorithm reads the pulse widths on channels 146 and 147 and returns these
      values in CVT elements 46 and 47
ALG:DEF 'ALG1','writecv( 1146, 46 ); writecv( 1147, 47 );'
INIT                                     start algorithm
...
SENS:DATA:CVT? (@46,47)                 read pulse widths from CVT
```

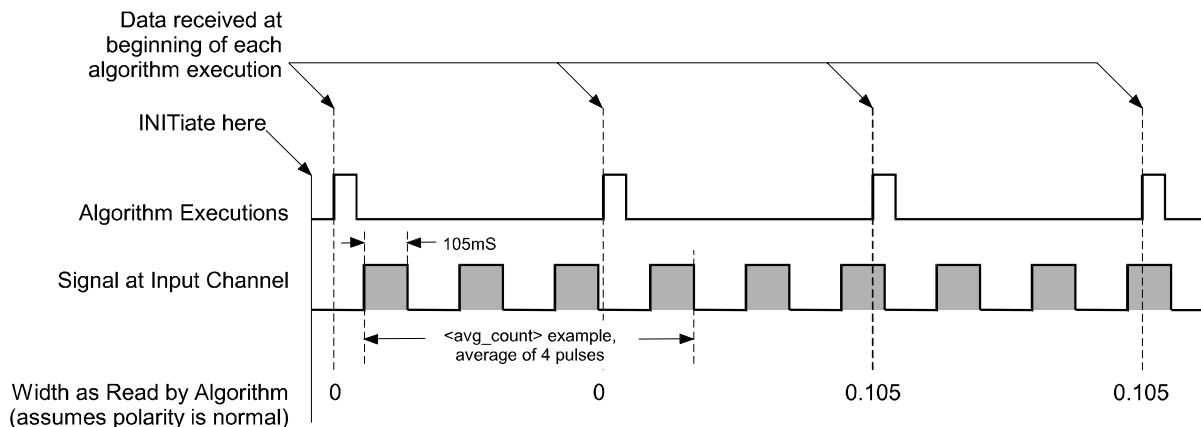


Figure 12. Measure Pulse Width

Sense Quadrature Position

This means that the E1538 will convert a digital quadrature signal pair into an absolute 24-bit count. The count value can be read by the algorithm.

The E1538's quadrature position function increments a counter value each time there is a transition on either of the quadrature channel pair. When the lower numbered channel's signal LEADS the higher numbered channel, the function counts up. When the lower numbered channel LAGS the higher numbered channel, the function counts down.

To configure a pair of channels to sense quadrature count use
[SENSe:]FUNcTion:QUADrature [<count_preset>,@<ch_list>]

- <count_preset> if included, allows presetting the absolute counter associated with the channel pair. All quadrature pairs in <ch_list> will be preset to the same value. If not included, the default count at algorithm start will be zero. <count_preset> can range from 0 to 16,777,215. The variable type is int32
- <ch_list> must always specify both channels of a pair. More than one pair can be specified. Both channels of any pair must be adjacent. <ch_list> can specify channels on more than one E1538. The channel numbers in <ch_list> must be in ascending order. The related error messages are:
3115, "Channels specified are not in ascending order."
3116, "Multiple channels specified are not grouped correctly."
3117, "Grouped channels are not adjacent."
3122, "This multiple channel function must not span multiple SCPs."

The algorithm reads the current count through the low numbered channel. The count is an unsigned 24-bit value ranging from 0 to 16,777,215. The counter can roll over from 16,777,2215 to 0, and roll under from 0 to 16,777,215 is 16,777,215.

To configure channels 42 and 43 as one quadrature pair, and channels 48 and 49 as another pair

```
*RST
SENS:FUNC:QUAD 8192,(@142,143)      pair 42&43 preset to count of
                                     8192
SENS:FUNC:QUAD 0,(@148,149)        pair 48&49 preset to 0
                                     algorithm will retrieve values from input channels and place in CVT elements
ALG:DEF 'ALG1','writecvt(1142,42); writecvt(1148,48);'
INIT                                start algorithm execution
begin loop                          loops between here and end loop
SENS:DATA:CVT? (@42,48)             get quadrature position count
display or otherwise use count info
end loop
```

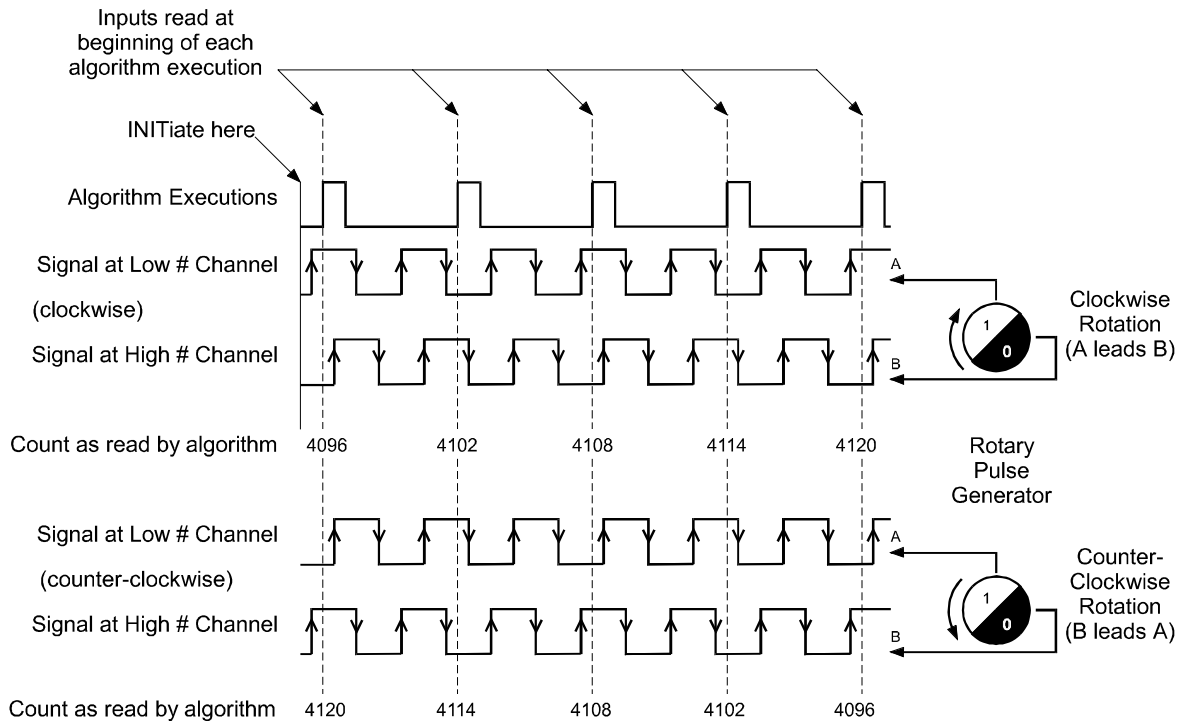


Figure 13. Sense Quadrature Position

Sense Rotational Velocity

This means that the E1538 will read the rotational velocity of a toothed wheel sensor. The E1538 measures tooth-to-tooth period and converts it into units of revolutions per second (RPS). This function can only be linked to the E1538's first channel. The function works for wheels that have either a missing, or an extra tooth to mark their index position. Figure 14 shows a wheel sensed with a variable reluctance sensor (using the VRS input option), but any wheel sensing method is applicable as long as it provides a digital output to the RVEL channel.

The value read by the algorithm can range from $\frac{1}{n_{teeth}}$ RPS to $\frac{100,000}{n_{teeth}}$ RPS.

As well as sensing rotational velocity, SENS:FUNC:RVEL provides the reference position to the SOUR:FUNC:RPULSE function that generates angular positioned pulses. See page 30 for more information on RPULSE.

To assign a channel to sense rotational velocity, use the command:
 [SENSe:]FUNCTION:RVELocity <n_teeth>,<index_type>,(@<ch_list>)

- <n_teeth> is the number of teeth that the wheel would have if it didn't have missing or extra teeth. For example, we would set <n_teeth> to 12 for the wheel shown in Figure 14, even though with the missing tooth, there are only 11. <n_teeth> can range from 3 to 255.

- *<index_type>* can be either of the strings "MISSing", or "EXTRa"
- *<ch_list>* must be the first channel on the SCP, but can contain more than one channel provided that each channel is on a separate E1538. See following note. The related Error Messages are:
3110, "Channel specified is invalid for RVELocity function.

Note Only one channel on any E1538 SCP can be assigned to the SENS:FUNC:RVEL function, and it must be the first channel on the SCP."

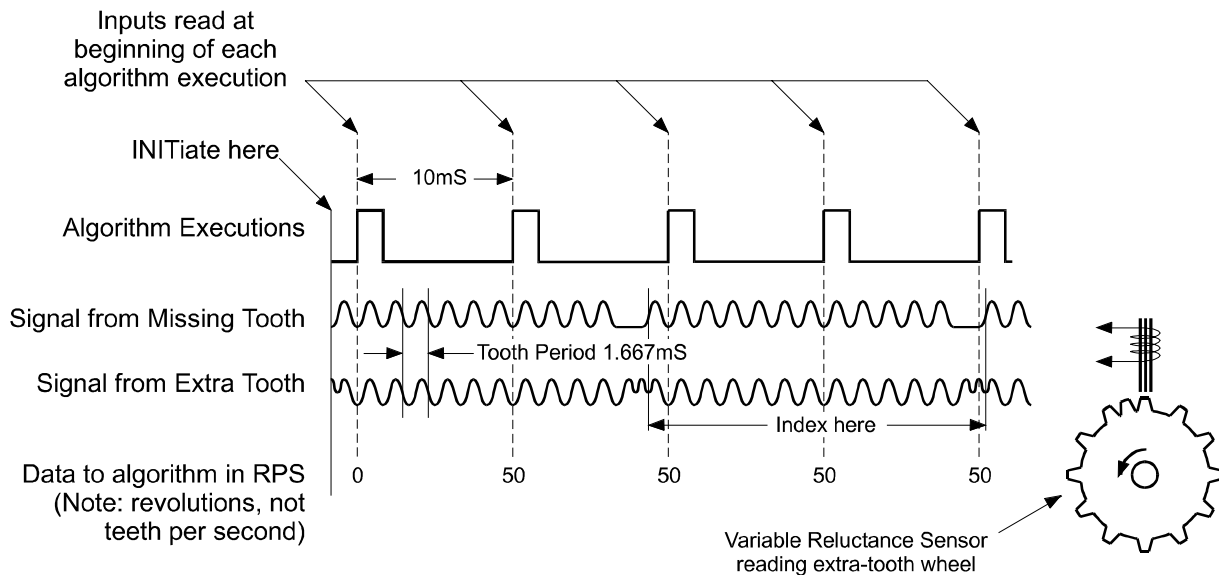


Figure 14. Sense Rotational Velocity

Example of Rotational Velocity Sense

Channel 40 senses RVEL and the algorithm reads and returns the velocity value in CVT element 40

```

*RST
SENSE:FUNC:RVEL 12,MISSING,(@140) 12 toothed wheel with one
                                     missing, from channel 40
ALG:DEF 'ALG1','writecvt(1140,40);' simply puts value from ch 40 into
                                     CVT element 40
INIT
loop always                          start the algorithm
SENS:DATA:CVT? (@40)                will loop from "end loop" to here
display the RVEL value               query the value from CVT 40
end loop

```

Programming Output Channels

This section deals with all aspects of programming output channel functions. Channels are configured for output with the I/O direction switches (see "Configuring Input-Output direction" on page 6). A related error message: 3124, "E1538 OE switch OFF conflicts with this command."

Controlling Output Polarity

Use `OUTPut:POLarity NORMal | INVerted,(@<ch_list>)` to configure output channel logic sense. The operations is as follows:

`OUTP:POL NORM` a logical 1 output from the algorithm, or generated within the E1538 (single or repetitive pulses) will turn off the output transistor (can be pulled up).

`OUT:POL INV` a logical 1 output from the algorithm, or generated within the E1538 (single or repetitive pulses) will turn off the output transistor (pulls low).

To configure channels 40 to 43 to drive their outputs low for a logic 1

```
OUTP:POL INV,(@140:143)
```

Output Static Digital State

This means setting a channel's digital state when an algorithm executes. To set individual channels to this function use the SCPI command `SOURce:FUNctIon[:SHAPe]:CONDition (@<ch_list>)`

To configure channels 40 through 43 as static digital outputs, send

