Lumencor

Light Engine Command Reference

Index of standard mode commands implemented on
AURA, CELESTA, RETRA, SPECTRA and ZIVA Light Engines

Lumencor, Inc.

Document Number 57-10018
Revision 4.5.5 072726

www.lumencor.com

Lumencor

Lumencor Light Engine Command Reference

Updated on 7/27/2026
Revision 4.5.5

Command parsing rules

Communication Protocols

System Information
- Get Version (GET VER)
- Get Number Of Channels (GET NUMCH)
- Get Model (GET MODEL)
- Get Serial Number (GET SN)
- Get Part Number (GET PARTNUM)
- Get Channel Map (GET CHMAP)
- Get Light Engine Status Code (GET STAT)
- Get Light Status Code (GET CHSTAT)
- Get Multiple Lights Status Codes (GET MULCHSTAT)
- Get Light Operating Time (GET OT)
- Get Multiple Light Operating Times (GET MULOT)
- Save Light Operating Time (SET SAVEOT)
- Get System Temperature (GET TEMP)
- Get System Temperature Data (GET TEMPDATA)
- Get Fan Status Code (GET FAN)
- Get Power Supply Current Measurement (GET SUPPLYCURRENT)
- Get Power Supply Power Measurement (GET SUPPLYPOWER)
- Get Error Description (GET ERRORTEXT)

System Control and Configuration
- Get IP Address (GET IP)
- SET IP Address (SET IP)
- Set User Variable (SET USERVAR)
- Get User Variable (GET USERVAR)
- Set Logging Level (SET LOGLVL)
- Get Logging Level (GET LOGLVL)
- Enable/Disable USB 5V Supply (SET USBPOWER)
- Check USB 5V Supply State (GET USBPOWER)
- Enable/Disable TTL Inputs (SET TTLENABLE)
- Check TTL Inputs Master Switch State (GET TTLENABLE)
- Set TTL Inputs Polarity (SET TTLPOL)

- Get TTL Inputs Polarity (GET TTLPOL)
- Get TTL Input Connector Pin (GET TTLPIN)
- Get Multiple TTL Input Connector Pins (GET MULTTLPIN)
- Set USB Port Mode (SET MODEUSB)
- Set RS232 Port Mode (SET MODECOM)
- Set power reading crosstalk compensation (SET CROSSTALK)
- Get power reading crosstalk compensation (GET CROSSTALK)
- Reset governor lock (RESET GOVERNOR)
- Set selected fiber / output (SET FBRSEL)
- Get selected fiber / output (GET FBRSEL)
- Reboot (REBOOT)
- Shutdown (SHUTDOWN)
- Wake up system (WAKEUP)
- Switch Standby mode (SLEEP)

Light Output Control

- Set Light State (ON/OFF) (SET CH)
- Get Light State (ON/OFF) (GET CH)
- Get Light TTL State (ON/OFF) (GET CHTTL)
- Get Actual Light State (ON/OFF) (GET CHACT)
- Set Multiple Light States (ON/OFF) (SET MULCH)
- Get Multiple Light States (ON/OFF) (GET MULCH)
- Get Multiple Light TTL States (ON/OFF) (GET MULCHTTL)
- Get Multiple Light Actual States (ON/OFF) (GET MULCHACT)
- Get Maximum Intensity Level (GET MAXINT)
- Set Light Intensity (SET CHINT)
- Get Light Intensity (GET CHINT)
- Set Multiple Light Intensities (SET MULCHINT)
- Get Multiple Light Intensities (GET MULCHINT)
- Set Multiple Light States (ON/OFF) & Intensities (SET MULCHPROP)
- Set Multiple Selected Light States (ON/OFF) & Intensities (SET MULCHPROPALT)
- Get Light Power Level (GET CHPWR)
- Get Multiple Light Power Levels (GET MULCHPWR)
- Get Light Estimated Power Output (GET CHPWRWATTS)
- Get Multiple Light Estimated Power Outputs (GET MULCHPWRWATTS)
- Set Light Power Regulation Lock State (SET PWRLOCK)
- Get Light Power Regulation Lock State (GET PWRLOCK)
- Set Multiple Light Power Regulation Lock States (SET MULPWRLOCK)
- Get Multiple Light Power Regulation Lock States (GET MULPWRLOCK)
- Set Light Power Reference (SET PWRREF)
- Get Light Power Reference (GET PWRREF)
- Set Multiple Light Power References (SET MULPWRREF)

- Get Multiple Light Power References (GET MULPWRREF)
- Get Average Power During Power Regulation (GET PWRAVG)
- Get Standard Deviation During Power Regulation (GET PWRDEV)
- Get Maximum Error Value During Power Regulation (GET PWRMAXERR)
- Set PID Mode (SET PIDMODE)
- Get PID Mode (GET PIDMODE)

