NetLinx Module Interface Specification

for

VCON HD2000 Audio/Video Conferencing Controller

AMX • 3000 Research Drive • Richardson, TX • 75082
469.624.8000 • 800.222.0193 • 469.624.7153 (fax)

TABLE OF CONTENTS

LIST OF TABLES

Revision History

Date	Initials	Comments
11/04/2004	CN	Initial Release v1.0

AMX • 3000 Research Drive • Richardson, TX • 75082
469.624.8000 • 800.222.0193 • 469.624.7153 (fax)

Introduction

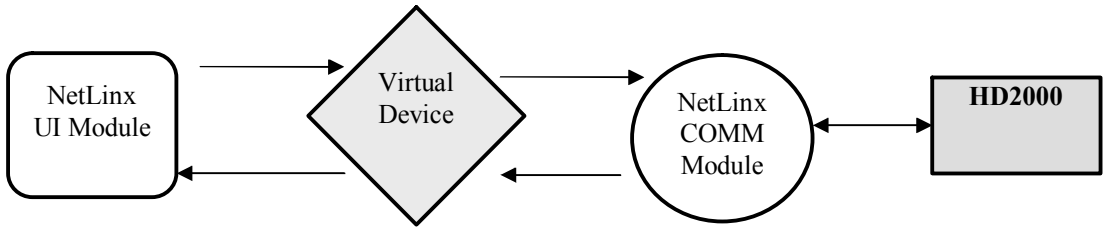
This is a reference manual to describe the interface provided between an AMX NetLinx system and a VCON HD2000. The HD2000 supports an IP protocol. The interface was tested using VCON HD2000 software version **0202.M05.D28.H12,** flash version **10030417**, and firmware version **19003500**.

Overview

The COMM module translates between the standard interface described below and the HD2000 IP protocol. It parses the buffer for responses from the HD2000, sends strings to control the HD2000, and receives commands from the UI module or telnet sessions.

A User Interface (UI) module is also provided. This module uses the standard interface described below and parses the string responses for feedback.

The following diagram gives a graphical view of the interface between the interface code and the NetLinx module.



Some functionality in the device interface may not be implemented in the API interface. In cases where device functions are desired but not API-supported, the PASSTHRU command may be used to send any and all device-protocol commands to the device. See the PASSTHRU command and the Adding Functions to Modules for more information.

A sample UI module and a touch panel file are provided in the module package. These are not intended to cover every possible application, but can be expanded as needed by a dealer to meet the requirements of a particular installation.

Implementation

To interface to the AMX **Vcon_HD_COMM** module, the programmer must perform the following steps:

1. Define the device ID for the **HD2000** that will be controlled.
2. Define the virtual device ID that the **Vcon_HD_COMM** COMM module will use to communicate with the main program and User Interface. NetLinx virtual devices start with device number 33001.
3. If a touch panel interface is desired, a touch panel file **VCON_HD2000.TP4** and module **Vcon_HD_UI.axs** have been created for testing.

4. The NetLinx **Vcon_HD_COMM** module must be included in the program with a DEFINE_MODULE command. This command starts execution of the module and passes in the following key information: the device ID of the **HD2000** to be controlled, and the virtual device ID for communicating to the main program.

An example of how to do this is shown below.

```
DEFINE_DEVICE
   dvDevice            = 0:5:0 // The HD2000 connected to the NetLinx IP port
   dvTouchPanel        = 128:1:0       // The touch panel used for output

    vdvDevice          = 33001:1:0    // The virtual device use for communication between the
                                      // Comm module interface and User_Interface (UI) module interface

DEFINE_VARIABLE
//Define arrays of button channels used on your own touch panel
integer nTP_BUTTONS[]={1,2,3,4,5,6,7}
integer nTP_TEXT[]={1,2,3,4,5,6,7}

DEFINE_START      // Place define_module calls to the very end of the define_start section.
// Comm module
DEFINE_MODULE 'Vcon_HD_COMM' Comm1(vdvDevice, dvDevice)
// Touch panel module
DEFINE_MODULE 'Vcon_HD_UI' TouchPanelA(vdvDevice, dvTouchPanel, nTP_BUTTONS,
nTP_TEXT)
```

In order to establish the initial connection, the user must first login to the specific HD2000 by providing the correct IP address to the communication module (see LOGIN= command). Upon initialization, the AMX Comm module will communicate with the **HD2000** and information will be exchanged.