```
*RST
SOUR:FUNC:COND (@140:143) default for all digital outputs
ALG:DEF 'ALG1',static float ch0=0, ch1=1, ch2=0, ch3=1; O140=ch0;
O141=ch1; O142=ch2; O143=ch3;
INIT
```

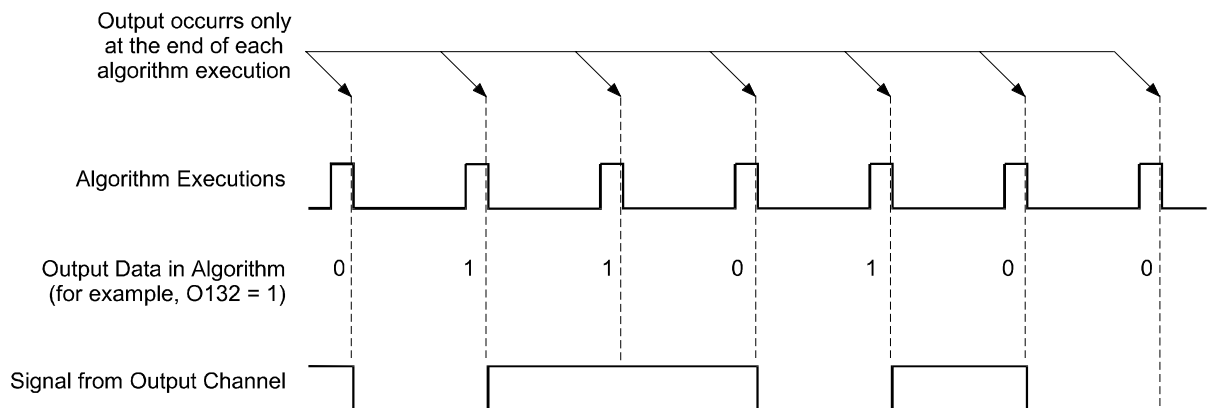


Figure 15. Output Static Levels

Variable Width Pulse Per Trigger

This means that the channel generates a pulse whose width is specified by the algorithm each time the algorithm executes. The value sent by the algorithm can range from 7.87 μ Sec to 7.812mSec.

The command sequence to set-up this mode is:

SOURce:FUNCtion:PULSe (@<ch_list>) to enable pulse generation.

*the following two commands return the E1538 to the Single pulse-per-trigger mode from either the FM or Pulse Width Modulation modes. Since Single pulse-per-trigger is the default pulse mode at power-up or after *RST, only *RST then SOUR:FUNC:PULS (@<ch_list>) are actually needed.*

SOURce:FM[:STATe] OFF,(@<ch_list>) to disable FM mode.

SOURce:PULM[:STATe] OFF,(@<ch_list>) to disable PWM mode.

To configure channel 44 to output a single controlled width pulse per trigger

*RST

*after *RST, sour:func:puls is all that is required to enable the default single pulse-per-trigger mode.*

SOUR:FUNC:PULS (@144) *channel sources pulses...*

ALG:DEF 'ALG1','static float outputpulse=0.001; O144=outputpulse;'

INIT *start alg execution*

...

ALG:SCAL 'ALG1','outputpulse',5E-4 *.5ms pulse*

ALG:UPDATE

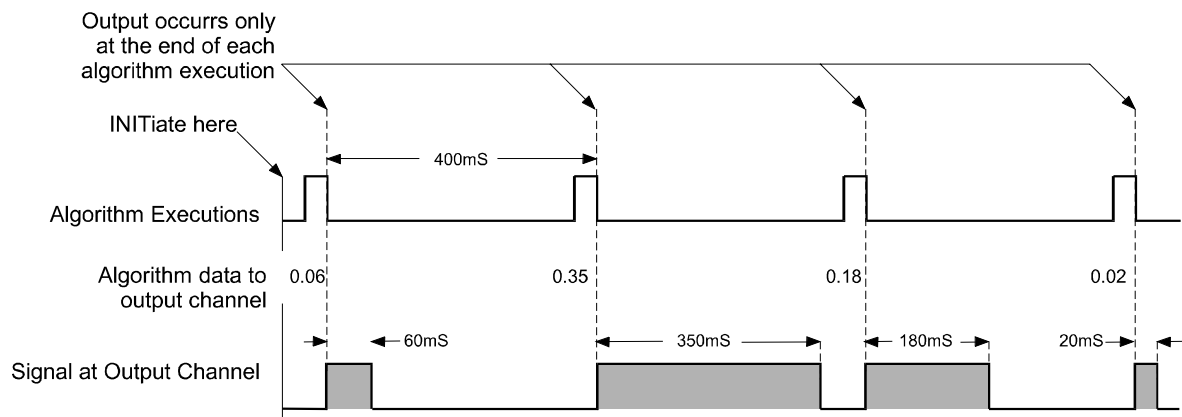


Figure 16. Output Variable Width Pulse per Trigger

Variable Width Pulse Train (PWM)

This means that the E1538 outputs a continuous train of pulses whose logic 1 pulse width is controlled by the algorithm. The frequency is set by a SCPI command before INIT. Use the following command sequence to set up this mode:

```
SOURce:FUNctio:n:PULSe (@<ch_list>) to enable pulse generation.
SOURce:PULM[:STATe] ON,(@<ch_list>) to select the PWM mode
SOURce:PULSe:PERiod <period>,(@<ch_list>) to set the pulse repetition
period (frequency = 1/<period>). <period> can range from 25µSec to
7.812mSec.
```

The pulse width value sent by the algorithm can range from $7.87\mu\text{Sec}$ to $\text{<period>} - 7.87\mu\text{Sec}$. Resolution within this range is 238.4nSec . 100% duty-cycle is output when the algorithm sends a value greater than or equal to <period> . 0% duty-cycle is output when the algorithm sends a value less than or equal to 0.

To configure channel 45 to output a variable pulse width continuous train

```
SOUR:FUNC:PULS (@145)           channel sources pulses...
SOUR:PULM ON,(@145)           and continuous PWM train
SOUR:PULS:PER .0005,(@145)    .5 msec period (2KHz freq)
```

The algorithm can now output a value to channel 45 to control pulse width of the logic 1 portion of the waveform:

```
O145 = 333E-6 /* channel 45 pulse width will be 333 µsec */
```

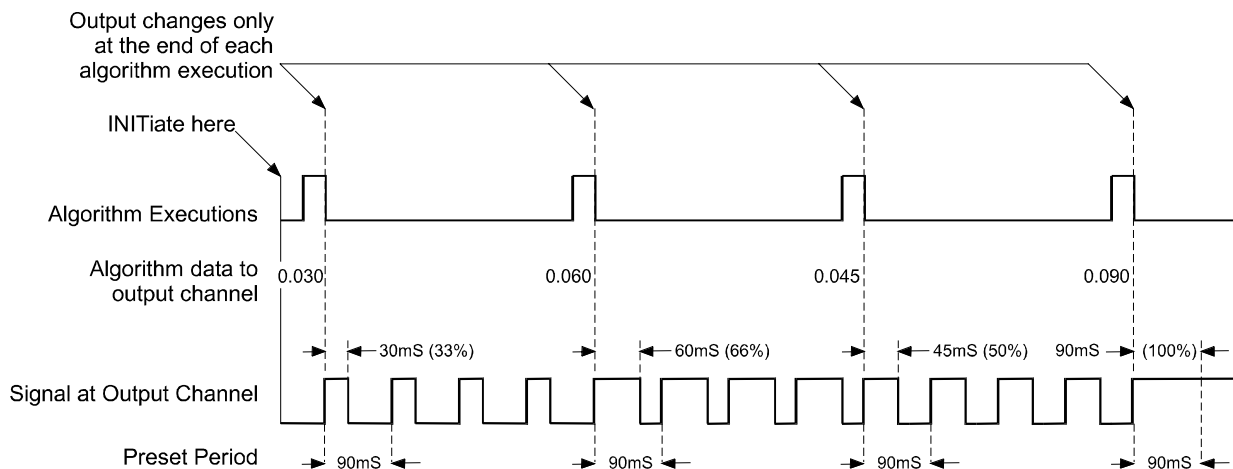


Figure 17. Output Pulse-Width-Modulated Signal

Variable Frequency Fixed Width Pulse Train (FM)

This means that the E1538 outputs a continuous train of pulses whose frequency is controlled by the algorithm. The logic 1 level pulse width is set by a SCPI command before INIT. Use the following command sequence:

```
SOURce:FUNCtion:PULSe (@<ch_list>) to enable pulse generation.
SOURce:FM[:STATe] ON,(@<ch_list>) to select the FM mode.
SOURce:PULSe:WIDTh <width>,(@<ch_list>) to pre-set the pulse width of the logic 1 portion of the waveform. <width> can range from 7.87µSec to 7.812mSec.
```

The frequency value sent by the algorithm can range from 128Hz to 40KHz.

The frequency resolution is $\frac{f_{out}^2}{4.194 \text{ MHz}}$

To configure channel 45 to output variable frequency continuous train with fixed pulse width

```
SOUR:FUNC:PULS (@145)           channel sources pulses...
SOUR:FM ON,(@145)              and continuous pulse train
SOUR:PULS:WIDT .001,(@145)     1 msec fixed pulse width
```

The algorithm can now output a frequency value to channel 45:

O145 = 250 /* channel 45 will source 250 Hz pulse train */

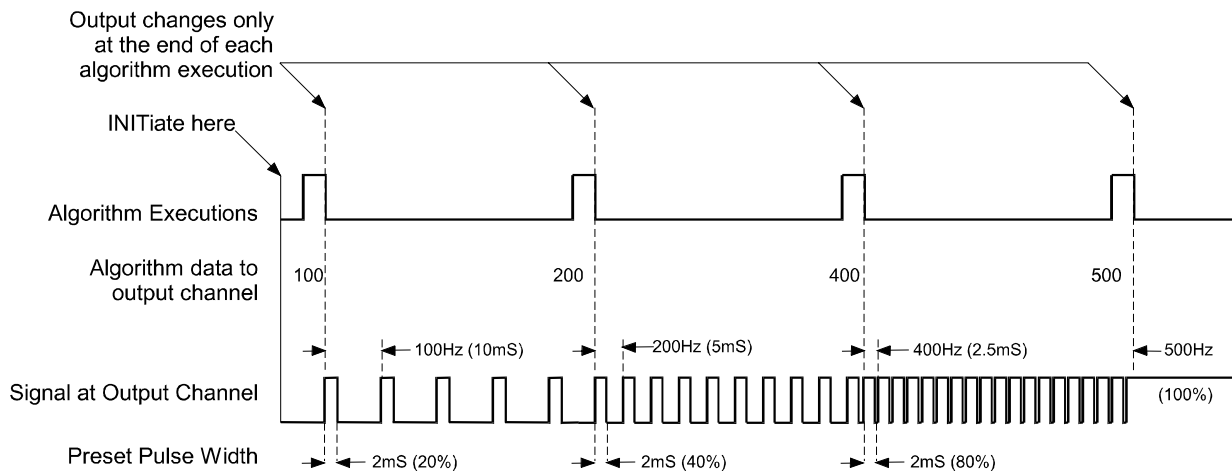


Figure 18. Output Fixed Pulse Width Variable Frequency (FM)

Variable Frequency Square-Wave Pulse Train (FM)

This means that the E1538 outputs a continuous train of pulses whose frequency is controlled by the algorithm. The duty-cycle of the waveform is always 50%. Use the following command sequence:

SOURce:FUNCtion:SQUare (@<ch_list>) to enable square-wave generation.
 SOURce:FM[:STATe] ON,(@<ch_list>) to select the FM mode.

The frequency value sent by the algorithm can range from 64Hz to 40KHz.

The frequency resolution is $\frac{f_{out}^2}{4.194 \text{ MHz}}$

To configure channel 45 to output variable frequency continuous train with 50% duty cycle (square wave)

SOUR:FUNC:SQUARE (@145) *channel sources square wave...*
 SOUR:FM ON,(@145) *and continuous PWM train*

The algorithm can now output a frequency value to channel 45:

O145 = 2000 /* channel 45 will source 2 KHz square wave */

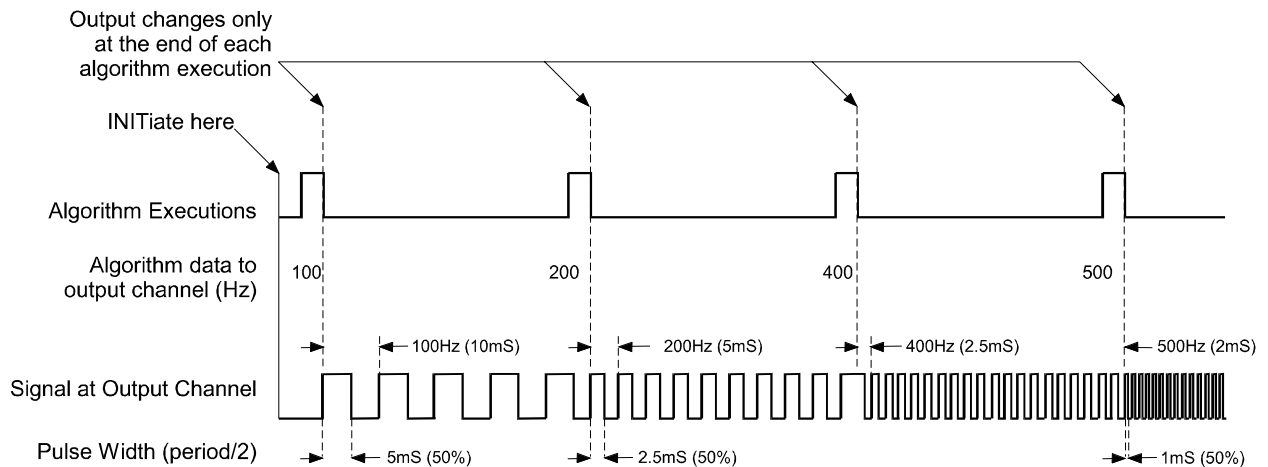


Figure 19. Output Square Wave Variable Frequency (FM)

Rotationally Positioned Pulse Output

This means that the E1538 will generate pulses which are positioned by angle (usually shaft angle). The rotational pulse function requires a rotational reference, and this is provided by the SENS:RVEL function from the SCP's first channel. There are four related commands that set up rotational pulses. Combinations of these commands can set up four different rotational pulse modes. Figure 20 shows these modes and the command sequence for each. Following Figure 20 is the command reference for all four commands. Following that are examples of each of the four modes.

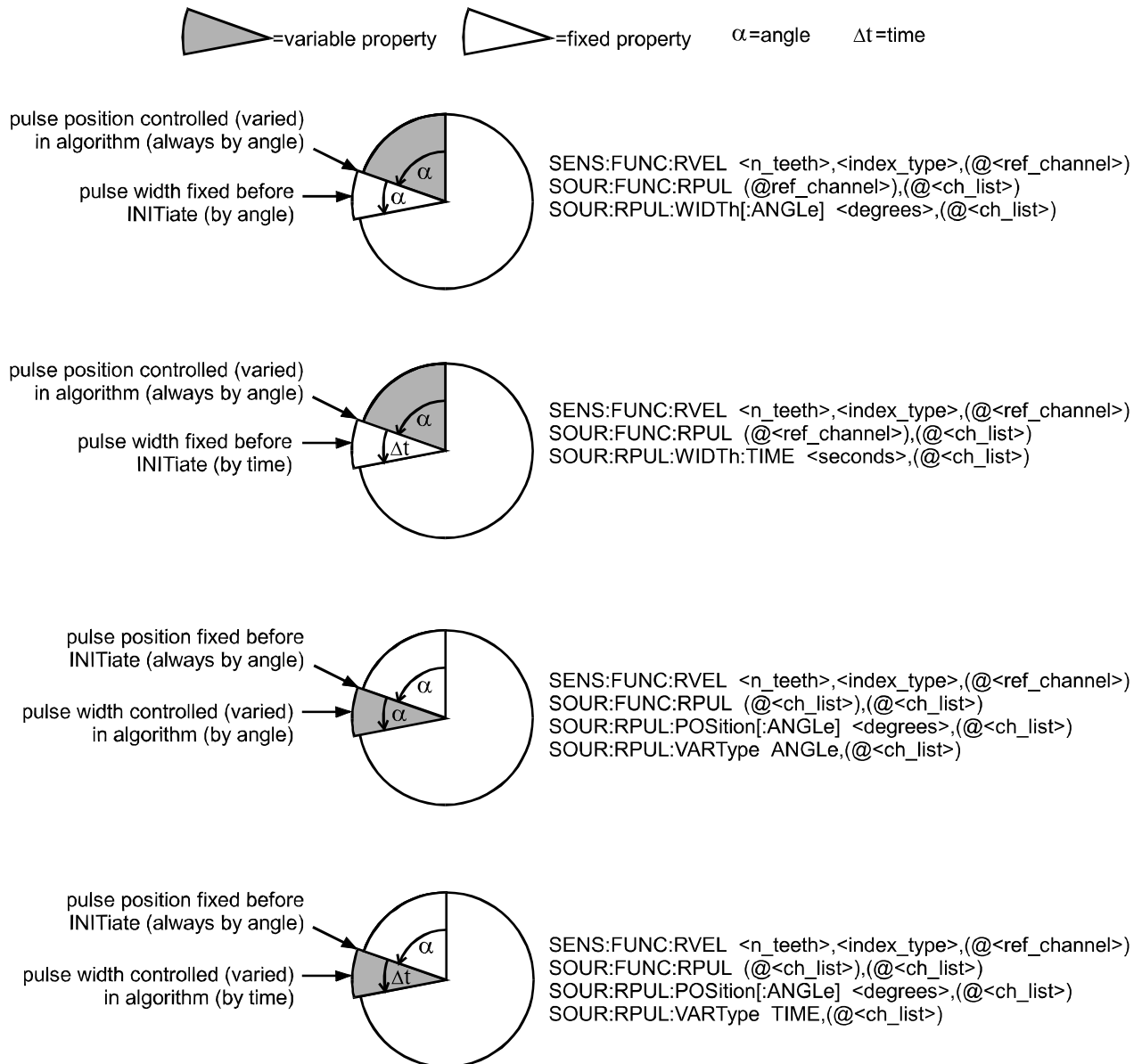


Figure 20. Four Modes of Rotationally Positioned Pulses

Rotational Pulse Command Usage

Use **SOURce:FUNCTION:RPULse** (@<ref_channel>),(@<ch_list>) to link channels in <ch_list> to the rotational pulse function. The channel in <ref_channel> will be linked to the SENS:FUNC:RVEL function to provide the rotational reference information to SOUR:FUNC:RPUL.

The commands:

SOURce:RPULse:POSition[:ANGLE] <degrees>,(@<ch_list>),
SOURce:RPULse:WIDTh[:ANGLE] <degrees>,(@<ch_list>), and
SOURce:RPULse:WIDTh:TIME <seconds>,(@<ch_list>) fix the channels' rotational pulse position (SOUR:RPUL:POS:ANGL), or the rotational pulse width (SOUR:RPUL:WIDT:ANGL and :TIME) before the INITiate command. The remaining pulse property - the property NOT specified in one of these commands - will be controlled within the algorithm.

- <ch_list> specifies the SOUR:FUNC:RPUL channel(s) that will be set to the property specified by the command syntax.
- For pulse position, <degrees> can range from -33,554,430 to 33,554,430 degrees, with a resolution of 1 degree. The pulse is positioned at <degrees> modulo 360.

For pulse width, <degrees> can range from 0 to 360 degrees, with a resolution of 1 degree.

- <time> specifies pulse width in seconds, ranging from .00000787 (7.87 μ S) to .015624 (15.624mS), with a resolution of 238.4nS

The command:

SOUR:RPULse:VARType ANGLE | TIME,(@<ch_list>) specifies the type of value that will be controlled (varied) by the algorithm.

- ANGLE specifies that the algorithm will send values of angle (in degrees) to the channel(s).

TIME specifies that the algorithm will send values of time (in seconds) to the channel(s).

- <ch_list> specifies the SOUR:FUNC:RPUL channel(s) that will be controlled (varied) by the algorithm.

Rotational Pulse Mode: Variable Angular Position, Preset Pulse Width (by angle)

In this mode, the angular position of the pulses is controlled by the algorithm, and the width (duration in degrees) is preset before INIT. See Figure 21. Use the following command sequence:

SOURce:FUNCtion:RPULse (@<ref_channel>),(@<ch_list>) to select the channels that will output angular positioned pulses, and to specify the reference channel.

SOURCe:RPULse:WIDTh[:ANGLE] <degrees>,(@<ch_list>) to preset the pulse width in degrees. The algorithm will control the angular pulse position.

Example of variable position, preset width (by angle):

Set up channel 40 as the reference channel, and channels 45 through 47 to output variable position pulses:

