What’s in this guide?

The top menu bar and the entries in the Table of Contents are all hyperlinks, just click on them to go to the topic.

We recommend you visit the TANDBERG web site regularly for updated versions of this guide. Go to: http://www.tandberg.com/docs

Table of Contents

Introduction

Introduction .......................................................... 5
The purpose of this document .................................. 5
Disclaimers and Notices ......................................... 5
Products covered in this guide ............................... 5
Other documents you might find useful ................. 5

TANDBERG Codec C40 ........................................... 6

What’s new in this version ....................................... 7
Software release notes ......................................... 7
User documentation ............................................. 7
New features and improvements ......................... 7
New commands .................................................. 7
Commands that have been changed ..................... 8
Commands that are removed ................................. 8

About the TANDBERG API

TANDBERG API .................................................. 10
Basic Principles .................................................. 10
The TANDBERG API-Engine ............................. 10
Structuring of Information ................................. 10
Addressing Using XPath or TANDBERG SimplePath .. 11
Feedback .......................................................... 11

Connecting to the codec ...................................... 12
Accessing XACLI ................................................. 12
Telnet/SSH login ............................................... 12
Serial port login .............................................. 12
Serial port configurations .................................. 12
Reboot .......................................................... 12
Hardware & Cabling (RS-232) ............................ 12
Troubleshooting (RS-232) ................................. 12

Value types and formats ..................................... 13
Root commands ............................................... 14
Main type of commands ..................................... 15
Configuration type commands ............................ 15
Command type commands .................................. 15
Status type commands ....................................... 15
Special type commands ..................................... 15
Feedback type command ................................... 15
Preferences type command ................................. 15

About xConfiguration ......................................... 16

xConfiguration operations .................................. 17
Return result parameters .................................. 17
xConfiguration Help .......................................... 17
xConfiguration Read ......................................... 17
xConfiguration Write ....................................... 17

About xCommand ............................................... 18

xCommand operations ....................................... 19
Return result parameters .................................. 19
xCommand Help .............................................. 19
xCommand Write ............................................ 19

About xStatus command .................................... 20
Query status information .................................. 21

About xHistory command ................................... 22

About xFeedback ............................................... 23

About xPreferences .......................................... 24
The xPreferences output modes ......................... 24

TANDBERG XML API service .............................. 25

Description of the xConfiguration commands

Description of the xConfiguration commands .......... 27
The Audio settings ........................................... 27
The Camera settings ........................................ 31
The Conference settings .................................. 33
The GPIO settings ......................................... 35
The H323 settings .......................................... 35
<table>
<thead>
<tr>
<th>The Network settings</th>
<th>The Call status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The NetworkServices</td>
<td>The Camera status</td>
</tr>
<tr>
<td>settings</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The Conference status</td>
</tr>
<tr>
<td>The Phonebook</td>
<td>The Diagnostics status</td>
</tr>
<tr>
<td>settings</td>
<td></td>
</tr>
<tr>
<td>The Provisioning</td>
<td>The GPIO status</td>
</tr>
<tr>
<td>settings</td>
<td></td>
</tr>
<tr>
<td>The SerialPort</td>
<td>The H323 status</td>
</tr>
<tr>
<td>settings</td>
<td></td>
</tr>
<tr>
<td>The SIP</td>
<td>The HTTP Feedback status</td>
</tr>
<tr>
<td>settings</td>
<td></td>
</tr>
<tr>
<td>The Standby</td>
<td>The Media Channel status</td>
</tr>
<tr>
<td>settings</td>
<td></td>
</tr>
<tr>
<td>The SystemUnit</td>
<td>The Network status</td>
</tr>
<tr>
<td>settings</td>
<td></td>
</tr>
<tr>
<td>The Time settings</td>
<td>The Preset status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Video settings</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Experimental</td>
<td>The SIP status</td>
</tr>
<tr>
<td>menu</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The Standby status</td>
</tr>
<tr>
<td></td>
<td>The System Unit status</td>
</tr>
<tr>
<td></td>
<td>The Video Input status</td>
</tr>
<tr>
<td></td>
<td>The Video Output status</td>
</tr>
<tr>
<td></td>
<td>The Video Layout status</td>
</tr>
<tr>
<td></td>
<td>The Video status</td>
</tr>
<tr>
<td></td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>Description of the</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>xStatus commands</td>
<td></td>
</tr>
<tr>
<td>xCommands with</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>parameters</td>
<td></td>
</tr>
<tr>
<td>The Audio commands</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Boot command</td>
<td>The Preset status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Call commands</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Call Log</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>commands</td>
<td></td>
</tr>
<tr>
<td>The CamCtrlPip</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>command</td>
<td></td>
</tr>
<tr>
<td>The Camera commands</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Dial command</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The DTMF command</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Far End Control commands</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The GPIO commands</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The HTTP Feedback commands</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Key commands</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Message command</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Phonebook</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>commands</td>
<td></td>
</tr>
<tr>
<td>The Presentation</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>commands</td>
<td></td>
</tr>
<tr>
<td>The Preset commands</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>The Provisioning</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>commands</td>
<td></td>
</tr>
<tr>
<td>The SString Send</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>command</td>
<td></td>
</tr>
<tr>
<td>The Standby</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>command</td>
<td></td>
</tr>
<tr>
<td>The SystemUnit</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>commands</td>
<td></td>
</tr>
<tr>
<td>The Video</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>commands</td>
<td></td>
</tr>
<tr>
<td>The Experimental</td>
<td>The Provisioning status</td>
</tr>
<tr>
<td>commands</td>
<td></td>
</tr>
</tbody>
</table>

**Appendices**

- Dynamic audio API
- The equalizer
- The Audio Console application
- The equalizer filter parameters
- The equalizer IIR filter
- The filter types
- Stereo
- Stereo in point to point call
- Stereo in Multisite
- Example of local stereo configuration
- Startup script
- Adding a startup script
- Remote Control TRC5–Key map
- Remote Control TRC5–One page overview
Chapter 1

Introduction
Introduction

You may already be familiar with the system integrator guides for the Codec C Series. To help you find the information you need, TANDBERG has split the guide into two smaller and more accessible parts listed on the right of this page.

The purpose of this document

The purpose of this document is to introduce you to the TANDBERG Application Programmer Interface (API) in general and to serve as a reference guide for the API commands.

Disclaimers and Notices

The objective of this documentation is to provide the reader with assistance in using and configuring the product. The capabilities of TANDBERG products and other manufacturers’ products change over time and so the required configuration may be different from that indicated here. If you have any suggestions for changes to this document, please feed them back to TANDBERG through your TANDBERG Authorized Service Representative.

If you need technical support, please contact your TANDBERG Authorized Service Representative.

The specifications for the product and the information in this Guide are subject to change at any time, without notice, by TANDBERG. Every effort has been made to supply complete and accurate information in this Guide; however, TANDBERG assumes no responsibility or liability for any errors or inaccuracies that may appear in this document.

TANDBERG® is a registered trademark belonging to Tandberg ASA. Other trademarks used in this document are the property of their respective holders.

This Guide may be reproduced in its entirety, including all copyright and intellectual property notices, in limited quantities in connection with the use of this product. Except for the limited exception set forth in the previous sentence, no part of this Guide may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronically, mechanically, by photocopying, or otherwise, without the prior written permission of TANDBERG.

www.tandberg.com
© 2010 TANDBERG
TANDBERG Codec C40

We are glad to introduce a new codec with the same design and technology principles as the Codec C90 and Codec C60. Supported with software version TC3.0.

The TANDBERG Codec C40 provides all the power required to transform any conference room to a HD video collaboration room. Designed for any standard HD integration project, the Codec C40 is the ideal solution for everyday video conferencing and collaboration solution. 1080p HD video, and Multisite™ features combine to make the Codec C40 ideal for a variety of applications.

The Codec C40 is ideal for standard meeting rooms, executive offices and team collaboration rooms.

- Full High Definition Video with up to 2 HD sources, and collaboration with optimal definition for the best video quality every time, regardless of environment.
- Highest Quality Audio with flexibility to add up to 2 microphones directly from the codec, and superior, full duplex audio with high quality stereo sound.
- Full APIs.
- Ensure successful, streamlined integration projects with standards-compliant professional connectors.
- The Codec C40 ties easily into the TANDBERG Total Solution for full management, firewall traversal and advanced services.

<table>
<thead>
<tr>
<th>Codec C40 and C60 Comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Codec</strong></td>
</tr>
<tr>
<td>Resolution</td>
</tr>
<tr>
<td>Audio inputs</td>
</tr>
<tr>
<td>Video inputs</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Video outputs</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>MultiSite</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Other</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
What’s new in this version

This section gives you an overview of the new and changed API commands for Codec C60/C40 in the TC3.1 software version.

New features and improvements

Callway
Callway is the first low cost subscription-based service for video calls. It delivers high-definition video and voice communications across the internet, connecting you with anyone who is using a standards-based video device, landline or mobile phone. Read more about Callway here: http://www.tandberg.com/callway

NAT support
With NAT (Network Address Translation) support on H.323, the TANDBERG firewall traversal technology creates a secure path through the firewall barrier, and enables proper exchange of audio/video data when connected to an external video conferencing system (when the IP traffic goes through a NAT router).

FECC on SIP
Far End Camera Control (FECC) is now supported on SIP.

Experimental settings
The Experimental settings are beta settings. These settings can be used ‘as is’, and are not fully documented.
NOTE! The Experimental settings WILL change.

New commands
Detailed descriptions of the API commands are found in this guide in the xConfiguration, xCommand, and xStatus sections.

xConfiguration
- xConfiguration Conference FarEndControl SignalCapability
- xConfiguration Conference MaxTransmitCallRate
- xConfiguration Conference MaxReceiveCallRate
- xConfiguration H323 NAT Mode
- xConfiguration H323 NAT Address
- xConfiguration Network VLAN Voice Mode
- xConfiguration Network VLAN Voice VlanId
- xConfiguration Network VLAN Voice Priority
- xConfiguration Phonebook Server Type
- xConfiguration Provisioning LoginName
- xConfiguration Provisioning Password
- xConfiguration Provisioning HttpMethod
- xConfiguration Video OSD MyContactsExpanded

xCommand
- xCommand Provisioning StartUpgrade
- xCommand Provisioning CompleteUpgrade
- xCommand Provisioning CancelUpgrade
- xCommand Video Layout AutoModeRemote SetLayoutFamily
- xCommand Video Layout AutoModeRemote Reset
- xCommand Video Layout AutoModeRemote List

xStatus
- xStatus Audio Output LocalOutput Name
- xStatus Audio Output LocalOutput Loudspeaker
- xStatus Audio Output LocalOutput Channels

Software release notes

For a complete overview of the news and changes, we recommend reading the TANDBERG TC Software Release Notes (TC3). Go to: http://www.tandberg.com/docs

User documentation

You may already be familiar with the user guides for the Codec C Series. To help you find the information you need, TANDBERG has split the guides into smaller and more accessible parts.

The Administrator Guide has been split into:
- Getting started guide
- Video conference room primer guide
- Camera user guide
- Administrator guide
- AD Drawings
- Legal and safety information guide

The System Integrator Guide has been split into:
- API guide
- Physical interfaces guide

The User Guide:
- User Guide for Codec C Series

The user documentation is available from our web site. Select your product from the drop down list to see an overview of the user documentation for that product.
Go to: http://www.tandberg.com/docs
New commands, continued...

- xStatus Audio Output LocalOutput Connector
- xStatus Audio Module Type
- xStatus Audio Module SoftwareID
- xStatus Audio Module HardwareID
- xStatus Audio Module Connector
- xStatus Call Duration
- xStatus MediaChannels Call IncomingVideoChannel Video FrameRate
- xStatus MediaChannels Call OutgoingVideoChannel Video FrameRate
- xStatus Provisioning Status
- xStatus Provisioning Reason
- xStatus Provisioning PendingUpgrade
- xStatus Video Input HDMI SignalState
- xStatus Video Input DVI SignalState
- xStatus Video Input Legacy SignalState
- xStatus Experimental Conference Site Capabilities FECC
- xStatus Experimental Audio StereoEchoCancellation Mode

Commands that have been changed

**xConfiguration**

- xConfiguration Provisioning Mode
  - CallWay has been added
- xConfiguration Time Zone
  - Added the (Coordinated Universal Time)/GMT
- xConfiguration SystemUnit MenuLanguage
  - Added the Danish and Dutch languages

Commands that are removed

**xConfiguration**

- xConfiguration Network VLAN Data Mode
- xConfiguration Network VLAN Data VlanId
- xConfiguration Network VLAN Data Priority
Chapter 2

About the TANDBERG API
Basic Principles
The heart of the API is the TANDBERG API-Engine. This is where all information is stored and processed.

The API-engine can be accessed by an easy-to-use Command Line Interface called XACLI using RS-232, Telnet or SSH, or by the TANDBERG XML API Service (TXAS) over HTTP/HTTPS.

Working with the API-engine is very similar to working with catalogues and files on a computer. All information is stored in a hierarchic tree structure which is accessible from different interfaces.

- When accessing the API-engine using XACLI (RS-232, Telnet or SSH), the information is formatted in a proprietary Command Line style or in XML formatting.
- When accessing the API-engine using the TXAS interface (HTTP/HTTPS), XML formatting is supported. This is similar to viewing files on a computer. Accessing catalogues on a Windows computer using the Command Prompt gives a different view than using Windows Explorer, but the information is the same.

About Telnet
Telnet is disabled by default. Before connecting to the codec using Telnet you will need to enable the interface via either RS-232 or SSH. The following command can be set from the Administrator settings menu or from the API command interface:

- xConfiguration NetworkServices
  Telnet Mode: On

The TANDBERG API-Engine
The TANDBERG API-Engine is optimized for easy, yet advanced, machine-machine interaction between a TANDBERG system and an external control application.

The main features can be summarized to:
1. Structuring of information
2. Addressing using XPath (XML Path Language) or TANDBERG SimplePath
3. Feedback

Structuring of Information
An application programming interface (API) can be seen as a gate where information is exchanged between two systems – a control application and a target system.

The control application transmits instructions to the target system, while the target system supplies information about how these instructions are executed, in addition to other system related information.

Consequently, the exchange of information can be divided into:
1. Information flowing from target. This we call READ information (R). The (R) should not be confused with the (r) used to indicate required parameters in the Commands tables.
2. Information flowing to target. This we call WRITE information (W).

Main types of information
If we look at the TANDBERG systems we can identify three main types of information
- READ information (R)
- WRITE information (W)
- READ/WRITE information (RW)

(R) READ information. This is Status Information about the system and system processes, i.e. information generated by the system.

Typical examples include: status about ongoing calls, network status, conference status etc. All status information is structured in a hierarchy, making up a database constantly being updated by the system to reflect process changes.

(W) WRITE information. This is Command information the user/control application supply to initiate an action.

Typical examples include: instructing the system to place a call, assigning floor to a specific site, disconnecting a call etc.

A command is usually followed by a set of parameters to specify how the given action is to be executed.

(RW) READ/WRITE information. This is Configuration Information defining system settings. This information can both be supplied and read by the user/control application. Typical examples include: default call rate, baud rate of a serial port, enabling/disabling of various features etc.

All configuration information is structured in a hierarchy making up a database of system settings. But for the Configuration information, the data in the database can only be updated by the user/control application.
Addressing Using XPath or TANDBERG SimplePath

To address information in the hierarchic structure of Status and Configuration information, the TANDBERG systems support abbreviated XML Path Language (XPath) and a proprietary notation called TANDBERG SimplePath (only available using XACLI).

This allows the user/control application to address everything from a single element of data (for example the call rate of a specific call) to larger parts of the hierarchy (for example all information available for a given call).

Using XPath

Addressing the 1st DNS Server Address of the 1st Network:
Each level is separated with a slash ("/"). Item numbers are added in brackets after the element name:

- Network[1]/DNS Server[1]/Address

Example:
```
xConfiguration Network[1]/DNS Server[1]/Address
*c xConfiguration Network 1 DNS Server 1 Address: "test"
OK
```

Using TANDBERG SimplePath

Addressing the 1st DNS Server Address of the 1st Network:
Both levels and item numbers are separated with white spaces:

- Network 1 DNS Server 1 Address

Example:
```
xConfiguration Network 1 DNS Server 1 Address
*c xConfiguration Network 1 DNS Server 1 Address: "test"
OK
```

Feedback

Feedback is an extremely powerful feature where the TANDBERG system actively returns updated status and configuration information to the user/control application whenever changes occur.

The user/control application can specify what parts of the status and configuration hierarchies it wants to monitor by using XPath. The user/control application can thereby limit the amount of information it receives from the target system to only those parts being of interest for the given application. This will also reduce the load on the link connecting the systems.

Feedback is supported on both XACLI (RS-232/Telnet/SSH) and TXAS (HTTP/HTTPS) simultaneously.

The system uses TANDBERG SimplePath when presenting configurations.

XPath and TANDBERG SimplePath are described thoroughly later in this section of the manual.

The structuring of information together with XPath and TANDBERG SimplePath for addressing, makes up powerful features as the ability to search and setting of multiple instances of a configuration.
Connecting to the codec

Accessing XACLI

XACLI can be accessed through Telnet and SSH via the LAN interface or through the COM port by connecting a serial cable to the serial interface connector, referred to as the COM port.

The COM port (RS-232) is a 9-pin, female, D-sub connector located on the back of the TANDBERG Codec C–Series. The connector is marked with the text: Camera Control.

The port is configured as a DCE (Data Communications Equipment). The COM port (RS-232) is default set to 38400 baud, 8 data bits, none parity and 1 stop bit from factory. The port may also be referred to as the Data port.

Telnet/SSH login

Telnet is by default disabled. This can be changed with a configuration command: xConfiguration NetworkServices Telnet Mode: On/Off

- xConfiguration NetworkServices Telnet Mode: On

Telnet/SSH login

- User name is: admin
- Default password is: TANDBERG

Serial port login

The serial port is password protected by default. The password protection may be configured.

- User name is: admin
- Default password is: TANDBERG

Serial port configurations

On the serial port the baud rate and password protection may be configured.

The configuration command for the baud rate is:

xConfiguration SerialPort BaudRate: <9600/19200/38400/57600/115200>

- xConfiguration SerialPort BaudRate: 38400

The configuration command for login required is:

xConfiguration SerialPort LoginRequired: <On/Off>

- xConfiguration SerialPort LoginRequired: On

Reboot

The system requires a reboot for the changes to baud rate and password protection to take effect.

NOTE: When system boots up the baud rate of the boot messages is 38400 regardless of the baud rate set in the codec application.

Hardware & Cabling (RS-232)

The pin outs for the RS-232 are defined in the tables to the right. Observe that the DTE (Data Terminal Equipment), could be a PC or any other device capable of serial communication.

Cable. A straight-through cable should be used between the TANDBERG RS-232 port and the DTE. The lower table shows the recommended cable-wiring scheme when connecting the Codec C–Series to a PC through RS-232.

DTR and RTS are ignored. DSR, CD, and CTS are always asserted, while RI is not used.

Troubleshooting (RS-232)

If communication cannot be established between the PC/terminal and the TANDBERG Codec data port, the following should be checked:

1. Verify that the serial cable is a straight-through 9-pin to 9-pin cable.
2. Confirm that the configuration of the PC/terminal’s serial RS-232 port is identical to the configuration of the TANDBERG RS-232 port.
3. Verify that the PC/terminal’s serial RS-232 port is working properly by connecting it back-to-back to another PC/terminal and send characters in both directions.

| COM port (RS-232) |
|---|---|---|
| Pin | Signal name | Direction |
| 1 | Carrier detect, CD | From DCE |
| 2 | Receive data, RXD | From DCE |
| 3 | Transmit data, TXD | To DCE |
| 4 | Data terminal ready, DTR | From DCE |
| 5 | Signal GND | |
| 6 | Data set ready, DSR | From DCE |
| 7 | Ready to send, RTS | To DCE |
| 8 | Clear to send, CTS | From DCE |
| 9 | Ring indicator, RI | From DCE |

| Cable wiring (RS-232) TANDBERG DCE <-> PC |
|---|---|---|
| TANDBERG DCE 9 pin | Direction | PC DTE, 9 pin |
| 1 CD | --> | 1 CD |
| 2 RD | --> | 2 RD |
| 3 TD | <-- | 3 TD |
| 4 DTR | <-- | 4 DTR |
| 5 GND | <-- | 5 GND |
| 6 DSR | --> | 6 DSR |
| 7 RTS | <-- | 7 RTS |
| 8 CTS | --> | 8 CTS |
| 9 RI | --> | 9 RI |
**Value types and formats**

The system supports the following value types:

- Integer values
- Literal values
- String values
- E164 string values (strings only containing digits, '#' and '*')
- IPv4 Address values
- *IPv6 Address values
- IPv4 or *IPv6 Address values

### Formats for values types

<table>
<thead>
<tr>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Integer values:</strong> &lt;x..y&gt;</td>
<td>Defines the valid range for an integer input. x = min value, y = max value.</td>
</tr>
<tr>
<td>&lt;1..100&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>Literal values:</strong> &lt;X/Y/..Z&gt;</td>
<td>Defines the possible values for a given configuration.</td>
</tr>
<tr>
<td>&lt;On/Off/Auto&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>String values:</strong> &lt;S: x, y&gt;</td>
<td>Defines that the valid input for this configuration is a String with minimum length x and maximum length of y characters.</td>
</tr>
<tr>
<td>&lt;S: 0, 49&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>IPv4 Address values:</strong> &lt;IPAddr&gt;</td>
<td>Defines that the input must be an IPv4 address.</td>
</tr>
<tr>
<td>&lt;IPAddr&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>IPv6 Address values:</strong> &lt;IPv6Addr: x, y&gt;</td>
<td>Defines that the input must be an IPv6 address with minimum length x and maximum length y.</td>
</tr>
<tr>
<td>&lt;IPv6Addr: 0, 43&gt;</td>
<td></td>
</tr>
<tr>
<td><strong>IPv4 or IPv6 Address values:</strong> &lt;IPv4v6Addr: x, y&gt;</td>
<td>Defines that the input must be an IPv4 or IPv6 address with minimum length x and maximum length y.</td>
</tr>
<tr>
<td>&lt;IPv4v6Addr: 0, 43&gt;</td>
<td></td>
</tr>
</tbody>
</table>

* Not supported in this version
Root commands

By typing `?` or `help` after connecting to the TANDBERG Codec C-Series using RS-232/Telnet/SSH, the system will list all supported root commands.

```
? - User Commands -
  help    xconfiguration    xfeedback    xgetxml    xhistory
  xstatus  xpreferences     xcommand    log

OK
```
Main type of commands

The XACLI is divided into three main types of commands, reflecting the information types supported by the TANDBERG API Engine. The main types are:

- Configuration type commands
- Status type commands
- Command type commands

Configuration type commands
Configuration type commands defines the system settings. Configuration type commands are either supplied or read by the user.

Example: Set default call rate, baud rate of a serial port and enabling/disabling of various features etc.

The configuration commands are structured in a hierarchy, making up a database of system settings.

Supported Configuration-type commands:
- xConfiguration

Command type commands
Command type commands instructs the system to perform an action. Command type commands are supplied by the user.

Example: instructing the system to place a call, assign floor to a specific site, disconnect a call etc.

A Command type command is usually followed by a set of parameters to specify how the given action is to be executed.

Supported Command-type commands:
- xCommand

Status type commands
Status type commands returns information about the system and system processes. Status type commands are read by the user.

Example: Information generated by the system about ongoing calls, network status, conference status etc.

All status information is structured in a hierarchy, making up a database constantly being updated by the system to reflect system and process changes.

Supported Status-type commands:
- xStatus
- xHistory

Special commands
In addition to the above sets of commands, XACLI supports the following set of special commands:

- Feedback type command
  The xFeedback command is used to specify what parts of the configuration and status hierarchies to monitor. Feedback will only be issued on the RS-232/Telnet/SSH session for which it is specified. If connecting to the TANDBERG codec with multiple sessions, each session can define feedback individually.

  More on this can be found in xfeedback.
  - xFeedback

- Preferences type command
  The xPreferences command is used to set various preferences for the RS-232/Telnet/SSH sessions. Each session can define preferences individually. IMPORTANT! This command has various settings to define the formatting of the XACLI output. It is therefore important to define settings to match the parser used on the control system. XACLI is designed to make parsing of data from the TANDBERG Codec C–Series very simple.

  More on this can be found in xpreferences.
  - xPreferences
About xConfiguration

The xConfiguration type commands define the system settings and are either supplied or read by the user. The xConfiguration commands are organized in a hierarchic tree structure.

To get an overview of accessible top-level configuration elements within the xConfiguration commands, enter `xConfiguration ?` or `xConfiguration help` after the xConfiguration command:

- `xConfiguration ?`
- `xConfiguration help`

To get an overview of all supported xConfiguration commands with the corresponding value space, enter `xConfiguration ??` after the xConfiguration command:

- `xConfiguration ??`

When issuing a xConfiguration command, the command consists of three parts:

1. The type of command: xConfiguration
2. The path: An address expression, terminated by a colon
3. The value: A value type

Example: `xConfiguration Audio Input HDMI 1 Mode: On`

The type
The path
The value

xConfiguration ?

- User Configurations -

AudioGPIO
CamerasH323
ConferenceNetwork
ExperimentalNetworkServices

OK

xConfiguration ??

*h xConfiguration Audio Volume: <0..100>
Sets the volume level [0-100] on the loudspeaker output in steps of 0.5dB from -34.5dB to 15dB.
Volume 0 = Off. The volume level bar which is displayed on screen, when using the remote control, goes from 0 to 20. Range: The volume level goes from 0 to 100. Volume level equals Audio gain value 0 equals 0 1 equals -34.5 dB 70 equals 0.0 dB 100 equals 15.0 dB

*h xConfiguration Audio Input Microphone [1..8] Type: <Microphone/Line>
The microphone inputs are intended for electret type microphones. The microphone inputs are balanced with 48 V phantom power. The microphone input can be set to line or microphone mode. Addresses the specific microphone. Microphone: Phantom voltage and pre-amplification is On. Line: Select Line when you have a standard balanced line input. The phantom voltage and pre-amplification is Off.

*h xConfiguration Audio Input Microphone [1..8] Mode: <On/Off>
By default, all inputs are enabled. Just plug in an audio source and it is active. Audio inputs that are On will automatically be mixed. Unconnected inputs will automatically be muted. Addresses the specific microphone. On: Turns the microphone On. Off: Connected but unused inputs should be set to Off to prevent audio/noise from the inputs.

OK
**xConfiguration operations**

Configuration type commands define system settings and are either supplied or read by the user.

**Return result parameters**

Three operations can be performed on xConfiguration:

- **xConfiguration Help**
  - Help text for this configuration is returned

- **xConfiguration Read**
  - \*c is used when returning the result of a read query

- **xConfiguration Write**
  - No return result parameter for configuration set (write)
  - Writes this value to the setting defined by the path.

**Example with xConfiguration Help:**

To get help on xConfiguration, type `?` or `help` after the configuration path (address expression):

```
xConfiguration <address expression> ?
xConfiguration H323 Profile 1 Gatekeeper Discovery ?
  \*h xConfiguration H323 Profile 1 Gatekeeper Discovery: <Manual/Auto>
  OK
```

**Example with xConfiguration Read:**

To read configurations from the system just type the root command (xConfiguration) followed by the path (address expression):

```
xConfiguration <address expression>
xConfiguration H323 Profile 1 Gatekeeper Discovery
  \*c xConfiguration H323 Profile 1 Gatekeeper Discovery: Manual
  OK
```

**Example with xConfiguration Write:**

To issue a command type a root command (xConfiguration) followed by a valid path (address expression). The path must be terminated with a colon before the value is added:

```
xConfiguration <address expression>: <value>
xConfiguration H323 Profile 1 Gatekeeper Discovery: Auto
  ** end
```
About xCommand

xCommand type commands instructs the system to perform an action. xCommand type commands are supplied by the user.

To get an overview of the supported xCommand type commands, type `?` or `help` after the `xCommand`:
- `xCommand ?`
- `xCommand help`

To get an overview of all supported xCommand commands with the corresponding value space, enter `??` after the `xCommand`:
- `xCommand ??`

When you type a command and `?` or `help` a list of the available parameters will show. Required parameters are identified by an `(r)` behind the parameter name.

```
xCommand ?
  - User Commands -
    Audio  Dial
    Boot  DTMFSend
    Call  Experimental
    CallLog  FarEndControl
    CamCtrlPip  GPIO
    Camera  HTTPFeedback

OK
```

```
xCommand ??
  *h xCommand Audio Microphones Mute
  *h xCommand Audio Microphones Unmute
  *h xCommand Audio Sound Play
    Sound(r): <Busy/CallWaiting/Dial/KeyTone/Ringing/SpecialInfo/TelephoneCall/VideoCall>
    Loop: <On/Off>
  *h xCommand Audio Sound Stop
  *h xCommand Audio Vumeter Start
    ConnectorType(r): <HDMI/Line/Microphone>
    ConnectorId(r): <1..8>
  *h xCommand Audio Vumeter Stop
    ConnectorType(r): <HDMI/Line/Microphone>
    ConnectorId(r): <1..8>
  *h xCommand Audio Setup Clear
    .
    .
    .
  OK
```
xCommand operations

Command type commands are used to instruct the system to perform a given action.

Return result parameters

The following operations can be performed on xCommand:

**Command Help**

- *h is used when returning the result of a help query

**Command Write**

- *r is used when returning the result of a write command

xCommand Help

To get help on a setting you can use a help query. Enter the path followed by ? or help.

- xCommand dial ?
  Returns a set of return values. See the example to the right.
- xCommand dial help
  As above.

Example with xCommand Help

To get help on xCommand, type ? or help after the command path (address expression):

```
xCommand <address expression> ?
```

```
xCommand Dial ?
*h xCommand Dial
   Number(r): <S: 0, 255>
   Protocol: <H323/Sip>
   CallRate: <64..6000>
   CallType: <Audio/Video>
   ForceNewConference: <False/True>
   ConferenceId: <0..65534>
OK
```

Example with xCommand Write

Dial a number with only the required parameter:

```
xCommand Dial Number: 95458458
```

```
OK
*r DialResult (status=OK):
   CallId: 2
   ConferenceId: 1
*r/end
OK
```
About xStatus commands

Status type commands returns information about the system and system processes. Status type commands are read by the user. All status information is structured in a hierarchy, making up a database constantly being updated by the system to reflect system and process changes.

To get an overview of the supported xStatus type commands, type `xStatus ?` or `help` after the `xStatus`:

- `xStatus ?`
- `xStatus help`

Return result parameters

The following operation can be performed on xStatus commands:

**xStatus Read**

- `*s` is used when returning the result of xStatus read query

<table>
<thead>
<tr>
<th>xStatus ?</th>
<th>- Status -</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>Experimental</td>
</tr>
<tr>
<td>Call[...]</td>
<td>GPIO</td>
</tr>
<tr>
<td>CallTransfer[...]</td>
<td>H323</td>
</tr>
<tr>
<td>Camera[1..7]</td>
<td>HTTPFeedback[1..3]</td>
</tr>
<tr>
<td>Conference</td>
<td>MediaChannels</td>
</tr>
<tr>
<td>Diagnostics</td>
<td>Network</td>
</tr>
</tbody>
</table>

OK
Query status information
Status type commands returns information about the system and system processes. You can query all information or just some of it.

To address status information enter the xStatus command followed by an address expression (XPath or TANDBERG SimplePath).

You can set up the xStatus read command to address all information or just some of it, see the examples to the right for illustrations.

Address status information with xStatus
To read status from the system just type the root command (xStatus) followed by the path (address expression):

Example 1: Query all ongoing Call information:

```
xstatus call
  *s Call 3 Status: Connected
  *s Call 3 Direction: Outgoing
  *s Call 3 Protocol: “h323”
  *s Call 3 RemoteNumber: ”firstname.lastname@company.com”
  *s Call 3 CallbackNumber: ”h323:firstname.lastname@company.com”
  *s Call 3 DisplayName: ”firstname.lastname@company.com”
  *s Call 3 TransmitCallRate: 768
  *s Call 3 ReceiveCallRate: 768
  *s Call 3 Encryption Type: ”None”
  *s Call 3 PlacedOnHold: False
  *s Call 3 Duration: 9
** end
```

OK

Example 2: Query the protocol for a call:

```
xstatus call protocol
  *s Call 3 Protocol: ”h323”
```

OK
About xHistory command

History type commands returns information about what has happened on the system. History type commands are read by the user.

All history information is structured in a hierarchy, making up a database constantly being updated by the system to reflect system and process changes.

To get an overview of the supported xHistory type commands, type ? or help after the xHistory:

- xHistory ?
- xHistory help

Status operations and the return result parameters

xHistory Log
- *h is used when returning the result of xHistory log query

Example with xHistory CallLogs

xhistory
*h xHistory CallLogs Call 1 CallId: 13
*h xHistory CallLogs Call 1 Protocol: "h323"
*h xHistory CallLogs Call 1 Direction: Outgoing
*h xHistory CallLogs Call 1 CallType: Video
*h xHistory CallLogs Call 1 RemoteNumber: "h323:firstname.lastname.office@company.com"
*h xHistory CallLogs Call 1 CallbackNumber: "h323:firstname.lastname.office@company.com"
*h xHistory CallLogs Call 1 DisplayName: "firstname.lastname@company.com"
*h xHistory CallLogs Call 1 CallRate: 768
*h xHistory CallLogs Call 1 DisconnectCauseValue: 1
*h xHistory CallLogs Call 1 DisconnectCause: "MC:Normal"
*h xHistory CallLogs Call 1 StartTime: "2010/04/14 11:04:14"
*h xHistory CallLogs Call 1 Duration: 184
*h xHistory CallLogs Call 1 Encryption: "None"
...
*h xHistory CallLogs Recent 6 CounterMissed: 0
*h xHistory CallLogs Recent 7 CounterMissed: 0
...
*h xHistory CallLogs Outgoing 29 Counter: 1
*h xHistory CallLogs Outgoing 30 Counter: 1
...
*h xHistory CallLogs Missed 50 Counter: 2
** end
About xFeedback

The xFeedback is a powerful feature on the Codec C90/C60/C40. It lets you subscribe to what you want to be notified about when changes occur on the system:

- This can be configuration changes like someone changes the name of the system.
- It might be events like key press from the remote control.
- Or it can be changes to the state of the system, like a call connecting or disconnecting.

The xFeedback command is used to specify what parts of the configuration and status hierarchies to monitor, and will only be issued on the RS-232/Telnet/SSH for which it is specified.

If connecting to the TANDBERG codec with multiple sessions, each session can define feedback individually.

CAUTION: We discourage registering all status changes as this may give too much feedback information than the control systems are able to handle.

Example with xFeedback

xFeedback ?
xFeedback help:
xFeedback Register XPathExpression - Registers feedback on expression XPathExpression
xFeedback Deregister XPathExpression - Deregisters feedback if registered on XPathExpression
xFeedback List - Generate list of currently registered XPathExpressions
xFeedback Help - Display this help text

xFeedback register Status/Audio
xFeedback register Configuration/Video
xFeedback register Event

xFeedback list
xFeedback deregister Event
xFeedback list
About xPreferences

The xPreferences command is used to set various preferences for the RS-232/Telnet/SSH sessions.

Each session can define preferences individually.

IMPORTANT! This command has various settings to define the formatting of the XACLI output. It is therefore important to define settings to match the parser used on the control system. XACLI is designed to make parsing of data from the TANDBERG Codec C–Series very simple.

To get an overview of the supported xPreferences commands and their value space, type ? or help after the xPreferences:

- xPreferences ?
- xPreferences help

The xPreferences output modes

- Terminal: Line based XACLI output for use with line based control systems
- XML: Pure XML output for use with control systems that understand XML. NOTE! This mode is to be considered experimental in version 1 of the software. Its format WILL change in next version.

```bash
xPreferences ?
xpreferences usage:
xpreferences outputmode <terminal/xml>
OK
```
TANDBERG XML API service

TXAS is a service provided by TANDBERG units for transmitting and receiving (transceiving) information encoded in XML format. The API uses HTTP(S) as the transport mechanism and connects to the normal web port (80). TXAS can be accessed by bare-bone HTTP requests where URL's uniquely identifies the request.