Command Interface

The interface code will control the audio/video conferencing controller via command events (NetLinx command *send_command)*. These commands will be sent to the module to affect control. Below are the commands supported.

Note: an '*' indicates an extension to the standard API.

Command	Description
AUDSOURCE=<value>	Select an audio input. <value>: 1 = Table-Top Mic 2 = Line Level (aux audio source) AUDSOURCE=1 **Note:** See device notes.
AUDSOURCE?	Query for the current audio input source. AUDSOURCE?
*AUTOANSWER=<mode>	Set the auto-answer mode for point-to-point calls. <mode>: 0 = Manual answer 1 = Auto answer 2 = Auto answer and Mute mic 3 = Auto reject AUTOANSWER=1
*AUTOANSWER?	Get the current auto-answer mode for point-to-point calls. AUTOANSWER?
*AUTOANSWER_MULTIPOINT=<mode>	Set the auto-answer mode for multipoint calls. <mode>: 0 = Manual answer 1 = Auto answer 2 = Auto reject AUTOANSWER_MULTIPOINT=1
*AUTOANSWER_MULTIPOINT?	Get the auto-answer mode for multipoint calls. AUTOANSWER_MULTIPOINT?
*BAN_OUTGOING=<mode>	Set the outgoing call block/ban feature. <mode>: 0 = off (outgoing calls allowed) 1 = on (outgoing calls blocked) BAN_OUTGOING=0
*BAN_OUTGOING?	Get the outgoing call block/ban state. BAN_OUTGOING?

*CALL_ACTIVE_ID?	Get the call IDs of all active calls. CALL_ACTIVE_ID?
*CALL_BITRATE?[<callID>]	Get the call's bitrate. <callID>: 0..n = ID of the call (optional). If omitted, call ID 0 is used by default. CALL_BITRATE?
*CALL_DIRECTION?[<callID>]	Get the call's direction. <callID>: 0..n = ID of the call (optional). If omitted, call ID 0 is used by default. CALL_DIRECTION?
*CALL_DURATION?[<callID>]	Get the call's duration. <callID>: 0..n = ID of the call (optional). If omitted, call ID 0 is used by default. CALL_DURATION?
*CALL_MAX?	Get the maximum number of concurrent calls supported by the VCON CALL_MAX?
CAMRECALL=<end> :<camera> :<value>	Recall a camera preset. This causes the camera to go to the saved position settings of the preset. <end> : N = near-end camera F = far-end camera <camera>: 1..4 = camera to affect (1 = Main Camera) <value> : 1..15 = preset number to recall CAMRECALL=F:1:4
CAMSAVE=<end> :<camera> :<value>	Save a camera preset. This causes the camera to store the position settings and go to those settings upon request. The zoom setting of the camera is not stored. <end> : N = near-end camera F = far-end camera <camera>: 1..4 = camera to affect (1 = Main Camera) <value> : 1..15 = preset number to save CAMSAVE=N:1:1
DEBUG=<value>	Enables or disables the debug feature. <value>: 0 = off 1 = on DEBUG=1
DEBUG?	Request the state of the debug feature. DEBUG?

