SnapAV Binary MoIP Controller Integration Protocol Document

Integration Protocol v1.2 rev20190522
Firmware 1.2.0.0

Overview

This integration protocol details how a third-party system can be used to control a SnapAV Binary MoIP Controller. With the controller online, the integration protocol will be listening for connections on **port 23 at the controllers IP address.** **NOTE: 10 simultaneous connections can be made at a time.** To get started, netcat or similar software can be used to initiate a connection and test any of the following protocol commands below.

Authentication

The protocol requires authentication before proceeding with commands. Once connected, a login prompt will be received and the third-party system must provide a valid username and password. If correct, login will be successful and other commands can be issued. If incorrect, the third-party system will be prompted for login again.

Specification

THIRD-PARTY SYSTEM <--> SnapAV Binary MoIP Controller
i.e. MoIP IP: 192.168.0.20 Port: 23

Integration

Message Structure
Command and response messages are standard ASCII text.
? – Request message
! – Control message
- Error message
~ - Unsolicited message
\n – End of command message, ASCII hex: 0x0A dec: 11

Protocol

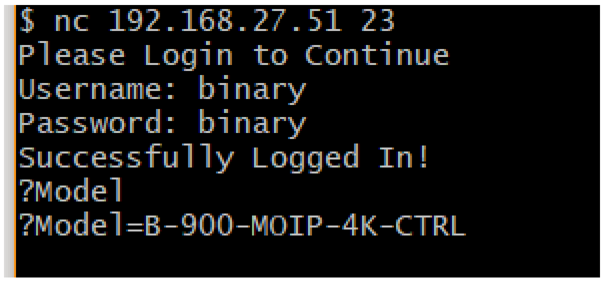
Protocol Command	**Description/Response**
?Firmware\n	Request Firmware Version. Response: `?Firmware=1.0.0.0\n`
?Receivers\n	Request all Receivers current inputs. Response: `?Receivers=1:3\n` Where 1 is the TX and 3 is the RX. This will be comma delimited for multiple devices.

?Devices\n	Request TX and RX count. Response: ?Devices=1,4\n Where 1 is the TX count and 4 is the RX count.
?Name=T\n Where T is 0/1	Request the names for either TX or RX. To request all the TX names, use 1 for the payload. To request all the RX names, use 0 for the payload. The response will be new line delimited for multiple devices where each lines format is as follows: ?Name=MODE,INDEX,NAME. Request for TX: ?Name=1\n Response for TX: ?Name=1,1,TX-D46A9121000B\n Request for RX: ?Name=0\n Response for RX: ?Name=0,1,RX-D46A91210620\n ?Name=0,2,Basement TV\n ?Name=0,3,Living Room TV\n ?Name=0,4,RX-D46A91210604\n
!Switch=TX,RX\n Where TX is the index of the Transmitter you want to switch and RX is the index of the Receiver you want the switch to happen on.	Switches the input on a Receiver to the desired Transmitter. Request to switch to Transmitter 1 on Receiver 2: !Switch=1,2\n Success Response: OK\n Error Response: #Error
!Resolution=RX,R\n Where RX is the Receiver you'd like to change the resolution of and R is one of the following: 0 = Pass through resolution from the source. 1 = 1080p 60Hz 2 = 1080p 50Hz 3 = 2160p 30Hz 4 = 2160p 25Hz	Changes the resolution on a given Receiver. Request to switch Receiver 1's resolution to Pass-Through: !Resolution=1,0\n Success Response: OK\n Error Response: #Error
!OSD=RX,MSG\n Where RX is the Receiver index you'd like to display MSG on. MSG must be plain ASCII Text.	Displays a plain text message on the display of the given Receiver. Request to display "Hello World" on Receiver 1: !OSD=1,Hello World\n Success Response: OK\n Error Response: #Error **NOTE: To clear the text, send !OSD=1,CLEAR\n**
!Reboot\n	Request to reboot the MoIP controller. Reboot Controller Request: !Reboot\n Success Response: OK\n Error Response: #Error

!CEC=RX,MODE\n Where RX is the Receiver index you'd like to control CEC on and MODE is one of the following: 0 = CEC OFF 1 = CEC ON	Controls CEC for a given Receiver. MODE must either be 0 for OFF or 1 for ON. Request CEC Off on Receiver 1: !CEC=1,0\n Success Response: OK\n Error Response: #Error
!Serial=TYPE,INDEX,BAUD,DATABITS,PARITY,STOPBITS,DATA\n type: 0 = output (RX), 1 = input (TX) index: device to send baud: integer baudrate data bits: 5, 6, 7, 8 parity: n = none, e = even, o = odd stop bits: 1, 2 data: hex data to send	Sends serial data to RX or TX serial port. Send to TX 2 at 9600-8n1 the characters "abc": !Serial=1,2,9600-8n1,61 62 63 Success Response: OK\n Error Response: #Error\n
!IR=TYPE,INDEX,PRONTOCODE\n type: 0 = output (RX), 1 = input (TX) index: device to send prontocode: Pronto Hex format string	Sends IR data to RX or TX IR Flasher. Send to TX 4 the pronto code 0000 006a 0022 0002 0160 00b2 0015 0017 0015 0017 0015 0043 0015 0017 0015 0017 0015 0017 0014 0018 0015 0017 0015 0043 0015 0043 0015 0017 0015 0043 0015 0043 0015 0043 0015 0043 0015 0043 0015 0017 0015 0017 0015 0017 0015 0043 0015 0017 0015 0017 0015 0017 0015 0017 0015 0043 0015 0043 0015 0043 0015 0017 0015 0044 0014 0044 0014 0044 0014 0044 0014 061d 015f 005a 0015 0eb5: !IR=1,4, 0000 006a 0022 0002 0160 00b2 0015 0017 0015 0017 0015 0043 0015 0017 0015 0017 0015 0017 0014 0018 0015 0017 0015 0043 0015 0043 0015 0017 0015 0043 0015 0043 0015 0043 0015 0043 0015 0043 0015 0017 0015 0017 0015 0017 0015 0043 0015 0017 0015 0017 0015 0017 0015 0017 0015 0043 0015 0043 0015 0043 0015 0017 0015 0044 0014 0044 0014 0044 0014 0044 0014 061d 015f 005a 0015 0eb5 Success Response: OK\n Error Response: #Error\n
~Serial=TYPE,INDEX,DATA\n TYPE: 0 = output (RX), 1 = input (TX) INDEX: device to send DATA: hex data received	Unsolicited serial data to the connected client. This data will be sent over the protocol without a request. The third-party system should always be handling these incoming messages. TX #2 sent characters "abc": ~Serial=1,2,61 62 63
~Receivers=TX,RX\n	Broadcasts all Receivers current inputs.

Where TX is the currently selected Transmitter index and RX is the Receiver index.	Response: `?Receivers=1:3\n` Where 1 is the TX and 3 is the RX. This will be comma delimited for multiple devices.
#Error\n	Sent whenever an invalid command was received or an internal device error has occurred. Consider this example with only 2 connected Transmitters and 5 connected Receivers: Request to switch Transmitter 2 to Receiver 6: !Switch=2,6\n Response: #Error Receiver 6 does not exist, therefore an error is returned.

Example:



```
$ nc 192.168.27.51 23
Please Login to Continue
Username: binary
Password: binary
Successfully Logged In!
?Model
?Model=B-900-MOIP-4K-CTRL
```