Bare-bone HTTP/HTTPS Access

The bare-bone HTTP mode uses a unique URL to identify the specific request. The contents of the HTTP body will be a XML document (or part of it).

Bare-bone HTTP(S) access is accomplished by passing arguments in the query string (after '?' in URL) in a GET request, or using the "application/x-www-form-urlencoded" content-type method of POSTing form data (Each argument starts with a name '=' and a value, and every parameter separated with '&' (and opt NL).)

```
/getxml
```

/getxml request returns an XML document based on the location parameter passed to the request. The elements (or complete document) matching the expression will be returned.

On incorrect XPath expression, a <Fault> element with a <XPathError> element will be returned.

```
getxml
REQUEST:
   /getxml
PARAM:
    location = XPath expression
```

```
/formputxml
```

This is most useful in a POST (to extend character limit of 255 of GET urls). It posts a Configuration or Command document to set the configurations or issue a command.

Like getxml, it has the data URL form-data encoded with one single parameter. The Content-Type of the document must be of type "application/x-www-form-urlencoded" and the body must be encoded accordingly (e.g. first line will be xmldoc=<then the document>).

```
/formputxml
REQUEST:
   /formputxml
PARAM:
    xmldoc   = "an XML document of Configuration, Directory or Command"
```

```
/putxml
```

Putxml is like formputxml+, put uses the complete BODY as argument (i.e. the content of the xmldoc parameter). The Content-type should be "text/xml" or "application/xml" (or "text/plain"), though no check at the moment. (Except for application/x-www-form-urlencoded which will cause a failure).

```
/putxml
REQUEST:
   /putxml
PARAM:
    HTTP BODY as argument
```
Chapter 3

Description of the xConfiguration commands
Description of the xConfiguration commands

In the following pages you will find a complete list of the xConfiguration commands. The examples shows either the default value or an example of a value.

We recommend you visit the TANDBERG web site regularly for updated versions of the manual.

Go to: http://www.tandberg.com/docs

The Audio settings

xConfiguration Audio Input HDMI [2] Level
Define the audio level of the HDMI input connector, in steps of 1dB.
See the Audio Level tables in the physical interfaces guide for the codec for a complete overview of the menu values represented in dB.

Value space: <-24..0>

Range: Select a value from -24 to 0dB.

Example: xConfiguration Audio Input HDMI 2 Level: 0

Determine whether or not the audio channels on the HDMI input should be enabled. The HDMI input 2 has two audio channels.

Value space: <On/Off>

On: Enable the audio channels on the HDMI input.
Off: Disable the audio channels on the HDMI input.

Example: xConfiguration Audio Input HDMI 2 Mode: On

xConfiguration Audio Input HDMI [2] VideoAssociation MuteOnInactiveVideo
Enable association of a video source to a HDMI audio input.

Value space: <On/Off>

On: A video source is associated, and the audio will be muted if the associated video source is not displayed.
Off: No video source is associated.

Example: xConfiguration Audio Input HDMI 2 VideoAssociation MuteOnInactiveVideo: Off

The Audio settings, continued...

xConfiguration Audio Input HDMI [2] VideoAssociation VideoInputSource
Select the associated video input source.

Value space: <1/2/3>

Range: Select one of the three video input sources.

Example: xConfiguration Audio Input HDMI 2 VideoAssociation VideoInputSource: 1

xConfiguration Audio Input Line [1..2] Equalizer ID
Select the audio input line equalizer ID.

Value space: <1..8>

Range: Select Equalizer ID 1 to 8.

Example: xConfiguration Audio Input Line 1 Equalizer ID: 1

xConfiguration Audio Input Line [1..2] Equalizer Mode
Set the audio input line equalizer mode.

Value space: <On/Off>

On: Enable the equalizer for the audio input line.
Off: No equalizer.

Example: xConfiguration Audio Input Line 1 Equalizer Mode: Off

xConfiguration Audio Input Line [1..2] VideoAssociation MuteOnInactiveVideo
Enable association of a video source to a Line audio input.

Value space: <On/Off>

On: A video source is associated, and the audio will be muted if the associated video source is not displayed.
Off: No video source is associated.

Example: xConfiguration Audio Input Line 1 VideoAssociation MuteOnInactiveVideo: Off

xConfiguration Audio Input Line [1..2] VideoAssociation VideoInputSource
Select the associated video input source.

Value space: <1/2/3>

Range: Select one of the three video input sources.

Example: xConfiguration Audio Input Line 1 VideoAssociation VideoInputSource: 1
The Audio settings, continued...

**xConfiguration Audio Input Line [1..2] Channel**
Define whether the Audio Line input is a mono signal or part of a multichannel signal.
- **Value space:** <Left/Right/Mono>
  - **Left:** The Audio Line input signal is the left channel of a stereo signal.
  - **Right:** The Audio Line input signal is the right channel of a stereo signal.
  - **Mono:** The Audio Line input signal is a mono signal.
- **Example:** `xConfiguration Audio Input 1 Channel: Left`

**xConfiguration Audio Input Line [1..2] Level**
Define the audio level of the Line input connector, in steps of 1dB.
See the Audio Level tables in the physical interfaces guide for the codec for a complete overview of the values represented in dB.
- **Value space:** <0..24>
  - **Range:** Select a value from 0 to 24dB.
- **Example:** `xConfiguration Audio Input Line 1 Level: 10`

**xConfiguration Audio Input Line [1..2] LoopSuppression**
NOTE! Codec C40/C60 does currently not support Loop Suppression, hence Loop Suppression can be set to Off only.
- **Value space:** <Off>
  - **Off:** Deactivate Loop Suppression.
- **Example:** `xConfiguration Audio Input Line 1 LoopSuppression: Off`

**xConfiguration Audio Input Line [1..2] Mode**
Set the audio input line mode.
- **Value space:** <On/Off>
  - **On:** Enable the Audio Line input.
  - **Off:** Disable the Audio Line input.
- **Example:** `xConfiguration Audio Input Line 1 Mode: On`

---

The Audio settings, continued...

**xConfiguration Audio Input Microphone [1..2]/[1..4] EchoControl Mode**
NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.
The echo canceller continuously adjusts itself to the audio characteristics of the room and compensates for any changes it detects in the audio environment. If the changes in the audio conditions are very significant the echo canceller may take a second or two to re-adjust.
- **Value space:** <On/Off>
  - **On:** Echo Control is normally set to On to prevent the far end from hearing their own audio. Once selected, echo cancellation is active at all times.
  - **Off:** Echo Control should be switched Off if external echo cancellation or playback equipment is used.
- **Example:** `xConfiguration Audio Input Microphone 1 EchoControl Mode: On`

**xConfiguration Audio Input Microphone [1..2]/[1..4] EchoControl NoiseReduction**
NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.
The system has a built-in noise reduction which reduces constant background noise (e.g. noise from air-conditioning systems, cooling fans etc.). In addition, a high pass filter (Humfilter) reduces very low frequency noise. NOTE! Requires the Echo Control Mode to be enabled for the microphone.
- **Value space:** <On/Off>
  - **On:** The Noise Reduction should be enabled in the presence of low frequency noise.
  - **Off:** Turn off the Noise Reduction.
- **Example:** `xConfiguration Audio Input Microphone 1 EchoControl NoiseReduction: On`

**xConfiguration Audio Input Microphone [1..2]/[1..4] Equalizer ID**
NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.
Select the audio input microphone equalizer ID.
- **Value space:** <1..8>
  - **Range:** Select Equalizer ID 1 to 8.
- **Example:** `xConfiguration Audio Input Microphone 1 Equalizer ID: 1`

**xConfiguration Audio Input Microphone [1..2]/[1..4] Equalizer Mode**
NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.
Set the audio input microphone equalizer mode.
- **Value space:** <On/Off>
  - **On:** Enable the equalizer for the audio input microphone.
  - **Off:** No equalizer.
- **Example:** `xConfiguration Audio Input Microphone 1 Equalizer Mode: Off`
The Audio settings, continued...

**xConfiguration Audio Input Microphone [1..2]/[1..4] VideoAssociation**

**MuteOnInactiveVideo**

NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.

Enable association of a video source to a microphone audio input.

Value space: `<On/Off>`

- **On**: A video source is associated, and the audio will be muted if the associated video source is not displayed.
- **Off**: No video source is associated.

**Example:**
```
xConfiguration Audio Input Microphone 1 VideoAssociation MuteOnInactiveVideo: On
```

**xConfiguration Audio Input Microphone [1..2]/[1..4] VideoAssociation**

**VideoInputSource**

NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.

Select the associated video input source.

Value space: `<1/2/3>`

**Range:** Select one of the three video input sources.

**Example:**
```
xConfiguration Audio Input Microphone 1 VideoAssociation VideoInputSource: 1
```

**xConfiguration Audio Input Microphone [1..2]/[1..4] Level**

NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.

Define the audio level of the Microphone input connector, in steps of 1dB.

See the Audio Level tables in the physical interfaces guide for the codec for a complete overview of the values represented in dB.

Value space: `<0..24>`

**Range:** Select a value from 0 to 24dB.

**Example:**
```
xConfiguration Audio Input Microphone 1 Level: 14
```

**xConfiguration Audio Input Microphone [1..2]/[1..4] Mode**

NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.

Set the audio input microphone mode.

Value space: `<On/Off>`

- **On**: Enable the microphone connector.
- **Off**: Disable the microphone connector.

**Example:**
```
xConfiguration Audio Input Microphone 1 Mode: On
```

**xConfiguration Audio Input Microphone [1..2]/[1..4] Type**

NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors.

The microphone connectors are intended for electret type microphones. The microphone connector can be set to line or microphone mode.

Value space: `<Microphone/Line>`

- **Microphone**: Select Microphone when you have 48 V Phantom voltage and the pre-amplification is On.
- **Line**: Select Line when you have a standard balanced line input. The phantom voltage and pre-amplification is Off.

**Example:**
```
xConfiguration Audio Input Microphone 1 Type: Line
```

The Audio settings, continued...

**xConfiguration Audio Output HDMI [1] Level**

Define the output level of the HDMI output connector, in steps of 1dB.

See the Audio Level tables in the physical interfaces guide for the codec for a complete overview of the values represented in dB.

Value space: `<-24..0>`

**Range:** Select a value from -24 to 0dB.

**Example:**
```
xConfiguration Audio Output HDMI 1 Level: 0
```

**xConfiguration Audio Output HDMI [1] Mode**

Determine whether or not the audio channel on the HDMI output should be enabled.

Value space: `<On/Off>`

- **On**: Enable the audio channel on the HDMI output.
- **Off**: Disable the audio channel on the HDMI output.

**Example:**
```
xConfiguration Audio Output HDMI 1 mode: On
```

**xConfiguration Audio Output Line [1..2] Channel**

Define whether the Audio Line output is a mono signal or part of a multichannel signal.

Value space: `<Left/Right/Mono>`

- **Left**: The Audio Line output signal is the left channel of a stereo signal.
- **Right**: The Audio Line output signal is the right channel of a stereo signal.
- **Mono**: The Audio Line output signal is a mono signal.

**Example:**
```
xConfiguration Audio Output Line 1 Channel: Left
```

The Audio settings, continued...
The Audio settings, continued...

xConfiguration Audio Output Line [1..2] Equalizer ID
Select the audio output line equalizer ID.
Value space: `<1..8>`
  Range: Select Equalizer ID 1 to 8.
Example: xConfiguration Audio Output Line 1 Equalizer ID: 1

xConfiguration Audio Output Line [1..2] Equalizer Mode
Set the audio output line equalizer mode.
Value space: `<On/Off>`
  On: Enable the equalizer for the audio output line.
  Off: No equalizer.
Example: xConfiguration Audio Output Line 1 Equalizer Mode: Off

xConfiguration Audio Output Line [1..2] Level
Define the output level of the Audio Output Line connector, in steps of 1dB.
See the Audio Level tables in the physical interfaces guide for the codec for a complete overview of the values represented in dB.
Value space: `<-24..0>`
  Range: Select a value from -24 to 0dB.
Example: xConfiguration Audio Output Line 1 Level: -10

xConfiguration Audio Output Line [1..2] Mode
Set the audio output line mode.
Value space: `<On/Off>`
  On: Enable the Audio Line output.
  Off: Disable the Audio Line output.
Example: xConfiguration Audio Output Line 1 Mode: On

xConfiguration Audio Output Line [1..2] Type
Determines if the Audio Line output is an analog or digital type output.
Value space: `<Auto/SPDIF>`
  Auto: If a TANDBERG Digital NAM is detected then SPDIF mode will be selected, otherwise analog mode will be selected.
  SPDIF: Set to SPDIF when you want the line output to be in digital mode.
Example: xConfiguration Audio Output Line 1 Type: Auto

The Audio settings, continued...

xConfiguration Audio Output Line [2] Type
Line output 2 is a dedicated analog output, hence type can be set to analog only.
Value space: `<Analog>`
  Analog: Can be set to analog only.
Example: xConfiguration Audio Output Line 1 Type: Analog

xConfiguration Audio SoundsAndAlerts KeyTones Mode
The system can produce a sound every time a key on the remote control is pressed.
Value space: `<On/Off>`
  On: There will be a sound indicator when pressing keys on the remote control.
  Off: The key tone on the remote control is switched off.
Example: xConfiguration Audio SoundsAndAlerts KeyTones Mode: Off

xConfiguration Audio SoundsAndAlerts RingTone
Select the ringtone for incoming calls.
Value space: `<Marbles/IceCrystals/Polaris/Alert/Discrete/Fantasy/Jazz/Nordic/Echo/Rhythmic>`
  Range: Select a tone from the list of ringtones.
Example: xConfiguration Audio SoundsAndAlerts RingTone: Jazz

xConfiguration Audio SoundsAndAlerts RingVolume
Set the ring tone volume for an incoming call. The value goes in steps of 5 from 0 to 100 (from -34.5dB to 15dB). Volume 0 = Off.
Value space: `<0..100>`
  Range: Select a value from 0 to 100.
Example: xConfiguration Audio SoundsAndAlerts RingVolume: 50

xConfiguration Audio Volume
Set the volume on the loudspeaker. The value goes in steps of 5 from 0 to 100 (from -34.5dB to 15dB). Volume 0 = Off.
Value space: `<0..100>`
  Range: Select a value from 0 to 100.
Example: xConfiguration Audio Volume: 70
The Camera settings

**xConfiguration Cameras PowerLine Frequency**
Applies to cameras supporting PowerLine frequency anti-flickering, i.e PrecisionHD 1080p cameras.
Value space: <Auto/50Hz/60Hz>
- **Auto**: Set to Auto to enable power frequency auto detection in the camera.
- **50Hz/60Hz**: Set to 50Hz or 60Hz.

*Example:* xConfiguration Cameras PowerLine Frequency: Auto

**xConfiguration Cameras Camera [1..7] Backlight**
The backlight functionality compensates for lights shining directly at the camera (usually the sun entering the window) to avoid a too dark image from the room.
Value space: <On/Off>
- **On**: Turn on the camera backlight.
- **Off**: Turn off the camera backlight.

*Example:* xConfiguration Cameras Camera 1 Backlight: Off

**xConfiguration Cameras Camera [1..7] Brightness Mode**
Define whether to control the camera brightness manually or to have it automatically adjusted by the system.
Value space: <Auto/Manual>
- **Auto**: The camera brightness is automatically set by the system.
- **Manual**: Enable manual control of the camera brightness, e.g. the level of the brightness level setting will be used for the camera.

*Example:* xConfiguration Cameras Camera 1 Brightness Mode: Auto

**xConfiguration Cameras Camera [1..7] Brightness Level**
Set the brightness level. **Note!** Requires the Brightness Mode to be set to Manual.
Value space: <1..31>
- **Range**: Select a value from 1 to 31.

*Example:* xConfiguration Cameras Camera 1 Brightness Level: 1

The Camera settings, continued...

**xConfiguration Cameras Camera [1..7] Flip**
With Flip mode (vertical flip) you can flip the image upside down.
Value space: <Auto/On/Off>
- **Auto**: When the camera is placed upside down the image is automatically flipped upside down. Use this setting with cameras that can be mounted upside down, and that can auto detect that the camera is mounted upside down.
- **On**: When enabled the video on screen is flipped. This setting is used with cameras that can be mounted upside down, but cannot auto detect that the camera is mounted upside down.
- **Off**: Display the video on screen the normal way.

*Example:* xConfiguration Cameras Camera 1 Flip: Off

**xConfiguration Cameras Camera [1..7] Focus Mode**
Set the camera focus mode.
Value space: <Auto/Manual>
- **Auto**: When set to Auto the focus will be updated throughout the call. When moving the camera, the system will use auto focus for a few seconds to set the right focus of the new camera position. After a few seconds auto focus is turned off to prevent continuous focus adjustments of the camera.
- **Manual**: Turn the autofocus off and adjust the camera focus manually.

*Example:* xConfiguration Cameras Camera 1 Focus Mode: Auto

**xConfiguration Cameras Camera [1..7] Gamma Mode**
Applies to cameras which supports Gamma mode. The Gamma Mode setting enables for gamma corrections. Gamma describes the nonlinear relationship between image pixels and monitor brightness. The TANDBERG PrecisionHD 720p camera supports Gamma Mode. Not supported by TANDBERG PrecisionHD 1080p camera.
Value space: <Auto/Manual>
- **Auto**: Auto is the default and the recommended setting.
- **Manual**: In severe light conditions, you may switch mode to manual and specify explicitly which gamma table to use by setting the Gamma Level.

*Example:* xConfiguration Cameras Camera 1 Gamma Mode: Auto

**xConfiguration Cameras Camera [1..7] Gamma Level**
Set the brightness level. **Note!** Requires the Brightness Mode to be set to Manual.
Value space: <0..7>
- **Range**: Select a value from 0 to 7.

*Example:* xConfiguration Cameras Camera 1 Gamma Level: 0
The Camera settings, continued...

**xConfiguration Cameras Camera [1..7] IrSensor**

The IR sensor LED is located in the front of the camera and flickers when the IR sensor is activated from the remote control. Both the TANDBERG camera and codec has an IR sensor. You would normally choose just one of them to be active at a time.

Value space:  <On/Off>

- **On**: Enable the IR sensor on the camera.
- **Off**: Disable the IR sensor on the camera.

Example: xConfiguration Cameras Camera 1 IrSensor: On

**xConfiguration Cameras Camera [1..7] Mirror**

With Mirror mode (horizontal flip) you can mirror the image on screen.

Value space:  <Auto/On/Off>

- **Auto**: When the camera is placed upside down the image is automatically mirrored. Use this setting with cameras that can be mounted upside down, and that can auto detect that the camera is mounted upside down.
- **On**: See the selfview in mirror mode, e.g. the selfview is reversed and the experience of selfview is as seeing yourself in a mirror.
- **Off**: See the selfview in normal mode, e.g. the experience of selfview is as seeing yourself as other people see you.

Example: xConfiguration Cameras Camera 1 Mirror: Off

**xConfiguration Cameras Camera [1..7] Whitebalance Mode**

Define whether to control the camera whitebalance manually or to have it automatically adjusted by the system.

Value space:  <Auto/Manual>

- **Auto**: When set to Auto, the camera will continuously adjust the whitebalance depending on the camera view.
- **Manual**: Set to Manual to enable manual control of the camera whitebalance, e.g. the level of the whitebalance level setting will be used for the camera.

Example: xConfiguration Cameras Camera 1 Whitebalance Mode: auto

The Camera settings, continued...

**xConfiguration Cameras Camera [1..7] Whitebalance Level**

Set the whitebalance level. NOTE! Requires the Whitebalance Mode to be set to manual.

Value space:  <1..16>

- **Range**: Select a value from 1 to 16.

Example: xConfiguration Cameras Camera 1 Whitebalance Level: 1

**xConfiguration Cameras Camera [1..7] DHCP**

Applies to cameras which supports DHCP. The TANDBERG PrecsionHD 1080p camera supports DHCP. The camera must be connected to a LAN. When set, the command enables support for SW upgrade of daisy chained cameras. It will enable the camera’s DHCP function and force start of MAC and IP address retrieval. Remember to reset the DHCP when the camera is no longer connected to a LAN.

Value space:  <On/Off>

- **On**: Enable DHCP in the camera. The camera is automatically re-booted. After re-boot the DHCP is started and the IP address will be retrieved. Run the command “xStatus Camera” for result.
- **Off**: Disable DHCP in the camera. NOTE! When camera is not connected to a LAN, this setting should be applied.

Example: xConfiguration Cameras Camera 1 DHCP: Off
The Conference settings

**xConfiguration Conference [1..1] AutoAnswer Mode**
Set the AutoAnswer mode.
Value space: <On/Off>
- **On**: Enable AutoAnswer to let the system automatically answer all incoming calls.
- **Off**: The incoming calls must be answered manually by pressing the OK key or the green Call key on the remote control.

**Example**: `xConfiguration Conference 1 AutoAnswer Mode: Off`

**xConfiguration Conference [1..1] AutoAnswer Mute**
The AutoAnswer Mute setting determines whether the microphone is muted when an incoming call is automatically answered. **NOTE! Requires the AutoAnswer Mode to be enabled.**
Value space: <On/Off>
- **On**: The incoming call will be muted when automatically answered.
- **Off**: The incoming call will not be muted.

**Example**: `xConfiguration Conference 1 AutoAnswer Mute: Off`

**xConfiguration Conference [1..1] AutoAnswer Delay**
Define how long (in seconds) an incoming call has to wait before it is answered automatically by the system. **NOTE! Requires the AutoAnswer Mode to be enabled.**
Value space: <0..50>
- **Range**: 0-50 seconds

**Example**: `xConfiguration Conference 1 AutoAnswer Delay: 0`

**xConfiguration Conference [1..1] MicUnmuteOnDisconnect**
The MicUnmuteOnDisconnect setting determines if the microphones should be automatically unmuted when all calls are disconnected. In a meeting room or other shared resource this could be done to prepare the system for the next user.
Value space: <On/Off>
- **On**: Un-mute the microphones after the call is disconnected.
- **Off**: If muted, let the microphones remain muted after the call is disconnected.

**Example**: `xConfiguration Conference 1 MicUnmuteOnDisconnect: On`

**xConfiguration Conference [1..1] DoNotDisturb Mode**
The Do Not Disturb setting determines whether or not there should be an alert on incoming calls.
Value space: <On/Off>
- **On**: All incoming calls will be rejected, with no alert. The calling side will receive a busy signal when trying to call the codec. A message will display on screen, telling that Do not disturb is turned on, together with an option to turn off the Do not disturb. When turning off the Do not disturb mode you will see a list of the calls that have been rejected.
- **Off**: The incoming calls will be alerted.

**Example**: `xConfiguration DoNotDisturb Mode: Off`

**xConfiguration Conference [1..1] IncomingMultisiteCall Mode**
Set the incoming Multisite call mode.
Value space: <Allow/Deny>
- **Allow**: Accept incoming calls to an already active call/conference. The incoming call will be added to the MCU conference.
- **Deny**: The system will not accept incoming calls when you are in a call. The calling side will receive a busy signal.

**Example**: `xConfiguration Conference 1 IncomingMultisiteCall Mode: Allow`

**xConfiguration Conference [1..1] FarEndControl Mode**
Lets you decide if the remote side (far end) should be allowed to select your video sources and control your local camera (pan, tilt, zoom).
Value space: <On/Off>
- **On**: Allows the far end to be able to select your video sources and control your local camera (pan, tilt, zoom). You will still be able to control your camera and select your video sources as normal.
- **Off**: Do not allow the far end to select your video sources or to control your local camera (pan, tilt, zoom).

**Example**: `xConfiguration Conference 1 FarEndControl Mode: On`

**xConfiguration Conference [1..1] FarEndControl SignalCapability**
Set the far end control (H.224) signal capability mode.
Value space: <On/Off>
- **On**: Enable the far end control signal capability.
- **Off**: Disable the far end control signal capability.

**Example**: `xConfiguration Conference 1 FarEndControl SignalCapability: On`
The Conference settings, continued...

**xConfiguration Conference [1..1] Encryption Mode**
Set the conference encryption mode. A padlock with the text “Encryption On” or “Encryption Off” displays on screen, for a few seconds, when the conference starts.
Value space: <BestEffort/On/Off>
- **BestEffort**: The system will use encryption whenever possible.
- **On**: The system will only allow calls that are encrypted.
- **Off**: The system will not use encryption.

**Example:**

```
xCongfiguration Conference 1 Encryption Mode: BestEffort
```

**xConfiguration Conference [1..1] DefaultCall Protocol**
Set the Default Call Protocol to be used when placing calls from the system.
Value space: <H323/SIP>
- **H.323**: Select H.323 to ensure that calls are set up as H.323 calls.
- **SIP**: Select SIP to ensure that calls are set up as SIP calls.

**Example:**

```
xCongfiguration Conference 1 DefaultCall Protocol: H323
```

**xConfiguration Conference [1..1] DefaultCall Rate**
Set the Default Call Rate to be used when placing calls from the system.
Value space: <64..6000>
- **Range**: 64-6000kbps

**Example:**

```
xCongfiguration Conference 1 DefaultCall Rate: 768
```

**xConfiguration Conference [1..1] MaxTransmitCallRate**
Specify the maximum transmit call rate to be used when placing or receiving calls.
Value space: <64..6000>
- **Range**: 64-6000kbps

**Example:**

```
xCongfiguration Conference 1 MaxTransmitCallRate: 6000
```

The Conference settings, continued...

**xConfiguration Conference [1..1] MaxReceiveCallRate**
Specify the maximum receive call rate to be used when placing or receiving calls.
Value space: <64..6000>
- **Range**: 64-6000kbps

**Example:**

```
xCongfiguration Conference 1 MaxReceiveCallRate: 6000
```

**xConfiguration Conference [1..1] VideoBandwidth Mode**
Set the conference video bandwidth mode.
Value space: <Dynamic/Static>
- **Dynamic**: The available transmit bandwidth for the video channels are distributed among the currently active channels. If there is no presentation, the main video channels will use the bandwidth of the presentation channel.
- **Static**: The available transmit bandwidth is assigned to each video channel, even if it is not active.

**Example:**

```
xCongfiguration Conference 1 VideoBandwidth Mode: Dynamic
```

**xConfiguration Conference [1..1] VideoBandwidth MainChannel Weight**
The available transmit video bandwidth is distributed on the main channel and presentation channel according to “MainChannel Weight” and “PresentationChannel Weight”. If the main channel weight is 2 and the presentation channel weight is 1, then the main channel will use twice as much bandwidth as the presentation channel.
Value space: <1..10>
- **Range**: Select a value from 0 to 10.

**Example:**

```
xCongfiguration Conference 1 VideoBandwidth MainChannel Weight: 5
```

**xConfiguration Conference [1..1] VideoBandwidth PresentationChannel Weight**
The available transmit video bandwidth is distributed on the main channel and presentation channel according to “MainChannel Weight” and “PresentationChannel Weight”. If the main channel weight is 2 and the presentation channel weight is 1, then the main channel will use twice as much bandwidth as the presentation channel.
Value space: <1..10>
- **Range**: Select a value from 0 to 10.

**Example:**

```
xCongfiguration Conference 1 VideoBandwidth PresentationChannel Weight: 5
```
The GPIO settings

xConfiguration GPIO Pin [1..4] Mode

NOTE! This command is not supported on Codec C40.

The four GPIO pins are configured individually. The state can be retrieved by "xStatus GPIO Pin [1..4] State". The default pin state is High (+12V). When activated as output, they are set to 0V. To activate them as input, they must be pulled down to 0V.

Value space: <InputNoAction/OutputManualState/OutputInCall/OutputMicrophonesMuted/>

- **InputNoAction**: The pin state can be set, but no operation is performed.
- **OutputManualState**: The pin state can be set by "xCommand GPIO ManualState Set PinX: <High/Low>" (to +12V or 0V, respectively).
- **OutputInCall**: The pin is activated when in call, deactivated when not in call.
- **OutputMicrophonesMuted**: The pin is activated when microphones are muted, deactivated when not muted.
- **OutputPresentationOn**: The pin is activated when presentation is active, deactivated when presentation is not active.
- **OutputAllCallsEncrypted**: The pin is activated when all calls are encrypted, deactivated when one or more calls are not encrypted.
- **InputMuteMicrophones**: When the pin is activated (0V), the microphones will be muted. When deactivated (+ 12V), the microphones are unmuted.

Example: xConfiguration GPIO Pin 1 Mode: InputNoAction

The H323 settings

xConfiguration H323 NAT Mode

The TANDBERG firewall traversal technology creates a secure path through the firewall barrier, and enables proper exchange of audio/video data when connected to an external video conferencing system (when the IP traffic goes through a NAT router). NOTE! NAT does not work in conjunction with gatekeepers.

Value space: <Auto/On/Off>

- **Auto**: The system will try to determine if the "NAT Address" or the real IP-address should be used within signalling. This is done to make it possible to place calls to endpoints on the LAN as well as endpoints on the WAN.
- **On**: The system will signal the configured "NAT Address" in place of its own IP-address within Q.931 and H.245. The NAT Server Address will be shown in the startup-menu as: "My IP Address: 10.0.2.1".
- **Off**: The system will signal the real IP Address.

Example: xConfiguration H323 NAT Mode: Off

xConfiguration H323 NAT Address

Enter the external/global IP-address to the router with NAT support. Packets sent to the router will then be routed to the system.

In the router, the following ports must be routed to the system’s IP-address:

- Port 1720
- Port 5555-5574
- Port 2326-2485

Value space: <S: 0, 64>

Format: String with a maximum of 64 characters.

Example: xConfiguration H323 NAT Address: ""

xConfiguration H323 Profile [1..1] Authentication Mode

Set the authentication mode for the H.323 profile.

Value space: <On/Off>

- **On**: If the H.323 Gatekeeper Authentication Mode is set to On and a H.323 Gatekeeper indicates that it requires authentication, the system will try to authenticate itself to the gatekeeper. NOTE! Requires the Authentication LoginName and Authentication Password to be defined on both the codec and the Gatekeeper.
- **Off**: If the H.323 Gatekeeper Authentication Mode is set to Off the system will not try to authenticate itself to a H.323 Gatekeeper, but will still try a normal registration.

Example: xConfiguration H323 Profile 1 Authentication Mode: Off
The H.323 settings, continued...

**xConfiguration H323 Profile [1..1] Authentication LoginName**

The system sends the Authentication Login Name and the Authentication Password to a H.323 Gatekeeper for authentication. The authentication is a one way authentication from the codec to the H.323 Gatekeeper, i.e. the system is authenticated to the gatekeeper. If the H.323 Gatekeeper indicates that no authentication is required, the system will still try to register. NOTE! Requires the H.323 Gatekeeper Authentication Mode to be enabled.

Value space: \[S: 0, 50\]

Format: String with a maximum of 50 characters.

Example: `xConfiguration H323 Profile 1 Authentication LoginName: ""`

**xConfiguration H323 Profile [1..1] Authentication Password**

The system sends the Authentication Login Name and the Authentication Password to a H.323 Gatekeeper for authentication. The authentication is a one way authentication from the codec to the H.323 Gatekeeper, i.e. the system is authenticated to the gatekeeper. If the H.323 Gatekeeper indicates that no authentication is required, the system will still try to register. NOTE! Requires the H.323 Gatekeeper Authentication Mode to be enabled.

Value space: \[S: 0, 50\]

Format: String with a maximum of 50 characters.

Example: `xConfiguration H323 Profile 1 Authentication Password: `

**xConfiguration H323 Profile [1..1] CallSetup Mode**

The H.323 Call Setup Mode defines whether to use a Gatekeeper or Direct calling when establishing H.323 calls.

NOTE! Direct H.323 calls can be made even though the H.323 Call Setup Mode is set to Gatekeeper.

Value space: \[\text{Direct/Gatekeeper}\]

Direct: An IP-address must be used when dialing in order to make the H.323 call.

Gatekeeper: The system will use a Gatekeeper to make a H.323 call. When selecting this option the H.323 Profile Gatekeeper Address and H.323 Profile Gatekeeper Discovery settings must also be configured.

Example: `xConfiguration H323 Profile 1 CallSetup Mode: Gatekeeper`

**xConfiguration H323 Profile [1..1] Gatekeeper Discovery**

Determines how the system shall register to a H.323 Gatekeeper.

Value space: \[\text{Manual/Auto}\]

- **Manual:** The system will use a specific Gatekeeper identified by the Gatekeeper’s IP-address.
- **Auto:** The system will automatically try to register to any available Gatekeeper. If a Gatekeeper responds to the request sent from the codec within 30 seconds this specific Gatekeeper will be used. This requires that the Gatekeeper is in auto discovery mode as well. If no Gatekeeper responds, the system will not use a Gatekeeper for making H.323 calls and hence an IP-address must be specified manually.

Example: `xConfiguration H323 Profile 1 Gatekeeper Discovery: Manual`

**xConfiguration H323 Profile [1..1] Gatekeeper Address**

Enter the IP address of the Gatekeeper. NOTE! Requires the H.323 Call Setup Mode to be set to Gatekeeper and the Gatekeeper Discovery to be set to Manual.

Value space: \[S: 0, 255\]

Format: Only the valid IP address format is accepted. An IP address that contains letters (192.a.2.0) or invalid IP addresses (192.0.1234.0) will be rejected.

Example: `xConfiguration H323 Profile 1 Gatekeeper Address: "10.47.1.58"`

**xConfiguration H323 Profile [1..1] H323Alias E164**

The H.323 Alias E.164 defines the address of the system, according to the numbering plan implemented in the H.323 Gatekeeper. The E.164 alias is equivalent to a telephone number, sometimes combined with access codes.

Value space: \[S: 0, 30\]

Format: Compact string with a maximum of 30 characters. Valid characters are 0–9, *, and #.

Example: `xConfiguration H323 Profile 1 H323Alias E164: "90550092"`

**xConfiguration H323 Profile [1..1] H323Alias ID**

 Lets you specify the H.323 Alias ID which is used to address the system on a H.323 Gatekeeper and will be displayed in the call lists. Example: “firstname.surname@company.com”, “My H.323 Alias ID”

Value space: \[S: 0, 49\]

Format: String with a maximum of 49 characters

Example: `xConfiguration H323 Profile 1 H323Alias ID: "firstname.surname@company.com"`
The H323 settings, continued...

**xConfiguration H323 Profile [1..1] PortAllocation**

The H.323 Port Allocation setting affects the H.245 port numbers used for H.323 call signalling.

Value space: **<Dynamic/Static>**

*Dynamic*: The system will allocate which ports to use when opening a TCP connection. The reason for doing this is to avoid using the same ports for subsequent calls, as some firewalls consider this as a sign of attack. When Dynamic is selected, the H.323 ports used are from 11000 to 20999. Once 20999 is reached they restart again at 11000. For RTP and RTCP media data, the system is using UDP ports in the range 2326 to 2487. Each media channel is using two adjacent ports, ie 2330 and 2331 for RTP and RTCP respectively. The ports are automatically selected by the system within the given range. Firewall administrators should not try to deduce which ports are used when, as the allocation schema within the mentioned range may change without any further notice.

*Static*: When set to Static the ports are given within a static predefined range [5555–6555].

**Example**: `xConfiguration H323 Profile 1 PortAllocation: Dynamic`

**xConfiguration Network [1..1] DNS Domain Name**

DNS Domain Name is the default domain name suffix which is added to unqualified names. Example: if the DNS Domain Name is “company.com” and the name to lookup is “MyVideoSystem”, this will result in the DNS lookup “MyVideoSystem.company.com”.

Value space: `<S: 0, 64>`

*Format*: String with a maximum of 64 characters.

**Example**: `xConfiguration Network 1 DNS Domain Name: ""`

**xConfiguration Network [1..1] DNS Server [1..5] Address**

Define the network addresses for DNS servers. Up to 5 addresses may be specified. If the network addresses are unknown, contact your administrator or Internet Service Provider.

Value space: `<S: 0, 64>`

*Format*: String with a maximum of 64 characters.

**Example**: `xConfiguration Network 1 DNS Server 1 Address: ""`

**xConfiguration Network [1..1] Assignment**

Define whether to use DHCP or Static IP assignment. NOTE! Changes to this setting requires a restart of the codec.