```
*RST
SENS:FUNC:RVEL 12,MISS,(@140)           sense rvel for reference channel
SOUR:FUNC:RPULSE (@140),( @145:147) 3 rotational pulse output chans
SOUR:RPULSE:WIDT:ANGL 15,( @145:147) preset pulse width to 15 degrees
```

Algorithm outputs pulses on all three channels with variable position.

```
ALG:DEF 'ALG1','static float Pos1, Pos2, Pos3;O145 = Pos1; O146 =
Pos2;O147 = Pos3;'
ALG:SCALAR'ALG1','Pos1',60           preset ch 45's pulse pos to 60°
ALG:SCALAR'ALG1','Pos2',180         preset ch 46's pulse pos to 180°
ALG:SCALAR'ALG1','Pos3',300         preset ch 47's pulse pos to 300°
ALG:UPDATE                           .
INIT                                  start algorithm execution
•
•
•                                     calculate values for NewPos(n)
ALG:SCALAR 'ALG1','Pos1',NewPos1     later, adjust channel 45's
                                       position while algorithm running
ALG:SCALAR 'ALG1','Pos2',NewPos2     later, adjust channel 46's
                                       position while algorithm running
ALG:SCALAR 'ALG1','Pos3',NewPos3     later, adjust channel 47's
                                       position while algorithm running
ALG:UPDATE                           values in update queue sent to
                                       variables
```

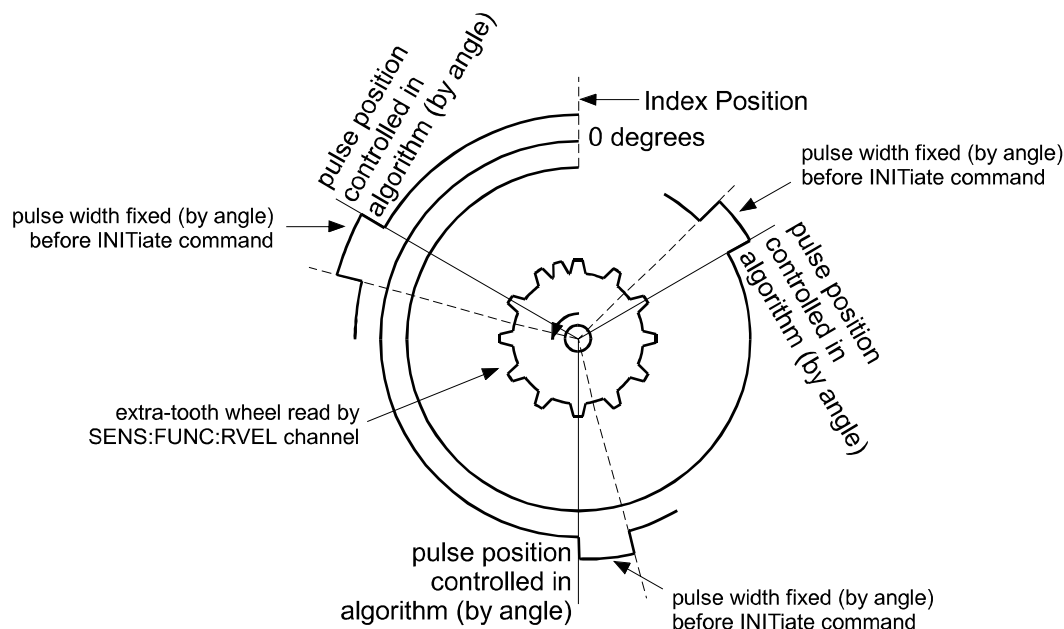


Figure 21. Variable Position, Width Preset by Angle

**Rotational Pulse Mode:
Variable Angular Position, Preset Pulse Width (by time)**

In this mode, the angular position of the pulses is controlled by the algorithm, and the width (duration in seconds) is preset before INIT. See Figure 22. Use the following command sequence:

SOURce:FUNCTION:RPULse (@<ref_channel>),(@<ch_list>) to select the channels that will output angular positioned pulses, and to specify the reference channel.

SOURce:RPULse:WIDTh:TIME <seconds>,(@<ch_list>) to preset the pulse width in seconds. The algorithm will control the angular pulse position.

Example of variable position, preset width:

Set up channel 40 as the reference channel, and channels 45 through 47 to output variable position pulses:

```
*RST
SENS:FUNC:RVEL 12,MISS,(@140)           sense rvel for reference channel
SOUR:FUNC:RPULSE (@140),( @145:147)     3 rotational pulse output chans
SOUR:RPULSE:WIDTh:TIME .001,( @145:147) preset pulse width to 1 millisec.
                                           Algorithm outputs pulses on all three channels with variable position.
ALG:DEF 'ALG1','static float Pos1, Pos2, Pos3;O145 = Pos1; O146 = Pos2;
O147 = Pos3;'
ALG:SCALAR'ALG1','Pos1',60                preset ch 45's pulse pos to 60°
ALG:SCALAR'ALG1','Pos2',180              preset ch 46's pulse pos to 180°
```



```

ALG:SCALAR'ALG1','Pos3',300
ALG:UPDATE
INIT
•
•
•
ALG:SCALAR 'ALG1','Pos1',NewPos1
ALG:SCALAR 'ALG1','Pos2',NewPos2
ALG:SCALAR 'ALG1','Pos3',NewPos3
ALG:UPDATE

```

preset ch 47's pulse pos to 300°
.
start algorithm execution

calculate values for NewPos(n)

later, adjust channel 45's
position while algorithm running
later, adjust channel 46's
position while algorithm running
later, adjust channel 47's
position while algorithm running
values in update queue sent to
variables

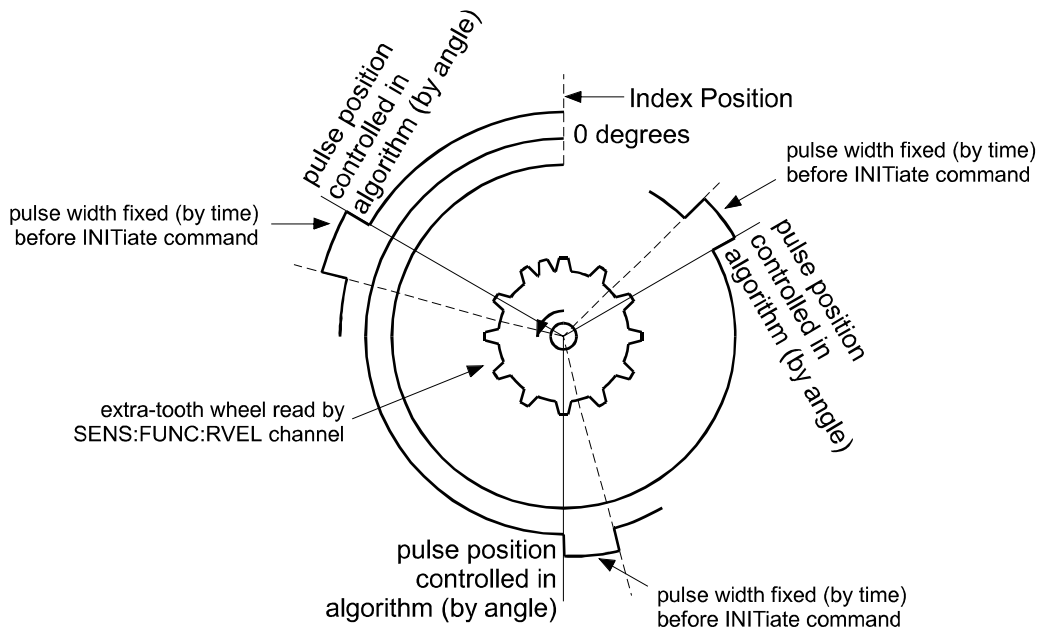


Figure 22. Variable Position, Width Preset by Time

**Rotational Pulse Mode:
Variable Pulse Width (by angle), Preset Angular Position**

In this mode, the angular pulse width is controlled by the algorithm, and the angular position is preset before INIT. See Figure 23. Use the following command sequence:

SOURCE:FUNCTION:RPULSE (@<ref_channel>),(@<ch_list>) to select the channels that will output angular positioned pulses, and to specify the reference channel.

SOURce:RPULse:POSition[:ANGLE] <degrees>,@<ch_list>, to preset the angular pulse position in degrees. The algorithm will control the pulse duration.

SOUR:RPULse:VARType ANGLE,@<ch_list> to set the type of value that will vary with algorithm control (in this case pulse width ANGLE).

Example of variable width (by angle), preset position:

Set up channel 40 as the reference channel, and channels 45 through 47 to output variable width pulses:

```
*RST
SENS:FUNC:RVEL 12,MISS,@140           sense rvel for reference channel
SOUR:FUNC:RPULSE (@140),(@145:147)    3 rotational pulse output chans
SOUR:RPULSE:POS:ANGL 20,@145         preset channel 45 pulse position
                                       to 20 degrees
SOUR:RPULSE:POS:ANGL 140,@146        preset channel 46 pulse position
                                       to 140 degrees
SOUR:RPULSE:POS:ANGL 260,@147        preset channel 47 pulse position
                                       to 260 degrees
SOUR:RPULSE:VART ANGL,@145:147      algorithm will control pulse
                                       width by ANGLE
                                       Algorithm outputs pulses on all three channels with variable width.
ALG:DEF 'ALG1','static float Width1, Width2, Width3;O145 = Width1; O146
= Width2; O147 = Width3;'
ALG:SCALAR'ALG1','Width1',5          preset ch 45's pulse width to 5°
ALG:SCALAR'ALG1','Width2',10        preset ch 46's pulse width to 10°
ALG:SCALAR'ALG1','Width3',15        preset ch 47's pulse width to 15°
ALG:UPDATE
INIT                                  start algorithm execution
•
•
•
ALG:SCALAR 'ALG1','Width1',NewWidth1 later, adjust channel 45's width
                                       while algorithm is running
ALG:SCALAR 'ALG1','Width2',NewWidth2 later, adjust channel 46's width
                                       while algorithm is running
ALG:SCALAR 'ALG1','Width3',NewWidth3 later, adjust channel 47's width
                                       while algorithm is running

