NCast Product Specification

Telepresenter
Serial Interface Implementation

Revision 3.7

Software Release 5.2.0

April 1st, 2010

Table of Contents - Serial Command Reference

1 Introduction

1.1 Purpose

The NCast Serial Specification outlines the proposed design implementation for an RS-232 interface to control the Telepresenter by external controllers such as a Crestron or AMX device. This same serial command set may be used on a Telnet connection using an IP network as the control link.

1.2 Document Overview

The following text provides a table describing the expected ASCII command text originating from the Telepresenter system.

2 Serial Port Interface

2.1 Serial Port Connector

The serial interface connector is located at the back of the Telepresenter. This is a Male DB9 type connector.

2.2 Serial Port Interface

The serial port interface conforms to the RS232 Specification and is configured as a DTE.

2.3 Serial Port Settings

The default speed for the serial link in each direction will be 38,400 bps. Each character will consist of one start bit, eight data bits and one stop bit. The high-order data bit will be normally unused and set to zero. There will be no parity bit.

Parameter	Setting
Data Bits per Second	38,400
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	None

2.4 Serial Pinouts

For proper operation of the serial interface a minimum of three connecting wires are needed: Rx Data, Tx Data and Ground

Pin	Signal	Description
1	DCD	Data Carrier Detect
2	Rx Data	Serial In, Received Data
3	Tx Data	Serial Out, Transmit Data
4	DTR	Data Terminal Ready
5	GND	Ground
6	DSR	Data Set Ready
7	RTS	Request to Send
8	CTS	Clear to Send
9	RI	Ring Indicate

2.5 Telnet Interface

The Telnet Interface is reached through use of a Telnet program or its equivalent. A TCP connection is made to the default Telnet port (currently port 7474) and an initial Id command is sent to initiate communication with the Telepresenter. The port number used may be changed on the Telnet configuration web page.

Example:

telnet telepresenter.mycompany.com 7474

IdTelnet,002,my.password

The password is set on the Telnet configuration page. An IP restriction may also be set. The IP restriction controls which IP addresses are permitted to connect to the Telnet port. As Telnet passwords are sent in clear-text, proper security controls should be in place to restrict the range of IP addresses which may issue commands to the Telepresenter.

An IP restriction entry is either a single IP address:

w.x.y.z

or a range of IP addresses:

w.x.y.*

or a subset of a network:

w.x.y.0/24

or a list (comma separated) of IP addresses or ranges:

a.b.c.d, m.n.o.*, w.x.y.z

To exit from a Telnet session, simply issue the Quit command (QT):

QT<lf>

The TCP connection will then be closed and a new login will be required.

If the Telnet TCP connection does not receive a keep-alive protocol response within the timeout period (15 minutes), the connection will be closed and must be re-opened by the Telnet client.

3 Command Format

3.1 Command Strings

A standard command consists of alpha-numeric characters followed by a line feed:

AA<lf>

where the first character is always an alphabetic character (A-Z, a-z) and <lf> stands for the line feed character (hex 0x0a). Examples:

L0<lf>	Display loopback is off.
C2<lf>	Change operation to Channel 2.

Upper-case and lower-case must be treated uniquely, so the commands

aa<lf>

Aa<lf>

aA<lf>

AA<lf>

represent four different commands in the command set.

Commands may be terminated with a carriage return-line feed <cr><lf> pair. The <cr> will be discarded. For example,

AA<cr><lf>

produces the same result as

AA<lf>

but

AA<cr>

Is not a valid command.

3.2 Command Response

After a command is sent to the Telepresenter, a command confirmation response is generated, indicating that the command has been received:

&<lf>

This response does not indicate that the command is finished or that it has completed execution. It simply confirms that the command string has been received by the Telepresenter.

3.3 Command Processing

After reception of the command string, the Telepresenter executes the command. Once processing has completed the Telepresenter will return the following strings:

+<command><lf>	Command has executed properly
-<command><lf>	Command has not executed properly

In the case of a command which returns a value, the command return value is sent in the lines between "&<lf>" and "+<command>"

Here is an example of dialog asking for the list of participants:

Controller:	"PL<lf>"
M4:	"&<lf>"
M4:	“=1,C,S,Paris<lf>”
M4:	“=2,P,R,New York<lf>”
M4:	“=3,P,R,London<lf>”
M4:	“=4,P,R,Chicago<lf>”
M4:	“+PL<lf>”

And if we want to switch to the Participant named Chicago, we use the command:

Controller:	"P4<lf>"
M4:	"&<lf>"
M4:	"+P4<lf>"

Here is an example of a complete dialog to start a Telepresenter on Channel 1:

Controller:	“IdSerial,002<lf>”
M4:	“&<lf>”
M4:	“@NCast,002<lf>”
M4:	“+IdSerial,002<lf>”
Controller:	"C1<lf>"
M4:	"&<lf>"
M4:	“+C1<lf>"

To end the Session, only the following command is required:

Controller:	"PE<lf>"
M4:	"&<lf>"
M4:	“+PE<lf>"

3.4 Essential Commands

In the table below, the minimal commands required to control a Telepresenter are the Id (Identification), Cn (Channel Start) and PE (Program End). Once those have been implemented the next level would probably include R0/R1 (Record Stop/Start). Finally, there might be a need to switch Main or Pip input devices (Gn, Vn) or do an input swap (SW). These few commands are the only ones required for simple operation of the unit.

4 Command Table

4.1 Command List

Featured below is a table defining all commands for the Serial Interface. Included are the ASCII commands, with the <lf> sequence assumed to be following every outlined command.

Function	Input Commands	Result if Processed
Audio Input Mic	A1	Enables Audio Mic Input
Audio Input Mic, Gain	A1,<num>	Enables Audio Mic Input, set Gain to 0-100
Audio Input Line	A2	Enables Audio Line Input
Audio Input Line, Gain	A2,<num>	Enables Audio Line Input, set Gain to 0-100
Audio Input USB	A3	Enable Audio Input from USB port
Border for PIP Off	b0	Set PIP window border off
Border for PIP On	b1	Set PIP window border on
Select Channel 1-100	C1…100	Selects Channel 1-100, and starts a session
Select Channel with Change of role	C<num>,<role>	Select Channel 1-100, and start a session with a given role.
Display Output - Auto	D0	Set display output to Auto
Display Output - VGA	D1	Set display output to VGA (640x480) @ 60 Hz. (DMT)
Display Output - SVGA	D2	Set display output to SVGA (800x600) @ 60 Hz. (DMT)
Display Output - XGA	D3	Set display output to XGA (1024x768) @ 60 Hz. (DMT)
Display Output - SXGA	D4	Set display output to SXGA (1280x960) @ 60 Hz. (DMT)
Display Output - SXGA	D5	Set display output to SXGA (1280x1024) @ 60 Hz. (DMT)
Display Output - UXGA	D6	Set display output to UXGA (1600x1200) @ 60 Hz. (DMT)
Display Output – GTF	D7-D11	Set display output to 1280x720, 1280x768, 1280x800, 1920x1080, 1920x1200 @60 Hz. (GTF)
Display Output - CVT	D12-D16	Set display output to 1280x720, 1280x768, 1280x800, 1920x1080, 1920x1200 @ 60 Hz. (GTF)
Display Output - CVT-RB	D17-D21	Set display output to 1280x720, 1280x768, 1280x800, 1920x1080, 1920x1200 @ 60 Hz. (CVT-RB)
Display Output - 861B	D22-D25	Set display output to 1280x720 @ 50 Hz., 1280x720 @ 60 Hz., 1920x1080 @ 50 Hz., 1920x1080 @ 60 Hz., (861B)
Display Aspect Ratio	d0	Set display aspect ratio to Auto.
Display Aspect Ratio	d1-d5	Set display aspect ratio to 5:4, 4:3, 5:3, 16:10, 16:9
Graphics Input Composite	G1	Enables Graphics Input from Composite Input