Value space: `<Static/DHCP>`

*Static*: When set the network assignment to Static you must configure the static IP settings. Configure the settings: Network IPv4 Address, Network IPv4 SubnetMask and Network IPv4 Gateway.

*DHCP*: The system addresses are automatically assigned by the DHCP server.

**Example**: `xConfiguration Network 1 Assignment: DHCP`

**xConfiguration Network [1..1] IPv4 Address**

Define the Static IP network address for the system. Only applicable if the Network Assignment is set to Static.

Value space: `<S: 0, 64>`

*Format*: Only the valid IP address format is accepted. An IP address that contains letters (192.a.2.0) or unvalid IP addresses (192.0.1234.0) will be rejected.

**Example**: `xConfiguration Network 1 IPv4 Address: "10.47.5.100"`

**xConfiguration Network [1..1] IPv4 Gateway**

Define the IP network gateway. Only applicable if the Network Assignment is set to Static.

Value space: `<S: 0, 64>`

*Format*: Compact string with a maximum of 64 characters.

**Example**: `xConfiguration Network 1 IPv4 Gateway: "10.47.5.100"`
The Network settings, continued...

**xConfiguration Network [1..1] IPv4 SubnetMask**

Define the IP network subnet mask. Only applicable if the Network Assignment is set to Static.

Value space: `<S: 0, 64>`

Format: Compact string with a maximum of 64 characters.

Example: `xConfiguration Network 1 IPv4 SubnetMask: “255.255.255.0”`

**xConfiguration Network [1..1] IPv4 QoS Mode**

The QoS (Quality of Service) is a method which handles the priority of audio, video and data in the network. The QoS settings must be supported by the infrastructure. Diffserv (Differentiated Services) is a computer networking architecture that specifies a simple, scalable and coarse-grained mechanism for classifying, managing network traffic and providing QoS priorities on modern IP networks.

Value space: `<Off/Diffserv>`

Off: No QoS method is used.

Diffserv: When you set the QoS Mode to Diffserv you must configure the Diffserv sub menu settings (Audio, Data, Signalling and Video).

Example: `xConfiguration Network 1 IPv4 QoS Mode: diffserv`

**xConfiguration Network [1..1] IPv4 QoS Diffserv Audio**

The Diffserv Audio defines which priority Audio packets should have in an IP network. Enter a priority, which ranges from 0 to 63 for the packets. The higher the number, the higher the priority. These priorities might be overridden when packets are leaving the network controlled by the local network administrator. NOTE! Requires the Network IPv4 QoS Mode to be set to Diffserv.

Value space: `<0..63>`

Audio: A recommended value is Diffserv Code Point (DSCP) AF41, which equals the value 34. If in doubt, contact your network administrator.

Example: `xConfiguration Network 1 IPv4 QoS Diffserv Audio: 0`

**xConfiguration Network [1..1] IPv4 QoS Diffserv Data**

The Diffserv Data defines which priority Data packets should have in an IP network. Enter a priority, which ranges from 0 to 63 for the packets. The higher the number, the higher the priority. These priorities might be overridden when packets are leaving the network controlled by the local network administrator. NOTE! Requires the Network IPv4 QoS Mode to be set to Diffserv.

Value space: `<0..63>`

Data: A recommended value is Diffserv Code Point (DSCP) AF23, which equals the value 22. If in doubt, contact your network administrator.

Example: `xConfiguration Network 1 IPv4 QoS Diffserv Data: 0`

**xConfiguration Network [1..1] IPv4 QoS Diffserv Signalling**

The Diffserv Signalling defines which priority Signalling packets should have in an IP network. Enter a priority, which ranges from 0 to 63 for the packets. The higher the number, the higher the priority. These priorities might be overridden when packets are leaving the network controlled by the local network administrator. NOTE! Requires the Network IPv4 QoS Mode to be set to Diffserv.

Value space: `<0..63>`

Signalling: A recommended value is Diffserv Code Point (DSCP) AF31, which equals the value 26. If in doubt, contact your network administrator.

Example: `xConfiguration Network 1 IPv4 QoS Diffserv Signalling: 0`

**xConfiguration Network [1..1] IPv4 QoS Diffserv Video**

The Diffserv Video defines which priority Video packets should have in an IP network. Enter a priority, which ranges from 0 to 63 for the packets. The higher the number, the higher the priority. These priorities might be overridden when packets are leaving the network controlled by the local network administrator. NOTE! Requires the Network IPv4 QoS Mode to be set to Diffserv.

Value space: `<0..63>`

Video: A recommended value is Diffserv Code Point (DSCP) AF41, which equals the value 34. If in doubt, contact your network administrator.

Example: `xConfiguration Network 1 IPv4 QoS Diffserv Video: 0`

**xConfiguration Network [1..1] IEEE8021X Mode**

The system can be connected to an IEEE 802.1X LAN network, with a port-based network access control that is used to provide authenticated network access for Ethernet networks.

Value space: `<On/Off>`

On: The 802.1X authentication is enabled.

Off: The 802.1X authentication is disabled (default).

Example: `xConfiguration Network 1 IEEE8021X Mode: Off`

**xConfiguration Network [1..1] IEEE8021X Identity**

The 802.1X Identity is the user name needed for 802.1X authentication.

Value space: `<S: 0, 64>`

Format: String with a maximum of 64 characters.

Example: `xConfiguration Network 1 IEEE8021X Identity: “”`
The Network settings, continued...

**xConfiguration Network [1..1] IEEE8021X Password**

The 802.1X Password is the password needed for 802.1X authentication.

Value space: <S: 0, 32>

*Format:* String with a maximum of 32 characters.

*Example:* xConfiguration Network 1 IEEE8021X Password: "***"

**xConfiguration Network [1..1] IEEE8021X AnonymousIdentity**

The 802.1X Anonymous ID string is to be used as unencrypted identity with EAP (Extensible Authentication Protocol) types that support different tunneled identity, like EAP-PEAP and EAP-TTLS. If set, the anonymous ID will be used for the initial (unencrypted) EAP Identity Request.

Value space: <S: 0, 64>

*Format:* String with a maximum of 64 characters.

*Example:* xConfiguration Network 1 IEEE8021X AnonymousIdentity: ""

**xConfiguration Network [1..1] IEEE8021X Eap Md5**

Set the Md5 (Message-Digest Algorithm 5) mode. This is a Challenge Handshake Authentication Protocol that relies on a shared secret. Md5 is a Weak security.

Value space: <On/Off>

*On:* The EAP-MD5 protocol is enabled (default).

*Off:* The EAP-MD5 protocol is disabled.

*Example:* xConfiguration Network 1 IEEE8021X Eap Md5: On

**xConfiguration Network [1..1] IEEE8021X Eap Peap**

Set the Peap (Protected Extensible Authentication Protocol) mode. Authenticates LAN clients without the need for client certificates. Developed by Microsoft, Cisco and RSA Security.

Value space: <On/Off>

*On:* The EAP-PEAP protocol is enabled (default).

*Off:* The EAP-PEAP protocol is disabled.

*Example:* xConfiguration Network 1 IEEE8021X Eap Peap: On

**xConfiguration Network [1..1] IEEE8021X Eap TTLS**

Set the TTLS (Tunneled Transport Layer Security) mode. Authenticates LAN clients without the need for client certificates. Developed by Funk Software and Certicom. Usually supported by Agere Systems, Proxim and Avaya.

Value space: <On/Off>

*On:* The EAP-TTLS protocol is enabled (default).

*Off:* The EAP-TTLS protocol is disabled.

*Example:* xConfiguration Network 1 IEEE8021X Eap TTLS: On

**xConfiguration Network [1..1] MTU**

Set the Ethernet MTU (Maximum Transmission Unit).

Value space: <400..1500>

*Range:* Select a value from 400 to 1500bytes.

*Example:* xConfiguration Network 1 MTU: 1500

**xConfiguration Network [1..1] Speed**

Set the Ethernet link speed.

Value space: <Auto/10half/10full/100half/100full/1000full>

*Auto:* Autonegotiate link speed.

*10half:* Force link to 10Mbps half-duplex.

*10full:* Force link to 10Mbps full-duplex.

*100half:* Force link to 100Mbps half-duplex.

*100full:* Force link to 100Mbps full-duplex.

*1000full:* Force link to 1Gbps full-duplex.

*Example:* xConfiguration Network 1 Speed: Auto

**xConfiguration Network [1..1] TrafficControl Mode**

Set the network traffic control mode to decide how to control the the video packets transmission speed.

Value space: <On/Off>

*On:* Transmit video packets at maximum 20Mbps. Can be used to smooth out bursts in the outgoing network traffic.

*Off:* Transmit video packets at link speed.

*Example:* xConfiguration Network 1 TrafficControl: On
The Network settings, continued...

### xConfiguration Network [1..1] VLAN Voice Mode
Set the VLAN voice mode.

Value space: `<Tagged/Untagged>`
- **Tagged**: The voice packets in the VLAN network are tagged with VlanId and Priority.
- **Untagged**: The voice packets in the VLAN network are untagged.

**Example:** `xConfiguration Network 1 VLAN Voice Mode: Untagged`

### xConfiguration Network [1..1] VLAN Voice VlanId
Set the VLAN voice ID.

Value space: `<0..4096>`
- **Range**: Select a value from 0 to 4096.

**Example:** `xConfiguration Network 1 VLAN Voice VlanId: 0`

### xConfiguration Network [1..1] VLAN Voice Priority
Set the VLAN voice priority.

Value space: `<0..7>`
- **Range**: Select a value from 0 to 7.

**Example:** `xConfiguration Network 1 VLAN Voice Priority: 0`

The NetworkServices settings

### xConfiguration NetworkServices H323 Mode
Determines whether the system should be able to place and receive H.323 calls. NOTE! Requires a restart of the codec.

Value space: `<On/Off>`
- **On**: Enable the possibility to place and receive H.323 calls (default).
- **Off**: Disable the possibility to place and receive H.323 calls.

**Example:** `xConfiguration NetworkServices H323 Mode: On`

### xConfiguration NetworkServices HTTP Mode
Set the HTTP mode to enable/disable access to the system through a web browser. The web interface is used for system management, call management such as call transfer, diagnostics and software uploads.

Value space: `<On/Off>`
- **On**: The HTTP protocol is enabled.
- **Off**: The HTTP protocol is disabled.

**Example:** `xConfiguration NetworkServices HTTP Mode: On`

### xConfiguration NetworkServices HTTPS Mode
HTTPS is a web protocol that encrypts and decrypts user page requests as well as the pages that are returned by the web server.

Value space: `<On/Off>`
- **On**: The HTTPS protocol is enabled.
- **Off**: The HTTPS protocol is disabled.

**Example:** `xConfiguration NetworkServices HTTPS Mode: On`

### xConfiguration NetworkServices HTTPS VerifyServerCertificate
When the system connects to an external HTTPS server (like a phonebook server or an external manager), this server will present a certificate to the system to identify itself. This setting tells the system if it should verify that the certificate is signed by a trusted Certificate Authority (CA). This requires that list of trusted CAs is uploaded to the system in advance.

Value space: `<On/Off>`
- **On**: Verify server certificates.
- **Off**: Do not verify server certificates.

**Example:** `xConfiguration NetworkServices HTTPS VerifyServerCertificate: Off`
The NetworkServices settings, continued...

**xConfiguration NetworkServices NTP Mode**
The Network Time Protocol (NTP) is used to synchronize the time of the system to a reference time server. The time server will subsequently be queried every 24th hour for time updates. The time will be displayed on the top of the screen. The system will use the time to timestamp messages transmitted to Gatekeepers or Border Controllers requiring H.235 authentication. The system will use the time to timestamp messages transmitted to Gatekeepers or Border Controllers that require H.235 authentication. It is also used for timestamping Placed Calls, Missed Calls and Received Calls.

Value space:  <Auto/Manual>

Auto: The system will use the NTP server, by which address is supplied from the DHCP server in the network. If no DHCP server is used, or the DHCP server does not provide the system with a NTP server address, the system will use the static defined NTP server address specified by the user.

Manual: The system will always use the static defined NTP server address specified by the user.

Example: xConfiguration NetworkServices NTP Mode: Manual

**xConfiguration NetworkServices NTP Address**
Enter the NTP Address to define the network time protocol server address. This address will be used if NTP Mode is set to Manual, or if set to Auto and no address is supplied by a DHCP server.

Value space:  <S: 0, 64>

Format:  String with a maximum of 64 characters.

Example: xConfiguration NetworkServices NTP Address: "1.tandberg.pool.ntp.org"

**xConfiguration NetworkServices SIP Mode**
Determines whether the system should be able to place and receive SIP calls. NOTE! Requires a restart of the codec.

Value space:  <On/Off>

On: Enable the possibility to place and receive SIP calls (default).

Off: Disable the possibility to place and receive SIP calls.

Example: xConfiguration NetworkServices SIP Mode: On

The NetworkServices settings, continued...

**xConfiguration NetworkServices SNMP Mode**
SNMP (Simple Network Management Protocol) is used in network management systems to monitor network-attached devices (routers, servers, switches, projectors, etc) for conditions that warrant administrative attention. SNMP exposes management data in the form of variables on the managed systems, which describe the system configuration. These variables can then be queried (set to ReadOnly) and sometimes set (set to ReadWrite) by managing applications.

Value space:  <Off/ReadOnly/ReadWrite>

Off: Disable the SNMP network service.

ReadOnly: Enable the SNMP network service for queries only.

ReadWrite: Enable the SNMP network service for both queries and commands.

Example: xConfiguration NetworkServices SNMP Mode: ReadWrite

**xConfiguration NetworkServices SNMP Host [1..3] Address**
Enter the address of up to three SNMP Managers. All traps will then be sent to the hosts listed. The system's SNMP Agent (in the codec) responds to requests from SNMP Managers (a PC program etc.). SNMP Traps are generated by the SNMP Agent to inform the SNMP Manager about important events. Can be used to send event created messages to the SNMP agent about different events like: system reboot, system dialling, system disconnecting, MCU call, packet loss etc. Traps can be sent to multiple SNMP Trap Hosts.

Value space:  <S: 0, 64>

Format:  String with a maximum of 64 characters.

Example: xConfiguration NetworkServices SNMP Host 1 Address: ""

**xConfiguration NetworkServices SNMP CommunityName**
Enter the name of the Network Services SNMP Community. The SNMP Community names are used to authenticate SNMP requests. The SNMP Community names are used to authenticate SNMP requests. The SNMP requests must have a 'password' (case sensitive) in order to receive a response from the SNMP Agent in the codec. The default password is "public". If you have the TANDBERG Management Suite (TMS) you must make sure the same SNMP Community is configured there too. NOTE! The SNMP Community password is case sensitive.

Value space:  <S: 0, 50>

Format:  String with a maximum of 50 characters.

Example: xConfiguration NetworkServices SNMP CommunityName: "public"

**xConfiguration NetworkServices SNMP SystemContact**
Enter the name of the Network Services SNMP System Contact.

Value space:  <S: 0, 50>

Format:  String with a maximum of 50 characters.

Example: xConfiguration NetworkServices SNMP SystemContact: ""
The NetworkServices settings, continued...

xConfiguration NetworkServices SNMP SystemLocation
Enter the name of the Network Services SNMP System Location.
Value space: <S: 0, 50>
Format: String with a maximum of 50 characters.
Example: xConfiguration NetworkServices SNMP SystemLocation: ""

xConfiguration NetworkServices Telnet Mode
Telnet is a network protocol used on the Internet or Local Area Network (LAN) connections.
Value space: <On/Off>
On: The Telnet protocol is enabled.
Off: The Telnet protocol is disabled. This is the factory setting.
Example: xConfiguration NetworkServices Telnet Mode: Off

xConfiguration Phonebook Server [1..5] ID
Enter a name for the external phonebook.
Value space: <S: 0, 64>
Format: String with a maximum of 64 characters.
Example: xConfiguration Phonebook Server 1 ID: ""

xConfiguration Phonebook Server [1..5] Type
Select the phonebook server type.
Value space: <VCS/TMS/Callway>
VCS: Select VCS if the phonebook is located on the TANDBERG Video Communication Server.
TMS: Select TMS if the phonebook is located on the TANDBERG Management Suite server.
Callway: Select Callway if the phonebook is to be provided by the Callway subscription service.
Contact your Callway provider for more information.
Example: xConfiguration Phonebook Server 1 Type: TMS

xConfiguration Phonebook Server [1..5] URL
Enter the address (URL) to the external phonebook server.
Value space: <S: 0, 255>
Format: String with a maximum of 255 characters.
The Provisioning settings

**xConfiguration Provisioning Mode**
Provides the possibility of managing the codec (endpoint) by using an external manager/management system.

Value space: `<Off/TMS/Callway>`
- **Off**: The system will not try to register to any management system.
- **TMS**: If set to TMS (TANDBERG Management System) the system will try to register with a TMS server. Contact your TANDBERG representative for more information.
- **Callway**: If set to Callway the system will try to register with the Callway subscription provider. Contact your Callway provider for more information.

Example: `xConfiguration Provisioning Mode: TMS`

**xConfiguration Provisioning ExternalManager Address**
Enter the IP Address to the External Manager/Management system. If an External Manager address and a path is configured, the system will post an HTTP message to this address when starting up. When receiving this HTTP posting the External Manager (typically a management system) can return configurations/commands to the unit as a result. If the DHCP Option 242 is returned in the DHCP response from the DHCP server the system will interpret this as the External Manager address to use.

Value space: `<S: 0, 64>`
- **Format**: Only the valid IP address format is accepted. An IP address that contains letters (192.a.2.0) or unvalid IP addresses (192.0.1234.0) will be rejected.

Example: `xConfiguration Provisioning ExternalManager Address: ""`

**xConfiguration Provisioning ExternalManager Path**
Set the path to the External Manager/Management system. If an External Manager address and a path is configured, the system will post an HTTP message to this address when starting up. When receiving this HTTP posting the External Manager (typically a management system) can return configurations/commands to the unit as a result. If the DHCP Option 242 is returned in the DHCP response from the DHCP server the system will interpret this as the External Manager address to use.

Value space: `<S: 0, 255>`
- **Format**: String with a maximum of 255 characters.

Example: `xConfiguration Provisioning ExternalManager Path: "tms/public/external/management/SystemManagementService.asmx"`

**xConfiguration Provisioning ExternalManager Protocol**
Determines whether or not to use secure management.

Value space: `<HTTP/HTTPS>`
- **HTTP**: Set to HTTP to disable secure management. Requires HTTP to be enabled in the xConfiguration NetworkServices HTTP Mode setting.
- **HTTPS**: Set to HTTPS to enable secure management. Requires HTTPS to be enabled in the xConfiguration NetworkServices HTTPS Mode setting.

Example: `xConfiguration Provisioning ExternalManager Protocol: HTTP`

**xConfiguration Provisioning HttpMethod**
Select the HTTP method to be used for the provisioning.

Value space: `<GET/POST>`
- **GET**: Select GET when the provisioning server supports GET.
- **POST**: Select POST when the provisioning server supports POST.

Example: `xConfiguration Provisioning HttpMethod: POST`

**xConfiguration Provisioning LoginName**
Enter the user id provided by the provisioning server. This is the user name part of the credentials used to authenticate towards the HTTP server when using HTTP provisioning.

Value space: `<S: 0, 80>`
- **Format**: String with a maximum of 80 characters.

Example: `xConfiguration Provisioning LoginName: ""`

**xConfiguration Provisioning Password**
Enter the password provided by the provisioning server. This is the password part of the credentials used to authenticate towards the HTTP server when using HTTP provisioning.

Value space: `<S: 0, 64>`
- **Format**: String with a maximum of 64 characters.

Example: `xConfiguration Provisioning Password: ""`
The SerialPort settings

**xConfiguration SerialPort BaudRate**

Specify the baud rate (data transmission rate, bits per second) for the COM 1 port on the codec. The default value is 38400.

Connection parameters for the COM port: Data bits: 8 Parity: None Stop bits: 1 Flow control: None.

Value space: <9600/19200/38400/57600/115200>

Range: Select a baud rate from the list of baud rates (bps).

Example: xConfiguration SerialPort BaudRate: 38400

**xConfiguration SerialPort LoginRequired**

The Serial Login setting determines whether or not there should be a login when connecting to the COM 1 port at the codec.

Value space: <On/Off>

On: Login is required when connecting to the codec through COM port.

Off: The user can access the codec through COM port without any login.

Example: xConfiguration SerialPort LoginRequired: On

The SIP settings

**xConfiguration SIP Profile [1..1] Authentication [1..1] LoginName**

This is the user name part of the credentials used to authenticate towards the SIP proxy.

Value space: <S: 0, 50>

Format: String with a maximum of 50 characters.

Example: xConfiguration SIP Profile 1 Authentication 1 LoginName: ""
The SIP settings, continued...

**xCconfiguration SIP Profile [1..1] Proxy [1..4] Address**

The Proxy Address is the manually configured address for the outbound proxy. It is possible to use a fully qualified domain name, or an IP address. The default port is 5060 for TCP and UDP but another one can be provided. If Outbound is enabled, multiple proxies can be addressed.

Value space: <S: 0, 255>

Format: Compact string with a maximum of 255 characters.

Example: xConfiguration SIP Profile 1 Proxy 1 Address: ""

**xCconfiguration SIP Profile [1..1] Proxy [1..4] Discovery**

Select if the SIP Proxy address is to be obtained manually or by using Dynamic Host Configuration Protocol (DHCP).

Value space: <Auto/Manual>

Auto: When Auto is selected, the SIP Proxy address is obtained using Dynamic Host Configuration Protocol (DHCP).

Example: xConfiguration SIP Profile 1 Proxy 1 Discovery: Manual

**xCconfiguration SIP Profile [1..1] Type**

Enables SIP extensions and special behaviour for a vendor or provider.

Value space: <Standard/Alcatel/Avaya/Cisco/Microsoft/Nortel/Experimental/Siemens>

Standard: Should be used when registering to standard SIP proxy like OpenSer.

Alcatel: Must be used when registering to an Alcatel-Lucent OmniPCX Enterprise R7 or later.

Avaya: Must be used when registering to a Avaya Communication Manager.

Cisco: Must be used when registering to a Cisco CallManager version 5 or later.

Microsoft: Must be used when registering to a Microsoft LCS or OCS server.

Nortel: Must be used when registering to a Nortel MCS 5100 or MCS 5200 PBX.

Experimental: Can be used if auto is not working. NOTE! This mode is for testing purposes only.

Example: xConfiguration SIP Profile 1 Type: Standard

**xCconfiguration SIP Profile [1..1] URI**

The SIP URI or number is used to address the system. This is the URI that is registered and used by the SIP services to route inbound calls to the system. A Uniform Resource Identifier (URI) is a compact string of characters used to identify or name a resource.

Value space: <S: 0, 255>

Format: Compact string with a maximum of 255 characters.

Example: xConfiguration SIP Profile 1 URI: “sip:firstname.lastname@company.com”

The Standby settings

**xCconfiguration Standby Control**

Determine whether the system should go into standby mode or not.

Value space: <On/Off>

On: Enter standby mode when the Standby Delay has timed out. NOTE! Requires the Standby Delay to be set to an appropriate value.

Off: The system will not enter standby mode.

Example: xConfiguration Standby Control: On

**xCconfiguration Standby Delay**

Define how long (in minutes) the system shall be in idle mode before it goes into standby mode. NOTE! Requires the Standby Control to be enabled.

Value space: <1..480>

Range: Select a value from 1 to 480 minutes.

Example: xConfiguration Standby Delay: 10

**xCconfiguration Standby BootAction**

Define the camera position after a restart of the codec.

Value space: <None/Preset1/Preset2/Preset3/Preset4/Preset5/Preset6/Preset7/Preset8/Preset9/Preset10/Preset11/Preset12/Preset13/Preset14/Preset15/RestoreCameraPosition/DefaultCameraPosition>

None: No action.

Preset1 to Preset15: After a reboot the camera position will be set to the position defined by the selected preset.

RestoreCameraPosition: After a reboot the camera position will be set to the position it had before the last boot.

DefaultCameraPosition: After a reboot the camera position will be set to the factory default position.

Example: xConfiguration Standby BootAction: DefaultCameraPosition

**xCconfiguration Standby StandbyAction**

Define the camera position when going into standby mode.

Value space: <None/PrivacyPosition>

None: No action.

PrivacyPosition: Turns the camera to a sideways position for privacy.

Example: xConfiguration Standby StandbyAction: PrivacyPosition
The SystemUnit settings

**xConfiguration SystemUnit CallLogging Mode**
Set the call logging mode for calls that are received or placed by the system. The call logs may then be viewed via the GUI or using the xHistory command.

Value space: <On/Off>

- **On**: Enable logging.
- **Off**: Disable logging.

**Example**: xConfiguration SystemUnit CallLogging Mode: On

**xConfiguration SystemUnit IrSensor Mode**
The System Unit IR Sensor setting determines whether the infrared receiver on the codec should be enabled or not. The IR sensor LED is located in the front of the codec and flickers when an IR signal is received from the remote control.

Value space: <On/Off/Auto>

- **On**: Enable the IR sensor on the codec.
- **Off**: Disable the IR sensor on the codec.
- **Auto**: Both the TANDBERG codec and camera has an IR sensor. The system will automatically disable the IR sensor on the codec if the IR sensor at camera is enabled. Otherwise the IR sensor on the codec will be enabled.

**Example**: xConfiguration SystemUnit IrSensor Mode: On

**xConfiguration SystemUnit MenuLanguage**
The setting is used to select the language for the GUI (Graphical User Interface).

Value space: <English/Norwegian/Swedish/German/French/Italian/Japanese/Russian/Spanish/Korean/Finnish/ChineseSimplified/ChineseTraditional/PortugueseBrazilian/Turkish/Polish/Danish/Dutch>

**Example**: xConfiguration SystemUnit MenuLanguage: English

**xConfiguration SystemUnit Name**
Enter a System Name to define a name of the system unit. If the H.323 Alias ID is configured on the system then this ID will be used instead of the system name. The system name will be displayed:

1) When the codec is acting as an SNMP Agent.
2) Towards a DHCP server.

Value space: <S: 0, 50>

- **Format**: String with a maximum of 50 characters.

**Example**: xConfiguration SystemUnit Name: "Meeting Room Name"
The Time settings, continued...

xConfiguration Time Zone

Set the time zone where the system is located, using Windows time zone description format.

Example: xConfiguration Time Zone: "GMT (Greenwich Mean Time : Dublin, Edinburgh, Lisbon, London)"

The Time settings,
The Video settings

**xConfiguration Video DefaultPresentationSource**

Define which video input source shall be used as the default presentation source (e.g. when you press the Presentation key on the remote control). The input source is configured to a video input connector.

Value space: <1..3>

Range: Select the video source to be used as the presentation source.

Example: xConfiguration Video DefaultPresentationSource: 3

**xConfiguration Video Input DVI [3]/[2..3] Type**

NOTE! Codec C40 has one DVI input (DVI-I 3). Codec C60 has two DVI inputs (DVI-I 2 and 3).
The official DVI standard supports both digital and analog signals. In most cases the default AutoDetect setting can detect whether the signal is analog RGB or digital. However, in some rare cases when DVI-I cables are used (these cables can carry both the analog and digital signals) the auto detection fails. This setting makes it possible to override the AutoDetect and select the correct DVI video input.

This setting should also be used if the video input is an analog component (YPbPr) type signal. This is used by some cameras (Sony EVI-HD1) and DVD/Blu-ray players. Since it is not possible to auto detect the difference between AnalogRGB and AnalogYPbPr, the AnalogYPbPr setting must be selected.

Value space: <AutoDetect/Digital/AnalogRGB/AnalogYPbPr>

- **AutoDetect**: Set to AutoDetect to automatically detect if the signal is analog RGB or digital.
- **Digital**: Set to Digital to force the DVI video input to Digital when using DVI-I cables with both analog and digital pins and AutoDetect fails.
- **AnalogRGB**: Set to AnalogRGB to force the DVI video input to AnalogRGB when using DVI-I cables with both analog and digital pins and AutoDetect fails.
- **AnalogYPbPr**: Set to AnalogYPbPr to force the DVI video input to AnalogYPbPr, as the component (YPbPr) signal cannot be auto detected.

Example: xConfiguration Video Input DVI 3 Type: AutoDetect

**xConfiguration Video Input Source 1 Connector**

Select which video input connector to be active on video input source 1.

Value space: <HDMI>

- **HDMI**: Select HDMI when you want to use the HDMI 1 as input source 1.

Example: xConfiguration Video Input Source 1 Connector: HDMI

**xConfiguration Video Input Source 2 Connector**

NOTE! Codec C40 has one DVI input (DVI-I 3). Codec C60 has two DVI inputs (DVI-I 2 and 3).

Select which video input connector to be active on video input source 2.

Value space: <HDMI/DVI>

- **HDMI**: Select HDMI when you want to use the HDMI 2 as input source 2.
- **DVI**: Select DVI-I when you want to use the DVI-I 2 as input source 2.

Example: xConfiguration Video Input Source 2 Connector: HDMI

**xConfiguration Video Input Source 3 Connector**

NOTE! Codec C40 has one DVI input (DVI-I 3). Codec C60 has two DVI inputs (DVI-I 2 and 3).

Select which video input connector to be active on video input source 3.

Value space: <DVI/Composite/YC>

- **DVI**: Select DVI-I when you want to use the DVI-I 3 as input source 3.
- **Composite**: Select Composite when you want to use the Composite as input source 3.
- **YC**: Select YC when you want to use the S-Video (YC) as input source 3. Connect to the two connectors marked Y/Comp and C.

Example: xConfiguration Video Input Source 3 Connector: DVI

**xConfiguration Video Input Source [1..3] Name**

Enter a name for the video input source 1 to 3.

Value space: <S: 0, 50>

Format: String with a maximum of 50 characters.

Example: xConfiguration Video Input Source 1 Name: ""

**xConfiguration Video Input Source [1..3] CameraControl Mode**

Set the camera control mode for the camera associated with the video source 1 to 3.

Value space: <On/Off>

- **On**: Enable camera control.
- **Off**: Disable camera control.

Example: xConfiguration Video Input Source 1 CameraControl Mode: On
The Video settings, continued...

xConfiguration Video Input Source [1..3] CameraControl CameraId

The CameraId represents the camera’s position in the Visca chain. NOTE! Requires the Video Input Source CameraControl Mode to be enabled.

Read more about cascaded cameras and Visca commands in the TANDBERG PrecisionHD 1080p User Guide. Go to: http://www.tandberg.com/docs.

Value space: <1..5>

Example: xConfiguration Video Input Source 1 CameraControl CameraId: 1

xConfiguration Video Input Source [1..3] OptimalDefinition Profile

Adjust how rapidly the system will increase the transmitted resolution when increasing the bandwidth. NOTE! Requires that the Video Input Source Quality is set to Motion.

Normal: Use this setting for normal to poorly lit environment. If the source is a camera with 1920x1080p60, the system will transmit 1920x720p60 at about 2.2Mb/sec and above with this setting set to normal.

Medium: Requires better than normal and consistent lighting and good quality video inputs. If the source is a camera with 1920x1080p60, the system will transmit 1920x720p60 at about 1.4Mb/sec and above with this setting set to medium.

High: Requires good lighting conditions for a good overall experience and good quality video inputs. If the source is a camera with 1920x1080p60, the system will transmit 1920x720p60 at about 1.1Mb/sec and above with this setting set to high.

Value space: <Normal/Medium/High>

See Table 1: Optimal definition for systems supporting 1080p and Table 2: Optimal definition for systems supporting 720p60.

Example: xConfiguration Video Input Source 1 OptimalDefinition Profile: Normal

<table>
<thead>
<tr>
<th></th>
<th>Normal</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Table 1: Optimal definition, for systems supporting 1080p</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w288p30</td>
<td>256kbit/s</td>
<td>128kbit/s</td>
<td>128kbit/s</td>
</tr>
<tr>
<td>w448p30</td>
<td>512kbit/s</td>
<td>384kbit/s</td>
<td>256kbit/s</td>
</tr>
<tr>
<td>w576p30</td>
<td>768kbit/s</td>
<td>512kbit/s</td>
<td>512kbit/s</td>
</tr>
<tr>
<td>720p30</td>
<td>1152kbit/s</td>
<td>512kbit/s</td>
<td>768kbit/s</td>
</tr>
<tr>
<td>1080p30</td>
<td>2560kbit/s</td>
<td>1920kbit/s</td>
<td>1472kbit/s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Normal</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Table 2: Optimal definition, for systems supporting 720p60</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w144p60</td>
<td>128kbit/s</td>
<td>128kbit/s</td>
<td>128kbit/s</td>
</tr>
<tr>
<td>w288p60</td>
<td>512kbit/s</td>
<td>384kbit/s</td>
<td>256kbit/s</td>
</tr>
<tr>
<td>w448p60</td>
<td>1152kbit/s</td>
<td>512kbit/s</td>
<td>768kbit/s</td>
</tr>
<tr>
<td>w576p60</td>
<td>1472kbit/s</td>
<td>768kbit/s</td>
<td>1152kbit/s</td>
</tr>
<tr>
<td>720p60</td>
<td>2240kbit/s</td>
<td>1920kbit/s</td>
<td>1152kbit/s</td>
</tr>
</tbody>
</table>

xConfiguration Video Input Source [1..3] OptimalDefinition Threshold60fps

For each video input, this setting tells the system the lowest resolution where it should transmit 60fps. So for all resolutions lower than this, the maximum transmitted framerate would be 30fps, while above this resolution 60fps would also be possible, if the available bandwidth is adequate.

Value space: <512_288/768_448/1024_576/1280_720/Never>

Example: xConfiguration Video Input Source 1 OptimalDefinition Threshold60fps: 1280 _ 720

xConfiguration Video Input Source [1..3] Quality

When encoding and transmitting video there will be a tradeoff between high resolution and high framerate. For some video sources it is more important to transmit high framerate than high resolution and vice versa. The Quality setting specifies whether to give priority to high frame rate or to high resolution for a given source.

Value space: <Motion/Sharpness>

Motion: Gives the highest possible framerate. Used when there is a need for higher frame rates, typically when a large number of participants are present or when there is a lot of motion in the picture.

Sharpness: Gives the highest possible resolution. Used when you want the highest quality of detailed images and graphics.

Example: xConfiguration Video Input Source 1 Quality: Motion

xConfiguration Video Layout ScaleToFrame

Define what to do if the aspect ratio of a video input source doesn’t match the aspect ratio of the corresponding image frame in a composition. For example if you have a 4:3 input source (like XGA) to be displayed on a 16:9 output (like HD720).

Value space: <Manual/MaintainAspectRatio/StretchToFit>

Manual: If the difference in aspect ratio between the video input source and the target image frame is less than the ScaleToFrameThreshold configuration (in percent), the image is stretched to fit. If not, the system will maintain the original aspect ratio.

MaintainAspectRatio: Will maintain the aspect ratio of the input source, and fill in black in the rest of the image frame (letter boxing or pillar boxing).

StretchToFit: Will stretch (horizontally or vertically) the input source to fit into the image frame.

Example: xConfiguration Video Layout ScaleToFrame: MaintainAspectRatio
The Video settings, continued...

xConfiguration Video Layout ScaleToFrameThreshold

Only applicable if the ScaleToFrame configuration is set to manual. If the difference in aspect ratio between the video input source and the target image frame is less than the ScaleToFrameThreshold configuration (in percent), the image is stretched to fit. If not, the system will maintain the original aspect ratio.

Value space: <0..100>
- Range: Select a value from 0 to 100 percent.

Example: xConfiguration Video Layout ScaleToFrameThreshold: 5

xConfiguration Video Layout Scaling

Define how the system shall adjust the aspect ratio for images or frames when there is a difference between the image and the frame it is to be placed in.

Value space: <On/Off>
- On: Let the system automatically adjust aspect ratio.
- Off: No adjustment of the aspect ratio.

Example: xConfiguration Video Layout Scaling: On

xConfiguration Video MainVideoSource

Define which video input source shall be used as the main video source.

Value space: <1..3>
- Range: Select the source to be used as the main video source.

Example: xConfiguration Video MainVideoSource: 1

xConfiguration Video Monitors

Set the monitor layout mode.

