ENDEVCO
AN AMPHENOL COMPANY

Model 6634D Signal Conditioner

Programmer's Manual

April 2024, Rev. 1.0

Table of Contents

Introduction

Remote operation of the 6634D signal conditioner from a host, that is, a terminal, controller, PC, or computer, is accomplished by sending commands to it through one of its remote control interfaces. This manual describes how to setup, configure, and operate the signal conditioner through each of the remote interfaces. The signal conditioner is controlled remotely using either Standard Commands for Programmable Instruments (SCPI), or 6634D-specific commands. Detailed information on the SCPI command set, and how the signal conditioner processes those commands is included in this manual.

Note
For more information regarding the SCPI programming language, visit http://www.ivifoundation.org/scpi/default.aspx.

The level of detail in this chapter is based on the assumption that the reader is familiar with the basics of data communication interfaces.

Computer Interfaces

The 6634D has an RS-232 remote control serial interface and an Ethernet remote control interface. The 6634D remote control interfaces are backward compatible with the 6634C remote control interface.

Ethernet Interface

The 6634D uses an IPv4 socket interface for remote control. The IP address is static; DHCP is not supported.

The factory Ethernet settings are described here. All settings may be changed via the front panel, the web interface or the SCPI remote control interface.

Parameter	Factory Setting
IP Address	192.168.1.100
Netmask	255.255.255.0
Gateway	192.168.1.1
Port	49808

RS-232 Interface

6634C

For background, the 6634C's RS-232 interface is fixed at 300 baud 7N1 (sic) with no flow control. The LOCAL/REMOTE pins (5 and 6) must be tied to ground (pin 7) during communication.

6634D

The RS-232 interface comes from the factory with 115200 baud 8N1 with no flow control. The baud rate is selectable from the front panel. The following baud rates are supported: 1200, 2400, 4800, 9600, 14400, 19200, 38400, 57600 and 115200.

The 8N1 protocol is not configurable.

The rear panel LOCAL/REMOTE pins (DB25 pins 5 and 6) are not used.

6634C Remote Control API

The 6634D's remote control API is backward-compatible with the 6634C's remote control API. See IM6634C for details.

The 6634C utilizes hardware "jumpers" to change configurations. The 6634D has no jumpers; all configurations are controlled via software. The 6634D has more features than the 6634C, all of which are controlled via software. For these reasons, the following 6634D's SCPI API is preferred over the 6634C's proprietary API.

Standard Commands for Programmable Instruments (SCPI)

This section describes the remote control Applications Programming Interface (API). The 6634D uses the Standard Commands for Programmable Instruments (SCPI) command language. SCPI is the industry-standard command language for test and measurement systems.

SCPI Syntax

The following notation is used to describe the command formats.