*DHCP=<mode>	Set the DHCP mode of the VCON unit. <mode>: 0 = off (use static IP) 1 = on (use dynamic IP) DHCP=0
*DHCP?	Get the DHCP mode of the VCON unit. DHCP?
DIAL=<address>[:<bitrate>]	Dial an address. The address can be an IP address, DNS name, H.323-ID, or E.164. Valid address characters are all numbers, all letters, dot, underscore, or dash. <address>: 0..9, a..z, A..Z, "_", "-", "." <bitrate>: optional call bitrate. If not used the VCON will use 384000. DIAL=10.0.0.6
*DIALDTMF=<callID>:<tone>	Simulates a touch-tone sound. Must be in a call for this command to execute. <callID>: 0..n = ID of the call to send the tones to <tone>: 0..9, *, # DIALDTMF=0:*
*FECC=<mode>	Set the far-end camera control. Enable/Disable remote party to control the local camera. <mode>: 0 = off (disable) 1 = on (enable) FECC=1
*FECC?	Get the far-end camera control status. FECC?
HOOK=<callID>:<state>	Change the hook-switch state of a call. <callID> : 0..n = call ID <state> : ON = put phone on-hook (disconnect a call) OFF = take phone off-hook (accept a call) REJECT = reject an incoming call HOOK=0:OFF
*IP_ADDRESS=<value>	Set the VCON IP address. Assumes the DHCP is turned off in the VCON. <value>: IP address in dot format IP_ADDRESS=10.0.0.6

*IP_ADDRESS?	Get the VCON IP address. Assumes there is an existing connection between AMX and the VCON already. If no connection between AMX and VCON exists, this command will fail. IP_ADDRESSS?
*KEY=<value>	This command will emulate the VCON HD2000 remote control keys. <value>: 1 = Zero key, 2 = One key, 3 = Two key, 4 = Three key, 5 = Four key, 6 = Five key, 7 = Six key, 8 = Seven key, 9 = Eight key, 10 = Nine key, 11 = Asterisk key, 12 = Pound key, 13 = Clear key, 14 = Display key, 15 = PIP key, 16 = Dial key, 17 = Hang-up key, 18 = Menu key, 19 = Multipoint key, 20 = Status key, 21 = Help key, 22 = OK key, 23 = Cancel key, 24 = Select key, 25 = Up key, 26 = Left key, 27 = Down key, 28 = Right key, 29 = Zoom In key, 30 = Zoom Out key, 31 = Mute Mic, 32 = Mute Speaker, 33 = Volume + key, 34 = Volume - key, 35 = Red A, 36 = Green B, 37 = Blue C, 38 = Yellow D KEY=25 **Note:** The key commands do not ramp. In other words, for example, if the Up key is used to move the camera, it will move one discrete unit and stop. If ramping is desired, it is recommended that the specific API command be used instead of this KEY= command.
LINE?[<callID>]	Query for the current call status. <callID>: 0..n =call ID number. Optional parameter. If not used, a call ID of 0 will be used. LINE?
*LOGIN=<IP address>	This command will establish IP communications between AMX and the VCON. <IP address>: IP address of the VCON to connect to. LOGIN=10.0.0.6 **Note:** If there is an active IP connection already established and this command is send, then the current IP connection will be terminated and the new connection will be attempted.
*LOGOUT	This command will terminate the IP connection. The ONLINE state will be reported as 0 (offline). LOGOUT

*MULTICAST_PASSWORD=<password> [:<callID>]	If requested to do so (see MULTICAST_CALL= feedback string), this command will send the required multicast password to the chair. <password>: password string (up to 49 characters) <callID>: 0..n = ID of your call. If not used, a call ID of 0 will be used instead. MULTICAST_PASSWORD=1234
*MULTICAST_TEXT=<text> [:<callID>]	Sends some text to the chair in a multicast call. <text>: text to send <callID>: 0..n = ID of your call. If not used, a call ID of 0 will be used instead. MULTICAST_TEXT=Hello chair
*MULTICAST_FLOOR=<state> [:<callID>]	Sends the appropriate floor command. <state>: REQUEST = request floor from chair REJECT = reject a floor grant from chair ACCEPT = accept a floor grant from chair <callID>: 0..n = ID of your call. If not used, a call ID of 0 will be used instead. MULTICAST_FLOOR=ACCEPT
*MUTE=<value>:<source>	Set the mute state of a source. <value>: 0 = mute off 1 = mute on <source>: 0 = Mic 1 = Speaker MUTE=0:0
*ONLINE?	Get the current state of communications between AMX and the device. ONLINE?
PAN=<end>:<camera>:<value>	Move (pans) the camera from side to side. <end> : N = near-end camera F = far-end camera <camera>: 1..4 = camera to affect (1 = Main Camera) <value> : + = begin pan right - = begin pan left S = stop pan PAN=N:1:+ PAN=N:1:S