Value space: <Single/Dual/DualPresentationOnly>
- Single: The same layout is shown on all monitors.
- Dual: The layout is distributed on two monitors.
- DualPresentationOnly: All participants in the call will be shown on the first monitor, while the presentation (if any) will be shown on the second monitor.

Example: xConfiguration Video Monitors: Single

The Video settings, continued...

xConfiguration Video OSD InputMethod InputLanguage

The codec can be enabled for Cyrillic input characters in the GUI (Graphical User Interface). NOTE! Requires that xConfiguration Video OSD InputMethod Cyrillic is set to On.

Value space: <Latin/Cyrillic>
- Latin: Latin characters can be entered when using the remote control (default).
- Cyrillic: Cyrillic characters can be entered using the remote control. NOTE! Requires a TANDBERG Remote Control TRC5 with Cyrillic fonts.

Example: xConfiguration Video OSD InputMethod InputLanguage: Latin

xConfiguration Video OSD InputMethod Cyrillic

This configuration is used to hide or show the Cyrillic mode as menu input language in the GUI (Graphical User Interface).

Value space: <On/Off>
- On: Cyrillic mode is available as a menu input language in the GUI. This will enable the setting xConfiguration Video OSD InputMethod InputLanguage.
- Off: Cyrillic mode is NOT available as a menu input language in the GUI.

Example: xConfiguration Video OSD InputMethod Cyrillic: Off

xConfiguration Video OSD Mode

The Video OSD (On Screen Display) Mode lets you define if information and icons should be displayed on screen.

Value space: <On/Off>
- On: Show the on screen menus, icons and indicators.
- Off: Hide the on screen menus, icons and indicators.

Example: xConfiguration Video OSD Mode: On
The Video settings, continued...

xConfiguration Video OSD Output
The Video OSD (On Screen Display) Output lets you define which monitor should display the on screen menus, information and icons. By default the OSD is sent to the monitor connected to the Video OSD Output 1. If you cannot see the OSD on screen, then you must re-configure the OSD Output. You can do this by entering a key sequence on the remote control, from the web interface, or by a command line interface.

Using the TANDBERG Remote Control TRC5: Press the Disconnect key followed by: * # * # 0 x # (where x is output 1 to 2).

Using the web interface: Open a web browser and enter the IP address of the codec. Open the Advanced menu and navigate to Video OSD Output and select the video output. Using a command line interface: Open a command line interface and connect to the codec (if in doubt of how to do this, see the API Guide for the codec). Enter the command: xConfiguration Video OSD Output [1..2] (select the OSD Output).

Value space: <1..2>

Range: Select 1 for HDMI 1 output, or select 2 for DVI-I 2 output.

Example: xConfiguration Video OSD Output: 1

xConfiguration Video OSD TodaysBookings
This setting can be used to display the systems bookings for today on the main OSD menu. This requires that the system is bookable by an external booking system, like TMS (TANDBERG Management Suite).

Value space: <On/Off>

On: Displays information about this systems bookings on screen.
Off: Do not display todays bookings.

Example: xConfiguration Video OSD TodaysBookings: Off

xConfiguration Video OSD MyContactsExpanded
Set how the local contacts will be displayed in the phone book dialog in the OSD (On Screen Display).

Value space: <On/Off>

On: The local contacts in the phone book will be shown in the top level of the phonebook dialog.
Off: The local contacts will be placed in a separate folder called MyContacts in the phonebook dialog.

Example: xConfiguration Video OSD MyContactsExpanded: Off

xConfiguration Video Output HDMI [1] MonitorRole
The HDMI monitor role describes what video stream will be shown on the monitor connected to the video output HDMI connector. Applicable only if the “Video > Monitors” configuration is set to dual.

Value space: <First/Second/PresentationOnly>

First: Show main video stream.
Second: Show presentation video stream if active, or other participants.
PresentationOnly: Show presentation video stream if active, and nothing else.

Example: xConfiguration Video Output HDMI 1 MonitorRole: First

xConfiguration Video Output HDMI [1] OverscanLevel
Some TV’s or other monitors may not display the whole image sent out on the systems video output, but cuts the outer parts of the image. In this case this setting can be used to let the system not use the outer parts of video resolution. Both the video and the OSD menu will be scaled in this case.

Value space: <Medium/High/None>

Medium: The system will not use the outer 3% of the output resolution.
High: The system will not use the outer 6% of the output resolution.
None: The system will use all of the output resolution.

Example: xConfiguration Video Output HDMI 1 OverscanLevel: None

xConfiguration Video Output HDMI [1] Resolution
Select the preferred resolution for the monitor connected to the video output HDMI connector. This will force the resolution on the monitor.

Value space: <Auto/640_480_60/800_600_60/1024_768_60/1280_1024_60/1280_720_60/1920_1080_60/1280_768_60/1360_768_60/1600_1200_60/1920_1200_60>

Auto: The system will automatically try to set the optimal resolution based on negotiation with the connected monitor.

Range: 640x480@60p, 800x600@60p, 1024x768@60p, 1280x1024@60p, 1280x720@60p, 1920x1080@60p, 1280x768@60p, 1360x768@60p, 1600x1200@60p, 1920x1200@60p

Example: xConfiguration Video Output HDMI 1 Resolution: 1920_1080_60
The Video settings, continued...

**xConfiguration Video Output DVI [2] MonitorRole**

The DVI monitor role describes what video stream will be shown on the monitor connected to the video output DVI-I connector. Applicable only if the “Video > Monitors” configuration is set to dual.

Value space: `<First/Second/PresentationOnly>`

- **First**: Show main video stream.
- **Second**: Show presentation video stream if active, or other participants.
- **PresentationOnly**: Show presentation video stream if active, and nothing else.

**Example:** `xConfiguration Video Output DVI 2 MonitorRole: Second`

**xConfiguration Video Output DVI [2] OverscanLevel**

Some TV's or other monitors may not display the whole image sent out on the systems video output, but cuts the outer parts of the image. In this case this setting can be used to let the system not use the outer parts of video resolution. Both the video and the OSD menu will be scaled in this case.

Value space: `<Medium/High/None>`

- **Medium**: The system will not use the outer 3% of the output resolution.
- **High**: The system will not use the outer 6% of the output resolution
- **None**: The system will use all of the output resolution.

**Example:** `xConfiguration Video Output DVI 2 OverscanLevel: None`

**xConfiguration Video Output DVI [2] Resolution**

Select the preferred resolution for the monitor connected to the video output DVI-I connector. This will force the resolution on the monitor.

Value space: `<Auto/640_480_60/800_600_60/1024_768_60/1280_1024_60/1280_720_60/1920_1080_60/1280_768_60/1360_768_60/1600_1200_60/1920_1200_60>`

- **Auto**: The system will automatically try to set the optimal resolution based on negotiation with the connected monitor.
- **Range**: 640x480@60p, 800x600@60p, 1024x768@60p, 1280x1024@60p, 1280x720@60p, 1920x1080@60p, 1280x768@60p, 1360x768@60p, 1600x1200@60p, 1920x1200@60p

**Example:** `xConfiguration Video Output DVI 2 Resolution: 1024_768_60`

The Video settings, continued...

**xConfiguration Video Output Composite [3] MonitorRole**

NOTE! This command is not supported on Codec C40.

The Composite monitor role describes what video stream will be shown on the monitor connected to the video output Composite connector. Applicable only if the monitor configuration is set to dual.

Value space: `<First/Second/PresentationOnly>`

- **First**: Show main video stream.
- **Second**: Show presentation video stream if active, or other participants.
- **PresentationOnly**: Show presentation video stream if active, and nothing else.

**Example:** `xConfiguration Video Output Composite 3 MonitorRole: First`

**xConfiguration Video Output Composite [3] OverscanLevel**

NOTE! This command is not supported on Codec C40.

Some TV's or other monitors may not display the whole image sent out on the systems video output, but cuts the outer parts of the image. In this case this setting can be used to let the system not use the outer parts of video resolution. Both the video and the OSD menu will be scaled in this case.

Value space: `<Medium/High/None>`

- **Medium**: The system will not use the outer 3% of the output resolution.
- **High**: The system will not use the outer 6% of the output resolution
- **None**: The system will use all of the output resolution.

**Example:** `xConfiguration Video Output Composite 3 OverscanLevel: None`

**xConfiguration Video Output Composite [3] Resolution**

NOTE! This command is not supported on Codec C40.

Select the preferred resolution for the monitor connected to the video output Composite connector. This will force the resolution on the monitor.

Value space: `<PAL/NTSC>`

- **Range**: Select PAL or NTSC resolution.

**Example:** `xConfiguration Video Output Composite 3 Resolution: NTSC`
xConfiguration Video Selfview
The Video Selfview setting determines whether or not the main video source (selfview) should be displayed on screen.

Value space:  <On/Off>
   On: Show self view on screen.
   Off: Hide self view on screen.

Example: xConfiguration Video Selfview: On

xConfiguration Video WallPaper
The Video Wallpaper setting determines whether or not a background picture should be displayed on screen when idle.

Value space:  <None/Growing/Summersky/Custom>
   None: No wallpaper will be displayed on screen.
   Summersky, Growing: Select one of the predefined wallpapers to be displayed on screen.
   Custom: The custom wallpaper must be uploaded to the codec from the web interface.

   1) On the video system: Find the IP address of the codec. Open the menu on screen and go to Home > Settings > System information to find the IP Address.
   2) On your computer: Open a web browser and enter the IP address of the codec. Select “Wallpaper” from the menu, browse for the file, and press the “Upload” button.
   3) On the video system: Open the menu on screen and go to Home > Settings > Wallpaper > Custom. Give it a few seconds to display the new picture. If the picture does not show, toggle once between “None” and “Custom” wallpaper to make the change take effect.

Example: xConfiguration Video Wallpaper: Summersky
The Experimental menu
The Advanced configurations menu has an option called Experimental. The settings within this menu can be used 'as is' and will not be fully documented. **NOTE!** The Experimental menu WILL change.

**xConfiguration Experimental Audio Input Microphone [1..2]/[1..4] Channel**
NOTE! Codec C40 has two microphone connectors. Codec C60 has four microphone connectors. Defines whether the signal from the microphone is a mono signal or part of a multichannel signal.
Value space: <Left/Right/Mono>
- **Left:** The Audio Microphone input signal is the left channel of a stereo signal.
- **Right:** The Audio Microphone input signal is the right channel of a stereo signal.
- **Mono:** The Audio Microphone input signal is a mono signal.

**Example:** xConfiguration Experimental Audio Input Microphone 1 Channel: Left

**xConfiguration Experimental Audio Input Microphone [1..4] EchoControl**

**HighPassFilter**
Value space: <On/Off>

**xConfiguration Experimental CapsetFilter**
Value space: <S: 0, 32>

**xConfiguration Experimental Conference [1..1] PacketLossResilience**
Value space: <On/Off>

**xConfiguration Experimental CustomSoftbuttons State [1..2] Softbutton [1..5] Type**
Value space: <NotSet/MainSource/PresentationSource/CameraPreset/Actions/SpeedDial>

**xConfiguration Experimental CustomSoftbuttons State [1..2] Softbutton [1..5] Value**
Value space: <S: 0, 255>

**xConfiguration Experimental SoftwareUpgrade Mode**
Value space: <Auto/Manual>

**xConfiguration Experimental SoftwareUpgrade ServerAddress**
Value space: <S: 0, 255>
Chapter 4

Description of the xCommand commands
The Audio commands

**xCommand Audio Equalizer Update**

The system has eight user defined equalizers, each made up of six second order IIR sections. Each of the eight equalizers can be applied to one or more of the audio input and output connectors on the codec. Each IIR section can be modified independently. There are five filter types, and the frequency response variations depend on some of the parameter variations. The Q-value for low pass, high pass, low shelf and high shelf filters should be set to 1/sqrt(2) in order to get maximally flat responses. The Q-value of the filter is defined as $Q = f_0/bw$. Where $f_0$ is the resonance frequency in Hz and $bw$ is the filter bandwith in Hz.

To switch off one of the six equalizer sections set the second order section to have a flat frequency response. This can be done by setting the filter type to “none” or by setting the filter type to “peaking” and the gain to “0” (zero).

We recommend using the Audio Console application, with the build in equalizer GUI, to modify the equalizers. The Audio Console is found at the TANDBERG Developer Zone web page. Go to: http://developer.tandberg.com/web/guest/tools/integrators/audio-console.

Read more about the equalizer, filter types, gain and Q-value in this API Guide. Go to the Appendices section.

**EqualizerId**: Select the equalizer 1 to 8.

**Section**: Select the section 1 to 6.

**FilterType**: Select the filter type.

**Frequency**: Set the frequency.

**Q**: Set the Q-value.

**Gain**: Set the gain.

**Parameters**:  
EqualizerId(r): <1..8>  
Section(r): <1..6>  
FilterType(r): <HighPass/HighShelf/LowPass/LowShelf/None/Peaking>  
Frequency(r): <S: 0, 32>  
Q(r): <S: 0, 32>  
Gain(r): <S: 0, 32>

**Example**:

```plaintext
xCommand Audio Equalizer List EqualizerId: 1  
  * AudioEqualizerListResult Equalizer 1 Section 1 FilterType: "Peaking"  
  * AudioEqualizerListResult Equalizer 1 Section 1 Frequency: "100"  
  * AudioEqualizerListResult Equalizer 1 Section 1 Q: "4"  
  * AudioEqualizerListResult Equalizer 1 Section 2 FilterType: "Peaking"  
  * AudioEqualizerListResult Equalizer 1 Section 2 Frequency: "200"  
  * AudioEqualizerListResult Equalizer 1 Section 2 Q: "4"  
  * AudioEqualizerListResult Equalizer 1 Section 3 FilterType: "Peaking"  
  * AudioEqualizerListResult Equalizer 1 Section 3 Frequency: "500"  
  * AudioEqualizerListResult Equalizer 1 Section 3 Q: "4"  
  *AudioEqualizerListResult Equalizer 1 Section 4 FilterType: "Peaking"  
  *AudioEqualizerListResult Equalizer 1 Section 4 Frequency: "1000"  
  *AudioEqualizerListResult Equalizer 1 Section 4 Q: "4"  
  *AudioEqualizerListResult Equalizer 1 Section 5 FilterType: "Peaking"  
  *AudioEqualizerListResult Equalizer 1 Section 5 Frequency: "2000"  
  *AudioEqualizerListResult Equalizer 1 Section 5 Q: "4"  
  *AudioEqualizerListResult Equalizer 1 Section 6 FilterType: "Peaking"  
  *AudioEqualizerListResult Equalizer 1 Section 6 Frequency: "5000"  
  *AudioEqualizerListResult Equalizer 1 Section 6 Q: "0"  
** end
```
The Audio commands, cont...

** end

xCommand Audio Microphones Mute

Mute all microphones.

** end

xCommand Audio Microphones Unmute

Unmute microphones.

** end

The Audio commands, cont...

xCommand Audio LocalInput Add

Create a local input and generates the local input id. A local input is a mix of input connectors with the following settings: Name, MixerMode, AGC, Mute and Channels.

InputId: A unique identifier for the local input.
Name: Choose a name that describes the mix of input connectors.
MixerMode Auto: The microphone with the strongest speaker is active and the others are strongly attenuated.
MixerMode Fixed: The input connector signals are mixed together with equal gains.
MixerMode GainShared: The microphones are given a normalized gain factor relative to the strongest speaker before being mixed together.
AGC: Automatic Gain Control.
Mute: Mutes the mix of input connectors.
Channels: Set channels to 1 to mix the input connectors into a mono signal. To mix the input connectors into a stereo signal, set channels to 2.

Parameters:

- InputId: \(<0..65534>\)
- Name: \(<S: 0, 255>\)
- MixerMode: \(<\text{Auto/Fixed/GainShared}>\)
- AGC: \(<\text{On/Off}>\)
- Mute: \(<\text{On/Off}>\)
- Channels: \(<1..2>\)

Example:

```
xCommand Audio LocalInput Add
OK
*r AudioLocalInputAddResult (status=OK):
  InputId: 2
** end
```
The Audio commands, cont...

**xCommand Audio LocalInput Update**

Update the settings of the local input given by the input ID.

**InputId**: A unique identifier for the local input.

**Name**: Choose a name that describes the mix of input connectors.

**MixerMode**
- **Auto**: The microphone with the strongest speaker is active and the others are strongly attenuated.
- **Fixed**: The input connector signals are mixed together with equal gains.
- **GainShared**: The microphones are given a normalized gain factor relative to the strongest speaker before being mixed together.

**AGC**: Automatic Gain Control.

**Mute**: Mutes the mix of input connectors.

**Channels**: Set channels to 1 to mix the input connectors into a mono signal. To mix the input connectors into a stereo signal, set channels to 2.

**Parameters**:
- **InputId(r)**: <0..65534>
- **Name(r)**: <S: 0, 255>
- **MixerMode(r)**: <Auto/Fixed/GainShared>
- **AGC(r)**: <On/Off>
- **Mute(r)**: <On/Off>
- **Channels(r)**: <1..2>

**Example**:

```
xCommand Audio LocalInput Update InputId: 2 Name: "Microphone" MixerMode: GainShared AGC: Off Mute: Off Channels: 1
```

**OK**

```
*r AudioInputGroupUpdateResult (status=OK):
** end
```

**xCommand Audio LocalInput AddConnector**

Attach an input connector to the local input given by the input ID. A connector is defined by its type and ID.

**InputId**: A unique identifier for the local input.

**ConnectorType**: Select the connector type.

**ConnectorId**: Select a connector.

**Parameters**:
- **InputId(r)**: <0..65534>
- **ConnectorType(r)**: <HDMI/Line/Microphone>
- **ConnectorId(r)**: <1..8>

**Example**:

```
xCommand Audio LocalInput AddConnector InputId: 3 ConnectorType: Line ConnectorId: 1
```

**OK**

```
*r AudioInputGroupAddConnectorResult (status=OK):
** end
```

**xCommand Audio LocalInput RemoveConnector**

Detach an input connector from the local input given by the input ID. A connector is defined by its type and ID.

**InputId**: A unique identifier for the local input.

**ConnectorType**: Select the connector type.

**ConnectorId**: Select the connector.

**Parameters**:
- **InputId(r)**: <0..65534>
- **ConnectorType(r)**: <HDMI/Line/Microphone>
- **ConnectorId(r)**: <1..8>

**Example**:

```
xCommand Audio LocalInput RemoveConnector InputId: 3 ConnectorType: Line ConnectorId: 1
```

**OK**

```
*r AudioInputGroupRemoveConnectorResult (status=OK):
** end
```

The Audio commands, cont...
The Audio commands, cont...

**xCommand Audio LocalOutput Add**

Create a local output and generate the local output id. A local output is a mix of local input and remote input signals. All connectors attached to the local output receive the same signal.

- **OutputId**: A unique identifier for the local output.
- **Name**: Choose a name that describes the local output.
- **Loudspeaker**: If one or more of the output connectors are connected to a loudspeaker, this signal should be a reference signal to the echo canceller. Hence set loudspeaker to On. NOTE! When microphone reinforcement is disabled there should only be one loudspeaker local output.
- **Channels**: Set channels to 1 to mix the local and remote inputs into a mono signal. To mix the inputs into a stereo signal, set channels to 2.

**Parameters:**
- OutputId: <0..65534>
- Name: <S: 0, 255>
- Loudspeaker: <On/Off>
- Channels: <1..2>

**Example:**

```plaintext
xCommand Audio LocalOutput Add
OK
*r AudioOutputGroupAddResult (status=OK):
  OutputId: 47
** end
```

**xCommand Audio LocalOutput Update**

Update the settings of the local output given by the output ID.

- **OutputId**: A unique identifier for the local output.
- **Name**: Choose a name that describes the local output.
- **Loudspeaker**: If one or more of the output connectors are connected to a loudspeaker, this signal should be a reference signal to the echo canceller. Hence set loudspeaker to On. NOTE! When microphone reinforcement is disabled there should only be one loudspeaker local output.
- **Channels**: Set channels to 1 to mix the local and remote inputs into a mono signal. To mix the inputs into a stereo signal, set channels to 2.

**Parameters:**
- OutputId(r): <0..65534>
- Name(r): <S: 0, 255>
- Loudspeaker(r): <On/Off>
- Channels(r): <1..2>

**Example:**

```plaintext
xCommand Audio LocalOutput Update OutputId: 5 Name: "Loudspeaker"
  Loudspeaker: On Channels: 2
OK
*r AudioOutputGroupUpdateResult (status=OK):
  ** end
```

**xCommand Audio LocalOutput Remove**

Remove the local output given by the output ID.

- **OutputId**: A unique identifier for the local output.

**Parameters:**
- OutputId(r): <0..65534>

**Example:**

```plaintext
xCommand Audio LocalOutput Remove OutputId: 6
OK
*r AudioOutputGroupRemoveResult (status=OK):
  ** end
```
The Audio commands, cont...

**xCommand Audio LocalOutput AddConnector**

Attach an output connector to the local output given by the output ID. A connector is defined by its type and ID.

OutputId: A unique identifier for the local output.
ConnectorType: Select the connector type.
ConnectorId: Select a connector.

Parameters:
- OutputId(r): <0..65534>
- ConnectorType(r): <HDMI/Line>
- ConnectorId(r): <1..8>

Example:
```
xCommand Audio LocalOutput AddConnector OutputId:5 ConnectorType: Line
ConnectorId:1
OK
```
```
*r AudioOutputGroupAddConnectorResult {status=OK};
** end
```

**xCommand Audio LocalOutput RemoveConnector**

Detach an output connector from the local output given by the output ID. A connector is defined by its type and ID.

OutputId: A unique identifier for the local output.
ConnectorType: Select the connector type.
ConnectorId: Select a connector.

Parameters:
- OutputId(r): <0..65534>
- ConnectorType(r): <HDMI/Line>
- ConnectorId(r): <1..8>

Example:
```
xCommand Audio LocalOutput RemoveConnector OutputId:5 ConnectorType: Line
ConnectorId:1
OK
```
```
*r AudioOutputGroupRemoveConnectorResult {status=OK};
** end
```

**xCommand Audio LocalOutput ConnectInput**

Connect a local input or remote input to a local output by giving their IDs as parameters.

OutputId: A unique identifier for the local output.
InputId: A unique identifier for the local input.
InputGain: Set a gain on the input signal in the range from -54dB to 15dB. The value -54dB equals Off.

Parameters:
- OutputId(r): <0..65534>
- InputId(r): <0..65534>
- InputGain: <-54..15>

Example:
```
xCommand Audio LocalOutput ConnectInput OutputId:6 InputId:3
OK
```
```
*r AudioOutputGroupConnectInputResult {status=OK};
** end
```

**xCommand Audio LocalOutput UpdateInputGain**

Update the gain of a local or remote input connected to a local output. The gain on the input signal is in the range from -54dB to 15dB. The value -54dB equals Off.

OutputId: A unique identifier for the local output.
InputId: A unique identifier for the local input.
InputGain: Set a gain on the input signal in the range from -54dB to 15dB. The value -54dB equals Off.

Parameters:
- OutputId(r): <0..65534>
- InputId(r): <0..65534>
- InputGain: <-54..15>

Example:
```
xCommand Audio LocalOutput UpdateInputGain OutputId:6 InputId:3
OK
```
```
*r AudioOutputGroupUpdateInputGainResult {status=OK};
** end
```
The Audio commands, cont...

**xCommand Audio LocalOutput DisconnectInput**
Disconnect a local or remote input from a local output.
OutputId: A unique identifier for the local output.
InputId: A unique identifier for the local or remote input.

**Parameters:**
- OutputId(r): <0..65534>
- InputId(r): <0..65534>

**Example:**
```
xCommand Audio LocalOutput DisconnectInput OutputId:6 InputId:3
OK
```

**xCommand Audio RemoteInput Update**
Update the automatic gain control setting of the remote input, given by the input ID.
InputId: A unique identifier for the remote input.
AGC: Set the Automatic Gain Control.

**Parameters:**
- InputId(r): <0..65534>
- AGC(r): <On/Off>

**Example:**
```
xCommand Audio RemoteInput Update InputId:9 AGC: Off
OK
```

**xCommand Audio RemoteOutput ConnectInput**
Connect a local or remote input to a remote output with their IDs as parameters. When a call is made a remote input and remote output pair is created.
OutputId: A unique identifier for the local output.
InputId: A unique identifier for the local or remote input.
InputGain: Set a gain on the input signal in the range from -54dB to 15dB. The value -54dB equals Off.

**Parameters:**
- OutputId(r): <0..65534>
- InputId(r): <0..65534>
- InputGain: <-54..15>

**Example:**
```
xCommand Audio RemoteOutput ConnectInput OutputId:10 InputId:8
OK
```

**xCommand Audio RemoteOutput UpdateInputGain**
Update the gain of a local or remote input connected to a remote output.
OutputId: A unique identifier for the local output.
InputId: A unique identifier for the local or remote input.
InputGain: Set a gain on the input signal in the range from -54dB to 15dB. The value -54dB equals Off.

**Parameters:**
- OutputId(r): <0..65534>
- InputId(r): <0..65534>
- InputGain: <-54..15>

**Example:**
```
xCommand Audio RemoteOutput UpdateInputGain OutputId:6 InputId:3
OK
```

The Audio commands, cont...
The Audio commands, cont...

**xCommand Audio RemoteOutput DisconnectInput**
Disconnect a local or remote input from a remote output with their IDs as parameters.
OutputId: A unique identifier for the local output.
InputId: A unique identifier for the local or remote input.

**Parameters:**
- OutputId(r): <0..65534>
- InputId(r): <0..65534>

**Example:**
```
xCommand Audio RemoteOutput DisconnectInput OutputId:10 InputId:8
```
```
* AudioRemoteOutputGroupDisconnectInputResult (status=OK):
** end
```

**xCommand Audio Setup Clear**
Remove all local inputs and local outputs.

**Parameters:**
No parameters.

**Example:**
```
xCommand Audio Setup Clear
```
```
* AudioSetupClearResult (status=OK):
** end
```

**xCommand Audio Sound Play**
Play the specified audio sound.

**Parameters:**
- Sound(r): <Busy/CallWaiting/Dial/KeyTone/Ringing/SpecialInfo/TelephoneCall/VideoCall>
- Loop: <On/Off>

**Example:**
```
xCommand Audio Sound Play Sound: Ringing
```
```
* AudioSoundPlayResult (status=OK):
** end
```

**xCommand Audio Sound Stop**
Stop playing audio sound.

**Parameters:**
No parameters.

**Example:**
```
xCommand Audio Sound Stop
```
```
* AudioSoundStopResult (status=OK):
** end
```

**xCommand Audio Vumeter Start**
Start collecting VU meter information for connector given by type and ID.

**Parameters:**
- ConnectorType(r): <HDMI/Line/Microphone>
- ConnectorId(r): <1..8>

**Example:**
```
xCommand Audio Vumeter Start ConnectorType: Microphone ConnectorId: 1
```
```
* AudioVumeterStartResult (status=OK):
** end
```

**xCommand Audio Vumeter Stop**
Stop collecting VU meter information for connector given by type and ID.

**Parameters:**
- ConnectorType(r): <HDMI/Line/Microphone>
- ConnectorId(r): <1..8>

**Example:**
```
xCommand Audio Vumeter Stop ConnectorType: Microphone ConnectorId: 1
```
```
* AudioVumeterStopResult (status=OK):
** end
```

The Audio commands, cont...
The Boot command

**xCommand Boot**
Reboot system.

**Parameters:**
No parameters.

**Example:**
```
xCMD Boot
*r BootResult (status=OK):
** endOK
CUIL reboot request, restarting
Connection closed by foreign host.
```

The Call commands, cont...

**xCommand Call Reject**
Reject incoming call. If no call id is specified, all incoming calls will be rejected. The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the Callid.

**Parameters:**
CallId: <0..65534>

**Example:**
```
xCMD Call Reject CallId:20
OK
*r CallRejectResult (status=OK):
** end
```

**xCommand Call Disconnect**
Disconnect a call. The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.

**Parameters:**
CallId(r): <0..65534>

**Example:**
```
xCMD Call Disconnect CallId:17
OK
*r DisconnectCallResult (status=OK):
** end
```

**xCommand Call DisconnectAll**
Disconnect all active calls.

**Parameters:**
No parameters.

**Example:**
```
xCMD Call DisconnectAll
OK
*r DisconnectAllResult (status=OK):
** end
```

The Call commands

**xCommand Call Accept**
Accept an incoming call. If no CallId is specified, all incoming calls will be accepted. The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.

**Parameters:**
CallId: <0..65534>

**Example:**
```
xCMD Call Accept CallId:19
OK
*r CallAcceptResult (status=OK):
** end
```
The Call Log commands

**xCommand CallLog Clear**
Clear call logs stored in the system. If a logtag is given as argument, that specific call is deleted from the logs. If no logtag is given then all call logs will be deleted. The LogTag values for the calls are found by issuing the xHistory CallLog Call command.

**Parameters:**
- LogTag: <0..2147483647>

**Example:**
```
xCommand CallLog Clear
*r ClearResult (status=OK):
** end
```

**xCommand CallLog Recent Delete**
Delete the call log of recent calls. If a logtag is given as argument, that specific call is deleted from the log. If no logtag is given, the complete recent calls log will be deleted. The LogTag values for recent calls are found by issuing the xHistory CallLog Recent command.

**Parameters:**
- LogTag: <0..2147483647>

**Example:**
```
xCommand CallLog Recent Delete
*r DeleteResult (status=OK):
** end
```

**xCommand CallLog Outgoing Delete**
Delete the call log of outgoing calls. If a logtag is given as argument, that specific call is deleted from the log. If no logtag is given, the complete outgoing calls log will be deleted. The LogTag values for outgoing calls are found by issuing the xHistory CallLog Outgoing command.

**Parameters:**
- LogTag: <0..2147483647>

**Example:**
```
xCommand CallLog Outgoing Delete LogTag:202
*r DeleteResult (status=OK):
** end
```

**xCommand CallLog Received Delete**
Delete the call log of received calls. If a logtag is given as argument, that specific call is deleted from the log. If no logtag is given, the complete received calls log will be deleted. The LogTag values for received calls are found by issuing the xHistory CallLog Received command.

**Parameters:**
- LogTag: <0..2147483647>

**Example:**
```
xCommand CallLog Received Delete LogTag:126
*r DeleteResult (status=OK):
** end
```

**xCommand CallLog Missed Delete**
Delete the call log of missed calls. If a logtag is given as argument, that specific call is deleted from the log. If no logtag is given, the complete missed calls log will be deleted. The LogTag values for missed calls are found by issuing the xHistory CallLog Missed command.

**Parameters:**
- LogTag: <0..2147483647>

**Example:**
```
xCommand CallLog Missed Delete LogTag:119
*r DeleteResult (status=OK):
** end
```

The Call Log commands, cont...

**xCommand CallLog Received Delete**
Delete the call log of received calls. If a logtag is given as argument, that specific call is deleted from the log. If no logtag is given, the complete received calls log will be deleted. The LogTag values for received calls are found by issuing the xHistory CallLog Received command.

**Parameters:**
- LogTag: <0..2147483647>

**Example:**
```
xCommand CallLog Received Delete LogTag:126
*r DeleteResult (status=OK):
** end
```

**xCommand CallLog Missed Delete**
Delete the call log of missed calls. If a logtag is given as argument, that specific call is deleted from the log. If no logtag is given, the complete missed calls log will be deleted. The LogTag values for missed calls are found by issuing the xHistory CallLog Missed command.

**Parameters:**
- LogTag: <0..2147483647>

**Example:**
```
xCommand CallLog Missed Delete LogTag:119
*r DeleteResult (status=OK):
** end
```
The CamCtrlPip command

xCommand CamCtrlPip

Show or hide the camera selfview in a small window (picture in picture).

Parameters:
- Mode(r): <On/Off>

Example:
- xCommand CamCtrlPip Mode: On
  *r CamCtrlPipResult (status=OK):
  ** end

The Camera commands

xCommand Camera PositionSet

Position the camera by defining the pan, tilt, zoom and focus parameters. If the camera is placed in a daisy chain you need to know the CameraId for the camera you want to address.

Parameters:
- CameraId(r): <1..7>
- Pan: <-65535..65535>
- Tilt: <-65535..65535>
- Zoom: <0..65535>
- Focus: <0..65535>

Example:
- xCommand Camera PositionSet CameraId:1 Pan:200 Tilt:300 OK
  *r CameraPositionSetResult (status=OK):
  ** end

xCommand Camera PanTiltReset

The camera is reset to its default values for pan and tilt. In a daisy chain you need to know the CameraId for the camera you want to address.

Parameters:
- CameraId(r): <1..7>

Example:
- xCommand Camera PanTiltReset CameraId:1 OK
  *r PanTiltResetResult (status=OK):
  ** end
xCommand Camera Ramp
Move the camera in a specified direction. The camera will move at specified speed until a stop command is issued. In a daisy chain, you need to know the CameraId for the camera you want to address. NOTE! You must run a stop command to stop the camera, see the example below.

CameraId: Give the camera id.
Pan: Move the camera to the Left or Right, followed by Stop.
PanSpeed: Set the pan speed.
Tilt: Move the camera Up or Down, followed by Stop.
TiltSpeed: Set the tilt speed.
Zoom: Zoom the camera In or Out, followed by Stop.
ZoomSpeed: Set the zoom speed.
Focus: Focus the camera Far or Near, followed by Stop.

Parameters:

- CameraId(r): <1..7>
- Pan: <Left/Right/Stop>
- PanSpeed: <1..15>
- Tilt: <Down/Stop/Up>
- TiltSpeed: <1..15>
- Zoom: <In/Out/Stop>
- ZoomSpeed: <1..15>
- Focus: <Far/Near/Stop>

Example:

```
xCommand Camera Ramp CameraId:1 Pan;left PanSpeed:1
OK
*r RampResult (status=OK):
** end
```

xCommand Camera TriggerAutofocus
Trigger the autofocus functionality. The camera must support autofocus functionality. In a daisy chain you need to know the CameraId for the camera you want to address.

Parameters:

- CameraId(r): <1..7>

Example:

```
xCommand Camera TriggerAutofocus CameraId:1
OK
*r TriggerAutofocusResult (status=OK):
** end
```

xCommand Camera ReconfigureCameraChain
Reinitialize the daisy chain of cameras and updates the CameraId parameter. The CameraId parameter holds information of which camera is sitting in what position in the camera chain.

Parameters:

- No parameters.

Example:

```
xCommand Camera ReconfigureCameraChain
*r ReconfigureCameraChainResult (status=OK):
** end
```
The Dial command

xCommand Dial

Dial out from the system. Returns information about the CallId and ConferenceId, which are required for some other commands.

Number(r): Enter the number or address.
Protocol: Select the H323 or SIP protocol.
CallRate: Set a call rate.
CallType: Select the audio or video call type.

Parameters:

  Number(r): <S: 0, 255>
  Protocol: <H323/SIP>
  CallRate: <64..6000>
  CallType: <Audio/Video>

Example:

  xCommand Dial Number:543210 Protocol:h323
  OK
  *r DialResult (status=OK):
    CallId: 2
    ConferenceId: 1
  ** end

The DTMF command

xCommand DTMFSend

Send DTMF tones to the far end.

CallId: The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallID.
DTMFString: Enter the DTMF string.

Parameters:

  CallId(r): <0..65534>
  DTMFString(r): <S: 0, 32>

Example:

  xCommand DTMFSend CallId:2 DTMFString:1234
  *r DTMFSendResult (status=OK):
  ** end
The Far End Control commands

**xCommand FarEndControl Camera Move**
Move the far end camera (the remote camera). NOTE! The far end camera will move in the specified direction until the stop command (ref: xCommand FarEndControl Camera Stop) is issued.
CallId: The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.
Value: Select the action for how to move the camera.

Parameters:
- CallId(r): <0..65534>
- Value(r): <Left/Right/Up/Down/ZoomIn/ZoomOut>

Example:
```
xCommand FarEndControl Camera Move CallId:3 Value:left
*r FECCMoveResult (status=OK):
** end
```

**xCommand FarEndControl Camera Stop**
Stop the far end camera after the xCommand FarEndControl Camera Move has been issued.
CallId: The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.