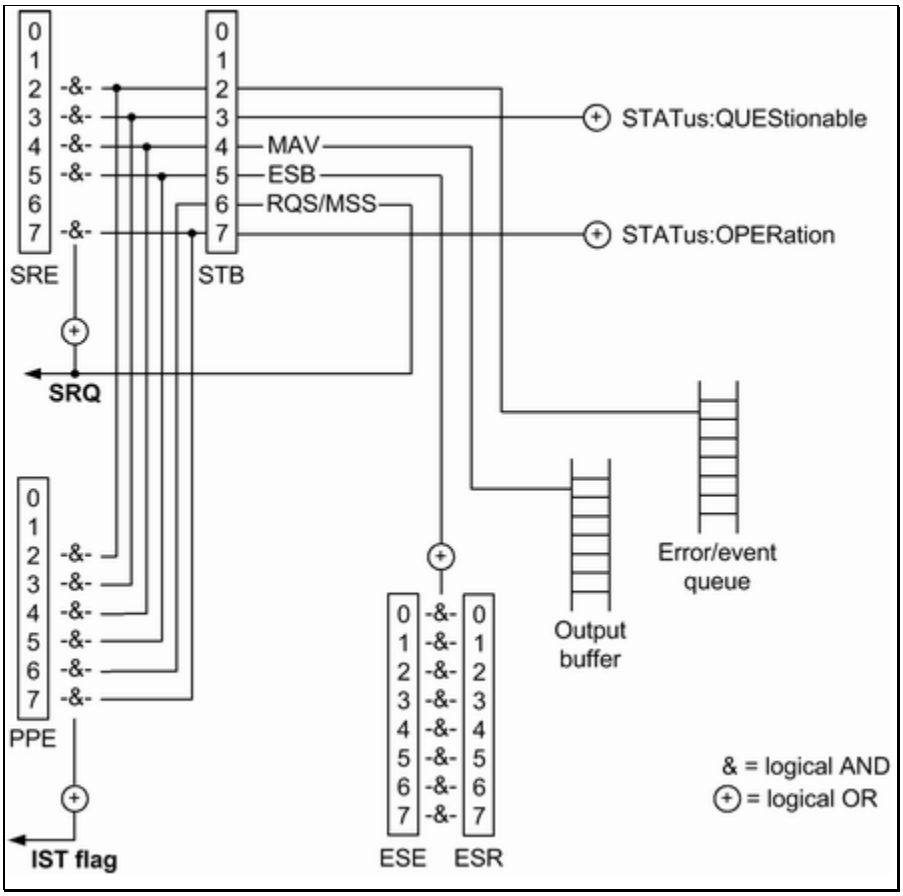
CAPITAL LETTERS	Capital letters are the minimally required letters of the command or query program header. Lowercase letters are the long form (complete spelling), which you can omit if desired. e.g. "CAL" is the same as "CALibrate" SCPI commands are not case sensitive. That means that "CALibrate" is the same as "CALIBRATE" and "calibrate" during communication.
<>	Angle brackets indicate that you must substitute a value for the enclosed parameter. For example "ACCELeration <value>", means ACCELeration needs a parameter. e.g. ACCELeration INTernal
[]	Square brackets are used to enclose optional information not required for execution of the command sequence. e.g. "[:INput]:ACCELeration?" can be replaced by "ACCEL?". The "[:INput]" is skipped.
|	The vertical bar can be read as "or" and is used to separate parameter options. e.g. "ACCELeration {INTernal | EXTernal}" means the command parameter is either INTernal or EXTernal.
{}	Braces (aka curly brackets) are used to enclose parameters within a command string.

Valid command terminators are:

- LF (Line Feed)
- CR (Carriage Return)
- CR LF (Carriage Return / Line Feed)

SCPI Registers

The following figure shows the hierarchy of the registers.



Introduction of the registers:

STB, SRE: The status byte (STB) register is at the highest level of the status reporting system. The mask register service request enable (SRE) is associated with the STB as its ENABle part if the STB is structured according to SCPI.

The STB provides a rough overview of the instrument status, collecting the information of the lower-level registers.

The STB receives its information from:

ESB: The summary bit of standard event status register indicates any enabled bit in the standard event status register (ESR). The standard event status enable (ESE) register is used as the ENABle part of the ESR.

Output buffer: Contains the messages that the instrument returns to the controller. It is not part of the status reporting system but determines the value of the MAV bit in the STB.

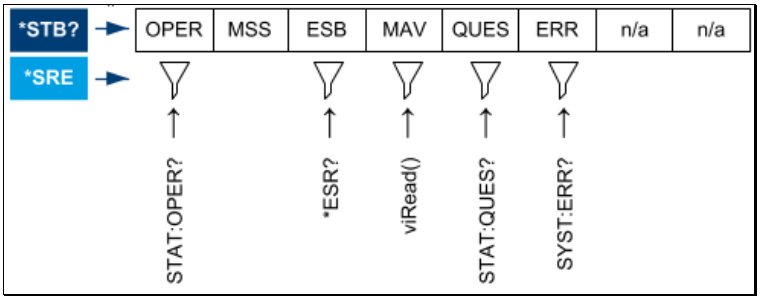
Error/event queue

Standard operation event status and **questionable event status** registers (**STATus:QUEStionable, STATus:OPERation**): defined by SCPI and contain detailed information on the instrument.

STB and SRE

The status byte (STB) provides a rough overview of the instrument status by collecting the pieces of information of the lower registers. The STB represents the highest level within the SCPI hierarchy.

The status byte (STB) is linked to the service request enable (SRE) register on a bit-by-bit basis.