Function	Input Commands	Result if Processed
Graphics Input S-Video	G2	Enables Graphics Input from S-Video Input
Graphics Input XGA	G3	Enables Graphics Input from XGA
Graphics Input DVI	G4	Enables Graphics Input from DVI input
Graphics Input Auto Detect	G5	Enables Graphics Input using Signal Auto Detect
Graphics Input Adjustments	G1,brt,cntr,sat,hue,s	Set Composite video input adjustment values: brightness (0-100), contrast (0-100), saturation (0-100), hue (0-100), sharpness (0-44).
Graphics Input Adjustments	G2,brt,cntr,sat,hue,s	Set S-Video input adjustment values: brightness (0-100), contrast (0-100), saturation (0-100), hue (0-100), sharpness (0-44).
Graphics Input Adjustments	G3,brt,cntr,sat,hue	Set graphics XGA input adjustment values: brightness (0-100), contrast (0-100), saturation (0-100), hue (0-100).
Graphics Input Adjustments	G4,brt,cntr,sat,hue	Set graphics DVI input adjustment values: brightness (0-100), contrast (0-100), saturation (0-100), hue (0-100).
Local Playback - Stop	l0	Stop archive playback (returns error 19 if already stopped). (Note: lowercase el-zero)
Local Playback - Resume	l1	Resume archive playback (returns error 18 if already running). (Note: lowercase el-one)
Local Playback - Start	l1,name	Start playing the archive. The name doesn't include an extension (.mp4, .wmv or .ogg) similar to the RL command. Before playback starts any current session is stopped, and any active playback is terminated. (Note: lowercase el-one).
Local Playback - Pause	l2	Pause playback (returns error 20 if already paused).
Local Playback – Get Duration	ld	Get duration (seconds) prefixed with "-" (returns error 19 if playback is stopped). (Note: lowercase el-dee).
Local Playback – Get Filename	lf	Get archive name prefixed with "-" (if playback is not stopped, otherwise returns error 19).
Local Playback Position - Percents	lp	Get position (percents) prefixed with "-" (returns error 19 if playback is stopped). (Note lowercase el-p).
Local Playback Seek - Absolute	ls,pos[%]	Absolute seek (seconds or percents) (returns error 19 if playback is stopped). (Note: lowercase el).
Local Playback Seek - Relative	ls,[+|-]pos	Relative seek (seconds) (returns error 19 if playback is stopped). (Note: lowercase el).
Local Playback Position - Time	lt	Get position (seconds) prefixed with "-" (returns error 19 if playback is stopped). (Note: lowercase el-tee).
Identification	Id<controller>,<rev>	Identify controller and software revision (I as in Item)
IP Address	Ip	Request IP Address of unit. (Note: Uppercase eye-p).
Captured Graphics Loopback OFF	L0	Do not display captured graphics locally

Function	Input Commands	Result if Processed
Captured Graphics Loopback ON	L1	Display captured graphics locally. This impacts the CPU at the transmitting site and should not be used for peak performance.
Mute OFF	M0	Mute off. Start transmission. If the channel is changed, M0 is set as the default condition.
Mute ON	M1	Mute on. Stop transmission.
Meter OFF	m0	Audio meter off.
Meter ON	m1	Audio meter on.
Meter Position	m<nw|ne|sw|se>	Audio meter position is top-left, top-right, bottom-left, bottom-right
Measured Audio Levels	ma	Report all audio levels in dB: local left, local right, network left, network right. Max value 0.0 dB, Min value –100.0 dB
Measured Audio Local	ml	Report local audio levels in dB: local left, local right. Max value 0.0 dB, Min value –100.0 dB
Measured Audio Network	mn	Report network audio levels in dB: network left, network right. Max value 0.0 dB, Min value –100.0 dB
Output Audio Gain	O0,<num>	Set Audio Output Gain to 0-100
Overlay Graphics Off	OG0	Disable all overlay graphics
Overlay Graphics 1	OG1,n	Enable (n=1) or disable (n=0) overlay graphic 1
Overlay Graphics 2	OG2,n	Enable (n=1) or disable (n=0) overlay graphic 2
Overlay Graphics 3	OG3,n	Enable (n=1) or disable (n=0) overlay graphic 3
Overlay Graphics 4	OG4,n	Enable (n=1) or disable (n=0) overlay graphic 4
Overlay Text Erase	OS0	Sets empty text on all four text overlays
Overlay Text Update	OS[1|2|3|4],TEXT	Set TEXT text on the given text overlay. TEXT can include special parameters (%H, %M, %S etc., \n for new line and \\ for \).
Overlay Text Off	OT0	Disables all four text overlays
Overlay Text On/Off	OT[1|2|3|4],n	Enable (n=1) or disable (n=0) text overlay [1|2|3|4]
Pass Floor Control	P1 … Pn	Pass floor control to Participant n. Fails if unit is not the coordinator. P0 returns control to the coordinator (issued by coordinator only).
Program End	PE	Program end. Ends the Active Session
Participant List	PL	Report all current participants in a collaborative session
Paticipant List of NCast units	PL,N	List only participants which have NCast floor control functionality (e.g. Telepresenters).
Participant List of Viewers	PL,V	List only participants which are passive viewers (e.g. desktop players).
Processor Reboot	PR	Reboot the Telepresenter.
Power Down, Shutdown	PS	Shuts down the Telepresenter.

Function	Input Commands	Result if Processed
Power Down, Shutdown when Idle	PSI	Shuts down the Telepresenter when Idle (transcoding, uploads have been completed) for at least 1 minute.
PIP Disabled	p0	PIP is Off.
PIP Enabled	p1	PIP is On.
Quit Telnet Session	QT	Quit the Telnet session and disconnect
Record Off	R0	Recording of this session is Off.
Record On	R1	Recording of this session is On.
Record Pause	R2	Recording of this session is Paused.
Record Continue	R3	Recording of this session is Continued.
Recording Contents	RC	Get the current recording details (title, presenter, channel name, start time, duration and size).
Record Disk	RD	Report disk information. Format is “u.uu t.tt”, with used space and total space in GB.
Record Filename	RF	Report the filename of the next file to be recorded. A blank entry means the name will be generated automatically. The filename will be reported without any extension and will be prefixed with a “>” character.
Record Filename	RF,name	Set the recording filename. This command must be executed before the recording has started. The “name” should not include the “.mp4” extension.
Recording Information	RI,description	Set description information for the current recording. This command must be executed after the current recording has started.
Recorded List	RL	Report list of recorded filenames. Each filename is returned as a single line with no extension and prefixed with a “>” character. All archives are reported, even if not yet ready and still being processed.
Recorded List with Descriptions	RL,SNFD...	Report list of recorded filenames with descriptions. Each list item is prefixed with “>”. Descriptors available are S – status (one of F=Ready, P=Processing, T=Transcoding, R=Recording, C=Captured, X=Corrupted), N – filename, F - format (mpeg4, h264, vc1, aac), D – duration. Use descriptors in any order or combination. See Section 5.9 for full details on all descriptors available.
Remove Recording	RM,pattern	Remove all archive files matching “pattern”, where “*” may be used to match any filename, and “?” may be used as a wildcard to match any individual character.
Recording Presenter	RP,presenter	Set presenter information for the current recording. This command must be executed after the current recording has started.
Recording Title	RT,title	Set title information for the current recording. This command must be executed after the current recording has started.