ALG:UPDATE
```

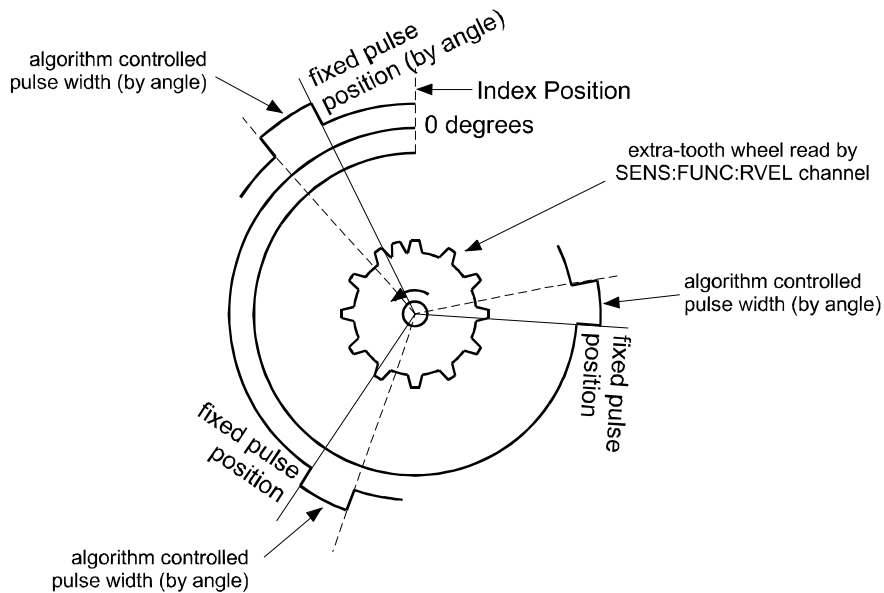


Figure 23. Fixed Position, Variable Width by Angle

**Rotational Pulse Mode:
Variable Pulse Width (by time), Preset Angular Position**

In this mode, the pulse duration (in seconds) is controlled by the algorithm, and the angular position is preset before INIT. See Figure 24. Use the following command sequence:

SOURce:FUNCtion:RPULse (@<ref_channel>),(@<ch_list>) to select the channels that will output angular positioned pulses, and to specify the reference channel.

SOURce:RPULse:POSition[:ANGLE] <degrees>,(@<ch_list>) to preset the angular pulse position in degrees. The algorithm will control the pulse duration.

SOUR:RPULse:VARType TIME,(@<ch_list>) to set the type of value that will vary with algorithm control (in this case pulse duration in seconds).

Example of variable width (by time), preset position:

Set up channel 40 as the reference channel, and channels 45 through 47 to output variable width pulses:

```
*RST
SENS:FUNC:RVEL 12,MISS,(@140)      sense rvel for reference channel
SOUR:FUNC:RPULSE (@140),(@145:147) 3 rotational pulse output chans
SOUR:RPULSE:POS:ANGL 20,(@145)     preset channel 45 pulse position
                                     to 20 degrees
```

SOUR:RPULSE:POS:ANGL 140,(@146)	<i>preset channel 46 pulse position to 140 degrees</i>
SOUR:RPULSE:POS:ANGL 260,(@147)	<i>preset channel 47 pulse position to 260 degrees</i>
SOUR:RPULSE:VART TIME,(@145:147)	<i>algorithm will control pulse duration by TIME</i>
<i>Algorithm outputs pulses on all three channels with preset duration.</i>	
ALG:DEF 'ALG1','static float Width1, Width2, Width3;O145 = Width1; O146 = Width2; O147 = Width3;'	
ALG:SCALAR'ALG1','Width1',.005	<i>preset ch 45 pulse width to 5ms</i>
ALG:SCALAR'ALG1','Width2',.010	<i>preset ch 46 pulse width to 10ms</i>
ALG:SCALAR'ALG1','Width3',.015	<i>preset ch 47 pulse width to 15ms</i>
ALG:UPDATE	
INIT	<i>start algorithm execution</i>
•	
•	<i>calculate NewWidth(n)</i>
•	
ALG:SCALAR 'ALG1','Width1',NewWidth1	<i>later, adjust channel 45's width while algorithm is running</i>
ALG:SCALAR 'ALG1','Width2',NewWidth2	<i>later, adjust channel 46's width while algorithm is running</i>
ALG:SCALAR 'ALG1','Width3',NewWidth3	<i>later, adjust channel 47's width while algorithm is running</i>
ALG:UPDATE	<i>values in update queue sent to variables</i>

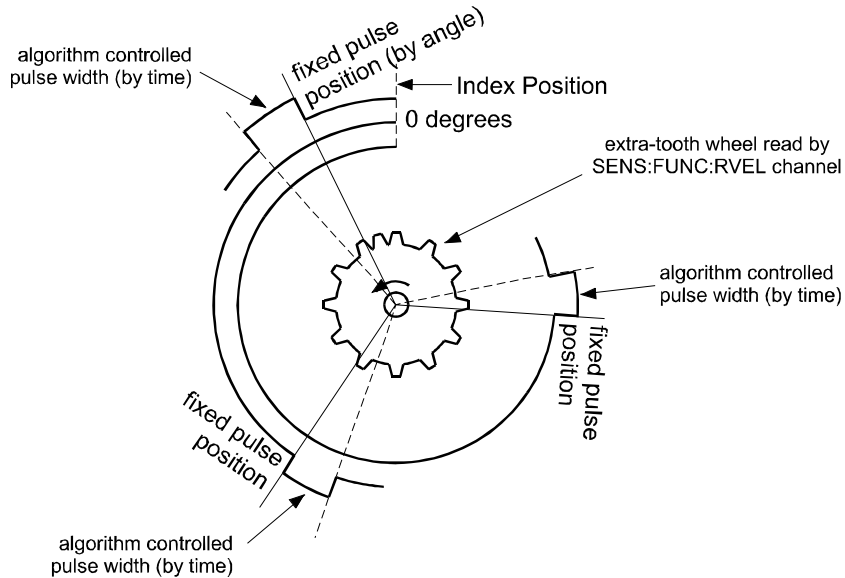


Figure 24. Fixed Position, Variable Width by Time

Stepper Motor Control

Use the command

SOURCE:FUNCTION:STEPper <preset_pos>,<mode>,<max_vel>,<min_vel>,<@ch_list>

to control stepper motors. The E1538 can operate 2 or 4 phase motors in full, and half step mode. Position values are sent from the algorithm to the first channel of a 2 or 4 channel "motor group". The algorithm reads the current position from the second channel of the group.

Four-phase stepper motors that require less than 100mA phase current can be directly driven by the SCP. See Figure 29 for a connection diagram that also shows the required user-supplied output protection components.

- <preset_pos> defines the position count at algorithm start-up.
- <mode> is used to select the stepping mode. the allowable values are:

Table 1. Stepping <mode> values

<mode> string	Stepping Mode	Speed	Channel
MFSFC2	Full	Full	2
MFSFC4	Full	Full	4
MFSHC2	Full	Half	2
MFSHC4	Full	Half	4
MHSFC2	Half	Full	4

- <min_vel> is specified in steps per second and is the beginning step rate at the start of the 14 or 38 step ramp-up to <max_vel>.

<max_vel> is specified in steps per second and is the maximum step rate that will be sent to the motor after ramp-up is complete.

Figure 25 shows the relationship between these parameters. A related error message: 3120, "Minimum velocity parameter must not exceed maximum velocity parameter."

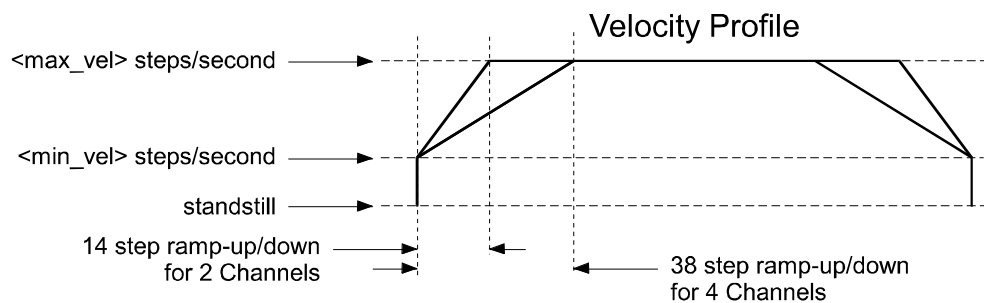


Figure 25. Relationship of min_vel, and max_vel

- <ch_list> specifies the channels that will control stepper motors. A

motor phase channel group can not be split across SCPs.

The algorithm sends new position values to the first channel in a motor-group. The algorithm reads the current position value from the second channel in the motor-group.

Example of full step, full speed, 4 phase stepper motor operation:

*RST

*preset count to 0, full step, half speed, 4 channel, min speed 64s/s,
max speed 256s/s (in half speed mode, actual speed=half specified speed)*

SOUR:FUNC:STEP 0,MFSFC4,128,512,(@144:147)

SENS:FUNC:VOLT (@100) *channel 0 reads voltage*

Algorithm reads voltage at channel 00, multiplies it by 100 to derive the value to send to the motor. Only when the expected motor position (previously sent to ch44) and the actual motor position (read from ch45) agree, is a new motor position is sent to ch44.

ALG:DEF 'ALG1','static float MotorDrive;MotorDrive = (I100 * 100) - 512;
/*5.12V =0 MtrDrv */

If (!(O144 - I145)) O144 = MotorDrive;'

INIT

start algorithm

The following figures show the step waveforms for the five built-in step modes.

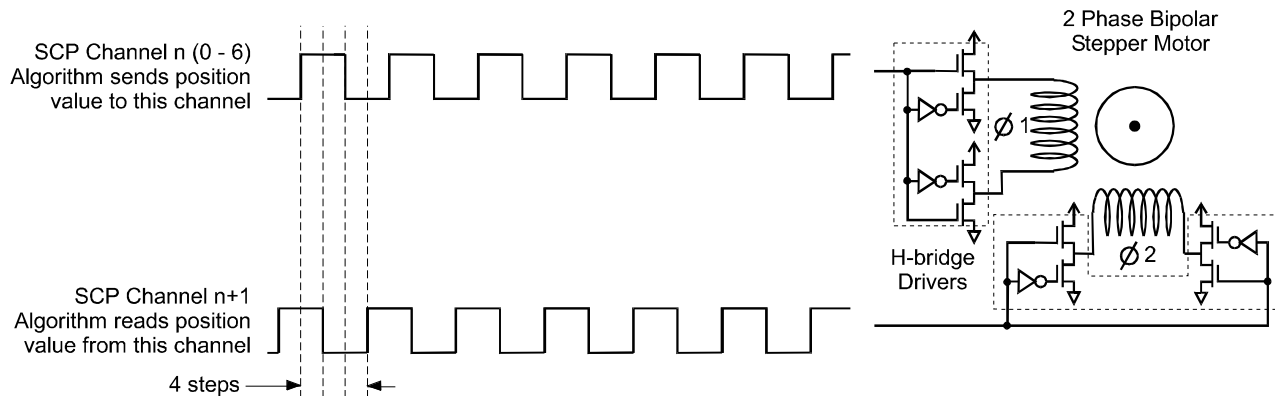


Figure 26. Full Step Mode, Full and Half Speed, 2-Channel

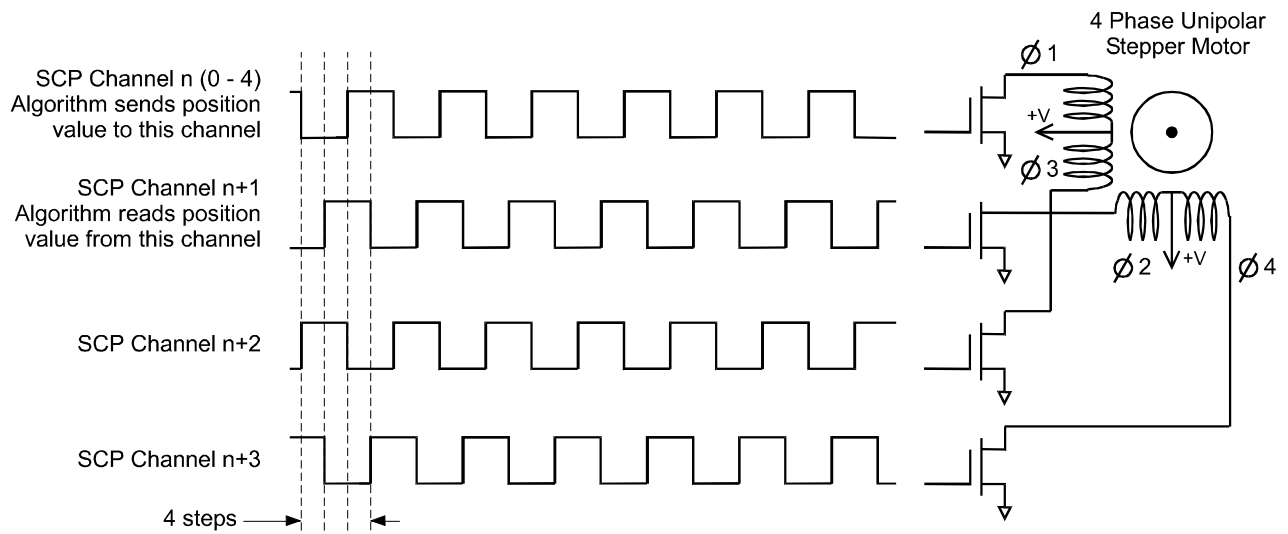


Figure 27. Full Step Mode, Full and Half Speed, 4-Channel

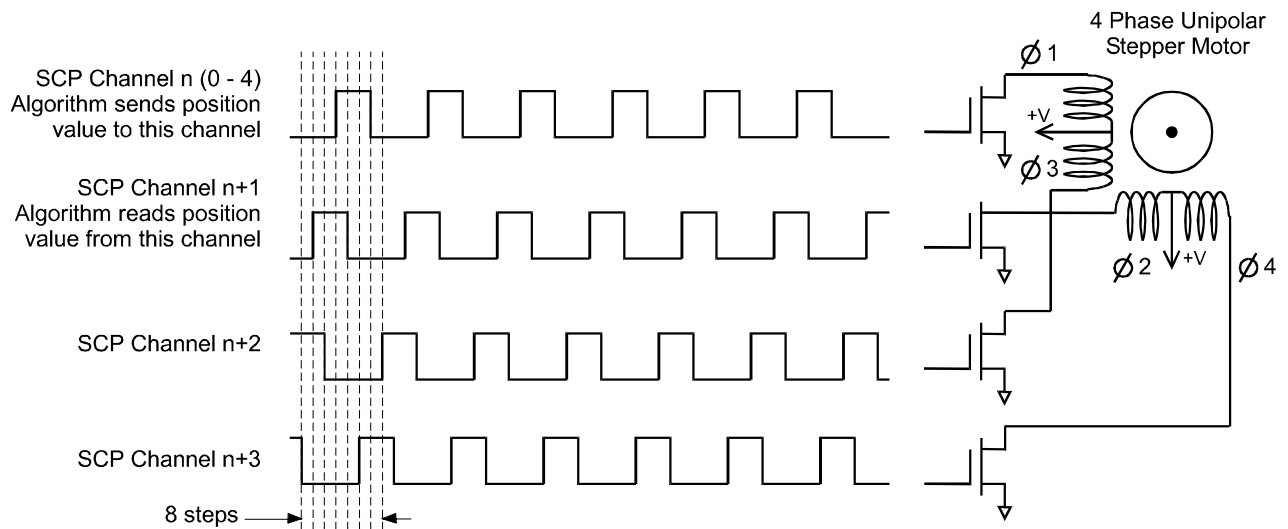


Figure 28. Half Step Mode, Full Speed, 4-Channel

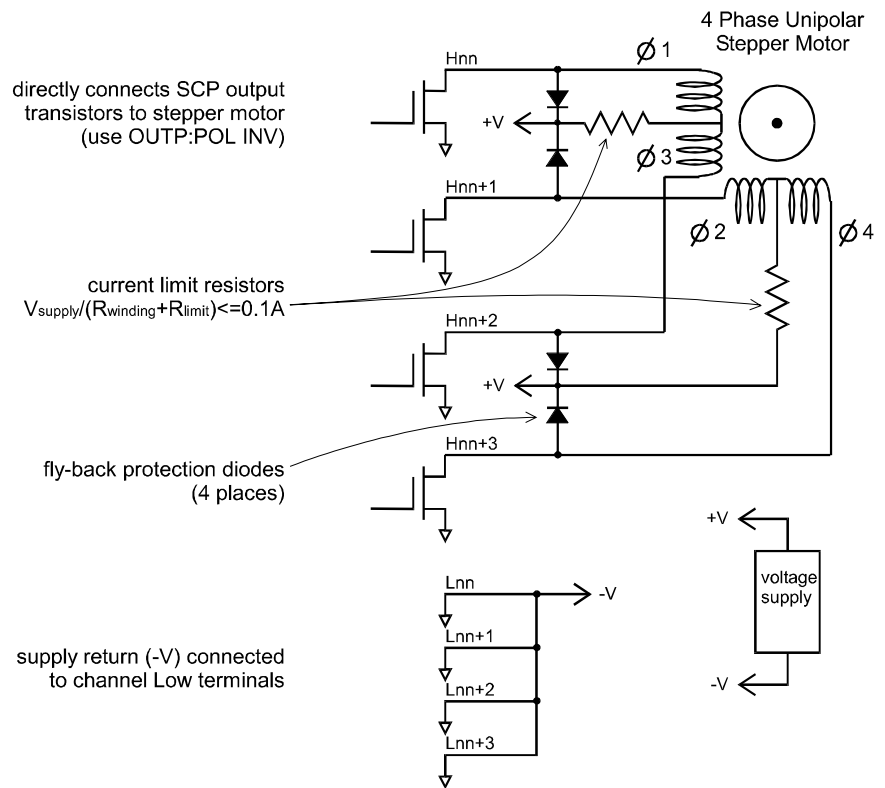


Figure 29. Directly Driving 4-Phase Stepper Motors

***RST and *TST (important!)**

The *RST and power-on condition (true also after *TST) for output-configured channels will output a logical one (open-drain output off). You should keep this behavior in mind when applying the E1415 to your system. It is best to have your system's digital inputs use a high (one) as their safe state.

SCPI Command Reference

Most of the E1538's commands were available since its introduction. A small number of commands are only available with E1538's built after February 1998. The following table indicates these new commands with an X in the "for newer units only" column. See "Identifying the Plug-on" on page 5

Table 2.

Command Syntax	For newer units only	Page Discussed
INPut:POLarity NORM INV,(@<ch_list>)		44
INPut:POLarity? (@<ch_list>)		44
INPut:THReshold[:LEVel] <level>,(@<ch_list>)		45
INPut:THReshold[:LEVel]? (@<channel>)		45
[SENSe:]FUNctIon:CONDition (@<ch_list>)		50
[SENSe:]FUNctIon:FREQuency (@<ch_list>)		51
SENSe:FREQuency:APERture <time>,(@<ch_list>)		47
SENSe:FREQuency:APERture? (@<channel>)		48
SENSe:FREQuency:LIMit:LOWer <freq_limit>,(@<ch_list>)	X	48
SENSe:FREQuency:LIMit:LOWer? (@<channel>)	X	50
[SENSe:]FUNctIon:PERiod (@<ch_list>)	X	51
SENSe:PERiod:APERture <time>,(@<ch_list>)	X	56
SENSe:PERiod:APERture? (@<channel>)	X	57
SENSe:PERiod:LIMit:UPPer <per_limit>,(@<ch_list>)	X	57
SENSe:PERiod:LIMit:UPPer? (@<channel>)	X	59
SENSe:PERiod:MODE APERture NPERiods,(@<ch_list>)	X	59
SENSe:PERiod:MODE? (@<channel>)	X	60
SENSe:PERiod:NPERiods <count>,(@<ch_list>)	X	60
SENSe:PERiod:NPERiods? (@<channel>)	X	61
SENSe:PERiod:RANGe[:UPPer] <count>,(@<ch_list>)	X	61
SENSe:PERiod:RANGe[:UPPer]? (@<channel>)	X	62
[SENSe:]FUNctIon:PWIDth <avg_count>,(@<ch_list>)		52
[SENSe:]FUNctIon:QUADrature [<preset_count>],(@<ch_list>)		52
[SENSe:]FUNctIon:RVELocity <n_teeth>,<index_type>,(@<ch_list>)		53
[SENSe:]FUNctIon:TOTalize (@<ch_list>)		55
[SENSe:]TOTalize:RESet:MODE INIT TRIG,(@<ch_list>)		62

Table 2.

Command Syntax	For newer units only	Page Discussed
[SENSe:]TOTAlize:RESet:MODE? (@<channel>)		63
OUTPut:POLarity NORM INV,(@<ch_list>)		46
OUTPut:POLarity? (@<channel>)		46
SOURce:FUNCTion:RPULse (@<ref_chan>),(@<ch_list>)		65
SOURce:RPULse:POSition[:ANGLE] <degrees>,(@<ch_list>)		72
SOURce:RPULse:POSition[:ANGLE]? (@<channel>)		73
SOURce:RPULse:WIDTh[:ANGLE] <degrees>,(@<ch_list>)		74
SOURce:RPULse:WIDTh[:ANGLE]? (@<channel>)		75
SOURce:RPULse:WIDTh:TIME <seconds>,(@<ch_list>)		75
SOURce:RPULse:WIDTh:TIME? (@<channel>)		76
SOURce:RPULse:VARType ANGLE TIME,(@<ch_list>)		73
SOURce:RPULse:VARType? (@<channel>)		74
SOURce:FUNCTion[:SHAPE]:CONDition (@<ch_list>)		66
SOURce:FUNCTion[:SHAPE]:PULSe (@<ch_list>)		66
SOURce:FM[:STATe] ON OFF,(@<ch_list>)		63
SOURce:FM[:STATe]? (@<channel>)		64
SOURce:PULSe:WIDTh <width>,(@<ch_list>)		71
SOURce:PULSe:WIDTh? (@<channel>)		72
SOURce:PULM[:STATe] ON OFF,(@<ch_list>)		69
SOURce:PULM[:STATe]? (@<channel>)		70
SOURce:PULSe:PERiod <period>,(@<ch_list>)		70
SOURce:PULSe:PERiod? (@<channel>)		71
SOURce:FUNCTion[:SHAPE]:SQUare (@<ch_list>)		67
SOURce:FM[:STATe] ON OFF,(@<ch_list>)		63
SOURce:FM[:STATe]? (@<channel>)		64
SOURce:FUNCTion:STEPper <preset_pos>,<mode>,<min_vel>,<max_vel>,(@<ch_list>)		67
SYSTem:CTYPe? (@<channel>)		12

INPut:POLarity

INPut:POLarity *<mode>*,*<ch_list>* sets logical input polarity on a digital SCP channel.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>mode</i>	discrete (string)	NORMal INVerted	none
<i>ch_list</i>	string	100 - 163	none

Comments

- If the channels specified are on an SCP that doesn't support this function, an error will be generated. See your SCP's User's Manual to determine its capabilities.
- **Related Commands:** for output sense; SOURce:PULSe:POLarity
- ***RST Condition:** INP:POL NORM for all digital SCP channels.
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage

INP:POL INV,(@140:143)

invert first 4 channels on SCP at SCP position 5. Channels 40 through 43

INPut:POLarity?

INPut:POLarity? *<channel>* returns the logical input polarity on a digital SCP channel.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- *<channel>* must specify a single channel.
- If the channel specified is on an SCP that doesn't support this function, an error will be generated. See your SCP's User's Manual to determine its capabilities.
- **Returned Value:** returns "NORM" or "INV". The type is **string**.
- **Send with VXIplug&play Function:** hpe14XX_cmdString_Q(...)

INPut:THReshold[:LEVel]

INPut:THReshold[:LEVel] <level>,@<ch_list> •allows programmatically setting the input threshold level for each input configured channel.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>level</i>	numeric (float32)	-46 VDC to +46 VDC	none
<i>ch_list</i>	string	100 - 163	none

Comments

- <level> can be set to a resolution of .375V
- . While input polarity is set to NORMAL, an input level higher than the threshold level is considered a logic one, and an input level lower than the threshold level is considered a logic zero. If input polarity is set to INVERTed, an input level higher than the threshold level is considered a logic zero and an input level lower than the threshold level is considered a logic one.

Note

The value sent for <level> will be rounded to the nearest multiple of 0.375 Volts. For instance, 5 would be 4.875, 10 would be 10.125, 9.5 would be 9.375, and 15 would be 15. The INP:THR:LEV? query will return the actual setting.

- **Related Commands:** INPut:POLarity, INP:THR:LEV?
- ***RST Condition:** INP:THR:LEV = 1.875
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

INPut:THReshold[:LEVel]?

INPut:THReshold[:LEVel]? (@<channel>) returns the threshold level set for <channel>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- <channel> must specify a single channel.

Note Because the E1538 rounds *<level>* to the nearest multiple of 0.375, the returned value can be different from the value sent.

- INP:THR:LEV? returns a numeric value between -46 and +46. The C-SCPI type is **float32**.
- **Related Commands:** INPut:POLarity, INP:THR:LEVel
- ***RST Condition:** INP:THR:LEV = 1.875
- **Send with VXIplug&play Function:** hpe14XX_cmdReal64_Q(...)

Usage To query the threshold level on the second channel at SCP position 4 send:
INP:THR:LEV? (@133) *query 2nd chan on SCP pos. 4*
enter statement here

OUTPut:POLarity

OUTPut:POLarity <select>,@<ch_list> sets the polarity on digital output channels in *<ch_list>*.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>select</i>	discrete (string)	NORMal INVerted	none
<i>ch_list</i>	string	100 - 163	none

- Comments**
- If the channels specified do not support this function, an error will be generated.
 - Related Commands: INPut:POLarity, OUTPut:POLarity?
 - ***RST Condition:** OUTP:POL NORM for all digital channels
 - **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage OUTP:POL INV,@144) *invert output logic sense on channel 44*

OUTPut:POLarity?

OUTPut:POLarity? (@<channel>) returns the polarity on the digital output channel in *<channel>*.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- *Channel* must specify a single channel
- **Returned Value:** returns one of NORM or INV. The type is **string**.
- **Send with VXIplug&play Function:** hpe14XX_cmdString_Q(...)

[SENSe:]FREQuency:APERture

[SENSe:]FREQuency:APERture <*aperture*>,<*ch_list*> sets the time allowed to determine signal frequency and return a reading to the algorithm. When APERture is large enough to contain more than one signal period, the SCP measures and averages the number of signal periods that will fit within this APERture time. If the specified APERture is less than the input signal period, the SCP stretches the aperture in order to measure at least one signal period. This is known as Adaptive Aperture.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>aperture</i>	numeric (float32)	.001 to 1 (.001 resolution)	seconds
<i>ch_list</i>	string	100 - 163	none

Comments

- For APERture to effect the measurement, SENS:FREQ:MODE must be set to APERture.
- If the channels specified are on an SCP that doesn't support this function, an error will be generated. See your SCP's User's Manual for its capabilities.
- Related Commands: SENS:FREQ:MODE, SENS:FREQ:NPERiods, SENS:FREQ:LIM:LOWer, SENS:FUNC:FREQ
- ***RST Condition:** .001 sec
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage

SENS:FREQ:APER .01,(@144)

set channel 44 aperture to 10msec

[SENSe:]FREQuency:APERture?

[SENSe:]FREQuency:APERture? <channel> returns the currently set APERture time.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- If the channel specified is on an SCP that doesn't support this function, an error will be generated. See your SCP's User's Manual for its capabilities.
- **Related Commands:** SENS:FREQ:APER
- **Returned Value:** returns numeric aperture in seconds, The type is float32.
- **Send with VXIplug&play Function:** hpe14XX_cmdReal64_Q(...)

SENSe:FREQuency:LIMit:LOWer

[SENSe:]FREQuency:LIMit:LOWer <freq_limit>,<ch_list> allows you to specify a frequency lower limit beyond which the E1538A will stop waiting for a signal transition and will return a frequency value of zero. Conceptually, this is an input signal period time-out.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>freq_limit</i>	numeric (float32)	.01667 to 250	Hz
<i>ch_list</i>	string	100 - 163	none

Comments

- <ch_list> must be channels on an E1538A SCP.

Note

Although SENS:FREQ:LIM:LOW can set a "timeout period" as long as 60 seconds (.01667Hz), the lowest frequency measurement supported by the E1538A is 1 Hertz.

- The period associated with the FREQ:LIMit:LOWer frequency is the time the SCP will allow for any single cycle period.

At any time during the frequency measurement, if the signal's period

exceeds the time-out period (i.e. frequency below LIMit:LOWer), then a frequency of 0 Hz will be returned to the E1415/19/22 algorithm.

- Typical use for this command is to allow the user to bound the period of time that is allowed for making a frequency measurement, thus, preventing the SCP from “hanging” during measurement.

In this case, typically, the LIMit:LOWer frequency would be the same as, or slower than the frequency associated w/the APERTure time.

- An unusual (but valid) use is to set the LIMit:LOWer frequency to be greater than the frequency associated with the APERTure period, which can provide a means to abort a frequency measurement if at any point during the measurement, the input waveform frequency is slower than the configured LIMit:LOWer frequency.

Note The lower limit set by SENS:FREQ:LIM:LOW is for a single signal period, not the sum of NPERiods. Unless at least one period of the input signal exceeds the limit value set, then NPERiods will be measured and averaged to return a reading. For instance if;