The STB corresponds to the EVENt part of a SCPI register, it indicates general instrument events. This register is cleared when it is read.

A special feature is that bit 6 acts as the summary bit of the remaining bits of the status byte.

The SRE corresponds to the ENABle part of a SCPI register. If a bit is set in the SRE and the associated bit in the STB changes from 0 to 1, a service request (SRQ) is generated.

Bit 6 of the SRE is ignored, because it corresponds to the summary bit of the STB.

Bit	Weight	Meaning
2	4	**Error queue summary** This bit is set when an entry is made in the error or event queue.
3	8	**Questionable register summary** The questionable status summary bit indicates a questionable instrument status, which can be further pinned down by polling the QUEStionable register.
4	16	**MAV bit**

Bit	Weight	Meaning
		The message available bit is set if a message is available and can be read from the output buffer. This bit can be used to transfer data automatically from the instrument to the controller.
5	32	**ESB bit** This summary bit of standard event status register; set if one of the bits in the standard event status (ESR) register is set and enabled in the standard event enable (ESE) register. Setting of this bit implies an error or an event which can be further pinned down by polling the event status register.
6	64	**MSS bit** The master summary status bit is set if one of the other bits of the STB is set together with its mask bit in the SRE register.
7	128	**Operation status summary** This bit is set if an EVENt bit is set in the OPERation status register and the associated ENABle bit is set to 1. A set bit indicates that the instrument is just performing an action. The type of action can be determined by querying the STATus:OPERation status register.

Related common commands

The STB is read out using the *STB? command or a serial poll.

The SRE can be set using the *SRE command and read using the *SRE? command.

ESR and ESE

The event status register (ESR) indicates general instrument states. It is linked to the standard event status enable (ESE) register on a bit-by-bit basis.

- The ESR corresponds to the CONDition part of a SCPI register indicating the current instrument state. However, reading the ESR deletes the contents.
- The ESE corresponds to the ENABle part of a SCPI register. If a bit is set in the ESE and the associated bit in the ESR changes from 0 to 1, the ESB bit in the status byte is set.

Bits in ESR register		
Bit	**Weight**	**Meaning**
0	1	**Operation complete** This bit is set on receipt of the *OPC command after all previous commands have been executed.
1	2	**Request control** This bit is set if the instrument requests the controller function.
2	4	**Query error** This bit is set if the controller wants to read data from the instrument without having sent a query. It is also set if the controller does not fetch requested data and sends new instructions to the instrument instead. The cause is often a query which is faulty and hence cannot be executed.
3	8	**Device-dependent error** This bit is set if a device-dependent error occurs. An error message with a number between -300 and -399 or a positive error number, which describes the error in greater detail, is entered into the error queue.
4	16	**Execution error** This bit is set if a received command is syntactically correct, but cannot be performed for other reasons. An error message with a number between -200 and -300, which describes the error in greater detail, is entered into the error queue.
5	32	**Command error** This bit is set if a command which is undefined or syntactically incorrect is received. An error message with a number between -100 and -200, which describes the error in greater detail, is entered into the error queue.
6	64	**User request**

Bits in ESR register		
Bit	**Weight**	**Meaning**
7	128	**Power on** (supply voltage on) This bit is set when the instrument is switched on.

Related common commands

The event status register (ESR) can be queried using *ESR? command.

The standard event status enable (ESE) register can be set using the *ESE command and read using *ESE? command.

Command Details

*CLS

Description Clears the following:

- Error Queue.
- Status Byte Register.
- Standard Event Status Register.
- Operation Status Event Register.
- Questionable Status Event Register.
- Questionable Limit Status Event Register.
- Questionable Limit Channel Status Event Register.

Parameters None

Examples
```
*CLS
```

Query None

*RST

Description Resets the signal conditioner to the factory default settings. The Ethernet settings are not affected.

Parameters None

Examples
```
*RST
```

Query None

*IDN?

Description Returns the unit identification in the following format:

<Manufacturer>,<Model>,<Serial Number>,<Software Revision>

Parameters None

Examples
```
*IDN?
returns
ENDEVCO,6634D,EN1234567, Build 1920 Feb  7 2024 13:04:37
```

*OPC

Description Sets the OPC bit (bit 0) of the Standard Event Status Register at the completion of all pending operations.