Function	Input Commands	Result if Processed
Swap Main and PIP	SW	Swap the Main and PIP inputs
Subtitle with Duration	s<duration>,<text>	Add subtitle with duration in milliseconds
Subtitle with Auto-duration	sa,<text>	Add subtitle with auto-duration
Subtitle without Duration	sn,<text>	Add subtitle without duration
Subtitle end Duration	s	End last subtitle added with sn,<text> command
Tunnel Off/On	t<0|1>	Setup N-Way client tunnel, 0=Off, 1=On
PIP Input Composite	V1	Enables PIP Input from Composite Input
PIP Input S-Video	V2	Enables PIP Input from S-Video Input
PIP Input XGA	V3	Enables PIP Input from XGA
PIP Input DVI	V4	Enables PIP Input from DVI input
PIP Input Auto Detect	V5	Enables PIP Input using Signal Auto Detect
Window Settings (main)	W<F|L|R| NW|NE|SW|SE| A|B|C|D|E|F|G|H|I|J>	Set graphics main window to F=full-screen, L=left side, R=right side, NW=top-left corner, NE=top-right corner, SW=bottom-left corner, SE=bottom-right corner, A=Custom-1, B=Custom-2, C=Custom-3, D=Custom-4, E=Custom-5, F=Custom-6, G=Custom-7, H=Custom-8, I=Custom-9, J=Custom-10
Window Settings (main)	W,x,y,w,h	Set graphics main window to (x,y,w,h)
Window Position	W<A|B|C|D|E| F|G|H|I|J>,x,y,w,h	Set custom window (A|B|C|D|E|F|G|H|I|J) to position (x, y) and size width x height. The position and size can be given in percents (0-100 for x and y, 1-100 for w and h) or pixels (0-1600 for x, 0-1200 for y, 128-1600 for w, 128-1200 for h).
Window Settings (PIP)	w<f|l|r| nw|ne|sw|se| a|b|c|d|e|f|g|h|i|j>	Set PIP window to f=full-screen, l=left side, r=right side, nw=top-left corner, ne=top-right corner, sw=bottom-left corner, se=bottom-right corner, a=Custom-1, b=Custom-2, c=Custom-3, d=Custom-4, e=Custom-5, f=Custom-6, g=Custom-7, h=Custom-8, i=Custom-9, j=Custom-10
Window Settings (PIP)	w,x,y,w,h	Set PIP window to (x,y,w,h)
Telepresenter Status	?	Displays current status of the Telepresenter.
Command Termination Character	<lf>	Required as the terminating character of a command line.
Command Termination Character	<cr>	Discarded

Note: The Telepresenter S3 does not support the following commands: D5, D6, S8, S11, S12, S13, S14, S15, S16, S17, S18.

4.2 Command Responses

Featured below is a table defining all command responses for the Serial Interface:

Response Type	Response String	Description
Status Response, current state of the controls followed by "*"	*G3,N:1, …	Readout of the current status of the Telepresenter
Command Completion Acknowledgment by Telepresenter	+L1	Telepresenter generates a reply because of an executed command from interface; acknowledgment will be "+" followed by the command.
Error	-C101	Error is a "-" reply followed by the command. In this example, Channel 101 has been selected, which is invalid
Change in State	!	Asynchronous interrupt from the Telepresenter indicates that a change in state has occurred, and a new status command must be issued.
Command Reply	&	A reply to the RS-232 interface from the Telepresenter indicating that a command has been received
Filename prefix	>	Prefix for an archive filename response from the RF, RL or RD commands.
Participant list prefix	=	Prefix used in response to a request for a participant list
Audio level prefix	$	Prefix used in response to a request for audio levels
Identification Reply	@NCast,<rev>	Identification reply with software revision number
IP Address Reply	#a.a.a.a,m.m.m.m, g.g.g.g,d.d.d.d	IP address reply, where a.a.a.a is the IP address of the unit, m.m.m.m is the netmask, g.g.g.g is the gateway and d.d.d.d is the DNS server

4.3 Common Problems

Customers who “cannot get a response from the unit” should check the items listed below:

1. The RS-232 interface does not need to be activated. It is enabled when the system boots. The IP serial interface needs to be enabled from the telnet administrative web page. It uses a non-standard telnet port.
2. The serial interface runs at 38,400 bps and not the more common 9600 bps. The interface does not auto-sense speed, so the settings must be (38400, 8, N, 1).
3. The initial command to the unit MUST be the “Id” command (e.g. “IdSerial,002<lf>” or “IdTelnet,002,pswd<lf>”). The unit will not respond until this command is received. See Section 5.2 below for additional information.
4. If testing the link from a Windows machine using Hyperterminal, note that the Enter key does not send the required <lf> code. A Ctrl-J key will issue the required <lf> character.

5 Special Functions

5.1 Purpose

The following is an additional description on special commands from the interface.

5.2 Identification Exchange

After initial power on the serial interface will wait for an Identification command. This informs the Telepresenter's software of the type of controller (e.g. Serial, Crestron, AMX, PTZ camera, etc.) which is driving the interface and the software revision level. (Note: Uppercase eye – d as in david).

Id<controller>,<rev>

The reply will be

@NCast,<rev>

Example:

Controller:	IdSerial,002<lf>
M4:	&<lf>
M4:	@NCast,002<lf>
M4:	+IdSerial,002<lf>

For Telnet sessions the form of the command is

Id<controller>,<rev>,<password>

Example:

Controller:	IdTelnet,002,my.favorite.password

5.3 Response Codes

5.3.1 Telepresenter Status

As was described in Section 4, the Serial Interface will make requests for the current state of the Telepresenter. The "?<lf>" character requests from the Telepresenter the current status. Following this, a reply is sent via the Telepresenter indicating the command was received. In this case, the character would be "+?<lf>". An example response acknowledging a readout of the current state would be an asterisk "*" followed by something similar to the following: "E:0,G3,C3,L1,M0,I1,S1,R0,T:C,P1,V:0,N:1" The readout is translated as: Error code 0 (successful completion), Graphics Input is XGA, Channel 3, Loopback enabled, Mute off, Input active, Sending graphics, Receive off, this unit is a Coordinator, Participant number 1 has floor control, no passive Viewers, and 1 unit currently in a collaboration session.

In future revisions of this product or specification additional status information may be added. A controller must ignore unknown letter codes and skip the following data until the next "," character or end of status.

Character	Description
A<num1>:<num2>	Audio input <source>:<gain> 1=mic, 2=line, 3=USB, gain 0-100
aw:h	Aspect ratio is w:h
b<0|1>	Border for PIP window: 0=Off, 1=On
C<num>	Current channel number
cwxh	Frame capture size is w (width) x h (height) in pixels
D<0|1|2|3|4|5|6|...>	Display output 0=auto, 1=VGA, 2=SVGA, 3=XGA, 4=SXGA, 5=SXGA, 6=UXGA, …
E:<num>	The letter for the last error code is "E" followed by ":" and then an error code number. The codes in the status field are without the "-" sign. D_ERR_NO_ERROR 0 D_ERR_INTERNAL_ERROR -1 D_ERR_BAD_SYNTAX -2 D_ERR_WRONG_PARAMETERS -3 D_ERR_ALREADY_COORDINATOR_SOMEWHERE -4 D_ERR_CONFERENCE_ALREADY_EXISTS -5 D_ERR_INVALID_NCCP_ADDRESS -6 D_ERR_INVALID_CHANNEL_NR -7 D_ERR_NO_SESSION -8 D_ERR_NO_NCCP_SESSION -9 D_ERR_NOT_COORDINATOR -10 D_ERR_RECORDING_IS_RUNNING -11 D_ERR_RECORDING_IS_STOPPED -12 D_ERR_RECORDING_IS_PAUSED -13 D_ERR_PERMISSION_DENIED -14 D_ERR_NO_SPACE -15 D_ERR_DISPLAY_MODE -16 D_ERR_CONNECTION_FAILED_ANNOUNCE -17 D_ERR_PLAYER_IS_PLAYING -18 D_ERR_PLAYER_IS_STOPPED -19 D_ERR_PLAYER_IS_PAUSED -20 After successful command execution the error code is cleared. This means that after getting the current status with a "?" command, the next "?" always returns "E:0".
G<num>	Selected graphics input, 1=Composite, 2=S-video, 3=VGA, 4=DVI, 5=Auto
I<0|1>	Main graphics or video input signal state, 1 if a signal was correctly detected, 0 otherwise
J<0|1>	PIP input signal state, 1 if a signal was correctly detected, 0 otherwise
L<0|1>	Loopback state, 1 when loopback is on, 0 otherwise
I<0|1|2>	Local player status: I0=stopped, I1=running, I2=paused
M<0|1>	Mute state, 1 when muted (no stream being sent), 0 otherwise
m<0|1>	Audio meter Off or On.
m<nw|ne|sw|se>	Audio meter position: nw=top-left, ne=top-right, sw=bottom-left, se=bottom-right
N:<num>	Number of NCast units in a collaboration or streaming session
O0:<num>	Output audio level, gain is in the range 0-100
o<1|2|3|4><0|1>	Overlay graphics [1|2|3|4] is Off=0 or On=1 (enabled and visible).
P<num>	Current floor control owner. Returned only when NCast collaboration session is active.