PASSTHRU=<string>	Allows user the capability of sending commands directly to whatever unit is attached without processing by the NetLinx module. User must be aware of the protocol implemented by the unit to use this command. This gives the user access to features which may not be directly supported by the module. For more information, see the "Adding Functions to Modules" section. The communication module adds the CR,LF (carriage return, line feed) at the end of each string automatically <string>: string to send to unit PASSTHRU=Dial 10.0.0.6
PHONEBOOK=<type> :<entry> :<number> [:<name>[:<bitrate>]]	Write an entry in the phone book. After this command is processed the updated phonebook is returned. <type> : 1 = local. Always 1. <entry> : 1 = phone book entry number. Always 1. <number>: IP address, DNS name, E164, H323-ID <name> : string = name to store for entry (optional). If omitted, the communication module will use the <number> parameter as the name. <bitrate>: 64000, 128000, 192000, 256000, 320000, 384000, 512000, 768000, 1024000, 1536000, 2048000, 3072000, and 4096000 bps. = bitrate of the phonebook entry (optional). If omitted, a default bitrate of 64000bps will be used. PHONEBOOK=1:1:10.0.0.6:Tom PHONEBOOK=1:1:10.0.0.6:Mary:128000 **Note:** If the <name> of an entry starts with a number, the VCON will automatically prefix the letter 'A' to the <name>. **Note:** The <name> of an entry has a maximum length supported by the device of 25 characters.
PHONEBOOK?<entry>	Query for a phonebook entry. No data is exchanged with the VCON. This command will retrieve the phone book listings as they are stored in the communication module. If no listings are stored in the communication module, data is exchanged in order to update the module. <entry>: 0 = All entries 1..1000 = phone book entry to return PHONEBOOK?2 **Note:** If changes are made to the phone book by other means then the specific AMX control system API commands, this query will return old data. Please use PHONEBOOKREFRESH to update. After each reboot of the AMX system the communication module automatically generates a refresh.

PHONEBOOKDEL=<entry>	Delete a phone book entry. After this command is processed the updated phonebook is returned. <entry>: 1..1000 = phone book entry/index to delete string = Name of Entry PHONEBOOKDEL=16 PHONEBOOKDEL=Tom **Note:** If the <entry> parameter consists of only numbers, then the communication module will search and remove the entry by its index number in the phone book. If the <entry> parameter has at least 1 non-numeric character in it, then the communication module will search and remove the entry by its name.
PHONEBOOKDIAL=<entry>	Dial a phone book entry. <entry>: 1..1000 = phone book entry/index to dial string = Name of Entry PHONEBOOKDIAL=16 PHONEBOOKDIAL=Tom **Note:** If the <entry> parameter consists of only numbers, then the communication module will search and dial the entry by its index number in the phone book. If the <entry> parameter has at least 1 non-numeric character in it, then the communication module will search and dial the entry by its name.
*PHONEBOOKREFRESH	This will re-sync/refresh the phone book entries stored in the communication module to match the phone book entries stored in the VCON. This command should only be used if changes to the phonebook have been made from the VCON (by using the remote control or remote KEY emulator command). PHONEBOOKREFRESH
*REBOOT	Reboots the VCON. After the VCON has finished rebooting the IP connection must be re-established. See LOGIN= command. REBOOT
*REMOTE_ADDRESS?[<callID>]	Get the far-end IP address or DNS name. <callID>: 0..n = Optional call ID. If not used, a default call ID of 0 will be used. REMOTE_ADDRESS?
*REMOTE_NAME?[<callID>]	Get the far-end name. <callID>: 0..n = Optional call ID. If not used, a default call ID of 0 will be used. REMOTE_NAME?