Parameters None

Examples
```
*OPC
```

Query *OPC? reads out the "1" at the completion of all pending operations. The query blocks the execution of the user program until execution of all previous instructions. E.g.

*OPC?
returns
1

*ESR?

Description Reads out the value of the Standard Event Status Register. Executing this command clears the register value.

Parameters None

Examples
```
*ESR?
returns
48
```

*ESE

Description Sets or reads out the value of the Standard Event Status Enable Register.

Parameters 0 to 255

Examples
```
*ESE 0
```

Query `*ESE?` reads out the value of the Standard Event Status Enable Register. E.g.

```
*ESE?
returns
0
```

*SRE

Description Sets or reads out the value of the Service Request Enable Register.

Parameters 0 to 255

Examples
```
*SRE 0
```

Query `*SRE?` reads out the value of the Service Request Enable Register. E.g.

```
*SRE?
returns
0
```

*STB?

Description Reads out the value of the Status Byte Register.

Parameters None

Examples
```
*STB?
returns
4
```

*TST?

Description Reads out the analyzer self-test result. 0 indicates no failures found. A non-zero value indicates one or more of failure conditions exist. The SYST:TEST? query returns a textual description of the failures.

Note: the query returns a non-zero value when it is issued until the instrument is ready..

Parameters None

Examples
```
*TST?
returns
0
```

*WAI

Description Waits till the completion of all pending commands.

Parameters None

Examples `*WAI 0`

SYSTem:ERRor[:NEXT]?

Description Reads out the error message when executing SCPI commands, from the FIFO (First In First Out) error queue stored in the Analyzer. The read-out error is deleted from the error queue. The *CLS command clears the error queue. The maximum size of the queue is 100 messages.

Parameters None

Examples
```
SYSTem:ERRor?
or
SYSTem:ERRor:NEXT?
returns
-113,"Undefined header"
```

SYSTem:ERRor:COUNt

Description Returns the number of erors in the error queue.

Parameters None

Examples
```
SYSTem:ERRor:COUNt?
returns
7
```

Query `*SRE?` reads out the value of the Service Request Enable Register. E.g.

```
*SRE?
returns
0
```

SYSTem:VERSion?

Description Returns the SCPI protocol version.

Parameters None

Examples
```
SYSTem:VERSion?
returns
1999.0
```

STATus:QUEStionable[:EVENt]?

Description Reads out the value of the Questionable Status Event Register.

Parameters None

Examples
```
STATus:QUEStionable:EVENt?
returns
0
```

STATus:QUEStionable:ENABle <numeric_value>

Description Sets or reads out the value of the Questionable Status Enable Register.

Parameters 0 to 65535

Examples
```
STATus:QUEStionable:ENABle 1
```

Query
```
STATus:QUEStionable:ENABle?
returns
0
```

STATus:PRESet

Description Resets all the status registers to the factory settings.

Parameters None

Examples
```
STAT:PRES
```

SYSTem:SerialNumber "<string>"

SYSTem:SN "<string>"

Description Sets the unit's serial number.

Parameters Serial number, in double quotes. The string is limited to 15 arbitrary characters.

Examples
```
SYSTem:SerialNumber "0123456789"
SYST:SN "arbitrary text"
```

Query
```
SYSTem:SerialNumber?
returns
"arbitrary text"
```

SYSTem:COMMunicate:NETwork:MAC "<string>"

Description Sets the Ethernet MAC address.

Parameters Ethernet MAC address, colon delimited octets, in double quotes.

Examples
```
SYST:COMM:NET:MAC "70:B3:D5:4C:A0:00"
```

Query
```
SYST:COMM:NET:MAC?
returns
"70:B3:D5:4C:A0:01"
```

[INput]:TYPE {SE | DIFF | IEPE | DRCC_ACCELeration | DRCC_VELocity | SE_VELocity_coil | DIFF_VELocity_coil}

Description Sets the sensor input type.

Parameters SE, DIFF, IEPE, DRCC_ACCELeration, DRCC_VELocity, SE_VELocity_coil, or DIFF_VELocity_coil.

