Axion DMX Controller Control System API

API Version 1.2

Minimum Firmware Required: 1.5.0

Protocol

A simple control interface is exposed on TCP port 4005. It is a line based control protocol using '\r\n' as the line terminator. It is also command/response protocol with no async messages currently.

Each command starts with the '>' character and is followed by an optional ID for command/response tracking. If given it should be 1-8 alphanumeric characters in length and ends with a ':'. Next is the command to perform. Following this is a comma separated list of parameters for the command. The number, or presence at all, is command dependent.

The response will start with the '#' character and is followed by the optional ID, and the colon, if given on the command. Next is the result code.

The result code of 'ok' is a command success. It may be followed by a comma separated list of results parameters. This is command dependent.

The result code of 'close' means the socket will close. Most likely the unit is rebooting or the connection was requested to be closed.

The result code of 'error' means there was an error and most likely will be followed with a comma and the error message.

Commands

Note: Almost all commands, other than 'getversion' and 'getmodel', require the connection to login with the 'login' command.

- getversion : Return the firmware version.
- getmodel : Return the model number.
- reboot : Reboot the unit. Should expect a close result code.
- setip : Set the IP settings. The first parameter is the interface, either 'eth' or 'wifi'. Then it's the ip, netmask, and gateway.
- getip : Get the IP settings. The provided parameter is the interface, either 'eth' or 'wifi'. The returned parameters are the ip, netmask, and gateway.
- setwifi : Set the Wifi network. First parameter is the SSID and the second is the Wifi password.
- getwifi : Get the current Wifi network.
- scanwifi : Return a list of available Wifi APs.
- login : Login and enable interaction. One parameter is expected, the system password.
- setpasswd : Change the system password.
- upgrade : Do a firmware upgrade. One parameter is expected, the next 64 bytes of the firmware image base64 encoded. The last block can be less. Once all the data has been sent, an empty 'upgrade' command should be sent to finish the upgrade. Expect a close result code if successful.
- close : Close the connection. Expect a close result code in response.

- stop : Stops a channel if it's currently ramping to a value. One parameter expected, the channel number (0-511).
- pause : Pause a channel. This applies to a current ramp and any future ramps. One parameter expected, the channel number (0-511).
- release : Release a number of paused channels. Can be given up to 16 channels to un-pause at the same time.
- setlevel : Set a channel level with possible delay and ramp. First parameter is required and is the channel number(0-255). Second parameter is the target level (0-255/65535). The third optional parameter is the ramp rate in levels per second. Fourth optional parameter is a start delay in milliseconds.
- getlevel : Get a channels current level. One parameter expected, the channel number(0-511). Returns the current level, which may be in mid ramp and may be a decimal value.
- save : Save a channels target level and ramp rate used to reach that level. First parameter is the channel number(0-511). The second parameter is the save slot to store it in (0-3).
- recall : Recall a saved target and ramp rate. First parameter is the channel number(0-511). The second parameter is the save slot to load from (0-3).
- savescene : Save one, or more, channels target level and ramp rate used to reach that level. First parameter is the save slot to store it in (0-3). While numbered the same, these slots are separate from unit slots. The remaining parameters, up to 8, are the channels to save.
- recallscene : Recall a saved list of channels. First parameter is the save slot to load from (0-3). While numbered the same, these slots are separate from unit slots. When restored all the saved channels will move to their saved target at the ramp rate they had when saved.
- setrawlevel : Sets a channel to a given level directly. The first parameter is the channel number(0-255). The second parameter is the level(0-255/65535).
- getrawlevel : Gets the channels raw level value. The first parameter is the channel number(0-255). The returned parameter is the level(0-255/65535).
- animate : Starts an animation with a set of channels and values. Only one animation is active at any time. Setting the animation to -1 will stop a currently running animation. The first parameter is the animation mode (see below). The next four parameters are the channels to use with the animation. A -1 can be given for a channel to not use that option. The next 8 parameters are factors in the animation. These are animation dependent and all may not be needed.
    - (0) Ramp to random target values. Up to four factors should be given, which are the ramp rates for the given channels.
    - (1) Roll around the HSV color cylinder. The first three channels are the R, G, B channels to use. The fourth channel isn't used. The first factor is the S value. This is the color saturation and could be described as how 'rich' the color is. It has a range of 0.0-1.0. The second factor is the V value. This is the intensity value and describes how bright the color is. It has a range of 0.0-1.0. The third factor is the ramp rate in roughly levels per second.
- setbitmode : Sets the DMX bus bit mode. Options are 8, 16be(big endian 16 bits), or 16le(little endian 16 bits).

- getbitmode : Gets the DMX bus bit mode. Options are 8, 16be(big endian 16 bits), or 16le(little endian 16 bits).

TCP Control Channel

The TCP control channel is exposed on TCP port 4005 and uses the protocol described above. There's a limit of 4 connections at any one time.

UDP Control Channel

The UDP control channel is exposed on UDP port 4005 and uses the protocol described above.

One or more commands may be in each packet, but the packet size is limited to 1024 bytes. Responses are sent back to the source of the UDP packet. All the commands in a packet will have their responses set back together.

Each packet is considered a new connection and will require the 'login' command if the other commands require it.

Since the commands 'getname', 'getmodel' and 'getversion' don't require a login, sending them in a packet to the broadcast address could be used as a simple discovery process.