```
SENS:FREQ:MODE NPERiods,<ch_list>
SENS:FREQ:NPERiods 255
SENS:FREQ:LIMit:LOWer 0.01667 ! 60 second period
...
INIT
```

As long as the input signal frequency is slightly greater than the LIMit:LOWer frequency, then the SCP will not time-out and will take 255 * (~60sec) = ~255 minutes to take a single frequency measurement.

Alternatively, if even one of the input waveforms has a frequency that is lower than the LIMit:LOWer frequency, then 0 Hz would be immediately returned to the E1415/19/22 algorithm.

- **Related Commands:** SENS:FREQ:APER, SENS:FREQ:LOW?
- ***RST Condition:** is “MINimum” frequency (i.e. 0.01667 Hz [period = 60sec]).
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

SENSe:FREQUency:LIMit:LOWer?

[SENSe:]FREQUency:LIMit:LOWer? <channel> returns the lower frequency limit currently set for <channel>

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- <channel> must be a single channel on an E1538A SCP.
- **Returned Value:** .01667 to 250, The type is **float32**.
- **Send with VXIplug&play Function:** hpe14XX_cmdReal64_Q(...)

[SENSe:]FUNCTION:CONDition

[SENSe:]FUNCTION:CONDition <ch_list> sets the SENSE function to input the digital state for channels in <ch_list>. See “Reading Static Digital State” on page 14.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>ch_list</i>	string	100 - 163	none

Comments

- The E1538 SCP senses the single bit digital state on each channel specified by this command.
- If the channels specified are not on a digital SCP, an error will be generated.
- Use the INPut:POLarity command to set input logical sense.
- **Related Commands:** INPut:POLarity
- ***RST Condition:** SENS:FUNC:COND and INP:POL NORM for all digital SCP channels.
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage

See “Reading Static Digital State” on page 14.

To set upper 4-bits of E1538 at SCP position 5 to digital inputs send:

[SENSe:]FUNCTION:FREQUENCY

[SENSe:]FUNCTION:FREQUENCY <*ch_list*> sets the SENSe function to frequency for channels in <*ch_list*>. Also configures the channels specified as digital inputs. See “About Period and Frequency Measurements” on page 16.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>ch_list</i>	string	100 - 163	none

Comments

- If the channels specified are on an SCP that doesn't support this function, an error will be generated. See your SCP's User's Manual for its capabilities.
- **Related commands:** SENS:FREQ:APER, SENS:FREQ:MODE
- ***RST Condition:** SENS:FUNC:COND and INP:POL NORM for all digital SCP channels
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage See “About Period and Frequency Measurements” on page 16.
SENS:FUNC:FREQ (@144)

[SENSe:]FUNCTION:PERIOD

[SENSe:]FUNCTION:PERIOD (@<*ch_list*>) sets the SENSe function to period for channels in <*ch_list*>. Also configures the channels specified as digital inputs. See “About Period and Frequency Measurements” on page 16.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>ch_list</i>	string	100 - 163	none

Comments

- If the channels specified are on an SCP that doesn't support this function, an error will be generated. See your SCP's User's Manual for its capabilities.
- **Related commands:** SENS:PER:APER, SENS:PER:NPER,

SENS:PER:MODE

- ***RST Condition:** SENS:FUNC:COND and INP:POL NORM for all digital SCP channels
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage See “About Period and Frequency Measurements” on page 16.
SENS:FUNC:PER (@144)

[SENSe:]FUNCTION:PWIDth

[SENSe:]FUNCTION:PWIDth <avg_count>,(@<ch_list>) configures channels to measure the input signal pulse width. See “Measure Pulse Width” on page 20.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>avg_count</i>	numeric (uint32)	1 to 255 MINimum MAXimum	seconds
<i>ch_list</i>	string	100 - 163	none

Comments

- <ch_list> must be channels on an E1538A SCP.
- <avg_count> sets the number of pulses to average when forming the pulse duration value. More counts give more accurate readings, but slower response to changing pulse widths.
- ***RST Condition:** SENS:FUNC:COND and INP:POL NORM for all digital SCP channels.
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage See “Measure Pulse Width” on page 20.
SENS:FUNC:PWID 10,(@146,147)

channels 46&47 meas pulse width

[SENSe:]FUNCTION:QUADrature

[SENSe:]FUNCTION:QUADrature [<preset_count>,(@<ch_list>)

configures a pair of E1538 channels to measure quadrature count. See “Sense Quadrature Position” on page 21.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>preset_count</i>	numeric (int32)	0 to 16,777,215	none
<i>ch_list</i>	string	100 - 163	none

Comments

- *<count_preset>* if included, allows presetting the absolute counter associated with the channel pair. All quadrature pairs in *<ch_list>* will be preset to the same value. If not included, the default count at algorithm start will be zero.
- *<ch_list>* must always specify both channels of a pair. More than one pair can be specified. Both channels of any pair must be adjacent. *<ch_list>* can specify channels on more than one E1538. The channel numbers in *<ch_list>* must be in ascending order. The related error messages are:
3115, "Channels specified are not in ascending order."
3116, "Multiple channels specified are not grouped correctly."
3117, "Grouped channels are not adjacent."
3122, "This multiple channel function must not span multiple SCPs."
- The algorithm reads the current count through the low numbered channel. The count is an unsigned 24-bit value ranging from 0 to 16,777,215. The counter can roll over from 16,777,2215 to 0, and roll under from 0 to 16,777,215 is 16,777,215.
- ***RST Condition:** SENS:FUNC:COND and INP:POL NORM for all digital SCP channels.
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage

See "Sense Quadrature Position" on page 21.
SENS:FUNC:QUAD 8192,(@142,143)

pair 42&43 will return quad count (on ch 142), count preset to 8192

[SENSe:]FUNCTION:RVELocity

[SENSe:]FUNCTION:RVELocity <n_teeth>,<index_type>,(@<ch_list>)

configures the first channel on E1538s to measure the rotational velocity of a toothed wheel sensor. The E1538 measures tooth-to-tooth period and converts it into units of revolutions per second (RPS). See "Sense Rotational Velocity" on page 22.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>N_teeth</i>	numeric (int32)	3 to 255	none
<i>index_type</i>	string	MISSing EXTRA	none
<i>ch_list</i>	string	100 - 163	none

Comments

- This function can only be linked to the E1538's first channel. The function works for wheels that have either a missing, or an extra tooth to mark their index position. Figure 14 on page 23 shows a wheel sensed with a variable reluctance sensor (using the VRS input option), but any wheel sensing method is applicable as long as it provides a digital output to the RVEL channel.
- The value read by the algorithm can range from $\frac{1}{n_{teeth}}$ RPS to $\frac{100,000}{n_{teeth}}$ RPS.
- As well as sensing rotational velocity, SENS:FUNC:RVEL provides the reference position to the SOUR:FUNC:RPULse function that generates angular positioned pulses. See page 30 for more information on RPULse.
- *<n_teeth>* is the number of teeth that the wheel would have if it didn't have missing or extra teeth. For example, we would set *<n_teeth>* to 12 for the wheel shown in Figure 14 on page 23, even though with the missing tooth, there are only 11.
- *<index_type>* can be either of the strings "MISSing", or "EXTRA"
- *<ch_list>* must be the first channel on the SCP, but can contain more than one channel provided that each channel is on a separate E1538. See following note. The related Error Messages are:
3110, "Channel specified is invalid for RVELocity function.

Note

Only one channel on any E1538 SCP can be assigned to the SENS:FUNC:RVEL function, and it must be the first channel on the SCP.

- ***RST Condition:** SENS:FUNC:COND and INP:POL NORM for all digital SCP channels.
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage

See "Sense Rotational Velocity" on page 22.
SENSE:FUNC:RVEL 12,MISSING,(@140)

12 toothed wheel with one missing, from channel 40

SENSe:]FUNcTion:TOTalize

[SENSe:]FUNcTion:TOTalize <ch_list> sets the SENSe function to TOTalize for channels in <ch_list>. See “Totalize Positive or Negative Edge State Changes” on page 15.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
ch_list	string	100 - 163	none

Comments

- The totalize function counts rising edges of digital transitions at Frequency/Totalize SCP channels. The counter is 24 bits wide and can count up to 16,777,215.
- The SENS:TOT:RESET:MODE command controls which events will reset the counter.
- If the channels specified are not on a Frequency/Totalize SCP, an error will be generated.
- **Related Commands:** SENS:TOT:RESET:MODE, INPUT:POLARITY
- ***RST Condition:** SENS:FUNC:COND and INP:POL NORM for all digital SCP channels.
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage

See “Totalize Positive or Negative Edge State Changes” on page 15.
SENSe:]FUNcTion:TOT (@134)

channel 34 is a totalizer

[SENSe:]PERiod:APERture

[SENSe:]PERiod:APERture <aperture>,<ch_list> sets the time allowed to determine signal period and return a reading to the algorithm. When APERTure is large enough to contain more than one signal period, the SCP measures and averages the number of signal periods that will fit within this APERTure time. If the specified APERTure is less than the input signal period, the SCP stretches the aperture in order to measure at least one signal period. This is known as Adaptive Aperture.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>aperture</i>	numeric (float32)	see text	seconds
<i>ch_list</i>	string	100 - 163	none

Comments

- The range for <aperture> is dependent on SENS:PER:RANGE:
 - When SENS:PER:RANGE is 1, <aperture> can range from .001 to 1 second.
 - When SENS:PER:RANGE is 4, <aperture> can range from .004 to 4 seconds.
- For APERTure to effect the measurement, SENS:PER:MODE must be set to APERTure.
- If the channels specified are on an SCP that doesn't support this function, an error will be generated. See your SCP's User's Manual for its capabilities.
- Related Commands: SENS:PER:MODE, SENS:PER:NPERiods, SENS:PER:LIM:UPPer, SENS:PER:RANGE, SENS:FUNC:PER
- ***RST Condition:** .001 sec
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage SENS:PER:APER .01,(@144)

set channel 44 aperture to 10msec

[SENSe:]PERiod:APERture?

[SENSe:]PERiod:APERture? *<channel>* returns the currently set APERture time.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- If the channel specified is on an SCP that doesn't support this function, an error will be generated. See your SCP's User's Manual for its capabilities.
- Related Commands: SENS:PER:APER
- **Returned Value:** returns numeric aperture in seconds, The type is **float32**.
- **Send with VXIplug&play Function:** hpe14XX_cmdReal64_Q(...)

SENSe:PERiod:LIMit:UPPer

[SENSe:]PERiod:LIMit:UPPer *<per_limit>*,*<ch_list>* allows you to specify a period upper limit beyond which the E1538A will stop waiting for a signal transition and will return a period value of zero. Conceptually, this is an input signal period time-out.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>per_limit</i>	numeric (float32)	.004 to 60	Sec
<i>ch_list</i>	string	100 - 163	none

Comments

- *<ch_list>* must be channels on an E1538A SCP.

Note

Although SENS:PER:LIM:UPP can set a "timeout period" as long as 60 seconds, the longest supported periods are 1 second or 4 seconds depending on the setting of SENSe:PERiod:RANGe.

- ...:LIMit:UPPer period is the maximum time the SCP will allow for any cycle period.

At any time during the period measurement, if the cycle period exceeds

the LIMit:UPPer period, then a period of 0 sec will be returned to the E1415/19/22 algorithm.

(Note: A period value of 0 sec is used as a special token that must be tested for in the user's provided E1415/19/22 algorithm.)

- Typical use for this command is to allow the user to bound the period of time that is allowed for making a period measurement, thus, preventing the SCP from “hanging” during measurement. In this case, typically, the upper period limit would be the same as, or longer than the APERture time.
- An unusual (but valid) use is to set the LIMit:UPPer period to be less than the APERture period, which can provide a means to abort a period measurement if at any point during the measurement, the input waveform period is longer than the configured LIMit:UPPer.

Note The upper limit set by SENS:PER:LIM:UPPer is for a single signal period, not the sum of NPERiods. Unless at least one period of the input signal exceeds the limit value set, then NPERiods will be measured and averaged to return a reading. For instance if;