Parameters:
- CallId(r): <0..65534>

Example:
```
xCommand FarEndControl Camera Stop CallId:3
*r FECCMoveResult (status=OK):
** end
```

**xCommand FarEndControl Preset Activate**
Move the far end camera to a camera preset position.
CallId: The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.
PresetId: The PresetId for far end control must be retrieved from the far end codec.

Parameters:
- CallId(r): <0..65534>
- PresetId(r): <1..15>

Example:
```
xCommand FarEndControl Preset Activate CallId:3 PresetId:1
*r FECCPresetActivateResult (status=OK):
** end
```

**xCommand FarEndControl Preset Store**
Store the far end camera position to a camera preset.
CallId: The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.
PresetId: The PresetId for far end control must be retrieved from the far end codec.

Parameters:
- CallId(r): <0..65534>
- PresetId(r): <0..15>

Example:
```
xCommand FarEndControl Preset Store CallId:3 PresetId:1
*r FECCPresetStoreResult (status=OK):
** end
```
The Far End Control commands, cont...

**xCommand FarEndControl Source Select**

Select which video input source to use as the main source on the far end system.

- **CallId**: The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.
- **SourceId**: Select a video input source on the far end.

**Parameters:**

- CallId(r): <0..65534>
- SourceId(r): <0..15>

**Example:**

```
xCommand FarEndControl Source Select CallId:3 SourceId:1
*r FECCSelectSourceResult (status=OK):
** end
```

The GPIO commands

**xCommand GPIO ManualState Set**

NOTE: This command is not supported on Codec C40.

Set the status of a GPIO pin that is configured to OutputManualState mode (ref. xConfiguration GPIO Pin [1..4] Mode).

**Parameters:**

- Pin1: <High/Low>
- Pin2: <High/Low>
- Pin3: <High/Low>
- Pin4: <High/Low>

**Example:**

```
xCommand GPIO ManualState Set Pin1: High
*r GpioStateSetResult (status=OK):
** end
```
The HTTP Feedback commands

**xCommand HttpFeedback Register**
Register the system to a HTTP(S) server to return XML feedback over HTTP(S) to specific URLs.
FeedbackSlot: You can have up to three slots for feedback.
ServerUrl: Define the URL for the HTTP(S) server.
Expression[1..15]: What parts of the Status and Configuration XML documents to monitor are specified by XPath expressions. You can have from 1 to 15 XPath expressions.

**Parameters:**
- FeedbackSlot: <1..3>
- ServerUrl: <S: 1, 256>
- Expression[1..15]: <S: 1, 256>

**Example:**
xCommand HttpFeedback Register FeedbackSlot:1 ServerUrl:10.47.19.41 Expression[1]:Status/Video Expression[2]:Status/Audio Expression[3]:Status/Call Expression[4]:Status/Conference OK
*r FeedbackRegisterResult (status=OK):
FeedbackSlot: 1
** end

**xCommand HttpFeedback Deregister**
Deregister XML feedback over HTTP(S).

**Parameters:**
- FeedbackSlot: <1..3>

**Example:**
xCommand HttpFeedback Deregister FeedbackSlot:1 OK
*r FeedbackDeregisterResult (status=OK):
FeedbackSlot: 1
** end

The Key commands

**xCommand Key Click**
Emulates a remote control key press, followed by a key release.

**Parameters:**
- Key(r): <0/1/2/3/4/5/6/7/8/9/C/Call/Disconnect/Down/F1/F2/F3/F4/F5/Grab/Home/Layout/Left/Mute/MuteMic/Ok/PhoneBook/Presentation/Right/Selfview/Square/SrcAux/SrcCamera/SrcDocCam/SrcPc/SrcVcr/Star/Up/VolumeDown/VolumeUp/ZoomIn/ZoomOut>

**Example:**
xCommand Key Click Key:Down
*r KeyClickResult (status=OK):
** end

**xCommand Key Press**
Emulates a remote control key press without releasing it. The Key Press command must be followed by a Key Release command to emulate releasing the key.

**Parameters:**
- Key(r): <0/1/2/3/4/5/6/7/8/9/C/Call/Disconnect/Down/F1/F2/F3/F4/F5/Grab/Home/Layout/Left/Mute/MuteMic/Ok/PhoneBook/Presentation/Right/Selfview/Square/SrcAux/SrcCamera/SrcDocCam/SrcPc/SrcVcr/Star/Up/VolumeDown/VolumeUp/ZoomIn/ZoomOut>

**Example:**
xCommand Key Press Key:Home
*r KeyPressResult (status=OK):
** end

**xCommand Key Release**
Emulates a remote control key release. The Key Release command is issued after a Key Press command.

**Parameters:**
- Key(r): <0/1/2/3/4/5/6/7/8/9/C/Call/Disconnect/Down/F1/F2/F3/F4/F5/Grab/Home/Layout/Left/Mute/MuteMic/Ok/PhoneBook/Presentation/Right/Selfview/Square/SrcAux/SrcCamera/SrcDocCam/SrcPc/SrcVcr/Star/Up/VolumeDown/VolumeUp/ZoomIn/ZoomOut>

**Example:**
xCommand Key Release Key:Home
*r KeyReleaseResult (status=OK):
** end
The Message command

**xCommand Message Alert Display**

Display a message on screen, for a specified duration of time (in seconds). NOTE! If Duration is not set, the command must be followed by xCommand Message Alert Clear.

Use the xFeedback commands to monitor the feedback from the user. Read more about the xFeedback commands in the API introduction on TANDBERG Developer Zone web site. Go to: http://developer.tandberg.com/web/guest/howtos/cseries-api/feedback.

Text: Enter the message to be displayed.
Duration: Set how long (in seconds) the message is to be displayed on the screen. If set to 0 (zero) the message will not disappear until a xCommand Message Alert Clear message has been sent.

**Parameters:**
- Text(r): <S: 0, 255>
- Duration: <0..3600>

**Example:**
xCommand Message Alert Display Text: “The meeting will end in 5 minutes.”
Duration: 20
OK
*r MessageAlertDisplayResult (status=OK):
** end

**xCommand Message Alert Clear**

Remove the message which was displayed using the xCommand Message Alert Display command. This is required when the Duration parameter is not set.

**Parameters:**
- No parameters.

**Example:**
xCommand Message Alert Clear
OK
*r MessageAlertClearResult (status=OK):
** end

The Message commands, cont...

**xCommand Message Prompt Display**

Display a small window on screen with a title, text and up to five options for response from the user. The message will display on screen until the user gives a response, or until the system receives the following command xCommand Message Prompt Clear.

Use the xFeedback commands to monitor the feedback from the user. Read more about the xFeedback commands in the API introduction on TANDBERG Developer Zone web site. Go to: TANDBERG Developer Zone.

Title: Enter the title of the message.
Text: Enter the message.
FeedbackId: To identify the feedback enter a FeedbackId.
Option.1 to Option.5: Enter the text to appear on the feedback options.

**Parameters:**
- Title: <S: 0, 255>
- Text(r): <S: 0, 255>
- FeedbackId: <S: 0, 255>
- Option.1: <S: 0, 255>
- Option.2: <S: 0, 255>
- Option.3: <S: 0, 255>
- Option.4: <S: 0, 255>
- Option.5: <S: 0, 255>

**Example:**
xCommand Message Prompt Display Title: “Meeting extension” Text: “The meeting is about to end. Do you want to extend the meeting?” Option.1: “No” Option.2: “Yes, extend with 5 minutes” Option.3: “Yes, extend with 10 minutes”
OK
*r MessagePromptDisplayResult (status=OK):
** end
The Message commands, cont...

**xCommand Message Prompt Clear**
Remove the window displayed using the xCommand Message Alert Display command.
Use the xFeedback commands to monitor the feedback from the user. Read more about the xFeedback commands in the API introduction on TANDBERG Developer Zone web site. Go to: http://developer.tandberg.com/web/guest/howtos/cseries-api/feedback.
FeedbackId: The FeedbackId corresponds to the FeedbackId given by the xCommand Message Prompt Display command.

**Parameters:**
- FeedbackId: <S: 0, 255>

**Example:**
```
xCommand Message Prompt Clear
OK
```
```
*r MessagePromptClearResult (status=OK):
** end
```

**xCommand Message Prompt Response**
Give a response to the xCommand Message Prompt Display.
Use the xFeedback commands to monitor the feedback from the user. Read more about the xFeedback commands in the API introduction on TANDBERG Developer Zone web site. Go to: TANDBERG Developer Zone.
FeedbackId: The FeedbackId corresponds to the FeedbackId given by the xCommand Message Prompt Display command.
OptionId: The OptionId corresponds to the OptionId given as possible responses in the xCommand Message Prompt Display command.

**Parameters:**
- FeedbackId: <S: 0, 255>
- OptionId: <0..5>

**Example:**
```
xCommand Message Prompt Response OptionId: 1
OK
```
```
*r MessagePromptResponseResult (status=OK):
** end
```

The Phonebook commands

**xCommand Phonebook Folder Add**
Add a folder to the local phonebook, where phonebook entries can be stored. Returns the FolderId (localGroupId-3), which is a unique Id of the folder.
Name(r): The name of the folder.
ParentFolderId: A unique identifier for the parent folder, which was created when a previous xCommand Phonebook Folder Add command was issued.

**Parameters:**
- Name(r): <S: 0, 255>
- ParentFolderId: <S: 0, 255>

**Example:**
```
xCommand Phonebook Folder Add Name: locationA
OK
```
```
*r PhonebookFolderAddResult (status=OK):
Name: localGroupId-3
** end
```

**xCommand Phonebook Folder Modify**
Modify an existing phonebook folder.
FolderId: A unique identifier for the folder, which was created when the xCommand Phonebook Folder Add command was issued.
Name(r): The name of the contact.
ParentFolderId: A unique identifier for the parent folder, which was created when the xCommand Phonebook Folder Add command was issued.

**Parameters:**
- FolderId(r): <S: 0, 255>
- Name: <S: 0, 255>
- ParentFolderId: <S: 0, 255>

**Example:**
```
xCommand Phonebook Folder Modify FolderId:localGroupId-3 Name:locationB
OK
```
```
*r PhonebookFolderModifyResult (status=OK):
** end
```
The Phonebook commands, cont...

**xCommand Phonebook Folder Delete**

Delete an existing folder from the local phonebook.
FolderId: A unique identifier for the folder, which was created when the xCommand Phonebook Folder Add command was issued.

**Parameters:**
- FolderId(r): <S: 0, 255>

**Example:**

```
xCommand Phonebook Folder Delete FolderId:localGroupId-3
OK
```

```
*r PhonebookFolderDeleteResult (status=OK):
** end
```

**xCommand Phonebook Contact Add**

Add new contact to the local phonebook. Stored internally in the system. Returns the ContactId (Name: localContactId-1), which is a unique id of the contact.

Name: The name of the contact.
FolderId: A unique identifier for the folder, which was created when the xCommand Phonebook Folder Add command was issued.
ImageURL: The URL to an image.
Title: The title of the contact.
Number: The phone number or address of the contact.
Protocol: Select H323 or SIP protocol.
CallRate: Set a call rate.
Device: Set which type of device to call to.

**Parameters:**
- Name(r): <S: 0, 255>
- FolderId: <S: 0, 255>
- ImageURL: <S: 0, 255>
- Title: <S: 0, 255>
- Number: <S: 0, 255>
- Protocol: <H323/SIP>
- CallRate: <0..65534>
- Device: <Mobile/Other/Telephone/Video>

**Example:**

```
xCommand Phonebook Contact Add Name:"John Doe" Number:12345
OK
```

```
*r PhonebookContactAddResult (status=OK):
Name: localContactId-1
** end
```
The Phonebook commands, cont...

xCommand Phonebook Contact Modify

Modify the contact details of an existing contact in the local phonebook.

ContactId: A unique identifier for the contact, which was created when the xCommand Phonebook Contact Add command was issued.
Name: The name of the contact.
FolderId: A unique identifier for the folder, which was created when the xCommand Phonebook Folder Add command was issued.
ImageURL: The URL to an image.
Title: The title of the contact.

Parameters:

- ContactId(r): <S: 0, 255>
- Name: <S: 0, 255>
- FolderId: <S: 0, 255>
- ImageURL: <S: 0, 255>
- Title: <S: 0, 255>

Example:

```
xCommand Phonebook Contact Modify ContactId:localContactId-1 Name:"John Doe - office"
OK
```
```
*r PhonebookContactModifyResult (status=OK):
** end
```

The Phonebook commands, cont...

xCommand Phonebook ContactMethod Add

Add details about the call setup to an existing contact in the local phonebook. Returns the ContactMethodId (Name: 1), which is a unique id of the contact method.

ContactId: A unique identifier for the contact, which was created when the xCommand Phonebook Contact Add command was issued.
Device: Set which type of device to call to.
Number(r): The phone number or address of the contact.
Protocol: Select H323 or SIP protocol.
CallRate: Set a call rate.

Parameters:

- ContactId(r): <S: 0, 255>
- Device: <Mobile/Other/Telephone/Video>
- Number(r): <S: 0, 255>
- Protocol: <H323/SIP>
- CallRate: <0..65534>

Example:

```
xCommand Phonebook ContactMethod Add ContactId:localContactId-2 Number:54321 Protocol:H323
OK
```
```
*r PhonebookContactMethodAddResult (status=OK):
Name: 1
** end
```

The Phonebook commands, cont...

xCommand Phonebook Contact Delete

Delete an existing contact from local phonebook.

ContactId: A unique identifier for the contact, which was created when the xCommand Phonebook Contact Add command was issued.

Parameters:

- ContactId(r): <S: 0, 255>

Example:

```
xCommand Phonebook Contact Delete ContactId:localContactId-1
OK
```
```
*r PhonebookContactDeleteResult (status=OK):
** end
```

The Phonebook commands, cont...

xCommand Phonebook ContactMethod Delete

Delete details about the call setup to an existing contact in the local phonebook.

ContactId: A unique identifier for the contact, which was created when the xCommand Phonebook Contact Add command was issued.

ContactMethodId: A unique identifier for the contact method, which was created when the xCommand Phonebook ContactMethod Add command was issued.

Parameters:

- ContactId(r): <S: 0, 255>
- ContactMethodId(r): <S: 0, 255>

Example:

```
xCommand Phonebook ContactMethod Delete ContactId:localContactId-2 ContactMethodId:1
OK
```
```
*r PhonebookContactMethodDeleteResult (status=OK):
** end
```
The Phonebook commands, cont...

**xCommand Phonebook Search**


**PhonebookId:** The value of the ID tag for which phonebook server to use. See xConfiguration Phonebook Server. Not necessary to use.

**PhonebookType:** Which phone book to search in. Either the local phone book or the corporate phonebook.

**SearchString:** Search for entries containing specified string (not begins with). If no FolderId is specified, the search will yield search results from ALL folders/phonebook directories. This parameter is optional for TC 2.0 and higher.

**SearchField:** Currently not in use

**FolderId:** Search only in the specified folder. FolderId (string) is listed in the ResultSet of a search result containing folders.

**Offset:** Get records starting with this offset in a search. Default 0. Used together with Limit to support paging.

**Limit:** Limit the number of records in the result set to this number. E.g. Limit: 10 will only give a ResultSet of 10 entries (Contacts + Folders) although the total number of hits may be greater.

**Parameters:**

- **PhonebookId:** <S: 0, 255>
- **PhonebookType:** <Corporate/Local>
- **SearchString:** <S: 0, 255>
- **SearchField:** <Name/Number>
- **FolderId:** <S: 0, 255>
- **Offset:** <0..65534>
- **Limit:** <0..65534>

**Example:**

```plaintext
xCommand Phonebook Search PhonebookType:Corporate Limit:2
FolderId:"corporate_001"
OK
* ResultSet ResultInfo TotalRows: 25
* ResultSet Contact 1 Name: "/tmp"
* ResultSet Contact 1 ContactId: "e_9664921"
* ResultSet Contact 1 ContactMethod 1 ContactMethodId: "1"
* ResultSet Contact 1 ContactMethod 1 Number: "H323:/John.Doe.Office@company.com"
* ResultSet Contact 1 ContactMethod 2 Number: "SIP:/John.Doe.Office@company.com"
* ResultSet Contact 1 ContactMethod 2 Protocol: SIP
* ResultSet Contact 2 Name: "AST Lounge"
* ResultSet Contact 2 ContactId: "e_9468400"
* ResultSet Contact 2 ContactMethod 1 ContactMethodId: "3"
* ResultSet Contact 2 ContactMethod 1 Number: "H323:John.Doe.Home@company.com"
* ResultSet Contact 2 ContactMethod 2 Number: "SIP:John.Doe.Home@company.com"
* ResultSet Contact 2 ContactMethod 2 Protocol: SIP
** end
```
The Presentation commands

**xCommand Presentation Start**
Open a media stream from the selected presentation source.
PresentationSource: Select the video input source to be used for presentation.

**Parameters:**
- PresentationSource: <1..5>

**Example:**
```plaintext
xCommand Presentation Start PresentationSource:2
OK
*r PresentationStartResult (status=OK):
** end
```

**xCommand Presentation Stop**
Stop the media stream from the presentation source.

**Parameters:**
- No parameters.

**Example:**
```plaintext
xCommand Presentation Stop
OK
*r PresentationStopResult (status=OK):
** end
```

The Preset commands

**xCommand Preset Store**
Store the current camera position. The system may hold 15 predefined camera positions (presets).

**Parameters:**
- PresetId(r): <1..15>
- Type(r): <All/Camera>
- Description: <S: 0, 255>

**Example:**
```plaintext
xCommand Preset Store PresetId:3 Type:Camera Description:"Left view"
OK
*r PresetStoreResult (status=OK):
** end
```

**xCommand Preset Activate**
Activate one of the local presets.

**Parameters:**
- PresetId(r): <1..15>

**Example:**
```plaintext
xCommand Preset Activate PresetId:3
OK
*r PresetActivateResult (status=OK):
** end
```

**xCommand Preset Clear**
Delete an existing preset.

**Parameters:**
- PresetId(r): <1..15>

**Example:**
```plaintext
xCommand Preset Clear PresetId:3
OK
*r PresetClearResult (status=OK):
** end
```
The Provisioning commands

**xCommand Provisioning StartUpgrade**

The codec software can be upgraded from the provisioning server. When starting the upgrade the software is automatically downloaded and installed. A reboot of the codec is required to complete the software upgrade.

AutoComplete On: The software upgrade is automatically completed, with a reboot of the codec.

AutoComplete Off: The software upgrade is not completed, and must be completed by running the xCommand Provisioning CompleteUpgrade command.

**Parameters:**
- AutoComplete: <On/Off>

**Example:**
```
xCommand Provisioning StartUpgrade AutoComplete:On
```
```
*r StartUpgradeResult (status=OK):
  ** end
```

**xCommand Provisioning CompleteUpgrade**

Completes the software upgrade. This will reboot the codec. NOTE! This command is required if the xCommand Provisioning StartUpgrade has been run with the AutoComplete set to Off.

**Parameters:**
- None

**Example:**
```
xCommand Provisioning CompleteUpgrade
```
```
*r CompleteUpgradeResult (status=OK):
  ** end
```

**xCommand Provisioning CancelUpgrade**

Cancel a software update in progress.

**Parameters:**
- None

**Example:**
```
xCommand Provisioning CancelUpgrade
```
```
*r CancelUpgradeResult (status=OK):
  ** end
```

The SString Send command

**xCommand SStringSend**

Send data to the far end, e.g. for control systems. Uses the H.224 data channel (UDP).

Message: Enter the message to be sent to the far end.

CallId: The CallId is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.

**Parameters:**
- Message(r): <S: 1, 256>
- CallId: <0..65534>

**Example:**
```
xCommand SStringSend CallId:4 Message:"This is a test"
```
```
*r SStringSendResult (status=OK):
  ** end
```
The Standby command

**xCommand Standby Activate**
Set the system in standby mode, which will turn off the video outputs and put the camera into sleep mode.

**Parameters:**
No parameters.

**Example:**
```
xCommand Standby Activate
*r ActivateResult (status=OK):
** end
```

**xCommand Standby Deactivate**
Bring the system out of standby mode.

**Parameters:**
No parameters.

**Example:**
```
xCommand Standby Deactivate
*r DeactivateResult (status=OK):
** end
```

**xCommand Standby ResetTimer**
The system will enter standby mode, after the time specified by Delay parameter (in minutes), when left idle. Note that next time the system is left idle, the standby delay set by the Advanced configuration menu (or by xConfiguration Standby Delay), will be valid.

**Parameters:**
Delay: <1..480>

**Example:**
```
xCommand Standby ResetTimer Delay:10
*r ResetResult (status=OK):
** end
```

The SystemUnit commands

**xCommand SystemUnit OptionKey Add**
Add an option key to support additional features.

**Parameters:**
Key(r): <S: 16, 24>

**Example:**
```
xCommand SystemUnit OptionKey Add Key:******************
*r OptionKeyResult (status=OK):
** end
```

**xCommand SystemUnit ReleaseKey Add**
Add software release key. Used to enable new software (applicable for main software releases).

**Parameters:**
Key(r): <S: 16, 24>

**Example:**
```
xCommand SystemUnit ReleaseKey Add Key:******************
*r ReleaseKeyResult (status=OK):
** end
```

**xCommand SystemUnit AdminPassword Set**
Set an administrator password to restrict access the codec. After a restart of the codec this password will also apply to the web interface.

**Parameters:**
Password(r): <S: 0, 255>

**Example:**
```
xCommand SystemUnit AdminPassword Set Password:**********
*r AdminPasswordSetResult (status=OK):
** end
```
The System Unit commands, cont...

**xCommand SystemUnit MenuPassword Set**
Set a menu password to restrict access to Advanced menu settings.

**Parameters:**
- Password(r): <S: 0, 255>

**Example:**
```
xCommand SystemUnit MenuPassword Set Password:**********
```
```
*r MenuPasswordSetResult (status=OK):
** end
```

**xCommand SystemUnit MenuPassword Validate**
Validate that the supplied password is correct.

**Parameters:**
- Password(r): <S: 0, 255>

**Example:**
```
xCommand SystemUnit MenuPassword Validate Password:**********
```
```
*r MenuPasswordValidateResult (status=OK):
** end
```

**xCommand SystemUnit DateTime Set**
Set the date and time for the system, if not available from NTP (Network Time Protocol).

**Parameters:**
- Year: <2008..2037>
- Month: <1..12>
- Day: <1..31>
- Hour: <0..23>
- Minute: <0..59>
- Second: <0..59>

**Example:**
```
xCommand SystemUnit DateTime Set Year:2009 Month:7 Day:3 Hour:12 Minute:0 Second:0
```
```
*r DateTimeSetResult (status=OK):
** end
```

**xCommand SystemUnit DateTime Get**
Read the time and date from the system.

**Parameters:**
- No parameters.

**Example:**
```
xCommand SystemUnit DateTime get
```
```
*r DateTimeGetResult (status=OK):
Year: 2009
Month: 7
Day: 3
Hour: 12
Minute: 0
Second: 0
** end
```

**xCommand SystemUnit ConfigurationProfile SaveCurrentConfigurationAs**
Save the current system settings into a configuration profile. Assign a name to the new profile. The name is the unique identifier of the profile.

**Parameters:**
- Name(r): <S: 0, 255>

**Example:**
```
xCommand SystemUnit ConfigurationProfile SaveCurrentConfigurationAs Name:"My ConfigurationProfile _1"
```
```
*r ConfigurationProfileSaveCurrentConfigurationResult (status=OK):
** end
```
The System Unit commands, cont...

**xCommand SystemUnit ConfigurationProfile List**

List configuration profiles that has been stored in the system.

**Parameters:**

No parameters.

**Example:**

```plaintext
xCommand SystemUnit ConfigurationProfile List
```

```plaintext
 r ConfigurationProfileListResult (status=OK):
   Profile: My_ConfigurationProfile_1
   Profile: My_ConfigurationProfile_2
** end
```

**xCommand SystemUnit ConfigurationProfile Change**

Select a previously saved configuration profile. Will be active after next system boot. Can be cancelled before the reboot, by the xCommand SystemUnit ConfigurationProfile CancelChange command.

**Parameters:**

- Name(r): <S: 0, 255>

**Example:**

```plaintext
xCommand SystemUnit ConfigurationProfile Change Name: "My_ConfigurationProfile_1"
```

```plaintext
 r ConfigurationProfileChangeResult (status=OK):
   Warning: New configuration profile will be active after next boot.
** end
```

**xCommand SystemUnit ConfigurationProfile CancelChange**

Cancel the xCommand SystemUnit ConfigurationProfile Change command, that would otherwise take effect after next system boot.

**Parameters:**

- Name(r): <S: 0, 255>

**Example:**

```plaintext
xCommand SystemUnit ConfigurationProfile CancelChange Name: "My_ConfigurationProfile_1"
```

```plaintext
 r ConfigurationProfileCancelChangeResult (status=OK):
** end
```

**xCommand SystemUnit FactoryReset**

Reset the codec to factory default settings, followed by an automatic reboot of the codec.

The call logs will be deleted and all system parameters will be reset to default values. All files that have been uploaded to the codec will be deleted. The Release key and Option key will not be affected.

**Parameters:**

- Confirm(r): <Yes>

**Example:**

```plaintext
xCommand SystemUnit FactoryReset Confirm: Yes
```

```plaintext
 r FactoryResetConfirmResult (status=OK):
** end
```
The Video commands

**xCommand TStringSend**

Send data to far end, e.g. for Telepresence control systems. Uses the H.245 control channel (TCP). Works with H.323 calls only.

Message: Enter the message to be sent to the far end. CallId: The CalId is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.

**Parameters:**
- CallId(r): <0..65534>
- Message(r): <S: 1, 1450>

**Example:**

```plaintext
xCommand TStringSend CallId:1 Message: "This is an example"
```

The Video commands, cont...

**xCommand Video PictureLayoutSet**

Select the screen layout mode.

**Parameters:**
- CallId(r): <0..65534>
- LayoutFamily(r): <Auto/Equal/Fullscreen/PresentationLargeSpeaker/PresentationSmallSpeaker>

**Example:**

```plaintext
xCommand Video PictureLayoutSet LayoutFamily: Equal
```

**xCommand Video Layout AutoMode SetLayoutFamily**

Set the default picture layout (defined by the LayoutFamily parameter) for the specified situation (defined by the Monitors, Selfview, Presentation and NumberOfCalls parameters).

**Parameters:**
- Monitors(r): <Dual/Single>
- Selfview(r): <On/Off>
- Presentation(r): <16_9/4_3/Off>
- NumberOfCalls(r): <0..3>
- LayoutFamily(r): <Auto/Equal/Fullscreen/PresentationLargeSpeaker/PresentationSmallSpeaker>

**Example:**

```plaintext
xCommand Video Layout AutoMode SetLayoutFamily Monitors: Single Selfview: On
```
xCommand Video Layout AutoMode Reset

Reset all default picture layouts to the system defaults.

Parameters:
No parameters.

Example:
```
xCommand Video Layout AutoMode Reset
*r VideoLayoutAutoModeResetResult (status=OK):
** end
```

xCommand Video Layout AutoMode List

Shows the current default picture layouts.

Parameters:
No parameters.

Example:
```
xCommand Video Layout AutoMode List
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_4_3 1 NumberOfCalls 0 LayoutFamily: "fullscreen"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_4_3 1 NumberOfCalls 1 LayoutFamily: "fullscreen"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_4_3 1 NumberOfCalls 2 LayoutFamily: "equal"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_4_3 1 NumberOfCalls 3 LayoutFamily: "equal"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_16_9_1 NumberOfCalls 0 LayoutFamily: "fullscreen"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_16_9_1 NumberOfCalls 1 LayoutFamily: "presentationsmallspeaker"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_16_9_1 NumberOfCalls 2 LayoutFamily: "presentationsmallspeaker"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_16_9_1 NumberOfCalls 3 LayoutFamily: "presentationsmallspeaker"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_4_3 1 NumberOfCalls 0 LayoutFamily: "fullscreen"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_4_3 1 NumberOfCalls 1 LayoutFamily: "fullscreen"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_4_3 1 NumberOfCalls 2 LayoutFamily: "fullscreen"
*r VideoLayoutAutoModeListResult LayoutAutoMode 1 SingleMonitor 1 SelfviewOff 1 Presentation_4_3 1 NumberOfCalls 3 LayoutFamily: "fullscreen"

//Comment: Displays the complete list of the auto mode layouts.//
** end
```

xCommand Video Layout AutoModeRemote SetLayoutFamily

Set the remote layout family in auto mode.
PresentationCapability: Set the presentation capability mode.
Presentation: Set the presentation mode.
NumberOfCalls: Set the number of calls from 1 to 3.
LayoutFamily: Select a layout family.

Parameters:
- PresentationCapability(r): <On/Off>
- Presentation: <On/Off>
- NumberOfCalls(r): <1..3>
- LayoutFamily(r): <Equal/FollowLocal/FullScreen/PresentationLargeSpeaker/PresentationSmallSpeaker>

Example:
```
xCommand Video Layout AutoModeRemote SetLayoutFamily PresentationCapability: On Presentation: Off NumberOfCalls: 1 LayoutFamily: Equal
*r VideoLayoutAutoModeRemoteSetLayoutFamilyResult (status=OK):
** end
```

xCommand Video Layout AutoModeRemote Reset

Reset the remote layouts in auto mode to the system defaults.

Parameters:
No parameters.

Example:
```
xCommand Video Layout AutoModeRemote Reset
*r VideoLayoutAutoModeRemoteResetResult (status=OK):
** end
```
The Video commands, cont...

**xCommand Video Layout AutoModeRemote List**

Shows a list of the remote layout families in auto mode.

**Parameters:**
No parameters.

**Example:**
```
xCommand Video Layout AutoModeRemote List
```
```
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithPresentationCapability 1 PresentationOff 1 NumberOfCalls 1 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithPresentationCapability 1 PresentationOff 1 NumberOfCalls 2 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithPresentationCapability 1 PresentationOff 1 NumberOfCalls 3 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithPresentationCapability 1 PresentationOn 1 NumberOfCalls 1 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithPresentationCapability 1 PresentationOn 1 NumberOfCalls 2 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithPresentationCapability 1 PresentationOn 1 NumberOfCalls 3 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithoutPresentationCapability 1 PresentationOff 1 NumberOfCalls 1 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithoutPresentationCapability 1 PresentationOff 1 NumberOfCalls 2 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithoutPresentationCapability 1 PresentationOff 1 NumberOfCalls 3 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithoutPresentationCapability 1 PresentationOn 1 NumberOfCalls 1 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithoutPresentationCapability 1 PresentationOn 1 NumberOfCalls 2 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithoutPresentationCapability 1 PresentationOn 1 NumberOfCalls 3 LayoutFamily: "followlocal"
xVideoLayoutAutoModeRemoteListResult LayoutAutoModeRemote 1
  WithoutPresentationCapability 1 PresentationOn 1 NumberOfCalls 3 LayoutFamily: "followlocal"
```

**xCommand Video Layout ListLayoutFamily**

Display a list of the layout families. If no LayoutFamilyId or DescriptorId is given, all layout families will be listed. Run the xCommand Video Layout ListLayoutFamily command to see the LayoutFamilyIds or DescriptorIds.

**Parameters:**
- LayoutFamilyId: A unique identifier of the layout family.
- DescriptorId: A unique identifier of the descriptor.

**Example:**
```
xCommand Video Layout ListLayoutFamily
  //Comment: Displays the complete list of layout families.//
```

**xCommand Video Layout ListLayoutGraphic**

Displays a list of all the defined layout graphics.

**Parameters:**
- LayoutGraphicId: A unique identifier of the layout graphic.
- FrameId: A unique identifier for the frame, which was created when the xCommand Video Layout Frame Add command was issued.

**Example:**
```
xCommand Video Layout ListLayoutGraphic
  //Comment: Displays the complete list of layout graphics.//
```

**end**
The Video commands, cont...

**xCommand Video Layout Add**

NOTE: This command is not supported on Codec C40.

Add a new empty video layout composition. Returns the LayoutId. When the command has been issued you can use the result, the LayoutId of the newly created layout, to add frames with different video sources to the layout.

**Parameters:**

- LayoutId: <1..2147483647>

**Example:**

```
xCommand Video Layout Add
OK
*r VideoLayoutAddResult (status=OK):
    LayoutId: 1
** end
```

**xCommand Video Layout Remove**

NOTE: This command is not supported on Codec C40.

Removes an existing video layout.

LayoutId: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.

**Parameters:**

- LayoutId(r): <1..2147483647>

**Example:**

```
xCommand Video Layout Remove LayoutId: 1
*r VideoLayoutRemoveResult (status=OK):
** end
```

**xCommand Video Layout RemoveAll**

NOTE: This command is not supported on Codec C40.

Removes all the existing video layouts.

**Parameters:**

- No parameters.

**Example:**

```
xCommand Video Layout RemoveAll
*r VideoLayoutRemoveAllResult (status=OK):
** end
```

**xCommand Video Layout Reset**

NOTE: This command is not supported on Codec C40.

Resets all the layout compositions to factory default.

**Parameters:**

- No parameters.

**Example:**

```
xCommand Video Layout Reset
*r VideoLayoutResetResult (status=OK):
** end
```
The Video commands, \textit{cont...}

\textbf{xCommand Video Layout Frame Add}

\textbf{NOTE:} This command is not supported on Codec C40.

Add a Video frame to an existing layout. Returns the FrameId. Select size and position of the frame, and the video source to be shown in the frame.

- \textbf{LayoutId}: A unique identifier for the layout, which was created when the \textit{xCommand Video Layout Add} command was issued.
- \textbf{FrameId}: A unique identifier of the frame.
- \textbf{PositionX}: The top-left X position of the frame.
- \textbf{PositionY}: The top-left Y position of the frame.
- \textbf{Width}: The width of the frame.
- \textbf{Height}: The height of the frame.
- \textbf{Layer}: The stacking order of the frames (1-5). Must be unique per frame per layout. Layer 5 will be on top.
- \textbf{Border}: Select whether or not the frame should have a border.
- \textbf{VideoSourceId}: A unique identifier of the video source.
  - If the VideoSourceType is set to RemoteMain the VideoSourceId is the CallId of the remote site.
  - If the VideoSourceType is set to LocalInput the VideoSourceId is the video input SourceId.
  - Otherwise set to 1.

\textbf{VideoSourceType}: Select the video source to be used as the content of the frame. The parameter arguments are explained below:

- Graphic: Not in use.
- LocalInput: Select one of the local input sources.
- LocalMain: Select what is currently the main source on the local side.
- LocalPresentation: Select what is currently the default presentation source on the local side.
- MostSpeaking: Select the current loudest speaker in the conference.
- OtherMain: Select another remote site in a call. This involves logic in the codec to ensure that a site does not see itself. You can add several frames with OtherMain. The source and the layout engine will automatically populate the frame content with one of the other sites main video, making sure that you do not see the same site more than once, nor yourself.
- OwnMain: Select that each site can see its own selfview.
- Presentation: Select what is currently the presentation source (there is always only one active H.239 content in a conference).
- RemoteMain: Select the main source of remote site. The CallId must then be specified in the VideoSourceId parameter.
- RemotePresentation: Select the presentation source of remote site. The CallId must then be specified in the VideoSourceId parameter.
- VideoFile: Not in use.

**Example:**

\begin{verbatim}
xCommand Video Layout Frame Add LayoutId:1 PositionX:100 PositionY:100 Width:9800 Height:9800 Layer:1 Border:off VideoSourceType:localInput VideoSourceId:1
\end{verbatim}

OK

* \textbf{VideoLayoutFrameAddResult (status=OK)}:

  FrameId: 1

** end

The Video commands, \textit{cont...}

\textbf{xCommand Video Layout Frame Remove}

\textbf{NOTE:} This command is not supported on Codec C40.

Remove a video frame from an existing layout.

- \textbf{LayoutId}: A unique identifier for the layout, which was created when the \textit{xCommand Video Layout Add} command was issued.
- \textbf{FrameId}: A unique identifier of the frame, which was created when the \textit{xCommand Video Frame Add} command was issued.