Examples
```
IN:TYPE DIFF_VELocity_coil
IN:TYPE DIFF
```

Query
```
INPUT:TYPE?
returns
DIFF
```

[:OUTput]:AC_FSo {1 | 5 | 10}

Description Sets the AC full scale output.

Parameters 1V, 5V or 10V.

Examples
```
OUTput:AC_FSo 10
```

Query
```
OUT:AC_FS?
returns
10
```

[:OUTput]:DC_FSo {1 | 5 | 10}

Description Sets the DC full scale output.

Parameters 1V, 5V or 10V.

Examples
```
OUTput:DC_FSo 10
```

Query
```
OUT:DC_FS?
returns
10
```

MEASure:VOLTage:DC?

Description Queries the DC output, in the current units.

Parameters n/a.

Query
```
MEAS:VOLT:DC?
returns
```
"0.443 g's rms" or “Low Signal” or “Overload”

[:INput]:ACCELeration {INTernal | EXTernal}

Description Sets the IEPE excitement current.

Parameters INTernal or EXTernal

Examples
```
IN:IEPE:CURR ON
```

Query
```
IN:ACCEL?
returns
```
"INTERNAL"

[:INput]:ACCELeration {INTernal | EXTernal}

Description Sets the acceleration input to the velocity integrator.

Parameters INTernal or EXTernal.

Examples
```
IN:ACCELeration INTernal
IN:ACCEL EXT
```

Query
```
IN:ACCEL?
returns
INT
```

[:INput]:VELocity {INTernal | EXTernal}

Description Sets the velocity input to the displacement integrator.

Parameters INTernal or EXTernal.

Examples
```
IN:VELocity INTernal
IN:VEL EXT
```

Query
```
IN:VEL?
returns
"INTERNAL"
```

[:INput]:DRCC:VOLTage {OFF | ON}

Description Sets the DRCC excitement voltage.

Parameters OFF or ON

Examples
```
IN:ACCELeration INTernal
IN:ACCEL EXT
```

Query
```
IN:DRCC:VOLT?
returns
"ON"
```

[:INput]:IEPE:CURRent {OFF | ON}

Description Sets the IEPE excitement current.

Parameters OFF or ON

Examples
```
IN:IEPE:CURR ON
```

Query
```
IN:IEPE:CURR?
returns
"ON"
```

[:FILTer]:EXTernal:ENABle {0 | 1}

Description Enables the user external filter between DB-25 pins BB-OUT and BB-IN.

Parameters 0 or 1

Examples
```
FILTer:EXTernal:ENABle 1
FILT:EXT:ENAB 1
```

Query
```
FILTer:EXTernal:ENABle?
or
FILT:EXT:ENAB?
return
1
```

UNITS {IMPERIAL | METRIC}

Description Sets the system's units.

Parameters IMPERIAL or METRIC

Examples
```
UNITS IMPERIAL
```

Query
```
UNITS?
returns
"IMPERIAL"
```

SENSitivity <number_value>

Description Sets the sensor's sensitivity.

Parameters Decimal number

Examples
```
SENS 123.45
```

Query
```
SENS?
returns
123.4500
```

[:FILTer]:PROGrammable {BYPASS | HPF | LPF | BPF}

Description Sets the programmable filter mode.

Parameters BYPASS, HPF (high pass filter), LPF (low pass filter) or BPF (bandpass filter)

Examples
```
FILT:PROG BPF
```

Query
```
FILT:PROG?
returns
"BPF"
```

[:FILTer]:PROGrammable:LPF <numeric_value>

Description Sets the low pass filter corner frequency, in Hz.

Parameters 50 to 10000

Examples
```
FILT:PROG:LPF 123
```

Query
```
FILT:PROG:LPF?
returns
123
```

[:FILTer]:PROGrammable:HPF <numeric_value>

Description Sets the high pass filter corner frequency, in Hz.

Parameters 5 to 500

Examples
```
FILT:PROG:HPF 250
```

Query
```
FILT:PROG:HPF?
returns
250
```

[:OUTput]:AC {ACCELeration | VELocity | DISPlacement}

Description Sets the AC output mode.

Parameters ACCELeration, VELocity or DISPlacement

Examples
```
OUT:AC VEL
```

Query
```
OUT:AC?
returns
"VEL"
```

[:OUTput]:DC {RMS | AVERAGE | PEAK}

Description Sets the DC output mode.