p<0|1>	PIP state, 0=Off, 1=On (enabled)
Q<0|1>	Processing Queue Inactive=0, Active=1
R<0|1>	Receiving and displaying a graphics stream, 1 when a stream is received and displayed, 0 otherwise. Note that when mute is on then a 0 will be returned here.
r<0|1|2>	Record state: 0 if not recording, 1 if recording, 2 if paused.
S<0|1>	Sending a graphics stream, 1 when a stream is being sent, 0 otherwise.
T:<N|C|P|W|R|S|D|A>	Current role: 'N' – there is no session, 'C' – NCast collaboration session coordinator, 'P' – NCast collaboration session participant, 'W' Waiting for collaboration session, 'R' – a streaming receiver, 'S' – a streaming sender, 'D' – a streaming sender and receiver (full-duplex mode), 'A' – Announce or Automatic Unicast.
t<1|2|3|4><0|1>	Text overlay [1|2|3|4] is Off=0, On=1 (enabled and visible).
U<0|1>	Uploading Inactive=0, Active=1
V<num>	Selected PIP input, 1=Composite, 2=S-video, 3=VGA, 4=DVI, 5=Auto
V:<num>	Number of passive viewers. Returned only when a streaming session is active.
Wx:y:w:h	Main window (x,y) position and width x height
wx:y:w:h	PIP window (x,y) position and width x height

5.3.2 Error Reports

The error function is the result of a reply from an invalid selection or request. For instance, if a user selects to view Channel 101 using the Serial Interface then it will generate an error as a request for Channel 101 is invalid. Therefore, in this instance an error will be reported with a "-" followed by the function "-C101<lf>". To determine the exact cause of the error via the error code, issue a status command.

5.3.3 Asynchronous Interrupt

An Asynchronous Interrupt indicates that a change in state has occurred. This is useful for reporting status changes as a result of operations using the web interface. The character for reporting a change in state is "!<lf>". The controller should issue a new status command to update its status conditions.

5.3.4 Command Execution Acknowledgment

Once new commands have been issued from the RS-232 Interface to the Telepresenter, an acknowledgment will be generated from the Telepresenter. This reply will only be generated by the Telepresenter in the case that the command was executed successfully. The reply for acknowledging a command will be "+" followed by the command. An example of this is the following: "+G1<lf>". The readout is translated as successful execution of selection of Graphic Input Composite 1.

5.3.5 Command Reply

After commands are sent to the Telepresenter, an initial reply will be generated by the Telepresenter to indicate the command was received. This reply applies to all commands. An example character of the command is as follows: "&".

5.4 Commands for Channel Change

The following commands change the Telepresenter to the requested channel and starts an active session with the new settings:

C<num>

or

C<num>,<role>

where

<num> stands for the channel number, 1 to the number of defined channels

<role> stands for what we want to do on a channel (C D P S R)

No other characters are permitted and there should be no spaces in this command string.

5.4.1 Becoming a Coordinator

C1,C

This is the command that starts a session on Channel 1 and becomes a coordinator. If the command succeeds it means that the box is on Channel 1 and has started a confer

If command fails, one the following error conditions were encountered:

a - there already was a coordinator

b - the channel does not allow the unit to be a coordinator

c - the box is a coordinator on some other channel, and has to stop the session first.

d - an unexpected error occured.

After c the box will be a coordinator (the state will not change). After all others the box will be first (1) probing to be a coordinator, and then (2) forwarded to idle state. It will be on idle state on channel that was selected.

5.4.2 Becoming a Participant

C1,P

This command will try to establish the box as a participant in a conference. After success the box will enter on a channel as a participant, and will start a session. That means that it will be in one of these states:

a - participant

b - waiting for a conference, if there is no coordinator on that channel

If this command fails, it can be one of these reasons:

a - the box is a coordinator on some channel, and has to stop it first

b - an unexpected error occurred.

After a, the box will still be a coordinator on the channel. After b, the box will be most probably be in the Idle State.

5.4.3 Becoming a Streaming Receiver

C1,R

Upon successful completion of the command we will be receiving data from Channel 1.

5.4.4 Becoming a Streaming Sender

C1,S

Upon successful completion of the command we will be sending data on Channel 1.

5.4.5 Becoming a Streaming Duplexer

C1,D

Upon successful completion of the command we will be in full-duplex streaming mode, sending and receiving data on Channel 1.

5.4.6 Changing Channels with Default Conditions

C1

This command says: go to the channel, and activate the default channel settings. If it will be a streaming receive channel, the command is equal to C1,R. The Mute condition is set to Off by default for any channel change.

5.5 Program End, Ending a session

The command to end a Session is “PE” or “Program End”. This operation is equal to Stop Session on the web interface. The box will disengage from all multicast groups.

5.6 Power Down, Shutdown

To invoke a “Power Shutdown” use command “PS”. It ends the session, and shuts down the Telepresenter. There is an additional command “PSI” which waits until the unit is idle (transcoding and uploading is finished).

5.7 Loopback: L0, L1

The Loopback command sets the variable Loopback to TRUE or FALSE. This will cause the display to go into local loopback mode if set to TRUE. Local loopback display generation consumes CPU resources and should not be used if absolute peak performance is required.

5.8 PL and P1, P2, P3, ...

The PL command request a participant list. The list will be returned in the following format:

=1,C,S,Unit1 Id

=2,P,R,Unit2 Id

=3,P,R,Unit3 Id

=4,P,R,Unit4 Id

=5,P,R,Unit5 Id

As mentioned earlier, P3 will switch to Unit3, and when Unit 3 will quit, the PL command will return

=1,C,S,Unit1 Id

=2,P,R,Unit2 Id

=3,P,R,Unit4 Id

=4,P,R,Unit5 Id

and then if 3 will enter the channel again as a participant, it will return:

=1,C,S,Unit1 Id

=2,P,R,Unit2 Id

=3,P,R,Unit4 Id

=4,P,R,Unit5 Id

=5,P,R,Unit3 Id

For a streaming session the list will be returned in the following format:

=1,N,S,Unit1 Id

=2,N,R,Unit2 Id

=3,V,R,IP address of viewer 3

=4,V,R,Host Name (DNS translated IP address of viewer 4)

The PL,N command will return:

=1,N,S,Unit1 Id

=2,N,R,Unit2 Id

The PL,V command will return:

=3,V,R,IP address of viewer 3

=4,V,R,Host Name (DNS translated IP address of viewer 4)

Character	Description
C	An NCast coordinator unit. The floor control can be passed to it. Use the “P0” command to automatically pass floor control to it without knowing its number in the participant list.
P	An NCast participant unit. The floor control can be passed to it.
N	An NCast unit in a streaming session. An attempt to pass floor control to this viewer will return an error.
V	A passive viewer which has no NCast floor control implemented. Desktop players connected to a session are marked with this character. An attempt to pass floor control to this viewer will fail. During a collaboration session the Telepresenter view web page is disabled so passive viewers won’t be returned in the PL response. This may change in future revisions of the software.
S	A unit is sending a graphics stream. Two units might have the “S” character which means that full duplex graphics is active.
R	A unit is receiving a graphics stream. It is not assured that a unit is displaying a stream because of its local settings.
D	A unit is receiving and sending a graphics stream. It might happen that two units will have a “D” character assigned, which means that a full-duplex graphics stream is active.