*REMOTE_TERMINAL_TYPE?[<callID>]	Get the far-end terminal type. <callID>: 0..n = Optional call ID. If not used, a default call ID of 0 will be used. REMOTE_TERMINAL_TYPE?
*STREAMING=<event>[:<address>]	Starts, stops, or resumes streaming. <event>: 0 = stop 1 = start 2 = resume <address>: Optional IP address or Session ID. If starting a unicast stream, the <address> is the IP address of the far-side. If stopping or resuming a unicast stream, the <address> is the streaming session ID (1..n). If broadcast streaming is used, then the <address> parameter need not be used. STREAMING=1 (start broadcast) STREAMING=1:10.0.0.6 (start unicast) ... STREAMING=0:10.0.0.6 (stop unicast) STREAMING=0 (stop broadcast) **Note:** Unicast streaming sessions have a watchdog mechanism. Users must call the "resume" function periodically at intervals smaller than the interval returned by the "resume" command; otherwise, the unicast session will be terminated by the HD2000. To find out this interval of time, simply send a "resume" streaming command, and the time interval will be returned in milliseconds. **Note:** In order to start a unicast stream session, the broadcast streaming session must be started first, followed by the unicast stream. If stopping, the unicast stream must be stopped first and then the broadcast. See above example. **Note:** See device notes.
TILT=<end>:<camera>:<value>	Move (tilt) the camera up and down. <end> : N = near-end camera F = far-end camera <camera>: 1..4 = camera to affect (1 = Main Camera) <value> : + = begin tilt up - = begin tilt down S = stop tilt TILT=N:1:+
*TIMEOUT=<time>	Set the timeout period that the communication module will use to determine when its time to send the next command if no reply is received to the previous command sent. Adjust this time as needed according to your installation needs and your network's response time. For more details see "Programming Notes" section. <time>: 3..120 = time in seconds TIMEOUT=20 (default at reboot)
*TIMEOUT?	Get the current timeout value from the communication module. TIMEOUT?

VERSION?	Query for the current version number of the NetLinx module. VERSION?
*VIDMUTE=<value>	Set the video mute. This command will display to the far-side a predefined bitmap instead of your local video. <value>: 0 = off (send local video to far-side) 1 = on (mute local video) VIDMUTE=0
VIDSOURCE=<value>[:<end>]	Select a video input. <value>: 1..4 = video source (1 = Main Camera) <end>: Optional parameter. If omitted, near-end is used. 0 = local (near-end) 1 = far-end VIDSOURCE=2
VIDSOURCE?[<end>]	Query for the current video input source. <end>: Optional parameter. If omitted, near-end is used. 0 = local (near-end) 1 = far-end VIDSOURCE?
VOL=<value>	Adjust the volume level. <value>: + = increment the current volume level - = decrement the current volume level 0..100 = API volume range, in percent (%) VOL=+ VOL=28 **Note:** Please note that due to the internal volume range of the device taking values from 0..99 and this API's range of 0..100, an API volume percent of 99 will be treated as 100.
VOL?	Query for the current volume level. VOL?
ZOOM=<end>:<camera>:<value>	Zoom camera in (near/tele) or out (far/wide). <end> : N = near-end camera F = far-end camera <camera>: 1..4 = camera to affect (1 = Main Camera) <value> : + = begin zoom in (tele) - = begin zoom out (wide) S = stop zoom ZOOM=N:1:+

Table 1 – Send Command Definitions

String Feedback

The NetLinx module will provide feedback to the interface code for a/v conference controller changes via string events. Below are the strings supported.