Parameters RMS, AVERAGE or PEAK

Examples
```
OUT:DC AVERAGE
```

Query
```
OUT:DC?
returns
"AVERAGE"
```

[:OUTput]:FSo <numeric_value>

Description Sets the full scale output in the current units.

Parameters Numeric value.

Examples
```
OUT:FS 51.23
```

Query
```
OUT:FS?
returns
51.2300
```

ALERT:LEVel <numeric_value>

Description Sets the alert alarm level.

Parameters 0 to 100

Examples
```
ALERT:LEV 50
```

Query
```
ALERT:LEV?
returns
50
```

WARNing:LEVel <numeric_value>

Description Sets the warning alarm level.

Parameters 0 to 100

Examples
```
WARNing:LEV 75
```

Query
```
WARNing:LEV?
returns
75
```

ALARM:RESET

Description Resets both the alert and warning alarms.

Parameters None.

Examples
```
ALARM:RESET
```

Query
```
ALARM?

returns one of the following:

"NONE"
"ALERT"
"WARNING"
"ALERT|WARNING"
```

CALibrate

Description Initiates the calibration sequence.

Parameters None.

Examples
```
CAL
```

Query
```
n/a
```

SETTINGS:STORE <numeric_value>

Description Stores the current system settings.

Parameters 1 to 10

Examples
```
SETTINGS:STORE 2
```

Query
```
n/a
```

SETTINGS:RECALL <numeric_value>

Description Restores the specified system settings.

Parameters 1 to 10

Examples
```
SETTINGS:RECALL 1
```

Query
```
n/a
```

SETTINGS:CURRENT?

Description Returns the current system settings.

Parameters n/a.

Examples
```
SETTINGS:RECALL 1
```

Query
```
SETTINGS:CURRENT?
returns
1
```

[SYSTem][:COMMunicate]:NETwork:ADDRess "<string>"

Description Sets the Ethernet IP address.

Parameters IP address, in double quotes.

Examples
```
SYST:COMM:NET:ADDR "192.168.1.100"
```

Query
```
SYST:COMM:NET:ADDR?
returns
"192.168.1.100"
```

[SYSTem][:COMMunicate]:NETwork:GATE "<string>"

Description Sets the Ethernet gateway address.

Parameters Gateway address, in double quotes.

Examples
```
SYST:COMM:NET:GATE "192.168.1.1"
```

Query
```
SYST:COMM:NET:GATE?
returns
```

```
"192.168.1.1"
```

[SYSTem][:COMMunicate]:NETwork:SUBNet "<string>"

Description Sets the Ethernet subnet.

Parameters Subnet, in double quotes.

Examples
```
SYST:COMM:NET:SUBN "255.255.255.0"
```

Query
```
SYST:COMM:NET:SUBN?
returns
"255.255.255.0"
```

[SYSTem][:COMMunicate]:NETwork:PORT <numeric_value>

Description Sets the Ethernet remote control port number.

Parameters 49152 to 65535

Examples
```
SYST:COMM:NET:PORT 49808
```

Query
```
SYST:COMM:NET:PORT?
returns
49808
```

Examples

System Reset

```
*RST;*CLS;*OPC?
```

The 6634D uses the SCPI standard API to report errors. (The 6634C API uses the "E?XXXX;" query to report errors.)

Error Queries

Clear all errors

```
*CLS
```

Read the ESR:

```
*ESR?
```

Response

```
0
```

0 = no errors

Get error count:

```
SYSTem:ERRor:COUNt?
```

Response

```
0
```

Execute two bad commands:

```
NONSENSE:FOO?
*NONSENSE?
```

Query the error count:

```
SYSTem:ERRor:COUNt?
```

Response

```
2
```

There are 2 errors in the queue.