If the controller interface receives a ”!" character (indicating a change of state) it then must check if the N or V value has changed (number of NCast units or desktop viewers). If there has been a change of N or V it has to download the current participants table again.

Use “P0” to switch control back to the coordinator and regain floor control at the coordinator’s site. The P0 command can only be issued from the coordinator (the coordinator unconditionally takes control back).

5.9 Recording Commands

There are a group of commands which control or affect the recording process and the creation of archive files:

R0 Stop recording

R1 Start recording

R2 Pause recording

R3 Continue recording

The Channel must be enabled for manual or automatic recording for these commands to have effect.

The RL command returns a list of all archive files, even those not in the Ready state:

RL

&

>20051110-045140-001

>20051110-045202-001

>20051110-045205-001

>20051110-045208-001

>20051110-045210-001

+RL

The “>” is a command response prefix and is not part of the filename.

An alternate form of the RL command uses descriptors (S, N, F, D, ...):

RL,SNF

will return:

>F,20090305-150858-022.wmv,wmv

while:

RL,ND

will return:

>20090305-150858-022.wmv,00:30:00

where the following descriptors are available:

- **S – the current status of the file:**
  F = Finished or Ready
  P = Processing (multiplexing of video and audio)
  T = Transcoding to another format or resolution
  C = Captured and waiting for processing
  R = Recording and not yet complete
  X = Corrupted, some error occurred with the file
- **N – filename with extension**
- **F – format (mp4, wmv, ogg)**
- **D – duration as HH:MM:SS**
- **T – title**
- **P – presenter**
- **C – channel name**
- **R – start time as YYYY-MM-DD HH:MM:SS**
- **L – size in MB**
- **A – audio format (aac, wma, vorbis)**
- **V – video format (mpeg4, h264, vc1, theora)**
- **s – frame size as WxH**
- **a – aspect ratio as W:H**
- **b – bitrate in kbps**
- **f – framerate**

The filename of the next archive file to be created may be set with the RF command. This command must be issued prior to the start of recording:

RF,filename

&

+RF,filename

Do not include the .mp4 extension. An RF command with no argument returns the current filename. A blank filename implies that the filename will be generated automatically:

RF

&

>filename

+RF

To set title information for the presentation detail file, use the RT command after recording has commenced:

RT,title info about the current presentation

To setup information about the current presenter or speaker associated with the recording, use the RP command after recording has commenced:

RP,John Smith – CEO

Set description information for the current recording. This command must be executed after the current recording has started:

RI,description

Recording information may be retrieved with the RC command:

RC

&

>Discussion on Telepresenter Features

>NCast CEO

>Streaming 1

>2006-04-10 07:15:51

>00:00:05

>0.04

+RC

Starting from the top it lists:

- title
- presenter
- channel name
- start time as YYYY-MM-DD HH:MM:SS
- duration as HH:MM:SS
- size in MB

Recordings may be removed with the RM command:

RM,*	Removes all archive files
RM,test*	Removes all archive files starting with “test”

RM,20051111* Removes all archives created on 11 Nov 2005.

Disk space available for recording will be reported with the RD command:

RD

&

>0.00 111.78

+RD

Where the first number is space used in GB, and the second number is space available in GB.

5.10 Audio Level Commands

There are three commands which allow remote readout of the audio meter levels. This will permit easy monitoring of audio activity on the Telepresenter from a centralized control point or network operations room.

ma – Report all audio levels (local left, local right, net left, net right):

ma

&

$-26.6,-36.0,-100.0,-100.0

+ma

ml – Report local audio levels (left, right):

ml

&

$-26.6,-35.9

+ml

mn – Report network audio levels (left, right):

mn

&

$-100.0,-100.0

+mn

All levels are reported in a dB scale (maximum value is 0.0 dB, minimum is –100.0 dB). Power is calculated separately for the local and net streams and for the left and right channels. The power is calculated over 1024 samples regardless of the selected audio format.

The calculation used is as follows:

Denote the samples as:

s(1), s(2), s(3), ..., s(1024)

where

-32768 <= s(i) <= 32767

First we calculate the absolute maximum of samples:

ams = max(abs(s(1)), abs(s(2)), ..., abs(s(1024)))

Then final power in a dB scale:

if ams == 0 (this can happen for net stream)

```
        power = -100.0
    else
        power = 10 * log10((ams*ams) / (32768 * 32768))
```

In last line the smallest value we can get is:

10 * log10(1/(32768*32768)) = -90.308998

and largest value is:

10 * log10((32768*32768)/(32768*32768)) = 0.000

5.11 Display Output Modes

The list of display modes has been extended to include many different types of CRT and LCD displays. Five different timings standards are available:

- DMT - VESA Display Monitor Timing
- GTF - VESA Generalized Timing Formula
- CVT - VESA Coordinated Video Timings
- CVT-RB - VESA Coordinated Video Timings with Reduced Blanking
- 861B – CEA/EIA-861B

5.12 Local Display Playback of Archives

Once recorded, archives may be played back on the local display. The following commands allow the serial interface to be used as a playback controller (Note: lowercase el):

l0 - stop archive playback, returns error 19 if already stopped

l1 - resume archive playback, returns error 18 if already running

l1,name - start playing archive, name doesn't include extension (.mp4, .wmv or .ogg). Before playback starts we stop the session and previous playback (if active)

l2 - pause playback, returns error 20 if already paused

lf - get archive name prefixed with "-" (if playback is not stopped, otherwise returns error 19)

ld - get duration (secs) prefixed with "-", returns error 19 if playback is stopped

lp - get position (percents) prefixed with "-", returns error 19 if playback is stopped

lt - get position (secs) prefixed with "-", returns error 19 if playback is stopped

ls,[+|-]pos - relative seek (secs), returns error 19 if playback is stopped

ls,pos[%] - absolute seek (secs or percents), returns error 19 if playback is stopped

5.13 Subtitle Commands

There does not appear to be a single, widely adopted standard for recording timed-event information for use either in sub-titles or chaptering. Some client players use various forms of .txt files and others use .xml files.

The Telepresenter implementation has adopted one format which is simple to use and has some existing support in the open-source community and works with several players.

The format is an ".srt" text file type which you may read about here:

http://en.wikipedia.org/wiki/SubRip

New serial commands have been added to receive text information and automatically create ".srt" files during a recording.

These files will be:

- Available on the Archives page for download
- Available for download via our http interface
- Uploaded automatically via the FTP upload service

Upon receipt of this file a program can easily extract the timing information of interest and create a web page with playback start points of interest. For example, Quicktime uses the "STARTTIME" parameter:

http://www.apple.com/quicktime/tutorials/embed2.html

<embed src="sample.mp4" width="320" height="240" starttime="00:15:22.5">

There are four serial commands which deal with subtitles:

sa,text - adds a subtitle with auto-duration, where the subtitle duration is based on the number of characters in “text”, for example:

sa,This is test subtitle

sduration,text - adds a subtitle with given duration in msec. The duration must be > 0, for example:

s250,This is test subtitle

sn,text - adds a subtitle without duration. The subtitle will end when the next subtitle command is passed or recording is stopped, for example:

sn,This is first subtitle

s - ends last subtitle

All commands work in real-time, as the subtitles are synchronized to what is seen on the screen.

For chaptering we suggest use of the sn,text command like this:

sn,Chapter 1

sn,Chapter 2

sn,Chapter 3

...