String	Description
AUDSOURCE=<value>	Reports the current audio source input selected. <value>: 1 = Table-Top Mic 2 = Line Level (aux audio source) AUDSOURCE=1
AUTOANSWER=<mode>	Reports the current auto-answer mode for point-to-point calls. <mode>: 0 = Manual answer 1 = Auto answer 2 = Auto answer and Mute mic 3 = Auto reject AUTOANSWER=1
AUTOANSWER_MULTIPOINT=<mode>	Reports the current auto-answer mode for multipoint calls. <mode>: 0 = Manual answer 1 = Auto answer 2 = Auto reject AUTOANSWER_MULTIPOINT=1
BAN_OUTGOING=<mode>	Reports the current outgoing call ban status. <mode>: 0 = off (outgoing calls allowed) 1 = on (outgoing calls blocked) BAN_OUTGOING=0
CALL=<callID>:<state>	Reports the state of an outgoing call as it is processed by the far-side. <callID>: 0..n = ID of the call <state>: RING = call ringing at the far-end NOT_RESPONING = far-side not responding CALL=0:RING
CALL_ACTIVE_ID=<value>	Reports the call ID of all active calls. <value>: 0..n = ID of the call NONE = no active calls detected CALL_ACTIVE_ID=0

CALL_BITRATE=<callID> :<bitrate>	Reports the call's bitrate. <callID>: 0..n = ID of the call <bitrate>: bitrate of call CALL_BITRATE=0:64000
CALL_CHANNEL=<callID> :<channelID> :<state> :<direction> :<type>	Reports an active call's channel information. This information is reported at the begging of each call. <callID>: 0..n = ID of the call <channelID>: 0..n = ID of the channel <state>: 0 = Off 1 = On <direction>: 0 = RX 1 = TX <type>: AUDIO VIDEO DATA CALL_CHANNEL=0:0:1:0:AUDIO
CALL_DIRECTION=<callID> :<direction>	Reports the call's direction. <callID>: 0..n = ID of the call <direction>: RX (incoming) TX (outgoing) CALL_DIRECTION=0:TX
CALL_DURATION=<callID> :<duration>	Reports the call's duration. <callID>: 0..n = ID of the call <duration>: time in seconds CALL_DURATION=0:30
CALL_INFO=<callID>:<details>	Reports the call's information. This data is received with every incoming call. <callID>: 0..n = ID of the call <details>: far-end address followed by the far-end name/label. There is a space between the address and the name CALL_INFO=0:10.0.0.6 HQ
CALL_MAX=<value>	Reports the maximum number of concurrent calls supported by the VCON. <value>: 1..n CALL_MAX=3

DEBUG=<value>	Feedback on the state of the debug feature. <value>: 0 = off 1 = on DEBUG=1
DHCP=<mode>	Reports the current DHCP mode of the VCON HD2000 unit. <mode>: 0 = off (use static IP) 1 = on (use dynamic IP) DHCP=0
ERROR-<message>	Unsolicited feedback received is case of an error. Both the communication module and the device will report any errors. The communication module will report any command syntax errors while the device will report all other errors. If the device returns an error code, the following legend can be used (ErrorCode 1 = Parameter Invalid, ErrorCode 2 = Operation Not Supported, ErrorCode 3 = Operation Not Implemented, ErrorCode 4 = Operation Failed) <message>: description of the error message ERROR - STREAMINGSTOP ERROR 1 ERROR - Invalid Pan Command Parameter Used.
FECC=<mode>	Reports the current state of the far-end camera control. <mode>: 0 = off (disable) 1 = on (enable) FECC=1
HOOK=<state>:<callID> [:<reason>]	Reports the state of the hook. <state>: OFF = off-hook ON = on-hook (disconnected) <callID>: 0..n = ID of the call <reason>: Reported only if disconnected (on-hook). Reason for the disconnect. LOCAL = call disconnected by local party REMOTE = call disconnected by remote party BUSY = remote party is busy REJECT = remote party rejected call UNREACHABLE = remote party is unreachable NO_ANSWER = remote party does not answer UNKNOWN = call disconnected due to unknown reasons HOOK=OFF:0 HOOK=ON:0:LOCAL
IP_ADDRESS=<value>	Reports the VCON IP address. <value>: IP address in dot format IP_ADDRESS=10.0.0.6