Read the ESR:

```
*ESR?
```

Response:

```
48
```

48 decimal = 0011 0000 binary = 16 + 32 = Execution Error and Command Error

Query First Error:

```
SYSTem:ERRor?
```

Response:

```
-113,"Undefined header"
```

Query Second Error:

```
SYSTem:ERRor?
```

Response

```
-113,"Undefined header"
```

Query Errors again:

```
SYSTem:ERRor?
```

Response:

```
0,"No error"
```

Calibration

The 6634D uses the "`CALibrate`" command to initiate calibration, and the "*ESR?" query to get the status. The "*ESR?" command returns the bitmap defined here:

ESR Bit	Decimal	Description
0	1	Operation complete
1	2	Request Control
2	4	Query Error
3	8	Device Dependent Error
4	16	Execution Error (e.g. range error)
5	32	Command error (e.g. syntax error)
6	64	User Request
7	128	Power On

(The 6634C's 'C' command initiates calibration, and the 6634C transmits "C \x07" when calibration has completed.)

Initiate calibration:

```
*RST
*CLS
CAL
```

Query the status until the Operation Complete bit (1) is set:

```
*ESR?
0

*ESR?
0

*ESR?
0

*ESR?
1
```

Parse the "*ESR?" response for any errors.

Sample Session

This section describes how to setup the 6634D to collect data using a Model 6222S-100A differential PE accelerometer with the following characteristics:

- Sensitivity 100 pC/g
- Maximum 500 g pk

Most commands are followed by an error check and a query to verify the setting was committed.

The main commands are highlighted in yellow.

Command	Description
Initialization	
*CLS	Clear all errors
UNITS IMPERIAL	Set the units to Imperial
*ESR?	Verify the response is 0 (no errors)
UNITS?	Verify the units are "IMPERIAL"
Set the Sensor Input	
INPUT:TYPE DIFF	Set the Input Type to Differential PE
*ESR?	Verify the response is 0 (no errors)
INPUT:TYPE?	Verify the Input Type is "DIFF"
SENSITIVITY 100.0	Set the sensor sensitivity to 100
*ESR?	Verify the response is 0 (no errors)
SENSITIVITY?	Verify the sensitivity is 100.0
Set the Filters	
FILTER:EXTERNAL:ENABLE 0	Disable the external filter
*ESR?	Verify the response is 0 (no errors)
FILTER:EXTERNAL:ENABLE?	Verify the external filter setting is 0
FILTER:PROG BYPASS	Bypass the programmable filter
*ESR?	Verify the response is 0 (no errors)
FILTER:PROG?	Verify the programmable filter setting is "BYPASS"
Set the Integrators	
IN:ACCELERATION INTERNAL	Set the acceleration input to the velocity integrator to internal
*ESR?	Verify the response is 0 (no errors)
IN:ACCELERATION?	Verify the acceleration input to the velocity integrator is set to "INTERNAL"
IN:VELOCITY INTERNAL	Set the velocity input to the displacement integrator to internal
*ESR?	Verify the response is 0 (no errors)

IN:VELOCITY?	Verify the velocity input to the displacement integrator is set to "INTERNAL"
Set the Alarms	
ALERT:LEVEL 100	Disable alerts
*ESR?	Verify the response is 0 (no errors)
WARNING:LEVEL 100	Disable warnings
*ESR?	Verify the response is 0 (no errors)
Set the Outputs	
OUTPUT:AC_FSO 10	Set the AC full scale output to 10 volts
*ESR?	Verify the response is 0 (no errors)
OUTPUT:AC_FSO?	Verify the AC full scale output is 10 volts
OUTPUT:DC_FSO 10	Set the DC full scale output to 10 volts
*ESR?	Verify the response is 0 (no errors)
OUTPUT:DC_FSO?	Verify the DC full scale output is 10 volts
OUTPUT:FSO 500.0	Set the full-scale output to 500 g pk
*ESR?	Verify the response is 0 (no errors)
OUTPUT:FSO?	Verify the full-scale output is 500 g pk
Read the Acceleration RMS	
OUTPUT:AC ACCELERATION	Set the AC output to acceleration
*ESR?	Verify the response is 0 (no errors)
OUTPUT:AC?	Verify the AC output is "ACCEL"
OUTPUT:DC RMS	Set the DC output to RMS
*ESR?	Verify the response is 0 (no errors)
OUTPUT:DC?	Verify the DC output is "RMS"
MEASURE:VOLTAGE:DC?	Read the voltage output representing acceleration (g's) RMS. E.g. "0.443 g's rms"
Read the Average Velocity	
OUTPUT:AC VELOCITY	Set the AC output to velocity
*ESR?	Verify the response is 0 (no errors)
OUTPUT:AC?	Verify the AC output is "VEL"
OUTPUT:DC AVERAGE	Set the DC output to average
*ESR?	Verify the response is 0 (no errors)
OUTPUT:DC?	Verify the DC output is "AVERAGE"
MEASURE:VOLTAGE:DC?	Read the voltage output representing velocity (inches per second) average. E.g. "1.234 ips avg"
Read the Peak Displacement	
OUTPUT:AC DISPLACEMENT	Set the AC output to displacement
*ESR?	Verify the response is 0 (no errors)
OUTPUT:AC?	Verify the AC output is "DISP"