**Parameters:**

- \textit{LayoutId}: \textit{r} <1..2147483647>
- \textbf{FrameId}: \textit{r} <1..65535>

**Example:**

\begin{verbatim}
xCommand Video Layout Frame Remove LayoutId:1 FrameId:1
\end{verbatim}

* \textbf{VideoLayoutFrameRemoveResult (status=OK)}:

** end
The Video commands, cont...

**xCommand Video Layout Frame Update**

NOTE: This command is not supported on Codec C40.

Change the details of an existing frame in a video layout.

- **LayoutId**: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.
- **FrameId**: A unique identifier of the frame.
- **PositionX**: The top-left X position of the frame.
- **PositionY**: The top-left Y position of the frame.
- **Width**: The width of the frame.
- **Height**: The height of the frame.
- **Layer**: The stacking order of the frames (1-5). Must be unique per frame per layout. Layer 5 will be on top.
- **Border**: Select whether or not the frame should have a border.
- **VideoSourceId**: A unique identifier of the video source.
  - If the VideoSourceType is set to RemoteMain the VideoSourceId is the CallId of the remote site.
  - If the VideoSourceType is set to LocalInput the VideoSourceId is the video input SourceId.
  - Otherwise set to 1.
- **VideoSourceType**: Select the video source to be used as the content of the frame. The parameter arguments are explained below:
  - **Graphic**: Not in use.
  - **LocalInput**: Select one of the local input sources.
  - **LocalMain**: Select what is currently the main source on the local side.
  - **LocalPresentation**: Select what is currently the default presentation source on the local side.
  - **MostSpeaking**: Select the current loudest speaker in the conference.
  - **OtherMain**: Select another remote site in a call. This involves logic in the codec to ensure that a site does not see itself. You can add several frames with OtherMain. The source and the layout engine will automatically populate the frame content with one of the other sites main video, making sure that you do not see the same site more than once, nor yourself.
  - **OwnMain**: Select that each site can see its own selfview.
  - **Presentation**: Select what is currently the presentation source (there is always only one active H.239 content in a conference).
  - **RemoteMain**: Select the main source of remote site. The CallId must then be specified in the VideoSourceId parameter.
  - **RemotePresentation**: Select the presentation source of remote site. The CallId must then be specified in the VideoSourceId parameter.
  - **VideoFile**: Not in use.

**Parameters:**

- **LayoutId**: <1..2147483647>
- **FrameId**: <1..65535>
- **PositionX**: <0..10000>
- **PositionY**: <0..10000>
- **Width**: <1..10000>
- **Height**: <1..10000>
- **Layer**: <1..5>
- **Border**: <On/Off>
- **VideoSourceId**: <0..256>
- **VideoSourceType**: Graphic/LocalInput/LocalMain/LocalPresentation/MostSpeaking/OtherMain/OwnMain/Presentation/RemoteMain/RemotePresentation/VideoFile

**Example:**

```
xCommand Video Layout Frame Update LayoutId:1 FrameId:1
VideoSourceId:1
VideoSourceType:localInput
OK

* xVideoLayoutFrameUpdateResult (status=OK):
** end
```
The Video commands, cont...

**xCommand Video Layout Assign**

NOTE: This command is not supported on Codec C40.

Assign an existing layout to any local or remote output. The defined video composition will appear on the specified local output or in the specified remote video stream.

- **Callid**: A unique identifier for the call. The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the Callid.
- **OutputId**: A unique identifier for the local output, which was created when the xCommand Audio LocalOutput Add command was issued.
- **LayoutId**: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.

**Parameters:**
- **CallId(r)**: <0..65534>
- **OutputId(r)**: <0..65534>
- **LayoutId(r)**: <1..2147483647>

**Example:**
```
xCommand Video Layout Assign CallId:1 OutputId:1 LayoutId:2
OK
```

```
*r VideoLayoutAssignResult {status=OK}:
** end
```

**xCommand Video Layout AssignLocalOutput**

NOTE: This command is not supported on Codec C40.

Assign an existing layout to a local output. The defined video composition will appear on the specified local output.

- **OutputId**: A unique identifier for the local output, which was created when the xCommand Audio LocalOutput Add command was issued.
- **LayoutId**: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.

**Parameters:**
- **OutputId(r)**: <0..65534>
- **LayoutId(r)**: <1..2147483647>

**Example:**
```
xCommand Video Layout AssignLocalOutput OutputId:1 LayoutId:2
OK
```

```
*r VideoLayoutAssignLocalOutputResult {status=OK}:
** end
```

**xCommand Video Layout AssignCall**

NOTE: This command is not supported on Codec C40.

Assign the call layout (main stream) to the remote output. The defined video composition will appear on the remote main video stream.

- **Callid**: A unique identifier for the call. The CallID is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the Callid.
- **LayoutId**: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.

**Parameters:**
- **CallId(r)**: <0..65534>
- **LayoutId(r)**: <1..2147483647>

**Example:**
```
xCommand Video Layout AssignCall Callid:1 LayoutId:2
OK
```

```
*r VideoLayoutAssignCallResult {status=OK}:
** end
```

**xCommand Video Layout AssignPresentation**

NOTE: This command is not supported on Codec C40.

Assign the presentation layout (dual stream) to the remote output. The defined video composition will appear on the remote dual video stream.

- **LayoutId**: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.

**Parameters:**
- **LayoutId(r)**: <1..2147483647>

**Example:**
```
xCommand Video Layout AssignPresentation LayoutId:2
OK
```

```
*r VideoLayoutAssignPresentationResult {status=OK}:
** end
```
The Video commands, cont...

**xCommand Video Layout UnAssign**

NOTE: This command is not supported on Codec C40.

Remove the defined video layout, and go back to default.

CallId: A unique identifier for the call. The CallId is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.

OutputId: A unique identifier for the local output, which was created when the xCommand Audio LocalOutput Add command was issued.

Parameters:

- **CallId(r):** <0..65534>
- **OutputId(r):** <0..65534>

Example:

```
xCommand Video Layout UnAssign CallId: 1 OutputId: 1
  *r VideoLayoutUnassignResult (status=OK):
  ** end
```

**xCommand Video Layout UnAssignLocalOutput**

NOTE: This command is not supported on Codec C40.

Remove the defined video layout, and go back to default.

OutputId: A unique identifier for the local output, which was created when the xCommand Audio LocalOutput Add command was issued.

LayoutId: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.

Parameters:

- **OutputId(r):** <0..65534>
- **LayoutId(r):** <1..2147483647>

Example:

```
xCommand Video Layout UnAssignLocalOutput OutputId:1 LayoutId:2
  OK
  *r VideoLayoutUnAssignLocalOutputResult (status=OK):
  ** end
```

**xCommand Video Layout UnAssignCall**

NOTE: This command is not supported on Codec C40.

Remove the defined video layout, and go back to default.

CallId: A unique identifier for the call. The CallId is returned when the xCommand Dial command is run. During the call you can run the xStatus Call command to see the CallId.

LayoutId: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.

Parameters:

- **CallId(r):** <0..65534>
- **LayoutId(r):** <1..2147483647>

Example:

```
xCommand Video Layout UnAssignCall CallId:1 LayoutId:2
  OK
  *r VideoLayoutUnAssignCallResult (status=OK):
  ** end
```

**xCommand Video Layout UnAssignPresentation**

NOTE: This command is not supported on Codec C40.

Remove the defined video layout, and go back to default.

LayoutId: A unique identifier for the layout, which was created when the xCommand Video Layout Add command was issued.

Parameters:

- **LayoutId(r):** <1..2147483647>

Example:

```
xCommand Video Layout UnAssignPresentation LayoutId:2
  OK
  *r VideoLayoutUnAssignPresentationResult (status=OK):
  ** end
```
The Experimental commands
The Experimental commands can be used ‘as is’ and will not be further documented. NOTE! The Experimental commands WILL change.

**xCommand Experimental Audio LocalOutput Unmute**

The Experimental commands can be used ‘as is’. NOTE! The Experimental settings WILL change.

Unmute the local audio output.

Conferenceld: A unique identifier for the conference, which was created when the xCommand Dial was issued.

**Parameters:**

Conferenceld(r): <0..65534>

**Example:**

```plaintext
xCommand Experimental Audio LocalOutput Unmute ConferenceID:1
*r AudioLocalOutputUnmuteResult (status=OK):
** end
```

**xCommand Experimental Audio StereoEchoCancellation**

The Experimental commands can be used ‘as is’. NOTE! The Experimental settings WILL change.

Turn on/off the stereo echo cancellation.

**Parameters:**

Mode(r): <On/Off>

**Example:**

```plaintext
xCommand Experimental Audio StereoEchoCancellation Mode: Off
*r AudioStereoEchoCancellationModeResult (status=OK):
** end
```
Chapter 5

Description of the xStatus commands
The following pages will list an example of the xStatus commands and the response. Status type commands returns information about the system and system processes. You can query all information or just some of it.

We recommend you visit the TANDBERG web site regularly for updated versions of the manual. Go to: http://www.tandberg.com/doc

The Audio status

xStatus Audio

Shows the top level overview of the audio status. The identities of the LocalInput, RemoteInput, LocalOutput and RemoteOutput are used when querying additional information.

Example:

```
xStatus Audio
*s Audio Microphones Mute: Off
*s Audio Volume: 60
*s Audio Input LocalInput 1 Name: “Microphone”
*s Audio Input LocalInput 1 MixerMode: “GainShared”
*s Audio Input LocalInput 1 Mute: “Off”
*s Audio Input LocalInput 1 Channels: 1
*s Audio Input LocalInput 1 AGC: “On”
*s Audio Input LocalInput 1 Connector: “Microphone.1”
*s Audio Input LocalInput 1 Connector: “Microphone.2”
*s Audio Input LocalInput 1 Connector: “Microphone.3”
*s Audio Input LocalInput 1 Connector: “Microphone.4”
*s Audio Input LocalInput 2 Name: “PC input”
*s Audio Input LocalInput 2 MixerMode: “Fixed”
*s Audio Input LocalInput 2 Mute: “Off”
*s Audio Input LocalInput 2 Channels: 2
*s Audio Input LocalInput 2 AGC: “Off”
*s Audio Input LocalInput 2 Connector: “Line.1”
*s Audio Input LocalInput 2 Connector: “Line.2”
*s Audio Input LocalInput 3 Name: “HDMI input”
*s Audio Input LocalInput 3 MixerMode: “Fixed”
*s Audio Input LocalInput 3 Mute: “Off”
*s Audio Input LocalInput 3 Channels: 2
*s Audio Input LocalInput 3 AGC: “Off”
```

End of example.

** end
The Audio status, cont...

**xStatus Audio Microphones Mute**
Shows the microphones mute mode.

Value space of the result returned:
<On/Off>

Example:
```
xStatus Audio Microphones Mute
*s Audio Microphones Mute: Off
** end
```

**xStatus Audio Volume**
Shows the volume level (dB) of the loudspeaker output.

Value space of the result returned:
<0..100>

Example:
```
xStatus Audio Volume
*s Audio Volume: 70
** end
```

**xStatus Audio Input LocalInput [1..n] Name**
Shows the name of the local input. A local input is a mix of input connectors. You can run the command xStatus Audio Input LocalInput to find the identity of the input.

Value space of the result returned:
<S: 0, 255>

Example:
```
xStatus Audio Input LocalInput 1 Name
*s Audio Input LocalInput 1 Name: "Microphone"
** end
```

**xStatus Audio Input LocalInput [1..n] MixerMode**
Shows how the local input connectors have been mixed together. You can run the command xStatus Audio Input LocalInput to find the identity of the input.

- Auto: The microphone with the strongest speaker is active and the others are strongly attenuated.
- Fixed: The input connector signals are mixed together with equal gains.
- GainShared: The microphones are given a normalized gain factor relative to the strongest speaker before being mixed together.

Value space of the result returned:
<Auto/Fixed/GainShared>

Example:
```
xStatus Audio Input LocalInput 1 MixerMode
*s Audio Input LocalInput 1 MixerMode: "Auto"
** end
```

**xStatus Audio Input LocalInput [1..n] Mute**
Shows the mute mode for the local inputs. A local input is a mix of input connectors. You can run the command xStatus Audio Input LocalInput to find the identity of the input.

Value space of the result returned:
<On/Off>

Example:
```
xStatus Audio Input LocalInput 1 Mute
*s Audio Input LocalInput 1 Mute: "Off"
** end
```

**xStatus Audio Input LocalInput [1..n] Channels**
States if the local input channels are mixed into into a mono signal (1) or stereo signal (2). You can run the command xStatus Audio Input LocalInput to find the identity of the input.

Value space of the result returned:
<1..2>

Example:
```
xStatus Audio Input LocalInput 1 Channels
*s Audio Input LocalInput 1 Channels: 1
** end
```
The Audio status, *cont...

**xStatus Audio Input LocalInput [1..n] AGC**

Shows the AGC (Automatic Gain Control) mode on the local input.
You can run the command `xStatus Audio Input LocalInput` to find the identity of the input.

**Value space of the result returned:**

- `<On/Off>`

**Example:**

```
xStatus Audio Input LocalInput 1 AGC
*s Audio Input LocalInput 1 AGC: "On"
** end
```

**xStatus Audio Input LocalInput [1..n] Connector**

Lists the connectors that are attached to the local input.
You can run the command `xStatus Audio Input LocalInput` to find the identity of the input.

**Value space of the result returned:**

- `<Microphone.1/../Microphone.4/Line.1/Line.2/HDMI.2>`

**Example:**

```
xStatus Audio Input LocalInput 1 Connector
*s Audio Input LocalInput 1 Connector: "Microphone.1"
** end
```

**xStatus Audio Input RemoteInput [1..n] CallId**

Shows the CallId for the remote audio input.
You can run the command `xStatus Audio Input RemoteInput` to find the identity of the input.

**Value space of the result returned:**

- `<0..65534>`

**Example:**

```
xStatus Audio Input RemoteInput 8 CallId
*s Audio Input RemoteInput 8 CallId: 28
** end
```

**xStatus Audio Input RemoteInput [1..n] AGC**

Shows the AGC (Automatic Gain Control) mode on the remote input.
You can run the command `xStatus Audio Input RemoteInput` to find the identity of the input.

**Value space of the result returned:**

- `<On/Off>`

**Example:**

```
xStatus Audio Input RemoteInput 8 AGC
*s Audio Input RemoteInput 8 AGC: "Off"
** end
```

**xStatus Audio Output LocalOutput [1..n] Name**

Shows the name of the local output.
You can run the command `xStatus Audio Output LocalOutput` to find the identity of the output.

**Value space of the result returned:**

- `<S: 0, 255>`

**Example:**

```
xStatus Audio Output LocalOutput 4 Name
*s Audio Output LocalOutput 4 Name: "MyLocalOutput1"
** end
```

**xStatus Audio Output LocalOutput [1..n] Loudspeaker**

Shows the Loudspeaker mode on the local output. If one or more of the output connectors that are attached to the local output are connected to a loudspeaker, then this signal should be a reference signal to the echo canceller and Loudspeaker should be set to On.
You can run the command `xStatus Audio Output LocalOutput` to find the identity of the output.

**Value space of the result returned:**

- `<On/Off>`

**Example:**

```
xStatus Audio Output LocalOutput 4 Loudspeaker
*s Audio Output LocalOutput 4 Loudspeaker: "Off"
** end
```
The Audio status, cont...

xStatus Audio Output LocalOutput [1..n] Channels
States if the local output channels are mixed into a mono signal (1) or stereo signal (2).
You can run the command xStatus Audio Output LocalOutput to find the identity of the output.

Value space of the result returned:
<1..2>

Example:
```
xStatus Audio Output LocalOutput 4 Channels
*s Audio Output LocalOutput 4 Channels: "1"
** end
```

xStatus Audio Output LocalOutput [1..n] Connector
Lists the connectors that are attached to the local output.
You can run the command xStatus Audio Output LocalOutput to find the identity of the output.

Value space of the result returned:
<Line.1/Line.2/HDMI.1>

Example:
```
xStatus Audio Output LocalOutput 4 Connector
*s Audio Output LocalOutput 4 Connector: "Line.1"
** end
```

xStatus Audio Output LocalOutput [1..n] Input [1..n] Gain
Shows the gain (dB) on the input, when it is connected to the local output. Range from -54 dB to 15 dB, where -54 dB equals Off.
You can run the command xStatus Audio Output LocalOutput to find the identity of the output and input.

Value space of the result returned:
<-54..15>

Example:
```
xStatus Audio Output LocalOutput 4 Input 2 Gain
*s Audio Output LocalOutput 4 Input 2 Gain: 0
** end
```

xStatus Audio Output RemoteOutput [1..n] CallId
Shows the CallId for the remote audio output.
You can run the command xStatus Audio Output RemoteOutput to find the identity of the output.

Value space of the result returned:
<0..65534>

Example:
```
xStatus Audio Output RemoteOutput 9 CallId
*s Audio Output RemoteOutput 9 CallId: 28
** end
```

xStatus Audio Output RemoteOutput [1..n] Input [1..n] Gain
Shows the gain (dB) on the input, when input is connected to the remote output. Range from -54 dB to 15 dB, where -54 dB equals Off.
You can run the command xStatus Audio Output RemoteOutput to find the identity of the output and input.

Value space of the result returned:
<-54..15>

Example:
```
xStatus Audio Output RemoteOutput 9 Input 1 Gain
*s Audio Output RemoteOutput 9 Input 1 Gain: 0
** end
```

xStatus Audio Module [1..n] Type
Shows the audio module type. If the module type is DigitalNAM (Digital Natural Audio Module) you can also read the SoftwareId and HardwareId.

Value space of the result returned:
<DigitalNAM/Unknown>

Example:
```
xStatus Audio Module 1 Type
*s Audio Module 1 Type: DigitalNAM
** end
```
The Audio status, cont...

**xStatus Audio Module [1..n] SoftwareID**
Shows the SoftwareID of the DNAM dsp software.

*Value space of the result returned:*
<Integer value>

**Example:**
```
xStatus Audio Module 1 SoftwareID
*s Audio Module 1 SoftwareID: "114"
** end
```

**xStatus Audio Module [1..n] HardwareID**
Shows the DNAM HardwareID.

*Value space of the result returned:*
<S: 0, 255>

**Example:**
```
xStatus Audio Module 1 HardwareID
*s Audio Module 1 HardwareID: "B40F69"
** end
```

**xStatus Audio Module [1..n] Connector**
Shows which audio output connector the audio module is attached to.

*Value space of the result returned:*
<Line_out.1/Line_out.2>

**Example:**
```
xStatus Audio Module 1 Connector
*s Audio Module 1 Connector: "Line_out.1"
** end
```

The Call status

**xStatus Call**
Shows the top level overview of the call status. The call identity is used when query for additional information about the call.

**Example:**
```
xStatus Call
*s Call 27 Status: Connected
*s Call 27 Direction: Outgoing
*s Call 27 Protocol: "sip"
*s Call 27 RemoteNumber: "firstname.lastname@company.com"
*s Call 27 CallbackNumber: "sip:firstname.lastname@company.com"
*s Call 27 DisplayName: "Firstname Lastname"
*s Call 27 TransmitCallRate: 768
*s Call 27 ReceiveCallRate: 4000
*s Call 27 Encryption Type: "None"
*s Call 27 PlacedOnHold: False
*s Call 27 Duration: 2354
** end
```

**xStatus Call [1..n] Status**
Shows the status of a call. You can run the command xStatus Call to find the call identity.

*Value space of the result returned:*
<Connecting/Dialling/Ringing/Connected/Idle>

**Example:**
```
xStatus Call 27 Status
*s Call 27 Status: Connected
** end
```

**xStatus Call [1..n] Direction**
States the direction of the call initiation. You can run the command xStatus Call to find the call identity.

*Value space of the result returned:*
<Incoming/Outgoing>

**Example:**
```
xStatus Call 27 Direction
*s Call 27 Direction: Outgoing
** end
```
The Call status, cont...

**xStatus Call [1..n] Protocol**
Shows the call protocol of the incoming or outgoing call. You can run the command xStatus Call to find the call identity.

*Value space of the result returned:*

<H323/SIP>

*Example:*

```plaintext
xStatus Call 27 Protocol
*s Call 27 Protocol: "h323"
** end
```

**xStatus Call [1..n] RemoteNumber**
Shows the remote (far end) number or URI of an incoming or outgoing call. You can run the command xStatus Call to find the call identity.

*Value space of the result returned:*

<S: 0, 100>

*Example:*

```plaintext
xStatus Call 27 RemoteNumber
*s Call 27 RemoteNumber: "5585232"
** end
```

**xStatus Call [1..n] CallbackNumber**
Shows the remote (far end) number or URI of an incoming or outgoing call, including the call protocol, for call back. You can run the command xStatus Call to find the call identity.

*Value space of the result returned:*

<S: 0, 100>

*Example:*

```plaintext
xStatus Call 27 CallbackNumber
*s Call 27 CallbackNumber: "h323:firstname.lastname@company.com"
** end
```

The Call status, cont...

**xStatus Call [1..n] DisplayName**
Shows the name of the remote (far end) participant in an incoming or outgoing call. You can run the command xStatus Call to find the call identity.

*Value space of the result returned:*

<S: 0, 100>

*Example:*

```plaintext
xStatus Call 27 DisplayName
*s Call 27 DisplayName: "firstname.lastname@company.com"
** end
```

**xStatus Call [1..n] TransmitCallRate**
Shows the transmit bandwidth in the call in kilobits per second (kbps). You can run the command xStatus Call to find the call id.

*Value space of the result returned:*

<Integer value>

*Example:*

```plaintext
xStatus Call 27 TransmitCallRate
*s Call 27 TransmitCallRate: 768
** end
```

**xStatus Call [1..n] ReceiveCallRate**
Shows the receive bandwidth in the call in kilobits per second (kbps). You can run the command xStatus Call to find the call id.

*Value space of the result returned:*

<Integer value>

*Example:*

```plaintext
xStatus Call 27 ReceiveCallRate
*s Call 27 ReceiveCallRate: 4000
** end
```
The Call status, cont...

**xStatus Call [1..n] Encryption Type**
Shows the encryption type of the call. You can run the command `xStatus Call` to find the call identity.

- **Value space of the result returned:**
  <None/Aes-128>

- **Example:**
  ```
  xStatus Call 27 Encryption Type
  *s Call 27 Encryption Type: "None"
  ** end
  ```

**xStatus Call [1..n] PlacedOnHold**
Shows the placed on hold status of the call. You can run the command `xStatus Call` to find the call identity.

- **Value space of the result returned:**
  <True/False>

- **Example:**
  ```
  xStatus Call 27 PlacedOnHold
  *s Call 27 PlacedOnHold: False
  ** end
  ```

**xStatus Call [1..n] Duration**
Shows the duration of a call (in seconds). You can run the command `xStatus Call` to find the call identity.

- **Value space of the result returned:**
  <Integer value>

- **Example:**
  ```
  xStatus Call 27 Duration
  *s Call 27 Duration: 2354
  ** end
  ```

The Camera status

**xStatus Camera**
Shows the top level overview of the camera status.

- **Example:**
  ```
  xStatus Camera
  *s Camera 1 Connected: True
  *s Camera 1 HardwareID: "50000000"
  *s Camera 1 Manufacturer: "TANDBERG"
  *s Camera 1 Model: "PrecisionHD 1080p 12X"
  *s Camera 1 SoftwareID: "S01718-4.0FINAL [ID:40059] 2010-04-29"
  *s Camera 1 IpAddress: "-"  
  *s Camera 1 MacAddress: "-"
  *s Camera 1 Position Pan: 440
  *s Camera 1 Position Tilt: 69
  *s Camera 1 Position Zoom: 1
  *s Camera 1 Position Focus: 4581
  -  //continues with status for Camera 2-7//
  ** end
  ```

**xStatus Camera [1..7] Connected**
States if the camera is connected or not.

- **Value space of the result returned:**
  <True/False>

- **Example:**
  ```
  xStatus Camera 1 Connected
  *s Camera 1 Connected: True
  ** end
  ```

**xStatus Camera [1..7] HardwareID**
Shows the hardware identity of the camera.

- **Value space of the result returned:**
  <S: 0, 100>

- **Example:**
  ```
  xStatus Camera 1 HardwareID
  *s Camera 1 HardwareID: "50000000"
  ** end
  ```
**xStatus Camera [1..7] Manufacturer**
Shows the manufacturer of the camera.

**Value space of the result returned:**
\(<S: 0, 100>\)

**Example:**
```
xStatus Camera 1 Manufacturer
*s Camera 1 Manufacturer: "TANDBERG"
** end
```

**xStatus Camera [1..7] Model**
Shows the camera model.

**Value space of the result returned:**
\(<S: 0, 100>\)

**Example:**
```
xStatus Camera 1 Model
*s Camera 1 Model: "PrecisionHD 1080p 12X"
** end
```

**xStatus Camera [1..7] SoftwareID**
Shows the software identity of the camera.

**Value space of the result returned:**
\(<S: 0, 100>\)

**Example:**
```
xStatus Camera 1 SoftwareID
*s Camera 1 SoftwareID: "S01718-4.0FINAL [ID:40059] 2010-04-29"
** end
```

**xStatus Camera [1..7] IpAddress**
Shows the camera IP address.

**Value space of the result returned:**
\(<S: 0, 100>\)

**Examples:**
```
xStatus Camera 1 IpAddress
*s Camera 1 IpAddress: ""
** end
```

**xStatus Camera [1..7] MacAddress**
Shows the MAC (Media Access Control) address for the camera.

**Value space of the result returned:**
\(<S: 0, 100>\)

**Examples:**
```
xStatus Camera 1 MacAddress
*s Camera 1 MacAddress: ""
** end
```

**xStatus Camera [1..7] Position Pan**
Shows the current pan (move left and right) position of the camera. The value range depends on camera type.

**Value space of the result returned:**
\(<-65535..65535>\)

**Example:**
```
xStatus Camera 1 Position Pan
*s Camera 1 Position Pan: 514
** end
```
The Camera status, cont...

**xStatus Camera [1..7] Position Tilt**
Shows the current tilt (move up and down) position of the camera. The value range depends on camera type.

*Value space of the result returned:*
<-65535..65535>

*Example:*
```
xStatus Camera 1 Position Tilt
*s Camera 1 Position Tilt: 142
** end
```

**xStatus Camera [1..7] Position Zoom**
Shows the current zoom (zoom in and out) position of the camera. The value range depends on camera type.

*Value space of the result returned:*
<0..65535>

*Example:*
```
xStatus Camera 1 Position Zoom
*s Camera 1 Position Zoom: 1636
** end
```

**xStatus Camera [1..7] Position Focus**
Shows the current focus position of the camera. The value range depends on camera type.

*Value space of the result returned:*
<0..65535>

*Example:*
```
xStatus Camera 1 Position Focus
*s Camera 1 Position Focus: 4474
** end
```

The Conference status

**xStatus Conference**
Shows the top level overview of the conference status. The identity of the Conference Site can only be read during a call.

*Example:*
```
xStatus Conference
*s Conference Presentation Mode: Off
*s Conference Presentation Protocol: ""
*s Conference Presentation Resolution Height: 0
*s Conference Presentation Resolution Width: 0
*s Conference Presentation SiteId: 0
*s Conference Presentation LocalSource: 0
*s Conference Site 2 MicrophonesMuted: True
*s Conference Site 2 Capabilities Presentation: True
** end
```

**xStatus Conference Presentation Mode**
Shows the status of the secondary video stream.

*Value space of the result returned:*
<Off/Sending/Receiving>

*Example:*
```
xStatus Conference Presentation Mode
*s Conference Presentation Mode: Off
** end
```

**xStatus Conference Presentation Protocol**
Shows the video protocol used when transmitting the presentation.

*Value space of the result returned:*
<S: 0, 10>

*Example:*
```
xStatus Conference Presentation Protocol
*s Conference Presentation Protocol: "H264"
** end
```
xStatus Conference Presentation Resolution Height
Shows the height of the presentation.
Value space of the result returned:
<0..3000>
Example:

```
xStatus Conference Presentation Resolution Height
  *s Conference Presentation Resolution Height: 0
** end
```

xStatus Conference Presentation Resolution Width
Shows the width of the presentation.
Value space of the result returned:
<0..4000>
Example:

```
xStatus Conference Presentation Resolution Width
  *s Conference Presentation Resolution Width: 0
** end
```

xStatus Conference Presentation SiteId
Shows the identity of the system that sends the presentation.
Value space of the result returned:
<0..65535>
Example:

```
xStatus Conference Presentation SiteId
  *s Conference Presentation SiteId: 0
** end
```

xStatus Conference Presentation LocalSource
Shows the local video input source that is used when the presentation is sent from the local site.
Value space of the result returned:
<1..5>
Example:

```
xStatus Conference Presentation LocalSource
  *s Conference Presentation LocalSource: 0
** end
```

xStatus Conference Site [1..n] MicrophonesMuted
Lists the audio mute status for other participants in the conference.
Value space of the result returned:
<Ture/False>
Example:

```
xStatus Conference Site 2 MicrophonesMuted
  *s Conference Site 2 MicrophonesMuted: True
** end
```

xStatus Conference Site [1..n] Capabilities Presentation
Lists the presentation capabilities for other participants in the conference.
Value space of the result returned:
<Ture/False>
Example:

```
xStatus Conference Site 2 Capabilities Presentation
  *s Conference Site 2 Capabilities Presentation: True
** end
```
The Diagnostics status

xStatus Diagnostics

Shows the top level overview of the diagnostics. The example shows the status for an ongoing call. The identities of the call and channels are used when querying additional information.

Example:

```
xStatus Diagnostics
*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Jitter: 0
*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Packets: 132505
*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Loss: 0
*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Drop: 0
*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Bytes: 21200960
*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 ChannelRate: 64000
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Jitter: 6
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Packets: 123043
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Loss: 0
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Drop: 0
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Bytes: 94720991
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 ChannelRate: 493000
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Jitter: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Packets: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Loss: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Drop: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Bytes: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 ChannelRate: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Jitter: 4
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Packets: 123043
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Loss: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Drop: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Bytes: 94720991
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 ChannelRate: 453000
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Jitter: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Packets: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Loss: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Drop: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Bytes: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 ChannelRate: 0
```

The Diagnostics status, cont...

```
*s Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 ChannelRate: 0
*s Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 MaxJitter: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Jitter: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Packets: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Loss: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Drop: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Bytes: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 ChannelRate: 0
*s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 MaxJitter: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Jitter: 4
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Packets: 123043
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Loss: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Drop: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Bytes: 94720991
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 ChannelRate: 493000
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 MaxJitter: 4
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Jitter: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Packets: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Loss: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Drop: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 Bytes: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 ChannelRate: 0
*s Diagnostics Call 27 Channels OutgoingVideoChannel 334 Netstat 1 MaxJitter: 0
```

**end**
The Diagnostics status, cont...

**xStatus Diagnostics Call [1..n] Channels IncomingAudioChannel [1..n] Netstat 1 Jitter**

Shows the jitter at the present moment in the incoming channel, as specified by RFC 3550.

**Value space of the result returned:**

   <Integer value>

**Example:**

```plaintext
xStatus Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Jitter
```*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Jitter: 0 ** end

**xStatus Diagnostics Call [1..n] Channels IncomingAudioChannel [1..n] Netstat 1 Packets**

Shows the number of packets received in the incoming channel.

**Value space of the result returned:**

   <Integer value>

**Example:**

```plaintext
xStatus Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Packets
```*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Packets: 405 ** end

**xStatus Diagnostics Call [1..n] Channels IncomingAudioChannel [1..n] Netstat 1 Loss**

Shows the packets lost in the incoming channel.

**Value space of the result returned:**

   <Integer value>

**Example:**

```plaintext
xStatus Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Loss
```*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Loss: 96 ** end

The Diagnostics status, cont...

**xStatus Diagnostics Call [1..n] Channels IncomingAudioChannel [1..n] Netstat 1 Drop**

Shows the packets dropped in the incoming channel.

**Value space of the result returned:**

   <Integer value>

**Example:**

```plaintext
xStatus Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Drop
```*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Drop: 0 ** end

**xStatus Diagnostics Call [1..n] Channels IncomingAudioChannel [1..n] Netstat 1 Bytes**

Shows the number of bytes received in the incoming channel.

**Value space of the result returned:**

   <Integer value>

**Example:**

```plaintext
xStatus Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Bytes
```*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 Bytes: 129920 ** end

**xStatus Diagnostics Call [1..n] Channels IncomingAudioChannel [1..n] Netstat 1 ChannelRate**

Shows the bandwidth for the incoming channel.

**Value space of the result returned:**

   <Integer value>

**Example:**

```plaintext
xStatus Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 ChannelRate
```*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 ChannelRate: 128000 ** end
The Diagnostics status, cont...

**xStatus Diagnostics Call [1..n] Channels IncomingAudioChannel [1..n] Netstat 1 MaxJitter**

Shows the maximum jitter that has been measured during last time interval (5 seconds).

**Value space of the result returned:**

<Integer value>

**Example:**

```c
xStatus Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 MaxJitter
*s Diagnostics Call 27 Channels IncomingAudioChannel 327 Netstat 1 MaxJitter: 0
** end
```

**xStatus Diagnostics Call [1..n] Channels IncomingVideoChannel [1..n] Netstat 1 Jitter**

Shows the jitter at the present moment in the incoming channel, as specified by RFC 3550.

**Value space of the result returned:**

<Integer value>

**Example:**

```c
xStatus Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Jitter
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Jitter: 6
** end
```

**xStatus Diagnostics Call [1..n] Channels IncomingVideoChannel [1..n] Netstat 1 Packets**

Shows the number of packets received in the incoming channel.

**Value space of the result returned:**

<Integer value>

**Example:**

```c
xStatus Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Packets
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Packets: 38699
** end
```

**xStatus Diagnostics Call [1..n] Channels IncomingVideoChannel [1..n] Netstat 1 Loss**

Shows the packets lost in the incoming channel.

**Value space of the result returned:**

<Integer value>

**Example:**

```c
xStatus Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Loss
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Loss: 0
** end
```

**xStatus Diagnostics Call [1..n] Channels IncomingVideoChannel [1..n] Netstat 1 Drop**

Shows the packets dropped in the incoming channel.

**Value space of the result returned:**

<Integer value>

**Example:**

```c
xStatus Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Drop
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Drop: 0
** end
```

**xStatus Diagnostics Call [1..n] Channels IncomingVideoChannel [1..n] Netstat 1 Bytes**

Shows the number of bytes received in the incoming channel.

**Value space of the result returned:**

<Integer value>

**Example:**

```c
xStatus Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Bytes
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 Bytes: 0
** end
```
The Diagnostics status, cont...

xStatus Diagnostics Call [1..n] Channels IncomingVideoChannel [1..n] Netstat 1 ChannelRate

Shows the bandwidth for the incoming channel.

Value space of the result returned:

<Integer value>

Example:

xStatus Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 ChannelRate
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 ChannelRate: 0
** end

xStatus Diagnostics Call [1..n] Channels IncomingVideoChannel [1..n] Netstat 1 MaxJitter

Shows the maximum jitter that has been measured during last time interval (5 seconds).

Value space of the result returned:

<Integer value>

Example:

xStatus Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 MaxJitter
*s Diagnostics Call 27 Channels IncomingVideoChannel 330 Netstat 1 MaxJitter: 0
** end

xStatus Diagnostics Call [1..n] Channels IncomingVideoChannel [1..n] Netstat 1 Jitter

Shows the jitter at the present moment in the incoming channel, as specified by RFC 3550.

Value space of the result returned:

<Integer value>

Example:

xStatus Diagnostics Call 27 Channels IncomingVideoChannel 335 Netstat 1 Jitter
*s Diagnostics Call 27 Channels IncomingVideoChannel 335 Netstat 1 Jitter: 0
** end

xStatus Diagnostics Call [1..n] Channels IncomingDataChannel [1..n] Netstat 1 Packets

Shows the number of packets received in the incoming channel.

Value space of the result returned:

<Integer value>

Example:

xStatus Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 Packets
*s Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 Packets: 10
** end

xStatus Diagnostics Call [1..n] Channels IncomingDataChannel [1..n] Netstat 1 Loss

Shows the packets lost in the incoming channel.