```
SENS:PER:MODE NPERiods,<ch_list>
SENS:PER:NPERiods 255
SENS:PER:LIMit:UPPer 60
...
INIT
```

When the input waveform period is slightly less than the upper period limit, then the SCP will not time-out and will take
 $255 * (\sim 60\text{sec}) = \sim 255 \text{ minutes}$ to take a single period measurement.

Alternatively, if even one of the input waveforms has a period that exceeds the LIMit:UPPer period, then 0 sec would be immediately returned to the E1415/19/22 algorithm.

- ***RST Condition:** is “MAXimum” (i.e. 60.0sec).
- **.Send with VXIplug&play Function:** hpe14XX_cmd(...)

[SENSe:]PERiod:LIMit:UPPer?

[SENSe:]PERiod:LIMit:UPPer? <channel> returns the upper period limit currently set for <channel>

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- <channel> must be a single channel on an E1538A SCP.
- **RETURNS:** .004 to 60, Type is **float32**.

SENSe:PERiod:MODE

SENSe:PERiod:MODE <mode>,(@<ch_list>) selects the measurement interval mode the SCP will use to measure the signal period. This can be set as a fixed amount of time (APERTure), or a fixed number of signal periods (NPERiods).

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>mode</i>	discrete (string)	APERTure NPERiods	none
<i>ch_list</i>	string	100 - 163	none

Comments

- The SENS:PER:APERTure command sets the aperture value. The SENS:PER:NPERiods command sets the nperiods value.
- <ch_list> must be channels on E1538A SCPs.
- **Related Commands:** SENS:PER:APERTure, SENS:PER:NPERiods, SENS:FUNC:PER, SENS:PER:LIM:UPPer
- ***RST Condition:** is “APERTure”.
- **.Send with VXIplug&play Function:** hpe14XX_cmd(...)

SENSe:PERiod:MODE?

SENSe:PERiod:MODE? (@<channel>) returns the measurement interval mode currently set for period measurement.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- <channel> must be a single channel on an E1538A SCP.
- **RETURNS:** String value "APER" or "NPER", Type is **String**
- **.Send with VXIplug&play Function:** hpe14XX_cmdString_Q(...)

SENSe:PERiod:NPERiods

SENSe:PERiod:NPERiods <count>,@<ch_list> sets the number of signal periods to measure and average in order to compute the input signal period.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>count</i>	numeric (uint32)	1 to 255 MINimum MAXimum	seconds
<i>ch_list</i>	string	100 - 163	none

Comments

- <ch_list> must be channels on an E1538A SCP.
- This feature is only available when the NPERiods period mode is in use:
SENSe:PERiod:MODE NPERiods,<ch_list>
- **Related Commands:** SENSe:PERiod:MODE
- **RESET Condition:** is "MINimum" (i.e. 1).
- **.Send with VXIplug&play Function:** hpe14XX_cmd(...)

SENSe:PERiod:NPERiods?

SENSe:PERiod:NPERiods? (@<channel>) returns the number of signal periods the SCP will measure and average to calculate the signal period.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- <channel> must be single channel on an E1538A SCP.
- **RETURNS:** uint32 value which is the current period count configured.
- **Related Commands:** SENSe:PERiod:NPERiods
- ***RST condition:** SENSe:PERiod:NPERiods = 1
- **.Send with VXIplug&play Function:** hpe14XX_cmdInt32_Q(...)

[SENSe:]PERiod:RANGe[:UPPer]

[SENSe:]PERiod:RANGe[:UPPer] <range>,<ch_list> can extend the range of period measurement from the default 1 second maximum to a 4 second maximum.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>range</i>	numeric (float32)	1 4 MAXimum MINimum	Sec
<i>ch_list</i>	string	100 - 163	none

Comments

- <ch_list> must be channels on an E1538A SCP.
- Range 1.0 sec provides period measurement in the range:
10usec - 1sec

Range 4.0 sec provides period measurement in the range:
40usec - 4sec
- MINimum = 1.0, MAXimum = 4.0

Note 1538As have a possible settings conflict:

If SENSE:PERiod:RANGe = 1.0
 then settings conflict if 1.0sec < APERTure < 4.0sec.
 If SENSE:PERiod:RANGe = 4.0
 then settings conflict if 0.01sec <= APERTure < 0.04sec

The E1415/19/22 driver will report these Settings Conflicts at INITiate time with the following error message:
 3129, Incompatible Aperture and Range values,SCP[x]

- **Related Commands:** SENSE:PERiod:NPERiods
- ***RST Condition:** is 1
- **.Send with VXIplug&play Function:** hpe14XX_cmd(...)

[SENSE:]PERiod:RANGe[:UPPer]?

[SENSE:]PERiod:RANGe[:UPPer]? <channel> returns the current setting of upper period limit.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- <channel> must be single channel on an E1538A SCP.
- RETURNS: float32, configured upper period time range for given ch. Response is in seconds. Returned value either 1 or 4
- **.Send with VXIplug&play Function:** hpe14XX_cmdReal64_Q(...)

[SENSE:]TOTAlize:RESet:MODE

[SENSE:]TOTAlize:RESet:MODE <select>,<ch_list> sets the mode for resetting totalizer channels in <ch_list>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>select</i>	discrete (string)	INIT TRIGger	seconds
<i>ch_list</i>	string	100 - 163	none

Comments

- In the INIT mode the total is reset only when the INITiate command is executed. In the TRIGger mode the total is reset every time a new scan

is triggered.

- If the channels specified are not on a Frequency/Totalize SCP, an error will be generated.
- **Related Commands:** SENS:FUNC:TOT, INPUT:POLARITY
- ***RST Condition:** SENS:TOT:RESET:MODE INIT
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage SENS:TOT:RESET:MODE TRIG,(@134) *totalizer at channel 34 resets at each trigger event*

[SENSe:]TOTAlize:RESet:MODE?

[SENSe:]TOTAlize:RESet:MODE? <channel> returns the reset mode for the totalizer channel in <channel>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- *Channel* must specify a single channel.
- If the channel specified is not on a frequency/totalize SCP, an error will be generated.
- **Returned Value:** returns INIT or TRIG. The type is **string**.
- **Send with VXIplug&play Function:** hpe14XX_cmdString_Q(...)

SOURce:FM[::STATe]

SOURce:FM[::STATe] <enable>,(@<ch_list>) enables the Frequency Modulated mode for a PULSe channel.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>enable</i>	boolean (uint16)	1 0 ON OFF	none
<i>ch_list</i>	string	100 - 163	none

Comments

- This command is coupled with the SOURce:PULM:STATE command. If the FM state is ON then the PULM state is OFF. If the PULM state

is ON then the FM state is OFF. If both the FM and the PULM states are OFF then the PULSe channel is in the single pulse mode.

- If the channels specified are not on a Frequency/Totalize SCP, an error will be generated.
- Use SOURce:FUNCtion[:SHAPe]:SQUare to set FM pulse train to 50% duty cycle. Use SOURce:PULSe:PERiod to set the period
- ***RST Condition:** SOUR:FM:STATE OFF, SOUR:PULM:STATE OFF, SENS:FUNC:COND and INP:POL for all digital SCP channels
- **Related Commands:** SOUR:PULM[:STATe], SOUR:PULS:POLarity, SOUR:PULS:PERiod, SOUR:FUNC[:SHAPe]:SQUare
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage The variable frequency control for this channel is provided by the algorithm language. When the algorithm executes an assignment statement to this channel, the value assigned will be the frequency setting. For example:

O143 = 2000 /* set channel 43 to 2KHz */

SOURce:FM:STATe?

SOURce:FM:STATe? (@<channel>) returns the frequency modulated mode state for a PULSe channel.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- *Channel* must specify a single channel.
- If the channel specified is not on a Frequency/Totalize SCP, an error will be generated.
- **Returned Value:** returns 1 (ON) or 0 (OFF). The type is **uint16**.
- **Send with VXIplug&play Function:** hpe14XX_cmdInt16_Q(...)

SOURce:FUNction:RPULse

SOURce:FUNction:RPULse (@<ref_channel>),(@<ch_list>) links channels in <ch_list> to the rotational pulse function. The channel in <ref_channel> will be linked to the SENS:FUNC:RVEL function to provide the rotational reference information to SOUR:FUNC:RPUL.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>ref_channel</i>	string	100, 108, 116, 124, 132, 140, 148, 156	none
<i>ch_list</i>	string	100 - 163	none

Comments

- <ref_channel> must be a single channel and must be the first channel on the SCP. The channel specified in <ref_channel> must be linked to the SENS:FUNC:RVEL function before the INIT command is received. See page 22 for more on RVEL. The related error messages are:
3111, "multiple channels are specified in reference channel list."
3112, "Channel specified is invalid for RPULse reference channel."
3119, "RPULse reference channel must be defined as RVELocity type."
- Channels in <ch_list> must be higher numbered and on the same SCP as the channel specified in <ref_channel>. The related error messages are:
3113, "Channel specified is not in same SCP as reference channel."
3114, "First channel in SCP can not be used in RPULse output channel list."
3118, "Incomplete setup information for RPULse function. "

Notes

1. There must be one (and only one) channel on the same SCP that is set to SENSE:FUNction:RVELocity. This sense channel provides the rotational velocity and index reference that the SCP uses to position the output pulses at a desired rotational angle. This is the <ref_channel> seen above.
2. The lower velocity limit for RPULse is 108 teeth per Second (TPS) for extra-tooth wheels, and 384TPS for missing-tooth wheels. For example, a 60 tooth wheel would need to rotate at a minimum of 108RPM if it had an extra tooth, but at 384RPM minimum with a missing tooth.
3. Long duration pulses that begin and end within a wheel's missing tooth area can exhibit significant jitter. Use an extra tooth wheel for

these applications. See Figure 30 .

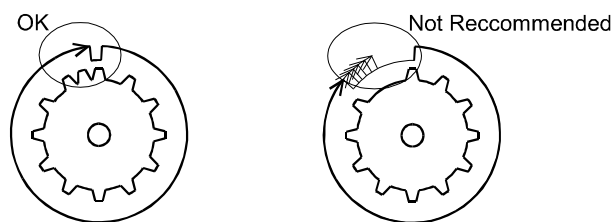


Figure 30. For Long Pulses Use Extra Tooth Wheel

- Send with VXIplug&play Function: hpe14XX_cmd(...)

Usage SENSE:FUNC:RPULSE (@108),(@114,115) *reference chan is 108, pulse output on channels 114 and 115*

SOURce:FUNCTION[:SHAPE]:CONDition

SOURce:FUNCTION[:SHAPE]:CONDition (@<ch_list>) sets the SOURce function to output digital patterns to bits in <ch_list>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
ch_list	string	100 - 163	none

Comments

- The E1533 SCP sources 8 digital bits on the channel specified by this command. The E1534 SCP can source 1 digital bit on each of the the channels specified by this command.
- Send with VXIplug&play Function: hpe14XX_cmd(...)

SOURce:FUNCTION[:SHAPE]:PULSe

SOURce:FUNCTION[:SHAPE]:PULSe (@<ch_list>) sets the SOURce function to PULSe for the channels in <ch_list>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
ch_list	string	100 - 163	none

Comments

- This PULSe channel function is further defined by the SOURce:FM:STATe and SOURce:PULM:STATe commands. If the FM state is enabled then the frequency modulated mode is active. If the

PULM state is enabled then the pulse width modulated mode is active. If both the FM and the PULM states are disabled then the PULSe channel is in the single pulse mode.

- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

SOURce:FUNCtion[:SHAPe]:SQUare

SOURce:FUNCtion[:SHAPe]:SQUare (@<ch_list>) sets the SOURce function to output a square wave (50% duty cycle) on the channels in <ch_list>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>ch_list</i>	string	100 - 163	none

Comments

- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage

The frequency control for these channels is provided by the algorithm language function:.

O143 = 2000 /* set channel 43 to 2KHz */

SOURce:FUNCtion:STEPper

SOURce:FUNCtion:STEPper <preset_pos>,<mode>,<max_vel>,<min_vel>,(@<ch_list>) controls stepper motors. The E1538 can operate 2 or 4 phase motors in full, and half step mode. Position values are sent from the algorithm to the first channel of a 2 or 4 channel "motor group". The algorithm reads the current position from the second channel of the group. See "Stepper Motor Control" on page 37.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>preset_pos</i>	discrete (string)	see text	none
<i>mode</i>		see text	none
<i>max_vel</i>			none
<i>min_vel</i>			none
<i>ch_list</i>	string	100 - 163	none

Comments

- <preset_pos> defines the position count at algorithm start-up. This is an unsigned 16-bit integer and can range from 0 to 65,535 for full speed modes ("SF"), or 0 to 32,767 for half speed modes ("SH").

- *<mode>* is used to select the stepping mode. the allowable values are:

Table 3. Stepping *<mode>* values

<i><mode></i> string	Stepping Mode	Speed	Channel
MFSFC2	Full	Full	2
MFSFC4	Full	Full	4
MFSHC2	Full	Half	2
MFSHC4	Full	Half	4
MHSFC2	Half	Full	4

Related error message:3127, "Undefined E1538 Stepper motor mode."

- The range of position values that an algorithm can send for the full-speed ("SF") mode is 0 to 65,535.
- The range of position values that an algorithm can send for the half-speed ("SH") mode is 0 to 32,767.
- *<min_vel>* is specified in steps per second and is the beginning step rate at the start of the 14 or 38 step ramp-up to *<max_vel>*. The *<min_vel>* should be a step rate that the motor can achieve from a standstill without missing a step. *<min_vel>* can range from 128 to 40,000 (64 to 40,000 for half speed "SH" modes).
- *<max_vel>* is specified in steps per second and is the maximum step rate that will be sent to the motor after ramp-up is complete. *<max_vel>* can range from 128 to 40,000 (64 to 40,000 for half speed "SH" modes).
- The increase in step rate from *<min_vel>* to *<max_vel>* will occur in 14 steps for a 2-Channel configuration, and will occur in 38 steps for a 4-channel configuration. Figure 25 shows the relationship between these parameters. A related error message: 3120, "Minimum velocity parameter must not exceed maximum velocity parameter."

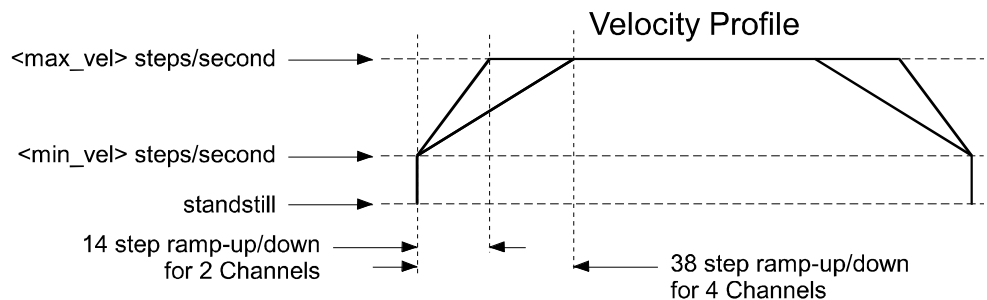


Figure 31. Relationship of *min_vel*, and *max_vel*

- Four-phase stepper motors that require less than 100mA phase current

can be directly driven by the SCP. See Figure 29 for a connection diagram that also shows the required user-supplied output protection components.

- *<ch_list>* specifies the channels that will control stepper motors. The channels referenced can be on more than one E1538. The channels must be in ascending order. Based on the *<mode>* parameter, the channels will be arranged into adjacent groups of 2 ("...C2"), or 4 ("...C4") channels. These groups can not be split across SCPs.

The algorithm can send new position values to the first channel in a motor-group. The algorithm will read the current position value from the second channel in the motor-group. Related error messages:

3115, "Channels specified are not in ascending order."

3116, "Multiple channels specified are not grouped correctly."

3117, "Grouped channels are not adjacent."

- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage See "Stepper Motor Control" on page 37.

*preset count to 0, full step, half speed, 4 channel, min speed 64s/s,
max speed 256s/s (in half speed mode, actual speed=half specified speed)*
SOUR:FUNC:STEP 0,MFSFC4,128,512,(@144:147)

SOURce:PULM[:STATe]

1SOURce:PULM[:STATe] <enable>,(@<ch_list>) enable the pulse width modulated mode for the PULSe channels in *<ch_list>*.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>enable</i>	boolean (uint16)	1 0 ON OFF	none
<i>ch_list</i>	string	100 - 163	none

Comments

- This command is coupled with the SOURce:FM command. If the FM state is enabled then the PULM state is disabled. If the PULM state is enabled then the FM state is disabled. If both the FM and the PULM states are disabled then the PULSe channel is in the single pulse mode.
- If the channels specified are not on a Frequency/Totalize SCP, an error will be generated.
- ***RST Condition:** SOUR:PULM:STATE OFF
- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

SOURce:PULM:STATe?

SOURce:PULM[:STATe]? (@<channel>) returns the pulse width modulated mode state for the PULSe channel in <channel>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments *Channel* must specify a single channel.

- **Returned Value:** returns 1 (on) or 0 (off). The type is **int16**.
- **Send with VXIplug&play Function:** hpe14XX_cmdInt32_Q(...)

SOURce:PULSe:PERiod

SOURce:PULSe:PERiod <period>,@<ch_list> sets the fixed pulse period value on a pulse width modulated pulse channel. This sets the frequency (1/period) of the pulse-width-modulated pulse train.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>period</i>	numeric (float32)	25E-6 to 7.8125E-3 (resolution 0.238μsec)	seconds
<i>ch_list</i>	string	100 - 163	none

- Comments**
- If the channels specified are not on a Frequency/Totalize SCP, an error will be generated.
 - ***RST Condition:** SOUR:FM:STATE OFF and SOUR:PULM:STATE OFF
 - **Related Commands:** SOUR:PULM:STATE, SOUR:PULS:POLarity
 - The variable pulse-width control for this channel is provided by the algorithm language. When the algorithm executes an assignment statement to this channel, the value assigned will be the pulse-width setting. For example:

```
O140 = .0025 /* set channel 43 pulse-width to 2.5 msec */
```

- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage SOUR:PULS:PER .005,(@140)

*set PWM pulse train to 200 Hz
on channel 40*

SOURce:PULSe:PERiod?

SOURce:PULSe:PERiod? (@<channel>) returns the fixed pulse period value on the pulse width modulated pulse channel in <channel>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- If the channels specified are not on a Frequency/Totalize SCP, an error will be generated.
- **Returned Value:** numeric period. The type is **float32**.
- **Send with VXIplug&play Function:** hpe14XX_cmdReal64_Q(...)

SOURce:PULSe:WIDTh

SOURce:PULSe:WIDTh <pulse_width>,@<ch_list> sets the fixed pulse width value on the frequency modulated pulse channels in <ch_list>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>pulse_width</i>	numeric (float32)	7.87E-6 to 7.8125E-3 (238.4E-9 resolution)	seconds
<i>ch_list</i>	string	100 - 163	none

Comments

- If the channels specified are not on a Frequency/Totalize SCP, an error will be generated.
- ***RST Condition:** SOUR:FM:STATE OFF and SOUR:PULM:STATE OFF
- **Related Commands:** SOUR:PULM:STATE, SOUR:PULS:POLarity
- The variable frequency control for this channel is provided by the algorithm language. When the algorithm executes an assignment statement to this channel, the value assigned will be the frequency setting. For example:

O143 = 2000 /* set channel 43 to 2KHz */

- **Send with VXIplug&play Function:** hpe14XX_cmd(...)

Usage SOUR:PULS:WIDTH 2.50E-3,(@143) *set fixed pulse width of 2.5 msec on channel 43*

SOURce:PULSe:WIDTh?

SOURce:PULSe:WIDTh? (@<ch_list>) returns the fixed pulse width value on a frequency modulated pulse channel.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- *Channel* must specify a single channel.
- If the channels specified are not on a Frequency/Totalize SCP, an error will be generated.
- **Returned Value:** returns the numeric pulse width. The type is **float32**.
- **Send with VXIplug&play Function:** hpe14XX_cmdReal64_Q(...)

SOURce:RPULse:POSition[:ANGLE]

SOURce:RPULse:POSition[:ANGLE] <degrees>,@<ch_list> sets the angular position of the rotational output pulse before the INIT command that starts algorithm execution. With the pulse position thus fixed, the pulse width (in angle or time depending on how SOUR:RPUL:VARType is set) will be controlled by the algorithm.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>degrees</i>	numeric (int32)	-33,554,430 to 33,554,430	degrees
<i>ch_list</i>	string	100 - 163	none

Comments

- Channels in <ch_list> must be referenced in a SOUR:FUNC:RPUL command before the next INIT command. Related error messages:
3113, "Channel specified is not in same SCP as reference channel."
3114, "First channel in SCP can not be used in RPULse output channel list."
- <degrees> has a resolution of 1 degree. The pulse is positioned at <degrees> modulo 360.

Usage SOUR:RPULSE:POS:ANGL 20,(@145) *preset channel 45 pulse position to 20 degrees*

SOURce:RPULse:POSition[:ANGLE]?

SOURce:RPULse:POSition[:ANGLE]? (@<channel>) returns the angular position set for <channel>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments • <channel> must specify a single channel only

Usage SOUR:RPULSE:POS:ANGL? (@145) *return pulse pos set for channel 45*

SOUR:RPULse:VARType

SOUR:RPULse:VARType <type>,@<ch_list> specifies the type of value that will be controlled (varied) by the algorithm. Depending on how the RPULse system is set up, the varied property can be either pulse position, or pulse width.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>type</i>	discrete (string)	TIME ANGLE	seconds
<i>ch_list</i>	string	100 - 163	none

Comments

- <type> specifies the that the algorithm will send values of either: ANGLE (in degrees) to the channel(s).
or
TIME (in seconds) to the channel(s).
- <ch_list> specifies the SOUR:FUNC:RPUL channel(s) that will be controlled (varied) by the algorithm. Channels in <ch_list> must be referenced in a SOUR:FUNC:RPUL command before the next INIT command. Related error messages:
3113, "Channel specified is not in same SCP as reference channel."
3114, "First channel in SCP can not be used in RPULse output channel list."

Usage SOUR:RPULSE:VART ANGL,(@145:147) *algorithm will control pulse width by ANGLE*

SOUR:RPULSE:VART TIME,(@143:144) *algorithm will control pulse width by TIME*

SOUR:RPULse:VARType?

SOUR:RPULse:VARType? (@<channel>) returns the type of value that will be controlled (varied) by the algorithm.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

- Comments**
- <ch_list> must specify a single channel only
 - Returns the string "TIME" | "ANGL"

Usage SOUR:RPULSE:VART? (@145) *returns the setting for chan 45*

SOURce:RPULse:WIDTh[:ANGLE]

SOURce:RPULse:WIDTh[:ANGLE] <degrees>,@<ch_list> sets the width of the rotational output pulse before the INIT command that starts algorithm execution. With the pulse width thus fixed, the pulse position (in angle or time depending on how SOUR:RPUL:VARType is set) will be controlled by the algorithm.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>degrees</i>	numeric (uint32)	0 to 360	degrees
<i>ch_list</i>	string	100 - 163	none

- Comments**
- Channels in <ch_list> must be referenced in a SOUR:FUNC:RPUL command before the next INIT command. Related error messages: 3113, "Channel specified is not in same SCP as reference channel." 3114, "First channel in SCP can not be used in RPULse output channel list."
 - <degrees> has a resolution of 1 degree.
 - Since the pulse width is specified in angle, changes in rotational velocity will not change the angular proportion of the pulse. Of course, changes in rotational velocity do effect the pulse width as regards time.

Usage SOUR:RPULSE:WIDTh:ANGL 260,(@147) *preset channel 47 pulse width to*

SOURce:RPULse:WIDTh[:ANGLE]?

SOURce:RPULse:WIDTh[:ANGLE]? (@<channel>) returns the width of the rotational output pulse currently set for <channel>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments

- <channel> must specify a single channel only.

Usage SOUR:RPULSE:WIDTh:ANGL? (@147) *return pulse width for channel 47*

SOURce:RPULse:WIDTh:TIME

SOURce:RPULse:WIDTh:TIME <seconds>,@<ch_list> sets the width of the rotational output pulse before the INIT command that starts algorithm execution. With the pulse width thus fixed, the pulse position (in angle or time depending on how SOUR:RPUL:VARType is set) will be controlled by the algorithm.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>seconds</i>	numeric (int32)	.00000787 to .015624	seconds
<i>ch_list</i>	string	100 - 163	none

Comments

- Channels in <ch_list> must be referenced in a SOUR:FUNC:RPUL command before the next INIT command. Related error messages: 3113, "Channel specified is not in same SCP as reference channel." 3114, "First channel in SCP can not be used in RPULse output channel list."
- <seconds> specifies pulse width in seconds, with a resolution of 238.4nS
- .Since the pulse width is specified in seconds, changes in rotational velocity will not change the time proportion of the pulse. Of course, changes in rotational velocity do effect the angular proportion of the pulse.

Usage SOUR:RPULSE:WIDTh:TIME .0040,@147) *preset channel 47 pulse width to 4 mSec*

SOURce:RPULse:WIDTh:TIME?

SOURce:RPULse:WIDTh:TIME? (@<channel>) returns the width of the rotational output pulse currently set for <channel>.

Parameters

Parameter Name	Parameter Type	Range of Values	Default Units
<i>channel</i>	string	100 - 163	none

Comments • <channel> must specify a single channel only.

Usage SOUR:RPULSE:WIDT:TIME? (@147) *return the pulse width set for channel 47*

Specifications

These specifications for the E1538A reflect its performance while installed on your VXI module.

General Specifications

Output Characteristics	Characteristic	Pull-Up Off	Pull-Up On (10K to Vcc)
	current source (logic 1)	0	.38mA @ 1.2V
	current sink (logic 0)	100 mA	100 mA
	Voltage (logic 1)	0	5V (no load)
	Voltage (logic 0)	0.5 Max sinking 100mA 0.1 Max sinking 20mA	0.5 Max sinking 100mA 0.1 Max sinking 20mA

Input Characteristics (VRS OFF for Chs0&1)	Characteristic	Pull-Up Off	Pull-Up On (10K to Vcc)
	Equivalent circuit	120K conn. to 0 Volts	9.2K Ω conn. to 4.6 Volts
	Maximum input low	programmable from -46 to 46 Volts ($\pm 0.5V$)	programmable from -46 to 46 Volts ($\pm 0.5V$)
	Minimum input high		

Input Isolation

No Isolation Provided

Cross-Talk Between Channels

A large signal on one channel has an effect on the accuracy of frequency measured on other channels as follows:

Sine-wave interfering signal up to 70Vpp	No degradation of specification
Square-wave Interfering signal <63Vpp	No degradation of specification
Square-wave Interfering signal >63Vpp to 70Vpp	Minimum input amplitude changes from 15mV to 18mV for frequency range of 1Hz to 10Khz (see Input Signal Characteristics spec.)

Maximum voltage applied to any input terminal

-48 Volts to 48 Volts

Maximum voltage applied to any output terminal

0 - 48 Volts (outputs are diode clamped at -0.3V)

Totalizer	Capacity	24 bits or 16,777,215
	Minimum Pulse Width	500nS
	Frequency Range	0-100 KHz

Frequency Measurement	Gate Time ($t_{aperture}$)	1 mSec to 1 Second, resolution $\frac{1}{f_{in}}$
	Range	$\frac{1}{t_{aperture}}$ to 100,000
	Accuracy	.01%
	Resolution (Hz)	$\frac{f_{input}}{t_{aperture} \times 4.194MHz}$
	Minimum Pulse Width	500 nS

Period Measurement	Gate Time ($t_{aperture}$)	40 μ Sec to 4 Second
	Range (SENS:PER:RANGE=1)	10 μ Sec to $t_{aperture}$ (1 sec max)
	(SENS:PER:RANGE=4)	40 μ Sec to $t_{aperture}$ (4 sec max)
	Accuracy	.01%
	Resolution	.2384 μ Sec
	Minimum Pulse Width	500 nSec

Time-out Mechanism (SENS:FREQ:LIM:LOW & SENS:PER:LIM:UPP) Programmable 4mSec to 60Sec

Aperture Time	The aperture time is the time allowed to average multiple period and frequency measurements	
	Function	Aperture Range
Frequency Period (2 ranges)		1mSec to 1 Sec 1mSec to 1 Sec (SENS:PER:RANGE 1) 4mSec to 4 Sec (SENS:PER:RANGE 4)

Auto-Gating (adaptive aperture)

1mSec to 1Sec for Frequency
1mSec to 1Sec for Perid (using SENS:PER:RANGE 1)
4mSec to 4Sec for Perid (using SENS:PER:RANGE 4)

Rotational Velocity Measure	Characteristics	Extra Tooth Wheel or Missing Tooth Wheel
	Range in RPS	$\frac{1}{n_{teeth}}$ to $\frac{100,000}{n_{teeth}}$
	Accuracy	.01%
	Resolution in RPS	$\frac{(n_{teeth} \times f)^2}{4.194MHz}$ where f is n_{teeth} per second
	Minimum Pulse Width	500 nS

Pulse Width Measure	Periods Averaged	1 to 255
	Range	5 μ S to 1 S
	Accuracy	$\pm(250nS+0.1\%)$
	Resolution	59.6 nSec

Frequency Source	Range	64 Hz to 40 KHz Square Wave 128 Hz to 40 KHz other shapes
	Accuracy	0.01%
	Resolution	$\frac{f_{out}^2}{4.194MHz}$

Pulse Source	Range	7.87 μ sec to 1/f-7.87 msec continuous pulse 7.87 μ sec to 7.812 msec single pulse per trigger
	Accuracy	0.01%
	Resolution	238.4 nsec

Rotational Pulse Source	Characteristics	Extra Tooth Wheel	Missing Tooth Wheel
	Position angle range	-33,554.430 to 33,554.430	
	Position resolution	1 degree up to 10,000 RPS	
	Position width range (angle)	0 to 360 degrees	
	Pulse width resolution (angle)	the larger of $\frac{\text{tooth} - \text{to} - \text{toothangle}}{128}$ or $360 \times \frac{238.4\text{nSec}}{\text{RotPeriod}}$	
	Pulse width range (time)	0 to Rotational Period (see note 3 on page 65)	
	Pulse width resolution (time)	238.4 nS	
	Minimum Rot. velocity	108 teeth per second	384 teeth per second