After recording is finished and there is at least one subtitle, an .srt subtitle file is created which can be downloaded from the web-page or HTTP interface. The .srt format is very simple, this example has two subtitles:

```
1
00:00:20,000 --> 00:00:24,400
Subtitle 1

2
```

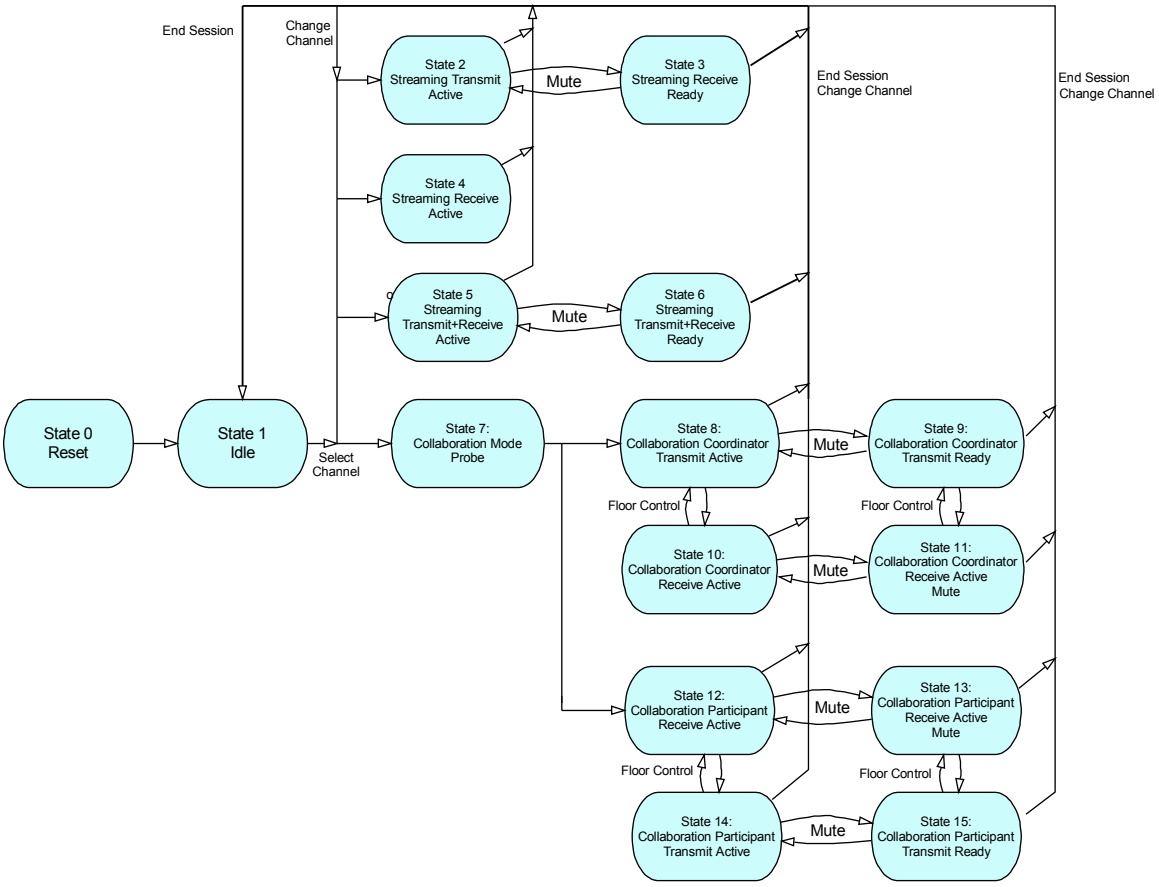
```
00:00:24,600 --> 00:00:27,800
Subtitle 2
```

Here is an example using the "sn" command only to record the timing of overlay pushbuttons using small Python application:

```
1
00:00:07,281 --> 00:00:07,281
Overlay 1 is 0

2
00:00:08,776 --> 00:00:08,776
Overlay 1 is 1

3
00:00:10,769 --> 00:00:10,769
Overlay 2 is 1

4
00:00:12,745 --> 00:00:12,745
Overlay 2 is 0

5
00:00:14,738 --> 00:00:14,738
Overlay 3 is 1
```

6 Operation Description

6.1 State Diagram



6.2 State Description

State Name	State Description
State 0: Reset	Channel settings are stored in session.conf file Power is off
State 1: Idle	Channel is selected Multicast addresses are not activated. Transmitter is Off Air NCast Logo is displayed on the Screen if no active input is detected or if the Loopback feature is disabled If Signal Detected, Display is in Loopback if enabled.
State 2: Streaming Transmit Active	Channel setup for streaming Transmitter is On Air Message “Check Input Signal” is displayed if no active input is detected. If Signal detected, Display is in loopback if enabled otherwise “Sending Graphics Stream” message is displayed.
State 3: Streaming Transmit Ready	Channel setup for streaming Transmitter is Off Air (muted) If Signal detected, Display is in loopback if enabled otherwise “Sending Graphics Stream” message is displayed. Message “Check Input Signal” is displayed if no active input is detected.
State 4: Streaming Receive Active	Channel setup for streaming Transmitter is Off Air Display shows received packets or message “Waiting for a graphics stream” is displayed.
State 5:Streaming Transmit+Receive Active	Channel setup for streaming Transmitter is On Air Display shows received packets or the message “Waiting for a graphics stream” is displayed.
State 6:Streaming Transmit+Receive Ready	Channel setup for streaming Transmitter is Off Air (muted) Display shows received packets or the message “Waiting for a graphics stream” is displayed.
State 7: Collaboration Mode Probe	Channel setup for NCCP Channel activity is being probed Transmitter is Off Air

State Name	State Description
	NCast Logo is displayed on the Screen if no active input is detected or Loopback feature is disabled If Signal Detected, Display is in Loopback if enabled.
State 8: Collaboration Coordinator Transmit Active	Channel setup for NCCP Unit is Coordinator Transmitter is On Air Coordinator has the floor Message "Check Input Signal" is displayed if no active input is detected. If Signal detected, Display is in loopback if enabled otherwise "Sending Graphics Stream" message is displayed.
State 9: Collaboration Coordinator Transmit Ready	Channel setup for NCCP Unit is Coordinator Transmitter is Off Air (muted) Coordinator has the floor If Signal detected, Display is in loopback if enabled otherwise "Sending Graphics Stream" message is displayed. Message "Check Input Signal" is displayed if no active input is detected.
State 10: Collaboration Coordinator Receive Active	Channel setup for NCCP Unit is Coordinator Transmitter is Off Air A Participant has the floor Display shows received packets or message "Waiting for a graphics stream" is displayed.
State 11: Collaboration Coordinator Receive Active (Mute On)	Channel setup for NCCP Unit is Coordinator Transmitter is Off Air (muted) A Participant has the floor Display shows received packets or message "Waiting for a graphics stream" is displayed.
State 12: Collaboration Participant Receive Active	Channel setup for NCCP Unit is a Participant Transmitter is Off Air Coordinator or some Participant has the floor Display shows received packets or message "Waiting for a graphics stream" is displayed.
State 13: Collaboration Participant	Channel setup for NCCP Unit is a Participant

State Name	State Description
Receive Active (Mute ON)	Transmitter is Off Air (muted) Coordinator or some Participant has the floor Display shows received packets or message "Waiting for a graphics stream" is displayed.
State 14: Collaboration Participant Transmit Active	Channel setup for NCCP Unit is a Participant Transmitter is On Air Participant has the floor Message "Check Input Signal" is displayed if no active input is detected. If Signal detected, Display is in loopback if enabled otherwise "Sending Graphics Stream" message is displayed.
State 15: Collaboration Participant Transmit Ready	Channel setup for NCCP Unit is a Participant Transmitter is Off Air (muted) Participant has the floor If Signal detected, Display is in loopback if enabled otherwise "Sending Graphics Stream" message is displayed. Message "Check Input Signal" is displayed if no active input is detected.