Value space of the result returned:

<Integer value>

Example:

xStatus Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 Loss
*s Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 Loss: 0
** end

xStatus Diagnostics Call [1..n] Channels IncomingDataChannel [1..n] Netstat 1 Drop

Shows the packets dropped in the incoming channel.

Value space of the result returned:

<Integer value>

Example:

xStatus Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 Drop
*s Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 Drop: 0
** end
The Diagnostics status, cont...

**xStatus Diagnostics Call [1..n] Channels IncomingDataChannel [1..n] Netstat 1 Bytes**
Shows the number of bytes received in the incoming channel.

*Value space of the result returned:*

<Integer value>

*Example:*

```plaintext
xStatus Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 Bytes
```

**xStatus Diagnostics Call [1..n] Channels IncomingDataChannel [1..n] Netstat 1 ChannelRate**
Shows the bandwidth for the incoming channel.

*Value space of the result returned:*

<Integer value>

*Example:*

```plaintext
xStatus Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 ChannelRate
```

**xStatus Diagnostics Call [1..n] Channels IncomingDataChannel [1..n] Netstat 1 MaxJitter**
Shows the maximum jitter that has been measured during last time interval (5 seconds).

*Value space of the result returned:*

<Integer value>

*Example:*

```plaintext
xStatus Diagnostics Call 27 Channels IncomingDataChannel 335 Netstat 1 MaxJitter
```

**xStatus Diagnostics Call [1..n] Channels OutgoingAudioChannel [1..n] Netstat 1 Jitter**
Shows the jitter at the present moment in the outgoing channel, as specified by RFC 3550.

*Value space of the result returned:*

<Integer value>

*Example:*

```plaintext
xStatus Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Jitter
```

**xStatus Diagnostics Call [1..n] Channels OutgoingAudioChannel [1..n] Netstat 1 Packets**
Shows the number of packets sent in the outgoing channel.

*Value space of the result returned:*

<Integer value>

*Example:*

```plaintext
xStatus Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Packets
```

**xStatus Diagnostics Call [1..n] Channels OutgoingAudioChannel [1..n] Netstat 1 Loss**
Shows the packets lost in the outgoing channel.

*Value space of the result returned:*

<Integer value>

*Example:*

```plaintext
xStatus Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Loss
```
The Diagnostics status, cont...

**xStatus Diagnostics Call [1..n] Channels OutgoingAudioChannel [1..n] Netstat 1 Drop**
Shows the packets dropped in the outgoing channel.

*Value space of the result returned:*

<integer value>

*Example:*

```
xStatus Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Drop
  *s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Drop: 0
  ** end
```

**xStatus Diagnostics Call [1..n] Channels OutgoingAudioChannel [1..n] Netstat 1 Bytes**
Shows the number of bytes sent in the media channel.

*Value space of the result returned:*

<integer value>

*Example:*

```
xStatus Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Bytes
  *s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 Bytes: 7653930
  ** end
```

**xStatus Diagnostics Call [1..n] Channels OutgoingAudioChannel [1..n] Netstat 1 ChannelRate**
Shows the bandwidth for the outgoing channel.

*Value space of the result returned:*

<integer value>

*Example:*

```
xStatus Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 ChannelRate
  *s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 ChannelRate: 127000
  ** end
```

**xStatus Diagnostics Call [1..n] Channels OutgoingAudioChannel [1..n] Netstat 1 MaxJitter**
Shows the maximum jitter that has been measured during last time interval (5 seconds).

*Value space of the result returned:*

<integer value>

*Example:*

```
xStatus Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 MaxJitter
  *s Diagnostics Call 27 Channels OutgoingAudioChannel 328 Netstat 1 MaxJitter: 0
  ** end
```

**xStatus Diagnostics Call [1..n] Channels OutgoingVideoChannel [1..n] Netstat 1 Jitter**
Shows the jitter at the present moment in the outgoing channel, as specified by RFC 3550.

*Value space of the result returned:*

<integer value>

*Example:*

```
xStatus Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Jitter
  *s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Jitter: 1
  ** end
```

**xStatus Diagnostics Call [1..n] Channels OutgoingVideoChannel [1..n] Netstat 1 Packets**
Shows the number of packets received in the outgoing channel.

*Value space of the result returned:*

<integer value>

*Example:*

```
xStatus Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Packets
  *s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Packets: 43096
  ** end
```
The Diagnostics status, cont...

xStatus Diagnostics Call [1..n] Channels OutgoingVideoChannel [1..n] Netstat 1 Loss
Shows the packets lost in the outgoing channel.

Value space of the result returned:
<Integer value>

Example:

```
xStatus Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Loss
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Loss: 0
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingVideoChannel [1..n] Netstat 1 Drop
Shows the packets dropped in the outgoing channel.

Value space of the result returned:
<Integer value>

Example:

```
xStatus Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Drop
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Drop: 0
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingVideoChannel [1..n] Netstat 1 Bytes
Shows the number of bytes sent in the media channel.

Value space of the result returned:
<Integer value>

Example:

```
xStatus Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Bytes
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 Bytes: 39111160
** end
```

The Diagnostics status, cont...

xStatus Diagnostics Call [1..n] Channels OutgoingVideoChannel [1..n] Netstat 1 ChannelRate
Shows the bandwidth for the outgoing channel.

Value space of the result returned:
<Integer value>

Example:

```
xStatus Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 ChannelRate
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 ChannelRate: 652000
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingVideoChannel [1..n] Netstat 1 MaxJitter
Shows the maximum jitter that has been measured during last time interval (5 seconds).

Value space of the result returned:
<Integer value>

Example:

```
xStatus Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 MaxJitter
*s Diagnostics Call 27 Channels OutgoingVideoChannel 331 Netstat 1 MaxJitter: 1
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingDataChannel [1..n] Netstat 1 Jitter
Shows the jitter at the present moment in the outgoing channel, as specified by RFC 3550.

Value space of the result returned:
<Integer value>

Example:

```
xStatus Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Jitter
*s Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Jitter: 0
** end
```
The Diagnostics status, cont...

xStatus Diagnostics Call [1..n] Channels OutgoingDataChannel [1..n] Netstat 1 Packets
Shows the number of packets sent in the outgoing channel.

Value space of the result returned:
<Integer value>

Example:
```
xStatus Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Packets
*s Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Packets: 2
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingDataChannel [1..n] Netstat 1 Loss
Shows the packets lost in the outgoing channel.

Value space of the result returned:
<Integer value>

Example:
```
xStatus Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Loss
*s Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Loss: 0
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingDataChannel [1..n] Netstat 1 Drop
Shows the packets dropped in the outgoing channel.

Value space of the result returned:
<Integer value>

Example:
```
xStatus Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Drop
*s Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Drop: 0
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingDataChannel [1..n] Netstat 1 Bytes
Shows the number of bytes sent in the media channel.

Value space of the result returned:
<Integer value>

Example:
```
xStatus Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Bytes
*s Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 Bytes: 107
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingDataChannel [1..n] Netstat 1 ChannelRate
Shows the bandwidth for the outgoing channel.

Value space of the result returned:
<Integer value>

Example:
```
xStatus Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 ChannelRate
*s Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 ChannelRate: 0
** end
```

xStatus Diagnostics Call [1..n] Channels OutgoingDataChannel [1..n] Netstat 1 MaxJitter
Shows the maximum jitter that has been measured during last time interval (5 seconds).

Value space of the result returned:
<Integer value>

Example:
```
xStatus Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 MaxJitter
*s Diagnostics Call 27 Channels OutgoingDataChannel 336 Netstat 1 MaxJitter: 0
** end
```
The GPIO status

**xStatus GPIO Pin [1..4] State**

NOTE: This command is not supported on Codec C40.

Shows the current state of each GPIO pin. The default state is High (+12V). When activated the state is Low (0V).

```
Example:
  xStatus GPIO Pin 1 State
  *s GPIO Pin 1 State: High
  ** end
```

The H323 status

**xStatus H323**

Shows the top level overview of the H323 status.

```
Example:
  xStatus H323
  *s H323 Gatekeeper Status: Registered
  *s H323 Gatekeeper Address: "192.0.2.20"
  *s H323 Gatekeeper Port: 1719
  *s H323 Gatekeeper Reason: ""
  ** end
```

**xStatus H323 Gatekeeper Status**

Shows the gatekeeper registration status.

**Value space of the result returned:**

<Registered/Inactive/Rejected>

```
Example:
  xStatus H323 Gatekeeper Status
  *s H323 Gatekeeper Status: Registered
  ** end
```

**xStatus H323 Gatekeeper Address**

Displays the IP address of the gatekeeper where the system is registered.

**Value space of the result returned:**

<S: 0, 100>

```
Example:
  xStatus H323 Gatekeeper Address
  *s H323 Gatekeeper Address: "192.0.2.20"
  ** end
```
The H323 status, cont...

**xStatus H323 Gatekeeper Port**
Shows the port which is used when connecting to on the gatekeeper.

*Value space of the result returned:*

<Integer value>

*Example:*

```
xStatus H323 Gatekeeper Port
*s H323 Gatekeeper Port: 1719
** end
```

**xStatus H323 Gatekeeper Reason**
Shows the reason for rejected registration.

*Value space of the result returned:*

<S: 0, 100>

*Example:*

```
xStatus H323 Gatekeeper Reason
*s H323 Gatekeeper Reason: ""
** end
```

**xStatus HTTPFeedback**
Shows the top level overview of the HTTP status.

*Example:*

```
xStatus HTTPFeedback
*s HttpFeedback 1 URL: "http://tms.group.company.com/tms/public/feedback/code.aspx"
*s HttpFeedback 1 Expression: "/History/CallLog/History"
*s HttpFeedback 1 Expression: "/Status/Call[Status='connected']"
*s HttpFeedback 1 Expression: "/Status/H323/Gatekeeper"
*s HttpFeedback 1 Expression: "/Status/Ethernet"
*s HttpFeedback 1 Expression: "/Event/CallSuccessful"
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: ""
*s HttpFeedback 1 Expression: "" - continues with HttpFeedback 2-3
** end
```

**xStatus HttpFeedback [1..3] URL**
Shows the URL (Uniform Resource Locator) of the HTTP server. There can be up to three HTTP servers, specified by the URL.

*Value space of the result returned:*

<S: 0, 100>

*Example:*

```
xStatus HttpFeedback 1 URL
** end
```
The HTTP Feedback status, cont...

**xStatus HttpFeedback [1..3] Expression**

Shows the feedback from the HTTP server. There can be up to 15 expressions for each URL. See the xCommand HttpFeedback commands for more information.

**Value space of the result returned:**

< S: 0..256 >

**Example:**

```
xStatus HttpFeedback 3 Expression
  *s HttpFeedback 3 Expression: "/History/CallLog/History"
  *s HttpFeedback 3 Expression: "/Status/Call[Status='connected']"
  *s HttpFeedback 3 Expression: "/Status/H323/Gatekeeper"
  *s HttpFeedback 3 Expression: "/Status/Ethernet"
  *s HttpFeedback 3 Expression: "/Event/CallSuccessful"
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  *s HttpFeedback 3 Expression: ""
  ** end
```

The Media Channel status

**xStatus MediaChannel**

Shows the top level overview of the media channel status. The example shows the status for an ongoing call. The identities of the call and channels are used when querying additional information.

**Example:**

```
xStatus Media
  *s MediaChannels Call 27 IncomingAudioChannel 327 Encryption Status: On
  *s MediaChannels Call 27 IncomingAudioChannel 327 Audio Protocol: AAC_LD
  *s MediaChannels Call 27 IncomingAudioChannel 327 Audio Mute: True
  *s MediaChannels Call 27 IncomingAudioChannel 327 Audio Channels: 1
  *s MediaChannels Call 27 IncomingVideoChannel 330 Encryption Status: On
  *s MediaChannels Call 27 IncomingVideoChannel 330 Video Protocol: H264
  *s MediaChannels Call 27 IncomingVideoChannel 330 Video FrameRate: 25
  *s MediaChannels Call 27 IncomingVideoChannel 330 Video ResolutionX: 352
  *s MediaChannels Call 27 IncomingVideoChannel 330 Video ResolutionY: 288
  *s MediaChannels Call 27 OutgoingAudioChannel 328 Encryption Status: On
  *s MediaChannels Call 27 OutgoingAudioChannel 328 Audio Protocol: AAC_LD
  *s MediaChannels Call 27 OutgoingAudioChannel 328 Audio Channels: 1
  *s MediaChannels Call 27 OutgoingVideoChannel 331 Encryption Status: On
  *s MediaChannels Call 27 OutgoingVideoChannel 331 Video Protocol: H264
  *s MediaChannels Call 27 OutgoingVideoChannel 331 Video FrameRate: 30
  *s MediaChannels Call 27 OutgoingVideoChannel 331 Video ResolutionX: 1024
  *s MediaChannels Call 27 OutgoingVideoChannel 331 Video ResolutionY: 576
  *s MediaChannels Call 27 OutgoingVideoChannel 334 Encryption Status: Off
  *s MediaChannels Call 27 OutgoingVideoChannel 334 ChannelRole: Presentation
  *s MediaChannels Call 27 OutgoingVideoChannel 334 Video Protocol: Off
  *s MediaChannels Call 27 OutgoingVideoChannel 334 Video FrameRate: 0
  *s MediaChannels Call 27 OutgoingVideoChannel 334 Video ResolutionX: 0
  *s MediaChannels Call 27 OutgoingVideoChannel 334 Video ResolutionY: 0
  ** end
```
The Media Channel status, cont...

**xStatus MediaChannels Call [1..n] IncomingAudioChannel [1..n] Encryption Status**

Shows the encryption status on the incoming channel.

Value space of the result returned:

<On/off>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingAudioChannel 327 Encryption Status
```

*s MediaChannels Call 27 IncomingAudioChannel 327 Encryption Status: Off

** end

**xStatus MediaChannels Call [1..n] IncomingAudioChannel [1..n] Audio Protocol**

Shows the audio algorithm for the incoming audio channel.

AACLD: The AAC-LD is a MPEG-4 Low Delay Audio Coder audio compression format.

G722: The G.722 algorithm is an ITU standard.

G7221: The G.722.1 algorithm is a licensed royalty-free ITU-T standard.

G711Mu: The G.711 Mu-law compression algorithm is used in North America and Japan.

G711A: The G.711 A-law compression algorithm is used in Europe and the rest of the world.

Value space of the result returned:

<AACLD/G722/G7221/G711Mu/G711A>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingAudioChannel 327 Audio Protocol
```

*s MediaChannels Call 27 IncomingAudioChannel 327 Audio Protocol: AACLD

** end

**xStatus MediaChannels Call [1..n] IncomingAudioChannel [1..n] Audio Mute**

Shows the audio mute status of the incoming audio channel.

Value space of the result returned:

<True/False>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingAudioChannel 327 Audio Mute
```

*s MediaChannels Call 27 IncomingAudioChannel 327 Audio Mute: True

** end

**xStatus MediaChannels Call [1..n] IncomingAudioChannel [1..n] Audio Channels**

Shows the number of incoming audio channels.

Value space of the result returned:

<Integer value>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingAudioChannel 327 Audio Channels
```

*s MediaChannels Call 27 IncomingAudioChannel 327 Audio Channels: 1

** end

**xStatus MediaChannels Call [1..n] IncomingVideoChannel [1..n] Encryption Status**

Shows the encryption status on the incoming channel.

Value space of the result returned:

<On/Off>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingVideoChannel 330 Encryption Status
```

*s MediaChannels Call 27 IncomingVideoChannel 330 Encryption Status: Off

** end

**xStatus MediaChannels Call [1..n] IncomingVideoChannel [1..n] ChannelRole**

Shows if the incoming channel is the main video channel or presentation channel.

Value space of the result returned:

<Main/Presentation>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingVideoChannel 330 ChannelRole
```

*s MediaChannels Call 27 IncomingVideoChannel 330 ChannelRole: Main

** end
The Media Channel status, cont...

**xStatus MediaChannels Call [1..n] IncomingVideoChannel [1..n] Video Protocol**

Shows the video algorithm for the incoming video channel.

H264: The H.264 algorithm is an ITU-T standard for video compression.


H263: The H.263 algorithm is an ITU-T standard for video compression.

H261: The H.261 algorithm is an ITU-T standard for video compression.

Value space of the result returned:

<H264/H263pp/H263/H261>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingVideoChannel 330 Video Protocol
*s MediaChannels Call 27 IncomingVideoChannel 330 Video Protocol: H264
** end
```

**xStatus MediaChannels Call [1..n] IncomingVideoChannel [1..n] Video FrameRate**

Shows the video frame rate of the incoming channel.

Value space of the result returned:

<Integer value>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingVideoChannel 330 Video FrameRate
*s MediaChannels Call 27 IncomingVideoChannel 330 Video FrameRate: 25
** end
```

**xStatus MediaChannels Call [1..n] IncomingVideoChannel [1..n] Video ResolutionX**

Shows the width (resolution in direction X) of the incoming video.

Value space of the result returned:

<Integer value>

Example:

```plaintext
xStatus MediaChannels Call 27 IncomingVideoChannel 330 Video ResolutionX
*s MediaChannels Call 27 IncomingVideoChannel 330 Video ResolutionX: 768
** end
```

**xStatus MediaChannels Call [1..n] OutgoingAudioChannel [1..n] Encryption Status**

Shows the encryption status on the outgoing channel.

Value space of the result returned:

<On/Off>

Example:

```plaintext
xStatus MediaChannels Call 27 OutgoingAudioChannel 328 Encryption Status
*s MediaChannels Call 27 OutgoingAudioChannel 328 Encryption Status: Off
** end
```

**xStatus MediaChannels Call [1..n] OutgoingAudioChannel [1..n] Audio Protocol**

Shows the audio algorithm for the outgoing audio channel.

AACLD: The AAC-LD is a MPEG-4 Low Delay Audio Coder audio compression format.

G722: The G.722 algorithm is an ITU standard.

G722.1: The G.722.1 algorithm is a licensed royalty-free ITU-T standard.

G711Mu: The G.711 Mu-law compression algorithm is used in North America and Japan.

G711A: The G.711 A-law compression algorithm is used in Europe and the rest of the world.

Value space of the result returned:

<AACLD/G722/G722.1/G711Mu/G711A>

Example:

```plaintext
xStatus MediaChannels Call 27 OutgoingAudioChannel 328 Audio Protocol
*s MediaChannels Call 27 OutgoingAudioChannel 328 Audio Protocol: AACLD
** end
```
xStatus MediaChannels Call [1..n] OutgoingAudioChannel [1..n] Audio Channels
Shows the number of outgoing audio channels.

Value space of the result returned:
<Integer value>

Example:
```
xStatus MediaChannels Call 27 OutgoingAudioChannel 328 Audio Channels
```
```
*s MediaChannels Call 27 OutgoingAudioChannel 328 Audio Channels: 1
** end
```

xStatus MediaChannels Call [1..n] OutgoingVideoChannel [1..n] Encryption Status
Shows the encryption status on the outgoing channel.

Value space of the result returned:
<On/Off>

Example:
```
xStatus MediaChannels Call 27 OutgoingVideoChannel 331 Encryption Status
```
```
*s MediaChannels Call 27 OutgoingVideoChannel 331 Encryption Status: Off
** end
```

xStatus MediaChannels Call [1..n] OutgoingVideoChannel [1..n] ChannelRole
Shows if the outgoing channel is the main video channel or presentation channel.

Value space of the result returned:
<Main/Presentation>

Example:
```
xStatus MediaChannels Call 27 OutgoingVideoChannel 331 ChannelRole
```
```
*s MediaChannels Call 27 OutgoingVideoChannel 331 ChannelRole: Main
** end
```

xStatus MediaChannels Call [1..n] OutgoingVideoChannel [1..n] Video Protocol
Shows the video algorithm for the outgoing video channel.
H264: The H.264 algorithm is an ITU-T standard for video compression.
H263: The H.263 algorithm is an ITU-T standard for video compression.
H261: The H.261 algorithm is an ITU-T standard for video compression.

Value space of the result returned:
<H264/H263pp/H263/H261>

Example:
```
xStatus MediaChannels Call 27 OutgoingVideoChannel 331 Video Protocol
```
```
*s MediaChannels Call 27 OutgoingVideoChannel 331 Video Protocol: “H264”
** end
```

xStatus MediaChannels Call [1..n] OutgoingVideoChannel [1..n] Video FrameRate
Shows the video frame rate of the outgoing channel.

Value space of the result returned:
<Integer value>

Example:
```
xStatus MediaChannels Call 27 OutgoingVideoChannel 331 Video FrameRate
```
```
*s MediaChannels Call 27 OutgoingVideoChannel 331 Video FrameRate: 30
** end
```

xStatus MediaChannels Call [1..n] OutgoingVideoChannel [1..n] Video ResolutionX
Shows the width (resolution in direction X) of the outgoing video.

Value space of the result returned:
<Integer value.>

Example:
```
xStatus MediaChannels Call 27 OutgoingVideoChannel 331 Video ResolutionX
```
```
*s MediaChannels Call 27 OutgoingVideoChannel 331 Video ResolutionX: 768
** end
```

The Media Channel status, cont...
The Media Channel status, cont...

**xStatus MediaChannels Call [1..n] OutgoingVideoChannel [1..n] Video ResolutionY**
Shows the height (resolution in direction Y) of the outgoing video.

**Value space of the result returned:**
<Integer value.>

**Example:**
```
xStatus MediaChannels Call 27 OutgoingVideoChannel 331 Video ResolutionY
```
```
*s MediaChannels Call 27 OutgoingVideoChannel 331 Video ResolutionY: 448
** end
```

The Network status

**xStatus Network**
Shows the top level overview of the network status.

**Example:**
```
xStatus Network
```
```
*s Network 1 Ethernet MacAddress: "00:50:60:02:E7:D3"
*s Network 1 Ethernet Speed: "1000full"
*s Network 1 IPv4 Address: "192.0.2.149"
*s Network 1 IPv4 SubnetMask: "255.255.255.0"
*s Network 1 IPv4 Gateway: "192.0.2.10"
*s Network 1 IPv4 DNS Domain Name: "www.example.com www.example.int"
*s Network 1 IPv4 DNS Server 1 Address: "192.0.2.60"
*s Network 1 IPv4 DNS Server 2 Address: "192.0.2.61"
*s Network 1 IPv4 DNS Server 3 Address: ""
*s Network 1 IPv4 DNS Server 4 Address: ""
*s Network 1 IPv4 DNS Server 5 Address: ""
*s Network 1 IPv4 MTU: 1500
** end
```

**xStatus Network 1 Ethernet MacAddress**
Shows the MAC (Media Access Control) address for the ethernet interface.

**Value space of the result returned:**
<S: 0, 100>

**Example:**
```
xStatus Network 1 Ethernet MacAddress
```
```
*s Network 1 Ethernet MacAddress: "00:50:60:02:FD:C7"
** end
```

**xStatus Network 1 Ethernet Speed**
Shows the Ethernet speed in Mbps. The speed can be in full-duplex or half-duplex.

**Value space of the result returned:**
<10half/10full/100half/100full/1000full>

**Example:**
```
xStatus Network 1 Ethernet Speed
```
```
*s Network 1 Ethernet Speed: "1000full"
** end
```
The Network status, cont...

**xStatus Network 1 IPv4 Address**
Shows the IP address that uniquely identifies this system.

**Value space of the result returned:**
  
  <S: 0, 100>

**Example:**

```
xStatus Network 1 IPv4 Address
*s Network 1 IPv4 Address: "192.0.2.149"
** end
```

**xStatus Network 1 IPv4 SubnetMask**
Shows the subnet mask which determines what subnet an IP address belongs to.

**Value space of the result returned:**

  
  <S: 0, 100>

**Example:**

```
xStatus Network 1 IPv4 SubnetMask
*s Network 1 IPv4 SubnetMask: "255.255.255.0"
** end
```

**xStatus Network 1 IPv4 Gateway**
Shows the address of the gateway.

**Value space of the result returned:**

  
  <S: 0, 100>

**Example:**

```
xStatus Network 1 IPv4 Gateway
*s Network 1 IPv4 Gateway: "192.0.2.10"
** end
```

**xStatus Network 1 IPv4 DNS Domain Name**
Shows the domain name.

**Value space of the result returned:**

  
  <S: 0, 100>

**Example:**

```
xStatus Network 1 IPv4 DNS Domain Name
*s Network 1 IPv4 DNS Domain Name: "www.example.com www.example.int"
** end
```

**xStatus Network 1 IPv4 DNS Server [1..5] Address**
Shows the IP address of the DNS server.

**Value space of the result returned:**

  
  <S: 0, 100>

**Example:**

```
xStatus Network 1 IPv4 DNS Server 1 Address
*s Network 1 IPv4 DNS Server 1 Address: "192.0.1.60"
** end
```

**xStatus Network 1 IPv4 MTU**
Shows the MTU (Maximum Transmission Unit) size for the network.

**Value space of the result returned:**

  
  <Integer value>

**Example:**

```
xStatus Network 1 IPv4 MTU
*s Network 1 IPv4 MTU: 1500
** end
```
The Preset status

**xStatus Preset**
Shows the top level overview of the camera presets status.

_Example:_

```
xStatus Preset
  *s Preset 1 Defined: True
  *s Preset 1 Type: All
  *s Preset 1 Description: “Zoom in”
  *s Preset 2 Defined: True
  *s Preset 2 Type: All
  *s Preset 2 Description: “Zoom out”
- //continues with Preset 3-15.//
** end
```

**xStatus Preset [1..15] Defined**
Shows if a camera preset is stored at this position.

**Value space of the result returned:**

<True/False>

_Example:_

```
xStatus Preset 1 Defined
  *s Preset 1 Defined: True
** end
```

**xStatus Preset [1..15] Type**
Shows the camera preset type.

**Value space of the result returned:**

<All/Camera>

_Example:_

```
xStatus Preset 1 Type
  *s Preset 1 Type: All
** end
```
The Provisioning status

**xStatus Provisioning**
Shows the top level overview of the Provisioning status.

Value space of the result returned:
<N/A>

Example:
```
xStatus Provisioning
* Provisioning Status: Provisioned
* Provisioning Reason: ""
* Provisioning PendingUpgrade: False
** end
```

**xStatus Provisioning Status**
Shows the status of the provisioning.
Failed: The provisioning failed.
AuthenticationFailed: The authentication failed.
Provisioned: The endpoint is provisioned.
Idle: The provisioning is not active.
NeedConfig: The endpoint needs to be configured.

Value space of the result returned:
<Failed/AuthenticationFailed/Provisioned/Idle/NeedConfig>

Example:
```
xStatus Provisioning Status
* Provisioning Status: Provisioned
** end
```

The Provisioning status, cont...

**xStatus Provisioning Reason**
Shows the cause when provisioning has failed.

Value space of the result returned:
<S: 0..80>

Example:
```
xStatus Provisioning Reason
* Provisioning Reason: ""
** end
```

**xStatus Provisioning PendingUpgrade**
Shows the status of the pending software upgrade.

Value space of the result returned:
<True/False>

Example:
```
xStatus Provisioning PendingUpgrade
* Provisioning PendingUpgrade: False
** end
```
The SIP status

xStatus SIP
Shows the top level overview of the SIP status.

Value space of the result returned:
<N/A>

Example:
```
xStatus SIP
*s SIP Proxy 1 Status: Active
*s SIP Proxy 1 Address: "192.0.2.50"
*s SIP Proxy 1 Secure: True
*s SIP Proxy 1 Verified: False
*s SIP Registration 1 Status: Registered
*s SIP Registration 1 Reason: ""
*s SIP Registration 1 URI: "firstname.lastname@company.com"
*s SIP Registration 1 Authentication: Off
*s SIP Profile 1 Proxy 1 Status: Active
*s SIP Profile 1 Proxy 1 Address: "192.0.2.50"
*s SIP Profile 1 Secure: True
*s SIP Profile 1 Verified: False
*s SIP Profile 1 Authentication: Off
*s SIP Profile 1 Registration 1 Status: Registered
*s SIP Profile 1 Registration 1 Reason: ""
*s SIP Profile 1 Registration 1 URI: "firstname.lastname@company.com"
** end
```

The SIP status, cont...

xStatus SIP Proxy 1 Status
Shows the status of the communication between the endpoint and the SIP Proxy server.

Active: The communication between the endpoint and the SIP Proxy is active.
DNSFailed: The attempt to establish communication to the DNS server failed.
Off: There is no communication between the endpoint and the SIP Proxy.
Timeout: The attempt to establish communication to the SIP Proxy timed out.
UnableTCP: The system is unable to use TCP as the transport method.
UnableTLS: The system is unable to use TLS as the transport method.
Unknown: The status of the communication is not known.

Value space of the result returned:
<Active/DNSFailed/Off/Timeout/UnableTCP/UnableTLS/Unknown>

Example:
```
xStatus SIP Proxy 1 Status
*s SIP Proxy 1 Status: Active
** end
```

xStatus SIP Proxy 1 Address
Shows the address of the SIP Proxy that the system communicates with.

Value space of the result returned:
<S: 0, 255>

Example:
```
xStatus SIP Proxy 1 Address
*s SIP Proxy 1 Address: "192.0.2.50"
** end
```

xStatus SIP Proxy 1 Secure
Shows the encryption status of the signalling with the SIP Proxy server.

Value space of the result returned:
<True/False>

Example:
```
xStatus SIP Proxy 1 Secure
*s SIP Proxy 1 Secure: True
** end
```
The SIP status, cont...

**xStatus SIP Proxy 1 Verified**
Not supported in this software version.

*Value space of the result returned:
  <True/False>*

*Example:*
  ```
  xStatus SIP Proxy 1 Verified
  *s SIP Proxy 1 Verified: False
  ** end
  ```

**xStatus SIP Registration [1..n] Status**
Shows the status of the registration to the SIP Proxy Server.

Deregister: The system is in the process of de-registering to the SIP Proxy.
Failed: The system failed to register to the SIP Proxy.
Inactive: The system is not registered to any SIP Proxy.
Registered: The system is registered to the SIP Proxy.
Registering: The system is in the process of registering to the SIP Proxy.

*Value space of the result returned:*
  <Deregister/Failed/Inactive/Registered/Registering>*

*Example:*
  ```
  xStatus SIP Registration 1 Status
  *s SIP Registration 1 Status: Registered
  ** end
  ```

**xStatus SIP Registration [1..n] Reason**
Shows a message to explain the reason why the SIP registration failed.

*Value space of the result returned:*
  <S: 0, 100>*

*Example:*
  ```
  xStatus SIP Registration 1 Reason
  *s SIP Registration 1 Reason: "404 Not Found"
  ** end
  ```

The SIP status, cont...

**xStatus SIP Registration [1..n] URI**
Shows the URI used for registration to the SIP Proxy server.

*Value space of the result returned:*
  <S: 0, 100>*

*Example:*
  ```
  xStatus SIP Registration 1 URI
  *s SIP Registration 1 URI: "firstname.lastname@company.com"
  ** end
  ```

**xStatus SIP Registration [1..n] Authentication**
Shows which authentication mechanism is used when registering to the SIP Proxy Server.

Digest: Uses the Digest access authentication method, as specified by RFC 2069.
NTLM: Uses the NTLM authentication method, which is a Microsoft authentication protocol.
Off: No authentication mechanism is used.

*Value space of the result returned:*
  <Digest/NTLM/Off>*

*Example:*
  ```
  xStatus SIP Registration 1 Authentication
  *s SIP Registration 1 Authentication: Off
  ** end
  ```
The SIP status, cont...

**xStatus SIP Profile 1 Proxy 1 Status**

Shows the status of the communication between the endpoint and the SIP Proxy server.

- **Active**: The communication between the endpoint and the SIP Proxy is active.
- **DNSFailed**: The attempt to establish communication to the DNS server failed.
- **Off**: There is no communication between the endpoint and the SIP Proxy.
- **Timeout**: The attempt to establish communication to the SIP Proxy timed out.
- **UnableTCP**: The system is unable to use TCP as the transport method.
- **UnableTLS**: The system is unable to use TLS as the transport method.
- **Unknown**: The status of the communication is not known.

**Value space of the result returned:**

```
<Active/DNSFailed/Off/Timeout/UnableTCP/UnableTLS/Unknown>
```

**Example:**

```
xStatus SIP Profile 1 Proxy 1 Status
*s SIP Profile 1 Proxy 1 Status: Active
** end
```

**xStatus SIP Profile 1 Proxy 1 Address**

Shows the address of the SIP Proxy that the system communicates with.

**Value space of the result returned:**

```
<S: 0, 255>
```

**Example:**

```
xStatus SIP Profile 1 Proxy 1 Address
*s SIP Profile 1 Proxy 1 Address: "192.0.2.50"
** end
```

**xStatus SIP Profile 1 Secure**

Shows the encryption status of the signalling with the SIP Proxy server.

**Value space of the result returned:**

```
<True/False>
```

**Example:**

```
xStatus SIP Profile 1 Secure
*s SIP Profile 1 Secure: True
** end
```

The SIP status, cont...

**xStatus SIP Profile 1 Verified**

Not supported in this software version.

**Value space of the result returned:**

```
<True/False>
```

**Example:**

```
xStatus SIP Profile 1 Verified
*s SIP Profile 1 Verified: False
** end
```

**xStatus SIP Profile 1 Authentication**

Shows which authentication mechanism is used when registering to the SIP Proxy Server.

- **Digest**: Uses the Digest access authentication method, as specified by RFC 2069.
- **NTLM**: Uses the NTLM authencitication method, which is a Microsoft authentication protocol.
- **Off**: No authentication mechanism is used.

**Value space of the result returned:**

```
<Digest/NTLM/Off>
```

**Example:**

```
xStatus SIP Profile 1 Authentication
*s SIP Profile 1 Authentication: Off
** end
```

**xStatus SIP Profile 1 Registration [1..n] Status**

Shows the status of the registration to the SIP Proxy Server.

- **Deregister**: The system is in the process of de-registering to the SIP Proxy.
- **Failed**: The system failed to register to the SIP Proxy.
- **Inactive**: The system is not registered to any SIP Proxy.
- **Registered**: The system is registered to the SIP Proxy.
- **Registering**: The system is in the process of registering to the SIP Proxy.

**Value space of the result returned:**

```
<Deregister/Failed/Inactive/Registered/Registering>
```

**Example:**

```
xStatus SIP Profile 1 Registration 1 Status
*s SIP Profile 1 Registration 1 Status: Registered
** end
```
The SIP status, cont...

**xStatus SIP Profile 1 Registration [1..n] Reason**
Shows a message to explain the reason why the SIP registration failed.

Value space of the result returned:
<\S: 0, 100>

Example:
```
xStatus SIP Profile 1 Registration 1 Reason
*s SIP Profile 1 Registration 1 Reason: "404 Not Found"
xStatus SIP Profile 1 Registration 1 Reason
** end
```

**xStatus SIP Profile 1 Registration [1..n] URI**
Shows the URI used for registration to the SIP Proxy server.

Value space of the result returned:
<\S: 0, 100>

Example:
```
xStatus SIP Profile 1 Registration 1 URI
*s SIP Profile 1 Registration 1 URI: "firstname.lastname@company.com"
xStatus SIP Profile 1 Registration 1 URI
** end
```

The Standby status

**xStatus Standby Active**
Shows if the system is in standby or not.

Value space of the result returned:
<\On/Off>

Example:
```
xStatus Standby Active
*s Standby Active: Off
** end
```
The System Unit status

xStatus SystemUnit
Shows the top level overview of the system unit status.