LINE=<callID>:<state>	Reports the current call state. <line> : 0..n = call ID <state>: RING = ringing CONN = connected IDLE = idle LINE=0:IDLE
LOGIN=<state>	Reports a successful login to VCON. <state>: OK LOGIN=OK
MULTICAST_CALL=<callID> :<state>	Reply received whenever in a multicast call event is issued. <callID>: 0..n = ID of the call <state>: CONN = connected (multicast call connected) INVALID_PASSWORD (multicast password is invalid) PASSWORD_REQUESTED (multicast requires a password) MULTICAST_CALL=2:CONN
MULTICAST_FLOOR=<state> :<callID> [:<type> [:<name>]]	Reply received whenever a multicast floor event is triggered. <state>: OFFER = the multicast chair offers the floor to this endpoint. REJECT = the multicast chair rejected this endpoint's floor request. CHANGED = fired when the floor is given to another endpoint GRANTED = fired when the multicast chair has granted the floor to this endpoint <callID>: 0..n = ID of the call <type>: AUDIO = type of floor VIDEO <name>: text = the name/label of the participant who now owns the floor MULTICAST_FLOOR=OFFER:2 MULTICAST_FLOOR=GRANTED:2:VIDEO MULTICAST_FLOOR=CHANGED:2:AUDIO:DEMOROOM **Note:** The <type> parameter is only reported if the floor has changed or it has been granted. The <name> parameter is only reported if the floor has changed.
ONLINE=<value>	Reports the current connection state between AMX and the device. <value>: 0 = offline 1 = online ONLINE=1

PHONEBOOK=<entry> :<name> :<address> :<bitrate>	Reports a phone book entry. <entry>: 1..1000 = phone book entry/index number <name> : string = entry name <address>: xxx-xxx-xxxx = entry IP address, DNS name, H.323-ID, or E.164 <bitrate>: established bitrate for the entry. Valid values are 64000, 128000, 192000, 256000, 320000, 384000, 512000, 768000, 1024000, 1536000, 2048000, 3072000, and 4096000 bps. PHONEBOOK=1:BugsBunny:10.0.0.6:64000
REMOTE_ADDRESS=<callID> :<value>	Reports the far-side IP address, DNS name, H.323-ID, or E.164 while in a call. <callID>: 0..n = ID of the call <value>: text specifying the address REMOTE_ADDRESS=0:10.0.0.6
REMOTE_NAME=<callID> :<value>	Reports the far-side name/label. <callID>: 0..n = ID of the call <value>: text specifying the name/label of the remote terminal REMOTE_NAME=0:HQ
REMOTE_TERMINAL_TYPE=<callID> :<value>	Reports the far-side terminal type. <callID>: 0..n = ID of the call <value>: TERMINAL GATEWAY MCU GATEKEEPER REMOTE_TERMINAL_TYPE=0:TERMINAL
SOFTWARE=<version>	This reply is received after each successful login to a VCON unit. <version> software version of the VCON unit connected. SOFTWARE=0202.M05.D28.H12
STREAMING=<value> [:<sessionID>]	Reports the state of the streaming feature. <value>: 0 = stopped 1 = started 2 = resume <sessionID>: 0..n = streaming session ID number. If resuming a session, the <sessionID> is not reported. STREAMING=0:0 STREAMING=2

STREAMING_TIMEOUT=<time>	Reports the current time interval after which the VCON will terminate a unicast streaming session. <time>: time in milliseconds. STREAMING_TIMEOUT=1800000
TIMEOUT=<time>	Reports the current timeout value used by the communication module. For more details see the "Programming Notes" section. <time>: 3..120 = time in seconds TIMEOUT=20 (default at reboot)
VERSION=<value>	Communication module version feedback. <value>: current version number in xx.yy format VERSION=1.0
VIDSOURCE=<value>:<end>	Reports the current video source input selected. <value>: 1..4 = video input(1 = Main Camera) <end>: 0 = local (near-end) 1 = far-end VIDSOURCE=1:0
VOL=<value>	Feedback detailing the current volume level. <value>: 0..100 = API volume range, in percent (%) VOL=56 **Note:** Please note that due to the internal volume range of the device taking values from 0..99 and this API's range of 0..100, an API volume percent of 99 will be treated as 100.