7 Sample Code

7.1 Python Program Example

The following code sample in Python (see http://www.python.org) provides an example of procedures which address the serial interface over the Telnet IP link (check the website for the latest versions):

```
#! /usr/bin/python

# Telnet Serial Interface Program

import sys
import socket
import time
import string
import random

class Telnet:
    def __init__(self, site, port, password):
        self.site = site
        self.port = port
        self.password = password
        self.debug = False
        self.errors = 0
        self.Connected = False
        self.tms = None
        self.tmsmodule = "tmstelnet"
        self.Telnet_ResetStatus()

    def Telnet_ResetStatus(self):
        self.Status = ""
        self.Alert = "0"
        self.AspectRatio = "?"
        self.AudioDevice = "?"
        self.AudioInput = "0"
        self.AudioLocalLeft = "-100.0"
        self.AudioLocalRight = "-100.0"
        self.AudioMeter = "0"
        self.AudioNetworkLeft = "-100.0"
        self.AudioNetworkRight = "-100.0"
        self.AudioOutput = "0"
        self.CaptureSize = "?"
        self.Channel = "?"
        self.Display = "?"
        self.Error = "0"
        self.GraphicsInput = "?"
        self.InputDetectMain = "0"
        self.InputDetectPIP = "0"
        self.Loopback = "0"
        self.Mute = "0"
        self.Noise = "?"
        self.NumberUnits = "0"
        self.Overlay_1 = "?"
        self.Overlay_2 = "?"
        self.Overlay_3 = "?"
```

```
        self.Overlay_4 = "?"
        self.Party = "0"
        self.PIPState = "0"
        self.ReceivingStream = "0"
        self.Recording = "0"
        self.SendingStream = "0"
        self.TypeRole = "?"
        self.VideoInput = "?"
        self.Viewers = "0"
        self.WindowMain = "?"
        self.WindowPIP = "?"

    def Telnet_Open(self):
        self.Connected = False

        try:
            s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
            s.connect((self.site, self.port))
        except socket.error, msg:
            self.errors = self.errors + 1
            self.tms.log(self.tmsmodule, "TMS Telnet error for site %s: %s" %
(self.site, msg))
            return None

#        s.setsockopt(socket.SOL_SOCKET, socket.SO_RCVTIMEO, 300)

 self.errors = 0
        self.Connected = True
        return s

    def Telnet_Close(self, s):
        self.Connected = False
        try:
            s.send('QT\n')
            s.close()
        except socket.error, msg:
            return

    def Telnet_Send(self, s, text):
        try:
            s.send(text)
        except socket.error, msg:
            self.tms.log(self.tmsmodule, "TMS Telnet error for site %s: %s" %
(self.site, msg))

    def Telnet_Receive(self, s):
        response = []
        alert = False
        while True:
            try:
                data = s.recv(1500)
            except socket.error, msg:
                self.tms.log(self.tmsmodule, "TMS Telnet error for site %s: %s" %
(self.site, msg))
                return (False, response, "0")
            rd = str(data).strip("'")
            commandwasgood = False
            done = False
            for r in rd.split("\n"):
                if r == '':
```

```
                continue
            if r[0] == '&':
                continue
            if r[0] == '+':
                commandwasgood = True
                done = True
                break
            if r[0] == '-':
                response.append(rd)
                commandwasgood = False
                done = True
                break
            if r[0] == '!':
                alert = "1"
                continue
            response.append(r[1:])
        if done :
            break
    return (commandwasgood, response, alert)

def Telnet_Command(self, s, command):

    if self.Connected:
        if self.debug:
            self.tms.log(self.tmsmodule, 'TMS Command Sent %s' % command)
        self.Telnet_Send(s, command)
    else:
        return (False, [], "0")

    if self.Connected:
        tr = self.Telnet_Receive(s)
    else:
        return (False, [], "0")

    commandstatus, response, ralert = tr

    if self.debug:
        self.tms.log(self.tmsmodule, 'TMS Command Received %s' % response)
    return tr

def Telnet_Initialize(self):

    # Clear out all Status indicators
    self.Telnet_ResetStatus()

    # Open the serial link
    self.s = self.Telnet_Open()
    if self.s == None:
        return False

    # Send initial sign-on command
    self.Telnet_Command(self.s,'IdTelnet,002,' + self.password + '\n')

    return self.Connected

def Telnet_Disconnect(self):
    if self.Connected:
        # Close serial link
        self.Telnet_Close(self.s)
```

```
        # Clear out all Status indicators
        self.Telnet_ResetStatus()

    def Telnet_Status(self):

        # Check if still connected
        if self.Connected == False:
            return ""
        # Send ? command
        self.CommandResult, self.Status, self.Alert =
self.Telnet_Command(self.s,'?\n')
        if not self.CommandResult:
            return ""
        if self.Status == []:
            return ""
        status_string = self.Status[0]
        for code in status_string[1:].split(","):
            if code[0] == "A":
                if code[1] == "1":
                    self.AudioDevice = "Microphone"
                if code[1] == "2":
                    self.AudioDevice = "Line"
                if code[1] == "3":
                    self.AudioDevice = "USB"
                self.AudioInput = code[3:]
            if code[0] == "a":
                self.AspectRatio = code[1:]
            if code[0] == "C":
                self.Channel = code[1:]
            if code[0] == "c":
                self.CaptureSize = code[1:]
            if code[0] == "D":
                if code[1] == "0":
                    self.Display = "Auto"
                if code[1] == "1":
                    self.Display = "VGA"
                if code[1] == "2":
                    self.Display = "SVGA"
                if code[1] == "3":
                    self.Display = "XGA"
                if code[1] == "4":
                    self.Display = "SXGA"
                if code[1] == "5":
                    self.Display = "UXGA"
                if code[1] == "6":
                    self.Display = "WUXGA"
            if code[0] == "E":
                self.Error = code[2:]
            if code[0] == "G":
                if code[1] == "1":
                    self.GraphicsInput = "Composite"
                if code[1] == "2":
                    self.GraphicsInput = "S-video"
                if code[1] == "3":
                    self.GraphicsInput = "XGA"
                if code[1] == "4":
                    self.GraphicsInput = "DVI"
                if code[1] == "5":
                    self.GraphicsInput = "Auto"
            if code[0] == "I":
```

```
        self.InputDetectMain = code[1]
    if code[0] == "J":
        self.InputDetectPIP = code[1]
    if code[0] == "L":
        self.Loopback = code[1]
    if code[0] == "M":
        self.Mute = code[1]
    if code[0] == "m":
        self.AudioMeter = code[1]
    if code[0] == "N":
        self.NumberUnits = code[1:]
    if code[0] == "n":
        if code[1] == "0":
            self.Noise = "Off"
        if code[1] == "1":
            self.Noise = "Low"
        if code[1] == "2":
            self.Noise = "Medium"
        if code[1] == "3":
            self.Noise = "High"
    if code[0] == "O":
        self.AudioOutput = code[3:]
    if code[0] == "o":
        if code[1] == "1":
            self.Overlay_1 = code[2]
        if code[1] == "2":
            self.Overlay_2 = code[2]
        if code[1] == "3":
            self.Overlay_3 = code[2]
        if code[1] == "4":
            self.Overlay_4 = code[2]
    if code[0] == "P":
        self.Party = code[1:]
    if code[0] == "p":
        self.PIPState = code[1]
    if code[0] == "R":
        self.ReceivingStream = code[1]
    if code[0] == "r":
        self.Recording = code[1]
    if code[0] == "S":
        self.SendingStream = code[1]
    if code[0] == "T":
        if code[2] == "N":
            self.TypeRole = "No Session"
        if code[2] == "C":
            self.TypeRole = "Coordinator"
        if code[2] == "P":
            self.TypeRole = "Participant"
        if code[2] == "W":
            self.TypeRole = "Waiting"
        if code[2] == "R":
            self.TypeRole = "Receiver"
        if code[2] == "S":
            self.TypeRole = "Sender"
        if code[2] == "D":
            self.TypeRole = "Full Duplex"
        if code[2] == "A":
            self.TypeRole = "Announce"
    if code[0] == "V" and code[1] == ":" :
        self.Viewers = code[2:]
```

```
            if code[0] == "V" and code[1] != ":" :
                if code[1] == "1":
                    self.VideoInput = "Composite"
                if code[1] == "2":
                    self.VideoInput = "S-video"
                if code[1] == "3":
                    self.VideoInput = "XGA"
                if code[1] == "4":
                    self.VideoInput = "DVI"
                if code[1] == "5":
                    self.VideoInput = "Auto"
            if code[0] == "W":
                self.WindowMain = code[1:]
            if code[0] == "w":
                self.WindowPIP = code[1:]

        return status_string

    def Telnet_SessionStart(self, channel):

        # Check if still connected
        if self.Connected == False:
            return False
        # Start Session on Channel
        command_status, response, alert = self.Telnet_Command(self.s, 'C%s\n' % chan-
nel)
        return command_status

    def Telnet_SessionStop(self):

        # Check if still connected
        if self.Connected == False:
            return False
        # End current Session
        command_status, response, alert = self.Telnet_Command(self.s,'PE\n')
        return command_status

    def Telnet_Record(self, option):

        # Check if still connected
        if self.Connected == False:
            return False
        # Execute recording option
        command_status, response, alert = self.Telnet_Command(self.s, 'R%s\n' % op-
tion)
        return command_status

    def Telnet_GraphicsInput(self, device):

        # Check if still connected
        if self.Connected == False:
            return False
        # Select an input device
        command_status, response, alert = self.Telnet_Command(self.s,'G%s\n' %
device)
        return command_status

    def Telnet_VideoInput(self, device):

        # Check if still connected
```

```
        if self.Connected == False:
            return False
        # Select an input device
        command_status, response, alert = self.Telnet_Command(self.s,'V%s\n' %
device)
        return command_status

    def Telnet_PIP(self, state):

        # Check if still connected
        if self.Connected == False:
            return False
        # Select PIP state
        command_status, response, alert = self.Telnet_Command(self.s,'p%s\n' % state)
        return command_status

    def Telnet_Swap(self):

        # Check if still connected
        if self.Connected == False:
            return False
        # Swap PIP
        command_status, response, alert = self.Telnet_Command(self.s,'SW\n')
        return command_status

    def Telnet_AudioInput(self, device):

        # Check if still connected
        if self.Connected == False:
            return False
        # Select an input device
        if device == "Line":
            code = "2"
        else:
            code = "1"
        command_status, response, alert = self.Telnet_Command(self.s,'A%s\n' % code)
        return command_status

    def Telnet_AudioInputGain(self, device, gain):

        # Check if still connected
        if self.Connected == False:
            return False
        # Select an input device, gain
        if device == "Line":
            code = "2"
        else:
            code = "1"
        command_status, response, alert = self.Telnet_Command(self.s,'A%s,%s\n' %
(code, gain))
        return command_status

    def Telnet_AudioOutputGain(self, gain):

        # Check if still connected
        if self.Connected == False:
            return False
        # Set output gain
        command_status, response, alert = self.Telnet_Command(self.s,'O0,%s\n' %
gain)
```

```
            return command_status

        def Telnet_AudioMeter(self, state):

            # Check if still connected
            if self.Connected == False:
                return False
            # Set meter on/off
            command_status, response, alert = self.Telnet_Command(self.s,'m%s\n' % state)
            return command_status

        def Telnet_AudioLevels(self):

            # Check if still connected
            if self.Connected == False:
                return False
            # Request update on audio level measurements
            command_status, response, alert = self.Telnet_Command(self.s,'ma\n')
            levelvalues = response[0]
            levels = levelvalues.split(",")
            self.AudioLocalLeft = levels[0]
            self.AudioLocalRight = levels[1]
            self.AudioNetworkLeft = levels[2]
            self.AudioNetworkRight = levels[3]
            return command_status

        def Telnet_Overlay_1(self, state):

            # Check if still connected
            if self.Connected == False:
                return False
            # Set overlay on/off
            command_status, response, alert = self.Telnet_Command(self.s,'OG1,%s\n' %
    state)
            return command_status

        def Telnet_Overlay_2(self, state):

            # Check if still connected
            if self.Connected == False:
                return False
            # Set overlay on/off
            command_status, response, alert = self.Telnet_Command(self.s,'OG2,%s\n' %
    state)
            return command_status

        def Telnet_Overlay_3(self, state):

            # Check if still connected
            if self.Connected == False:
                return False
            # Set overlay on/off
            command_status, response, alert = self.Telnet_Command(self.s,'OG3,%s\n' %
    state)
            return command_status

        def Telnet_Overlay_4(self, state):

            # Check if still connected
            if self.Connected == False:
```

```
            return False
        # Set overlay on/off
        command_status, response, alert = self.Telnet_Command(self.s,'OG4,%s\n' %
state)
        return command_status


    def Telnet_Debug(self,debug):
        self.debug = debug
```