Example:
xStatus SystemUnit
  *s SystemUnit ProductType: "TANDBERG Codec"
  *s SystemUnit ProductId: "TANDBERG Codec C60"
  *s SystemUnit ProductPlatform: "C60"
  *s SystemUnit Uptime: 597095
  *s SystemUnit Software Application: "Endpoint"
  *s SystemUnit Software Version: "TC3.1.0"
  *s SystemUnit Software Name: "s52000"
  *s SystemUnit Software ReleaseDate: "2010-04-30"
  *s SystemUnit Software MaxVideoCalls: 3
  *s SystemUnit Software ReleaseKey: "true"
  *s SystemUnit Software OptionKeys NaturalPresenter: "true"
  *s SystemUnit Software OptionKeys MultiSite: "true"
  *s SystemUnit Software OptionKeys PremiumResolution: "true"
  *s SystemUnit Hardware Module SerialNumber: "B1AD25AD0003"
  *s SystemUnit Hardware Module Identifier: "0"
  *s SystemUnit Hardware MainBoard SerialNumber: "PH0497201"
  *s SystemUnit Hardware MainBoard Identifier: "101401-3 [04]"
  *s SystemUnit Hardware VideoBoard SerialNumber: "PH0497874"
  *s SystemUnit Hardware VideoBoard Identifier: "101560-1 [02]"
  *s SystemUnit Hardware AudioBoard SerialNumber: "N/A"
  *s SystemUnit Hardware AudioBoard Identifier: ""
  *s SystemUnit Hardware BootSoftware: "U-Boot 2009.03-53"
  *s SystemUnit State System: Initialized
  *s SystemUnit State MaxNumberOfCalls: 3
  *s SystemUnit State MaxNumberOfActiveCalls: 3
  *s SystemUnit State NumberofActiveCalls: 1
  *s SystemUnit State NumberofSuspendedCalls: 0
  *s SystemUnit State NumberofInProgressCalls: 0
  *s SystemUnit State Subsystem Application: Initialized
  *s SystemUnit ContactInfo: "helpdesk@company.com"
** end

xStatus SystemUnit ProductType
Shows the product type.

Value space of the result returned:
<S: 0, 100>

Example:
xStatus SystemUnit ProductType
  *s SystemUnit ProductType: "TANDBERG Codec"
** end

xStatus SystemUnit ProductId
Shows the product identity.

Value space of the result returned:
<S: 0, 100>

Example:
xStatus SystemUnit ProductId
  *s SystemUnit ProductId: "TANDBERG Codec C60"
  ** end

xStatus SystemUnit ProductPlatform
Shows the product platform.

Value space of the result returned:
<S: 0, 100>

Example:
xStatus SystemUnit ProductPlatform
  *s SystemUnit ProductPlatform: "C60"
  ** end
The System Unit status, cont...

xStatus SystemUnit Uptime
Shows the number of seconds since the last restart of the codec.

Value space of the result returned:
<Integer value>

Example:
    xStatus SystemUnit Uptime
    "s SystemUnit Uptime: 597095"
    ** end

xStatus SystemUnit Software Application
Shows which software application is running on the codec.

Value space of the result returned:
<S: 0, 100>

Example:
    xStatus SystemUnit Software Application
    "s SystemUnit Software Application: "Endpoint"
    ** end

xStatus SystemUnit Software Version
Shows the software version installed on the codec.

Value space of the result returned:
<S: 0, 100>

Example:
    xStatus SystemUnit Software Version
    "s SystemUnit Software Version: "TC3.1.0"
    ** end

The System Unit status, cont...

xStatus SystemUnit Software Name
Shows the name of the software that is installed on the codec.

Value space of the result returned:

Example:
    xStatus SystemUnit Software Name
    "s SystemUnit Software Name: "s52000"
    ** end

xStatus SystemUnit Software ReleaseDate
Shows the release date of the software installed on the codec.

Value space of the result returned:
<S: 0, 100>

Example:
    xStatus SystemUnit Software ReleaseDate
    "s SystemUnit Software ReleaseDate: "2010-04-30"
    ** end

xStatus SystemUnit Software MaxVideoCalls
Shows the maximum number of simultaneous video calls that is supported.

Value space of the result returned:
<Integer value>

Example:
    xStatus SystemUnit Software MaxVideoCalls
    "s SystemUnit Software MaxVideoCalls: 3"
    ** end
The System Unit status, cont...

xStatus SystemUnit Software ReleaseKey
Shows if there is a valid releasekey for the software version that is installed on the codec.

Value space of the result returned:
<S: 0, 100>

Example:
```
xStatus SystemUnit Software ReleaseKey
*s SystemUnit Software ReleaseKey: "true"
** end
```

xStatus SystemUnit Software OptionKeys NaturalPresenter
Shows if the system has the option key installed that supports the NaturalPresenter functionality.

Value space of the result returned:
<S: 0, 100>

Example:
```
xStatus SystemUnit Software OptionKeys NaturalPresenter
*s SystemUnit Software OptionKeys NaturalPresenter: "true"
** end
```

xStatus SystemUnit Software OptionKeys MultiSite
Shows if the system has the option key installed that supports the MultiSite functionality.

Value space of the result returned:
<S: 0, 100>

Example:
```
xStatus SystemUnit Software OptionKeys MultiSite
*s SystemUnit Software OptionKeys MultiSite: "true"
** end
```

The System Unit status, cont...

xStatus SystemUnit Software OptionKeys PremiumResolution
Shows if the system has the option key installed that supports the PremiumResolution functionality.

Value space of the result returned:
<S: 0, 100>

Example:
```
xStatus SystemUnit Software OptionKeys PremiumResolution
*s SystemUnit Software OptionKeys PremiumResolution: "true"
** end
```

xStatus SystemUnit Hardware Module SerialNumber
Shows the serial number of the hardware module in the codec.

Value space of the result returned:
<S: 0, 100>

Example:
```
xStatus SystemUnit Hardware Module SerialNumber
*s SystemUnit Hardware Module SerialNumber: "B1AD25A00002"
** end
```

xStatus SystemUnit Hardware Module Identifier
Shows the revision of the hardware module in the codec.

Value space of the result returned:
<S: 0, 100>

Example:
```
xStatus SystemUnit Hardware Module Identifier
*s SystemUnit Hardware Module Identifier: "1"
** end
```
The System Unit status, cont...

**xStatus SystemUnit Hardware MainBoard SerialNumber**
Shows the serial number of the main board in the codec.

*Value space of the result returned:*
\(<S: 0, 100>\)

*Example:*
```
xStatus SystemUnit Hardware MainBoard SerialNumber
  *s SystemUnit Hardware MainBoard SerialNumber: "PH0528833"
  ** end
```

**xStatus SystemUnit Hardware MainBoard Identifier**
Shows the revision of the main board in the codec.

*Value space of the result returned:*
\(<S: 0, 100>\)

*Example:*
```
xStatus SystemUnit Hardware MainBoard Identifier
  *s SystemUnit Hardware MainBoard Identifier: "101400-5 [06]"
  ** end
```

**xStatus SystemUnit Hardware VideoBoard SerialNumber**
Shows the serial number of the video board in the codec.

*Value space of the result returned:*
\(<S: 0, 100>\)

*Example:*
```
xStatus SystemUnit Hardware VideoBoard SerialNumber
  *s SystemUnit Hardware VideoBoard SerialNumber: "PH0534914"
  ** end
```

**xStatus SystemUnit Hardware VideoBoard Identifier**
Shows the revision of the video board in the codec.

*Value space of the result returned:*
\(<S: 0, 100>\)

*Example:*
```
xStatus SystemUnit Hardware VideoBoard Identifier
  *s SystemUnit Hardware VideoBoard Identifier: "101410-4 [07]"
  ** end
```

**xStatus SystemUnit Hardware AudioBoard SerialNumber**
Shows the serial number of the audio board in the codec.

*Value space of the result returned:*
\(<S: 0, 100>\)

*Example:*
```
xStatus SystemUnit Hardware AudioBoard SerialNumber
  *s SystemUnit Hardware AudioBoard SerialNumber: "TBD"
  ** end
```

**xStatus SystemUnit Hardware AudioBoard Identifier**
Shows the revision of the audio board in the codec.

*Value space of the result returned:*
\(<S: 0, 100>\)

*Example:*
```
xStatus SystemUnit Hardware AudioBoard Identifier
  *s SystemUnit Hardware AudioBoard Identifier: "101420-2 [No objl.]"
  ** end
```
The System Unit status, cont...

**xStatus SystemUnit Hardware BootSoftware**
Shows the version of the boot software that is installed on the codec.

*Value space of the result returned:*

\(<S: 0, 100>\)

*Example:*

```
xStatus SystemUnit Hardware BootSoftware
*s SystemUnit Hardware BootSoftware: "U-Boot 2009.03-37"
** end
```

**xStatus SystemUnit State System**
Shows what state the system is in.

- **InCall:** The system is in a call.
- **Initialized:** The system is ready for use.
- **Initializing:** The system is initializing.
- **Multisite:** The system is in a Multisite conference.
- **Sleeping:** The system is in sleep mode.

*Value space of the result returned:*

\(<\text{InCall}/\text{Initialized}/\text{Initializing}/\text{Multisite}/\text{Sleeping}>\)

*Example:*

```
xStatus SystemUnit State System
*s SystemUnit State System: Initialized
** end
```

**xStatus SystemUnit State MaxNumberOfCalls**
Shows the the maximum number of simultaneous calls. Calls that are set on hold/transfer are not counted as active.

*Value space of the result returned:*

\(<0..5>\)

*Example:*

```
xStatus SystemUnit State MaxNumberOfActiveCalls
*s SystemUnit State MaxNumberOfActiveCalls: 3
** end
```

**xStatus SystemUnit State MaxNumberOfActiveCalls**
Shows the number of active calls.

*Value space of the result returned:*

\(<0..5>\)

*Example:*

```
xStatus SystemUnit State NumberOfActiveCalls
*s SystemUnit State NumberOfActiveCalls: 0
** end
```

**xStatus SystemUnit State NumberOfSuspendedCalls**
Shows the number of suspended calls.

*Value space of the result returned:*

\(<0..5>\)

*Example:*

```
xStatus SystemUnit State NumberOfSuspendedCalls
*s SystemUnit State NumberOfSuspendedCalls: 0
** end
```
The System Unit status, cont...

xStatus SystemUnit State NumberOfInProgressCalls
Shows the number of calls in progress.

Value space of the result returned:
<0..5>

Example:
```plaintext
xStatus SystemUnit State NumberOfInProgressCalls
*s SystemUnit State NumberOfInProgressCalls: 0
** end
```

xStatus SystemUnit State Subsystem Application
Shows the status of the sub system application.
Initialized: The sub-system application is initialized.
Initializing: The sub-system application is initializing.

Value space of the result returned:
<Initialized/Initializing>

Example:
```plaintext
xStatus SystemUnit State Subsystem Application
*s SystemUnit State Subsystem Application: Initialized
** end
```

xStatus SystemUnit ContactInfo
Shows the address which another system can dial to reach this system.

Value space of the result returned:
<S: 0, 100>

Example:
```plaintext
xStatus SystemUnit ContactInfo
*s SystemUnit ContactInfo: "firstname.lastname@company.com"
** end
```

The Video Input status

xStatus Video Input
Shows the top level overview of the video input status.

Example:
```plaintext
xStatus Video Input
*s Video Input Source 1 Resolution Height: 1080
*s Video Input Source 1 Resolution Width: 1920
*s Video Input Source 1 Resolution RefreshRate: 60
*s Video Input Source 2 Resolution Height: 0
*s Video Input Source 2 Resolution Width: 0
*s Video Input Source 2 Resolution RefreshRate: 0
*s Video Input Source 3 Resolution Height: 0
*s Video Input Source 3 Resolution Width: 0
*s Video Input Source 3 Resolution RefreshRate: 0
*s Video Input HDMI 1 Connected: True
*s Video Input HDMI 1 SignalState: OK
*s Video Input HDMI 2 Connected: True
*s Video Input HDMI 2 SignalState: OK
*s Video Input DVI 2 Connected: Unknown
*s Video Input DVI 2 SignalState: Unknown
*s Video Input DVI 3 Connected: Unknown
*s Video Input DVI 3 SignalState: Unknown
*s Video Input Legacy 3 Connected: False
*s Video Input Legacy 3 SignalState: Unknown
** end
```

xStatus Video Input Source [1..3] Resolution Height
Shows the resolution height (in pixels) for the video input source.

Value space of the result returned:
<0..3000>

Example:
```plaintext
xStatus Video Input Source 1 Resolution Height
*s Video Input Source 1 Resolution Height: 1080
** end
```
The Video Input status, cont...

**xStatus Video Input Source [1..3] Resolution Width**
Shows the resolution width (in pixels) for the video input source.

*Value space of the result returned:*
<0..4000>

*Example:*
```
xStatus Video Input Source 1 Resolution Width
*s Video Input Source 1 Resolution Width: 1920
** end
```

**xStatus Video Input Source [1..3] Resolution RefreshRate**
Shows the resolution refresh rate (Hz) for the video input source.

*Value space of the result returned:*
<0..300>

*Example:*
```
xStatus Video Input Source 1 Resolution RefreshRate
*s Video Input Source 1 Resolution RefreshRate: 50
** end
```

**xStatus Video Input HDMI [1..2] Connected**
Shows if there is something connected to the HDMI connector. Not all connections can be detected.

*Value space of the result returned:*
<True/False>

*Example:*
```
xStatus Video Input HDMI 1 Connected
*s Video Input HDMI 1 Connected: True
** end
```

**xStatus Video Input DVI [2..3] Connected**
Shows if there is something connected to the DVI connector. Not all connections can be detected.

*Value space of the result returned:*
<True/False>

*Example:*
```
xStatus Video Input DVI 2 Connected
*s Video Input DVI 2 Connected: False
** end
```

**xStatus Video Input HDMI [1..2] SignalState**
Shows the signal state for the HDMI input.

*Value space:*
<Unknown/OK/Unsupported>

*Example:*
```
xStatus Video Input HDMI 1 SignalState
*s Video Input HDMI 1 SignalState: OK
** end
```

**xStatus Video Input DVI [2, 3] SignalState**
Shows the signal state for the DVI-I input.

*Value space:*
<Unknown/OK/Unsupported>

*Example:*
```
xStatus Video Input DVI 2 SignalState
*s Video Input DVI 2 SignalState: OK
** end
```

The Video Input status, cont...

**xStatus Video Input Legacy [3] Connected**
Shows if there is something connected to the Legacy, which are the Y/Comp and C connectors. Not all connections can be detected.

*Value space of the result returned:*
- True/False

*Example:*
```c
xStatus Video Input Legacy 3 Connected
*s Video Input Legacy 3 Connected: False
** end
```

**xStatus Video Input Legacy [3] SignalState**
Shows the signal state for the Legacy, which are the Y/Comp and C inputs.
- Unknown: The signal format is unknown.
- OK: A signal is detected and the signal format is supported.
- Unsupported: A signal is detected, but the signal format is not supported.

*Value space:*
- Unknown/OK/Unsupported

*Example:*
```c
xStatus Video Input Legacy 3 SignalState
*s Video Input Legacy 3 SignalState: OK
** end
```

The Video Output status

**xStatus Video Output**
Shows the top level overview of the video output status.

*Example:*
```c
xStatus Video Output
*s Video Output HDMI 1 Resolution Height: 1080
*s Video Output HDMI 1 Resolution Width: 1920
*s Video Output HDMI 1 Resolution RefreshRate: 60
*s Video Output DVI 2 Resolution Height: 768
*s Video Output DVI 2 Resolution Width: 1024
*s Video Output DVI 2 Resolution RefreshRate: 60
*s Video Output Legacy 3 Resolution Height: 480
*s Video Output Legacy 3 Resolution Width: 720
*s Video Output Legacy 3 Resolution RefreshRate: 60
** end
```

**xStatus Video Output HDMI [1] Resolution Height**
Shows the resolution height (in pixels) for the video output HDMI.

*Value space of the result returned:*
- 120..3000

*Example:*
```c
xStatus Video Output HDMI 1 Resolution Height
<s Video Output HDMI 1 Resolution Height: 720
** end
```

**xStatus Video Output HDMI [1] Resolution Width**
Shows the resolution width (in pixels) for the video output HDMI.

*Value space of the result returned:*
- 176..4000

*Example:*
```c
xStatus Video Output HDMI 1 Resolution Width
<s Video Output HDMI 1 Resolution Width: 1280
** end
```
The Video Output status, cont...

Example:
<176.4000>

Value space of the result returned:
Shows the resolution width (in pixels) for the video output HDMI.

```
xStatus Video Output HDMI [1] Resolution Width
```
Example:
<1280>

Value space of the result returned:
Shows the resolution height (in pixels) for the video output DVI.

```
xStatus Video Output DVI [2] Resolution Height
```
Example:
<1280.3000>

Value space of the result returned:
Shows the resolution height (in pixels) for the video output Legacy (Composite).

```
xStatus Video Output Legacy [3] Resolution Height
```
Example:
<1280.3000>

Value space of the result returned:
Shows the resolution width (in pixels) for the video output Legacy (Composite).

```
xStatus Video Output Legacy [3] Resolution Width
```
Example:
<1280.3000>

The Video Output status, cont...
The Video Output status, cont...

**xStatus Video Output Legacy [3] Resolution RefreshRate**
Shows the resolution refresh rate (Hz) for the video output Legacy (Composite).

**Value space of the result returned:**
\(<1..300>\)

**Example:**

```
xStatus Video Output Legacy 3 Resolution RefreshRate
*s Video Output Legacy 3 Resolution RefreshRate: 60
** end
```

The Video Layout status

**xStatus Video Layout**
Shows the top level overview of the video status.

**Example:**

```
xStatus Video Layout
*s Video Layout Site 1 Output 1 FamilyName: "speaker"
*s Video Layout Site 1 Output 1 FullFamilyName: "speaker-sv-on"
*s Video Layout Site 1 Output 1 FamilyNumber: 1027
*s Video Layout Site 1 Output 1 GraphicName: "1top-1small"
*s Video Layout Site 1 Output 1 GraphicNumber: 1017
*s Video Layout Site 1 Output 1 Descriptor: 4
*s Video Layout Site 1 Output 1 DescriptorOutput: 1
*s Video Layout Site 1 Output 1 Frame 1 PositionX: 1333
*s Video Layout Site 1 Output 1 Frame 1 PositionY: 59
*s Video Layout Site 1 Output 1 Frame 1 Width: 7334
*s Video Layout Site 1 Output 1 Frame 1 Height: 7334
*s Video Layout Site 1 Output 1 Frame 1 Layer: 1
*s Video Layout Site 1 Output 1 Frame 1 VideoSourceType: "site"
*s Video Layout Site 1 Output 1 Frame 1 VideoSourceId: 27
*s Video Layout Site 1 Output 1 Frame 1 InputNumber: 1
*s Video Layout Site 1 Output 1 Frame 1 Filename: ""
-
- continues with the video layout status for the Sites [1..n], Outputs [1..3] and Frames [1..6].
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] FamilyName**
Shows the name of the video layout family.

**Value space of the result returned:**
\(<S: 0, 100>\)

**Example:**

```
xStatus Video Layout Site 1 Output 1 FamilyName
*s Video Layout Site 1 Output 1 FamilyName: "full"
** end
```
The Video Layout status, cont...

**xStatus Video Layout Site [1..n] Output [1..3] FullFamilyName**
Shows the name, included information about selfview on/off, for the video layout family.

Value space of the result returned:

```
<S: 0, 100>
```

Example:
```
xStatus Video Layout Site 1 Output 1 FullFamilyName
*s Video Layout Site 1 Output 1 FullFamilyName: "full-sv-on"
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] FamilyNumber**
Shows the number of the video layout family. The number identifies the layout family of the specified output.

Value space of the result returned:

```
<Integer value>
```

Example:
```
xStatus Video Layout Site 1 Output 1 FamilyNumber
*s Video Layout Site 1 Output 1 FamilyNumber: 1014
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] GraphicNumber**
Shows the number of the graphic layout. The number identifies the layout used right now at the specified output.

Value space of the result returned:

```
<Integer value>
```

Example:
```
xStatus Video Layout Site 1 Output 1 GraphicNumber
*s Video Layout Site 1 Output 1 GraphicNumber: 1037
** end
```

The Video Layout status, cont...

**xStatus Video Layout Site [1..n] Output [1..3] Descriptor**
Shows the descriptor of the layout.

Value space of the result returned:

```
<Integer value>
```

Example:
```
xStatus Video Layout Site 1 Output 1 Descriptor
*s Video Layout Site 1 Output 1 Descriptor: 3
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] DescriptorOutput**
Shows the descriptor output of the layout.

Value space of the result returned:

```
<Integer value>
```

Example:
```
xStatus Video Layout Site 1 Output 1 DescriptorOutput
*s Video Layout Site 1 Output 1 DescriptorOutput: 1
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] PositionX**
Shows the horizontal position of the upper left corner of the frame.

Value space of the result returned:

```
<0..10000>
```

Example:
```
xStatus Video Layout Site 1 Output 1 Frame 1 PositionX
*s Video Layout Site 1 Output 1 Frame 1 PositionX: 0
** end
```
The Video Layout status, *cont*

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] PositionY**

Shows the vertical position of the upper left corner of the frame.

*Value space of the result returned:*

```
<0..10000>
```

*Example:*

```
xStatus Video Layout Site 1 Output 1 Frame 1 PositionY
*s Video Layout Site 1 Output 1 Frame 1 PositionY: 0
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] Width**

Shows the width of the frame.

*Value space of the result returned:*

```
<0..10000>
```

*Example:*

```
xStatus Video Layout Site 1 Output 1 Frame 1 Width
*s Video Layout Site 1 Output 1 Frame 1 Width: 10000
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] Height**

Shows the height of the frame.

*Value space of the result returned:*

```
<0..10000>
```

*Example:*

```
xStatus Video Layout Site 1 Output 1 Frame 1 Height
*s Video Layout Site 1 Output 1 Frame 1 Height: 10000
** end
```

The Video Layout status, *cont*

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] Layer**

Shows the layer of the frame.

*Value space of the result returned:*

```
<1..6>
```

*Example:*

```
xStatus Video Layout Site 1 Output 1 Frame 1 Layer
*s Video Layout Site 1 Output 1 Frame 1 Layer: 1
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] VideoSourceType**

Describes the video source type in the frame.

*Value space of the result returned:*

```
<S:0..100>
```

*Example:*

```
xStatus Video Layout Site 1 Output 1 Frame 1 VideoSourceType
*s Video Layout Site 1 Output 1 Frame 1 VideoSourceType: "graphic"
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] VideoSourceId**

Shows the video source Id which is used when adding or updating frames. See the xCommand Video Layout Frame Add and xCommand Video Layout Frame Update.

*Value space of the result returned:*

```
<1..5>
```

*Example:*

```
xStatus Video Layout Site 1 Output 1 Frame 1 VideoSourceId
*s Video Layout Site 1 Output 1 Frame 1 VideoSourceId: 0
** end
```
The Video Layout status, *cont*

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] InputNumber**

Shows the layout input number.

*Value space of the result returned:*

<1..5>

**Example:**

```
xStatus Video Layout Site 1 Output 1 Frame 1 InputNumber
*% Video Layout Site 1 Output 1 Frame 1 InputNumber: 0
** end
```

**xStatus Video Layout Site [1..n] Output [1..3] Frame [1..6] Filename**

Shows the filename of the layout frame.

*Value space of the result returned:*

<S: 0, 200>

**Example:**

```
xStatus Video Layout Site 1 Output 1 Frame 1 Filename
*% Video Layout Site 1 Output 1 Frame 1 Filename: "/user/posters/wallpaper.png"
** end
```
The Experimental status
The Experimental commands can be used 'as is' and will not be further documented.

**NOTE!** The Experimental commands WILL change.

**xStatus Experimental**
Shows the top level overview of the Experimental status information.

**Example:**
```
xStatus Experimental
*s Experimental Conference Site 18 Capabilities FECC: On
*s Experimental Audio StereoEchoCancellation Mode: Off
*s Experimental Audio Input Connectors Microphone 1 Activity: True
*s Experimental Audio Input Connectors Microphone 2 Activity: False
*s Experimental Audio Input Connectors Microphone 3 Activity: False
*s Experimental Audio Input Connectors Microphone 4 Activity: False
*s Experimental Audio Input Connectors Line 1 Activity: False
*s Experimental Audio Input Connectors Line 2 Activity: False
*s Experimental Audio Input Connectors HDMI 2 Activity: False
** end
```

**xStatus Experimental Conference Site [1..n] Capabilities FECC**
NOTE! The Experimental command can be used 'as is'. The Experimental settings WILL change.
Shows the FECC (Far End Camera Control) status. This command is available only when in a conference.

**Value space of the result returned:**
<On/Off>

**Example:**
```
xStatus Experimental Conference
*s Experimental Conference Site 18 Capabilities FECC: On
** end
```

**xStatus Experimental Audio StereoEchoCancellation Mode**
NOTE! The Experimental command can be used 'as is'. The Experimental settings WILL change.
Shows the stereo echo cancellation mode.

**Value space of the result returned:**
<On/Off>

**Example:**
```
xStatus Experimental Audio StereoEchoCancellation Mode
*s Experimental Audio StereoEchoCancellation Mode: Off
** end
```

**xStatus Experimental Audio Input Connectors Microphone [1..4] Activity**
The Experimental commands can be used 'as is'. NOTE! The Experimental settings WILL change.
Shows if there is a signal on the connector.

**Value space of the result returned:**
<True/False>

**Example:**
```
xStatus Experimental Audio Input Connectors Microphone 1 Activity
*s Experimental Audio Input Connectors Microphone 1 Activity: True
** end
```

**xStatus Experimental Audio Input Connectors Line [1..2] Activity**
The Experimental commands can be used 'as is'. NOTE! The Experimental settings WILL change.
Shows if there is a signal on the connector.

**Value space of the result returned:**
<True/False>

**Example:**
```
xStatus Experimental Audio Input Connectors Line 1 Activity
*s Experimental Audio Input Connectors Line 1 Activity: False
** end
```

**xStatus Experimental Audio Input Connectors HDMI 2 Activity**
NOTE! The Experimental commands WILL change.
Shows if there is a signal on the connector.

**Value space of the result returned:**
<True/False>

**Example:**
```
xStatus Experimental Audio Input Connectors HDMI 2 Activity
*s Experimental Audio Input Connectors HDMI 2 Activity: False
** end
```
The Experimental status, cont...

**xStatus Experimental Audio Input Connectors HDMI [3, 4] Activity**

The Experimental commands can be used 'as is'. NOTE! The Experimental settings WILL change.

Shows if there is a signal on the connector.

**Value space of the result returned:**

<True/False>

**Example:**

```plaintext
xStatus Experimental Audio Input Connectors HDMI 3 Activity
*s Experimental Audio Input Connectors HDMI 3 Activity: False
** end
```

**xStatus Experimental Audiotracking Camera 1 Angle**

NOTE! The Experimental command can be used 'as is' and will not be further documented. The Experimental settings WILL change.

**Value space of the result returned:**

<Integer value>

**Example:**

```plaintext
xStatus Experimental Audiotracking Camera 1 Angle
*s Experimental Audiotracking Camera 1 Angle: 0
** end
```

**xStatus Experimental Audiotracking Camera 1 Quality**

NOTE! The Experimental command can be used 'as is' and will not be further documented. The Experimental settings WILL change.

**Value space of the result returned:**

<Integer value>

**Example:**

```plaintext
xStatus Experimental Audiotracking Camera 1 Quality
*s Experimental Audiotracking Camera 1 Quality: 0
** end
```
Chapter 6

Appendices
Dynamic audio API

With a set of xCommand Audio API commands, the audio system is fully configurable:

- A local input is defined as a mix of input connectors. A local input can be created and deleted. You can add and remove input connectors and update the local input settings.
- A local output is a mix of local input and remote input signals. All connectors attached to a local output receive the same signal.
- A remote input and remote output pair is created for each call.
- A remote output receives the signals from all local and remote Inputs, except the remote input from the same site.
- The mixer matrix of each local and remote output can easily be managed by disconnecting and connecting local and remote inputs.

Example 1:
The dynamic audio API offers a great flexibility when configuring the system and it is simple to use. To have audio in your calls you only need one local input with a microphone attached to it, and one local output with an output connector attached to it.

When the call is established a remote input and remote output pair are created and all the connections to these from the local audio system are set up automatically.

Local audio setup

REMOTE INPUT 3

REMOTE OUTPUT 4

LOCAL INPUT 1

LOCAL OUTPUT 2

MICROPHONE 1

MICROPHONE 2

Line 1

Line 2

The Local input 1 has been defined with two Microphones.

The Local output 2 has been defined with two Line outputs.

The audio from the Remote input 3 goes to the Local output 2.

The audio from the Local input 1 goes to the Remote output 4.
The equalizer

The system has 8 user defined equalizers, each made up of 6 second order IIR sections, see the illustration below. Each of these 8 equalizers can be applied to one or more of the audio input and output connectors on the codec.

The Audio Console application

We recommend using the Audio Console, with the build in equalizer GUI, to modify the equalizers. The Audio Console application is found at the TANDBERG Developer Zone web page. Go to: ► http://developer.tandberg.com/web/guest/tools/integrators/audio-console

The equalizer filter parameters

Each section (1 … 6) of each user defined equalizer (1 … 8) can be modified independently by setting the four parameters:

- Filter type (Peaking, low shelf, high shelf, low pass, high pass)
- Frequency
- Gain
- Q-value

To switch off one of the six equalizer sections; set the second order section to have a flat frequency response. This can be done by setting the filter type to “none” or by setting the filter type to “peaking” and the gain to “0” (zero).

The filter types

The illustrations below shows the 5 different filter types and frequency response variations dependent on some of the parameter variations. The Q-value for low pass, high pass, low shelf and high shelf filters should be set to \(1/\sqrt{2}\) in order to get maximally flat responses. The Q-value (or Q-factor) is defined as \(Q=f_0/bw\). Where \(f_0=\)resonance frequency in Hz; and \(bw=\)filter bandwidth in Hz.

The equalizer IIR filter

<table>
<thead>
<tr>
<th>Section 1</th>
<th>Section 2</th>
<th>Section 3</th>
<th>Section 4</th>
<th>Section 5</th>
<th>Section 6</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Diagram" /></td>
<td><img src="image2" alt="Diagram" /></td>
<td><img src="image3" alt="Diagram" /></td>
<td><img src="image4" alt="Diagram" /></td>
<td><img src="image5" alt="Diagram" /></td>
<td><img src="image6" alt="Diagram" /></td>
</tr>
</tbody>
</table>

1. **Peaking filter**
   - Frequency (Hz)
   - Gain

2. **Low pass and high pass filters**
   - Lowpass filter
   - Highpass filter

3. **Low shelf filter**
   - Gain = 6 dB
   - Gain = 4 dB
   - Gain = 2 dB
   - Gain = -2 dB
   - Gain = -4 dB
   - Gain = -6 dB

4. **High shelf filter**
   - Gain = 6 dB
   - Gain = 4 dB
   - Gain = 2 dB
   - Gain = -2 dB
   - Gain = -4 dB
   - Gain = -6 dB

The illustrations above show the frequency response variations dependent on some of the parameter variations. The Q-value for low pass, high pass, low shelf and high shelf filters should be set to \(1/\sqrt{2}\) in order to get maximally flat responses. The Q-value (or Q-factor) is defined as \(Q=f_0/bw\). Where \(f_0=\)resonance frequency in Hz; and \(bw=\)filter bandwidth in Hz.
Local stereo circuit example with API commands

The graphic gives a picture of how to configure the audio system for recording a stereo presentation. You can do the same from the API. The belonging xCommands and xConfigurations are listed below.

- **xCommand Audio LocalInput Update** Inputid:5 Name:"Presentation" MixerMode:Fixed AGC:On Mute:Off Channels:2
- **xCommand Audio LocalOutput Update** Outputid:6 Name:"Recorder" Loudspeaker:No Channels:2
- **xConfiguration Audio Input** Line 1 Channel: Left
- **xConfiguration Audio Input** Line 2 Channel: Right
- **xConfiguration Audio Output** Line 1 Channel: Left
- **xConfiguration Audio Output** Line 2 Channel: Right

**Local stereo circuit example**

![Local stereo circuit example](imageLink)

**Presentation**

The local input has two connectors: Line 1 and Line 2. When these two connectors are mixed together as a stereo signal, Line 1 is the left channel and Line 2 is the right channel.

**Recorder**

The local output has two connectors: Line 1 and Line 2. The local output pans the stereo signal according to the channel configuration of the connectors. Line 1 receives the left channel and Line 2 receives the right channel.

**Stereo**

Stereo in point to point call

**Receive stereo**

If the other participant sends stereo the codec will receive stereo. To play stereo the local output still needs to be configured correctly (see local stereo circuit example).

**Transmit stereo**

In order to send stereo, the other participant must be able to receive stereo. Also, the local input with the stereo signal must be configured correctly (see local stereo circuit example). In addition, there must be a signal on one of the connectors connected to the local input.

**Stereo in Multisite**

The TANDBERG Codec C60 will only send and receive mono signals in multisite.

**Example of local stereo configuration**

When you want to record a stereo presentation the audio system can be configured as shown in the illustration.

In the example the local input has two connectors: Line 1 and Line 2. When these two connectors are mixed together as a stereo signal, Line 1 is the left channel and Line 2 is the right channel.

And the local output has two connectors: Line 1 and Line 2. The local output pans the stereo signal according to the channel configuration of the connectors. Line 1 receives the left channel and Line 2 receives the right channel.
Startup script

You can add a startup script on the codec to execute certain commands from the API during boot up.

Adding a startup script

The startup script can be used to execute certain commands from the API during boot up. To enable this feature one must log in to the codec as root and follow the below points.

Login to the codec

1. Connect to the codec through the network, using a command line interface (ssh, telnet or scp) and login as root
2. Make a user directory using the following command: “mkdir /user/scripts.d”
3. Put an executable file (permission must be changed to executable) in this directory.

Example of the text in such a file:

```bash
#!/usr/bin/env tsh
xCommand Audio LocalInput Update InputId: 1 MixerMode:Fixed
```

About the startup script file

- The file must start with the following sequence:
  ```bash
  #!/usr/bin/env tsh
  ```
- The file can contain any xCommand or xConfiguration command
- The system will execute the commands/configurations in sequence.
- The file can have any name as long as it is placed in this directory.
- For multiple commands you must use Unix end of line (LF). Windows end of line will not work.
Remote Control TRC5—Key map

The TANDBERG Remote control TRC5 has the following button codes and IR signal parameters.

You will find a one page overview of the TANDBERG Remote Control TRC5 overleaf.

<table>
<thead>
<tr>
<th>Button codes - Remote control TRC5</th>
<th>Button codes - Remote control TRCS</th>
<th>IR Signal parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dec</td>
<td>Hex</td>
<td>Address</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0A</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0B</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0C</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0D</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>0E</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>0F</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>0</td>
</tr>
</tbody>
</table>
Remote Control TRC5—One page overview

**Microphone:** Press the key to toggle the microphones on/off.

**Volume:** Press the + or – on the key to adjust the codec volume.

**Mute:** Press the – on the key to mute the volume during an incoming call.

**OK/Select:** Press the key to confirm your choice or selection.

**Phone book:** Press the key to display the local Phone book.

**Home:** Press the key to show the menu on screen.

**Call:** Using the key:

- **INITIATE A CALL:** Select a name from the Phone book or enter the name, number or URI and press the Call key to initiate the call.
- **SHORTCUT TO RECENT CALLS:** Use the Call key as a shortcut to Recent Calls when the Call menu is not visible.

**Clear:** Press the key to remove characters in a text field.

**Waking up the system**

Grab the remote control and make sure your hand touches the rubber line sensors going along both sides of the remote control.

-or: Touch any key on the remote control.

**Presentation:** Press the key to show/hide a presentation.

**Zoom:** Press the + or – on the key to zoom the camera in and out.

**Arrows:**

- **Up/Down:** Use the and arrow keys to navigate in the menu.
- **Arrow Right:** Press the arrow key to expand the selected menu item or to move to the right in a text field.
- **Arrow Left:** Press the arrow key to go one step back in the menu or to move to the left in a text field.

**Layout:** Press the key to display the Layout menu, then select a view in the menu.

**End call/Standby:** Press the key to end a call, or when idle, press and hold the key to go into standby mode.

**Alphanumeric keypad:** Use the keypad in the same way as you would use a cellular phone.

- **0-9, a-z, period (.), @, space, #:** Press a key repeatedly to toggle between the options displayed on each key.
- **abc/123 #:** Press the # key to toggle between touch tones mode (long press), lower case characters and numbers.

**IR transmitter range (DIP switch setting)**

The IR transmitter has a short and long range. Open the battery cover and remove the batteries to set the DIP switch.

- **Short range (1 m):** Move the DIP switch down.
- **Longer range:** Move the DIP switch up.