Table 2 - String Feedback Definitions

Device Notes

- The AUDSOURCE= command does not properly select a source. It always selects "Table-Top Mic" regardless of the parameter passed. A workaround at this time is to use the remote control emulator command (KEY) to navigate to the appropriate menu and make the selection that way. This command has been implemented in this communication module version so that it can be used with future VCON firmware/software updates. This problem has been noted with the firmware/software version documented in the Introduction section of this document.
- If a broadcast stream is stopped while a unicast stream is in progress, you will not be able to stop the unicast stream unless you use the KEY= remote control emulator commands to navigate to the Network on-screen menu option and stop streaming that way.
- Presets are documented in the VCON HDK Integrator's Commands Guide as taking values from 0..9. Correct values are from 1..15.
- The DisplaySet command documented in the VCON HDK Integrator's Commands Guide on page 134 is not currently supported by the device, therefore it does not work and was not implemented in this communication module version.
- It has been noted that sometimes the device becomes unresponsive or slow while using repeated and fast KEY remote control emulator commands. Please refrain from ramping by using the KEY command whenever possible. If device becomes unresponsive or slow, reboot it.

Programming Notes

- The communication module implements a queue for sending commands to the device. A command will wait for the previous command to receive a reply before it is processed. If a command does not receive a reply within the timeout period, the device is declared offline and the next command in the queue is sent. The default timeout period is 20 seconds. To adjust this timeout period see TIMEOUT= command.
- After a successful connection to the VCON, the communication module will issue a 1 time query to the device. The data retrieved is as follows: Online state, Login state, VCON Software version, Audio Source, Video Source, Volume level, Auto-Answer mode for point-to-point calls, Auto-Answer mode for multipoint calls, Outgoing Call ban status, Active Call IDs, Line state, and all phonebook entries.
- The device does not always notify the communication module of changes made to the VCON without asking for it. Therefore, for example, phone book changes made from the remote control of the device will not be forwarded to the communication module automatically. Other commands such as Mute and Volume also fit into this category. Manual queries (where available) must be made in order to get the most updated information. The communication module does not constantly poll the device. Most call events (incoming call notifications, multicast floor states, line states, etc…) are however reported by the VCON automatically.
- Since there was no documented limit to the number of phone book entries that can be stored in the VCON, the communication module supports up to 1000 entries.
- Please limit all command parameters passed to the communication module to a maximum of 49 characters per parameter (unless otherwise noted).
- A sample user interface and touch panel file have been created as an example. While an attempt has been made to make this sample code as functional as possible, it may not fit the needs of your installation. Please feel free to modify the sample code as you see fit in order to

match the needs of a particular installation. The sample user interface code and touch panel file are presented as is.

Adding Functions to Modules

Commands to the device

This module supplies a mechanism to allow additional device features to be added to software using the module. This is the PASSTHRU command, which allows protocol strings to be passed through the module. The device-specific protocol must be known in order to use this feature.

As an example, suppose that a module for a projector has not implemented the 'white balance adjustment' feature. The command that the projector protocol requires is 03H, 10H, 05H, 14H, followed by a checksum. The documentation for the PASSTHRU command specifies that the module will automatically generate the checksum. In this case, the following string should be sent from the UI code to implement 'white balance adjustment'.

```
send_command vdvDevice, "'PASSTHRU=',$03,$10,$05,$14"
```

The reason to use PASSTHRU instead of sending a protocol string directly to the device port is that the device may require command queuing, calculation of checksums, or other internal processing, which would not be done if the string was sent directly. Because of this, it is best to filter all communication TO the device through the module. (The module documentation will indicate any processing that will be automatically done to the PASSTHRU string like checksum calculation.)

Additional Feedback from the device

The module documentation indicates what feedback is provided. If additional feedback is required, a CREATE_BUFFER for the device must be implemented in the user code to process the strings from the device manually. Note that the module will still be processing the response strings independently and sending the interpreted feedback up to the user code.