8 Revision History

Revision 3.7 – Updates for Rev. 5.2 features. Introduction of “l” commands for local playback (on the attached display only) of archive files. Additional error codes to support these operations. Update to the RL command to support reporting of archives. Status codes for the local player.

Revision 3.6 – Updates for Rev. 5.1 features. Remove all “S” commands to set capture (frame) size. Remove all “a” commands to set aspect ratio. Added “W” and “w” custom window definition commands. New option for USB audio input, USB status. New status formats for aspect ratio, frame size, main and PIP window positions and size.

Revision 3.5 – Updates for Rev. 5.0 features. Additional frame, aspect, main window and pip window variables and status. Text overlay commands. Additional shutdown commands and status. Recording list now includes descriptors.

Revision 3.4 – New commands added for PIP border (b0, b1) and audio meter position (mnw, mne, msw, mse). New command “RI” for recording description. Modified commands D1-D6 and new Display and Display Aspect Ratio commands. Added commands for real-time sub-title creation and/or chapter and event recording.

Revision 3.3 – New commands added to support remote audio level monitoring: ma, ml, mn. New status reporting codes: a, W, w, o. Disk space response now is prefixed. Overscan status removed. 3.3.1 Python example updated.

Revision 3.2 – New commands added to support the M3 Series 2: D6, S17, S18 and status codes c17, c18, and D6. The S0 command is no longer supported. Clarification of limits for the S3. Syntax fix for a, S and W commands.

Revision 3.1 – New commands added to support overlay graphics: OG0, OG1-OG4. Updated example code which now supports the Telepresenter M3. Split commands and status into two tables.

Revision 3.0 – Documentation upgraded to reflect the first release of the Telepresenter M3. Status line additions include “J”, “V”, “p”. Status line changes affect “c”, “D”, “G”, and “I”. Removal of “n”, “o”. Commands added or modified include “a”, “Gn”, “p”, “S”. “SW”, “V”, “W”, “w”. Commands changed are “D”, “S”. Commands removed are “N”, “o”.

Revision 2.9 – Addition of the “a”, “RC”, “t” commands. Changes to the “c”, “D”, “S” commands. Documentation fixes for the “RF” and “RL” commands.

Revision 2.8 – New commands added as of software Revision 2.8. See documentation for the “D”, “m”, “N”, “o”, “PR”, “RD”, “RF”, “RL”, “RT”, “RP”, “RF”, “RM” and “S” commands. Also, new status line information is available. The Telnet interface port is no longer fixed, and may be setup on the Telnet configuration page. Multiple Telnet connections are now permitted.

Revision 2.7 – Minor editorial corrections.

Revision 2.6 – The document has been renamed to reflect additional Telepresenter models. Picture of the unit was added. No changes to the protocols or commands.

Revision 2.5 – The ID command may now use the generic term “Serial” instead of a manufacturer specific term such as “Crestron”.

Revision 2.4 – Added documentation of the “Ip” command for IP address reporting. Added graphics selection command G4 for DVI input use. Note: Auto-detect of graphics input is now G5. Status command for G4 and G5 changes as well.

Revision 2.3 – New commands for audio input selection, audio gain, audio output gain and audio status reporting. See the “A1” and “A2” commands for audio selection and gain settings and status reports. Use the “O” command for output gain settings and status reports.

Revision 2.2 – Added commands for control of archiving. Start and stop of recording, pause, continue, and status of archive state are now implemented. See the “R” commands for control and the “r” command for status.