LEA AMPLIFIER MODULES - 2.0

SUMMARY

The LEA amplifier module suite, is a generic set of modules that allows you to control a majority of the API endpoints. Each symbol can interact with a different type of endpoint.

The main debug module controls the verbosity of the debug logging, as well as the global enable/disable state. Every other symbol has a debug flag where you can enable debug logging for that specific module. (Enable only for testing purposes to avoid unnecessary disk writes! Logs are saved in the \\user directory with the assembly name as the prefix, and a timestamp as the suffix.)

INPUTS

debug_enable

Enables debug logging for this symbol instance.

subscribe

Subscribes this symbol instance to real-time feedback from the device.

unsubscribe

Unsubscribes this symbol instance to real-time feedback from the device. (This module will no longer automatically receive real feedback from the device.)

get_value

Requests the current value from the device for the corresponding API endpoint.

push_value

Sends the desired value to the device. (Note: this is based on set_value/set_value$ for analog and serial API elements. For Bool endpoints, there is a discrete true/false digital input.)

** For Enum types, there is a discrete set digital for each settable value, and a corresponding digital output for each value as feedback.

OUTPUTS

initialized.fb

Indicates that the module is ready to be used, and has initialized properly.

is_sensor_element.fb

Indicates that the API element is a sensor (and thus cannot be set manually from the program).

is_control_element.fb

Indicates that the API element is a control point (and thus can be set manually from the program).

value.fb

This signal outputs the current known value of the API element. This value is updated automatically if a subscription has been set, or if the **get_value** input has been triggered. For analog values, this should be the full 16-bit unsigned range as a percentage of the allowed range.

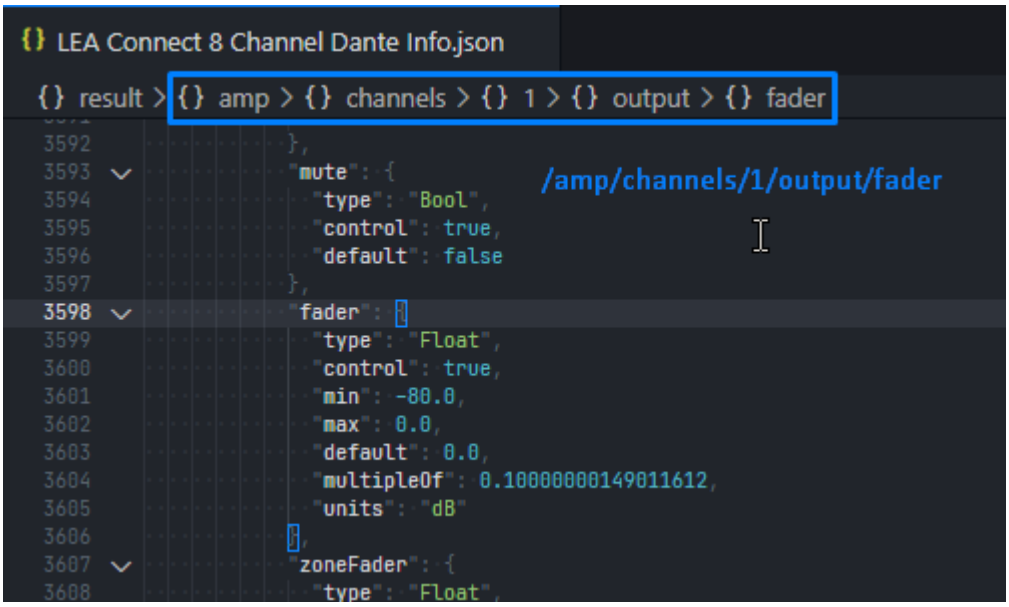
PARAMETERS

communicator

The LEA amplifier modules allow for 16 independent communicator groups. Typically, you would assign a single communicator per device that you want to control, and tell each corresponding symbol which communicator it is associated with. The main controller symbol specifies the IP address for the communicator group.

endpoint

The endpoint parameter is used to specify the API endpoint based on the API hierarchy. From the JSON API definition below, you can see that the endpoint for this fader is simply the levels of JSON up to the control point, separated by a forward slash.



Make sure you are using the correct symbol for the API endpoint you want to control! If you are using the incorrect symbol type for a specific endpoint, an exception is purposefully thrown at runtime. Check the error logs for any exception messages mentioning a symbol / endpoint mismatch to address any problems for production code. (String, Bool, and Float should use the generic type symbols, and anything as an Enum should be using one of the other symbols depending on the expected actions and feedback.)

NOTES

To ensure proper module feedback for each symbol, please make sure that you are subscribing each module based on the connected feedback from the controller symbol for the corresponding communicator group.

There should be only one controller per communicator group defined in a single program!