`OUTPUT:DC PEAK`	Set the DC output to PEAK
`*ESR?`	Verify the response is 0 (no errors)
`OUTPUT:DC?`	Verify the DC output is "PEAK"
`MEASURE:VOLTAGE:DC?`	Read the voltage output representing displacement (mils) peak. E.g. "0.151 mils pk"

6634C API Cross-Reference

The following table describes how to translate 6634C commands to 6634D SCPI commands.

Name	6634C API	6634D SCPI
Go Remote	Z*XXXX;	N/A
Go Local	Y*XXXX;	N/A
Become Listener/Talker	InnXXXX;	N/A
Enter Single-Ended PE input type	T0XXXX;	Input:TYPE:SE
Enter velocity coil input type	T1XXXX;	Input:TYPE:DRCC_VELocity
Enter RCC input type	T2XXXX;	Input:TYPE:DRCC_ACCELeration
Enter Differential PE input type	T3XXXX;	Input:TYPE:DIFF
Enter Sensitivity	Snnnnn;	SENSitivity nn.nn
Enter Acceleration Output Mode	M0XXXX;	OUTput:AC ACCELeration
Enter Velocity output mode	M1XXXX;	:OUTput:AC VELocity
Enter Displacement output mode	M2XXXX;	:OUTput:AC DISPlacement
Enter RMS as DC output Mode	D0XXXX;	:OUTput:DC RMS
Enter Average as DC output mode	D1XXXX;	:OUTput:DC AVERAGE
Enter Peak as DC output mode	D2XXXX;	:OUTput:DC PEAK
Enter Full Scale	Onnnnn;	:OUTput:FSo nnn.nn
Select Filter In	F1XXXX;	:FILTer:EXTernal:ENABle 1
Select Filter Out	F0XXXX;	:FILTer:EXTernal:ENABle 0
Enter Lower Cutoff	Lnnnnn;	:FILTer:PROGrammable:LPF nnn
Enter Upper Cutoff	Unnnnn;	:FILTer:PROGrammable:HPF nnn
Enter warning alarm level	Wnnnnn;	WARNing:LEVel nn
Enter alert alarm level	Annnnn;	ALERT:LEVel nn
Normal Mode	B0XXXX;	?
Peak Detector mode	B1XXXX;	?
Reset Both Alarms	RXXXXX;	ALARM:RESET
Calibrate System	CXXXXX;	CAL
Store Setup	PnnXXXX;	SETTINGS:STORE n
Recall Setup	QnnXXXX;	SETTINGS:RECALL n
Request Input Type	T?XXXX;	INput:TYPE?
Request Sensitivity	S?XXXX;	SENSitivity?
Request Output Mode	M?XXXX;	:OUTput:AC?
Request DC Output Mode	D?XXXX;	:OUTput:DC?
Request Full Scale	O?XXXX;	:OUTput:FSo?
Request Filter	F?XXXX;	:FILTer:EXTernal:ENABle?
Request Lower Cutoff	L?XXXX;	:FILTer:PROGrammable:LPF?
Request Upper Cutoff	U?XXXX;	:FILTer:PROGrammable:HPF?
Request Warning Alarm Level	W?XXXX;	WARNing:LEVel?
Request Alert Alarm Level	A?XXXX;	ALERT:LEVel?
Alarm condition Request	K?XXXX;	ALARM?
Errors that have occurred at power up	E?XXXX;	*ESR?

Data Request for Programmed Setup	V?XXXX;	MEASure:VOLTage:DC?
Address Request.	I?XXXX;	N/A