Multi-channel Modes
- Create Multi-channel Mode (CREATE MULTIMODE)
- Delete Multi-channel Mode (DELETE MULTIMODE)
- List all available Multi-channel Modes (GET MULTIMODES)
- Set Multi-channel Mode (SET MULTIMODE)
- Get Multi-channel Mode (GET MULTIMODE)
- Get channels corresponding to a given Multi-channel Mode (GET MULTIMODECHANNELS)
- Set Multi-channel Mode Intensity (SET MULTIMODEINT)
- Get Multi-channel Mode Intensity (GET MULTIMODEINT)
- Set Multi-channel Mode State: On/Off (SET MULTIMODESW)
- Get Multi-channel Mode State: On/Off (GET MULTIMODESW)
- Set Single Channel Intensity in Multi-Channel Mode (SET MULTIMODESINGLEINT)
- Set Single Channel State (ON/OFF) in Multi-Channel Mode (SET MULTIMODESINGLESW)
- Set Multi-channel Mode Power Regulation Lock State (SET MULTIMODEPWRLOCK)
- Get Multi-channel Mode Power Regulation Lock State (GET MULTIMODEPWRLOCK)
- Set Multi-channel Mode Power Reference (SET MULTIMODEPWRREF)
- Get Multi-channel Mode Power Reference (GET MULTIMODEPWRREF)
- Set Multi-channel Mode LUT (SET MULTIMODELUT)
- Get Multi-channel Mode LUT (GET MULTIMODELUT)
- Set Multi-channel Mode LUT for the second output (SET MULTIMODELUTB)
- Get Multi-channel Mode LUT for the second output (GET MULTIMODELUTB)
- Save Multi-channel Mode LUT (SAVE MULTIMODELUT)
- Get Maximum DAC value (GET MAXDAC)
- Get Channel DAC value (GET CHDAC)

Error codes

Channel status codes

REST Interface

Lumencor

Command parsing rules

- Each command sent to the Light Engine is a text string.
- A command is a list of tokens (words) separated by whitespace.
- Command contains two or more tokens. Number of tokens is variable.
- The first token for a TX (transmitted) command string is one of the two keywords GET, SET (to query or to set)
- First token for a RX (received) answer string is one of the two keywords A, E (for success or failure)
- Each TX command will cause the Light Engine to respond with a RX string. Depending on the command, typical response times are 1-10ms. If RX string is not received within 50ms, we can assume that the command failed
- Serial port commands must be terminated by either LF (ASCII 10) or CR (ASCII13).
- TCP commands should not have any terminating character at the end

Communication Protocols

LE Server supports serial communication over RS232, TCP (on port 8095) and HTTP (REST interface on default port) over Ethernet. Serial and TCP commands are used in the form presented in this reference. In order to send a command over the REST interface additional formatting is required. See REST Interface for more information.

System Information

Get Version (GET VER)

Returns the Light Engine firmware version.

Command:

```
GET VER
```

Answer:

```
A VER 1.0.6
```

Get Number Of Channels (GET NUMCH)

Returns the number of channels available in the Light Engine.

Command:
`GET NUMCH`

Answer:
`A NUMCH 4`

Get Model (GET MODEL)

Returns model name.

Command:
`GET MODEL`

Answer:
`A MODEL SPECTRAX`

Get Serial Number (GET SN)

Returns serial number.

Command:
`GET SN`

Answer:
`A SN 6678`

Get Part Number (GET PARTNUM)

Returns part number.

Command:
`GET PARTNUM`

Answer:
`A PARTNUM 90-10496`

Get Channel Map (GET CHMAP)

Returns ordered list of color mnemonics, e.g. VIOLET,BLUE,GREEN,RED, separated by spaces. The first channel ID is 0, the next one 1, etc. The list in this example should be interpreted as zero-based mapping of channel numbers to color mnemonics:

0 = VIOLET
1 = BLUE

2 = GREEN
3 = RED

Command:
`GET CHMAP`

Answer:
`A CHMAP VIOLET BLUE GREEN RED`

Get Light Engine Status Code (GET STAT)

Returns current status code from the Light Engine. "0" means all OK. Any other number will represent an error ID with a specific meaning:

0 - OK
1 - Fan malfunction
2 - High temperature (over 25 C)
3 - High temperature and fan malfunction
4 - Device safety lock active
5 - Invalid hardware configuration
6 - Standby mode (TECs disabled)
7 - TECs warming up

Command:
`GET STAT`

Answer:
`A STAT 3`

Get Light Status Code (GET CHSTAT)

Returns current status code of the individual light, specified by its ID number. Return value of "0" means the normal operation. Any other number will represent an error condition.

Command:
`GET CHSTAT 3`

Answer:
`A CHSTAT 0`

Get Multiple Lights Status Codes (GET MULCHSTAT)

Obtains multiple light channel status codes with a single command. Channel status codes are provided in the order of channel IDs.

Command:
`GET MULCHSTAT`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

Answer:
`A MULCHSTAT 0 0 0 0`
if successful or
`E MULCHSTAT`
if error occurred

Get Light Operating Time (GET OT)

Returns total time in milliseconds that the specified channel was in the ON state. This value accumulates through the entire lifetime of the Light Engine.

Command:
`GET OT 2`
Returns total operating time for channel 2

Answer:
`A OT 1890667`

Get Multiple Light Operating Times (GET MULOT)

Obtains multiple light channel operating times (total milliseconds that the channel was in the ON state) with a single command. Operating times are provided in the order of channel IDs.

Command:
`GET MULOT`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

Answer:
`A MULOT 1890667 4646464 311585 2213`
if successful or
`E MULOT`
if error occurred

Save Light Operating Time (SET SAVEOT)

Explicitly save operating time for all light channels. Operating times will be persisted on disk.

Command:
```
SET SAVEOT
```

Answer:
```
A SAVEOT
```

Get System Temperature (GET TEMP)

Returns current temperature in degrees Celsius.

Command:
```
GET TEMP
```

Answer:
```
A TEMP 26.2
```

Get System Temperature Data (GET TEMPDATA)

Returns current temperature in degrees Celsius, relative humidity (if appropriate sensor available) and dew point in degrees Celsius (if appropriate sensor is available).

Command:
```
GET TEMPDATA
```

Answer:
```
A TEMPDATA 26.2 30.2 12.5
```

Get Fan Status Code (GET FAN)

Returns the fan status code:
0 - Fan off
1 - Fan on / Low speed
2 - Fan on / High speed
3 - Fan malfunction

Command:
```
GET FAN
```

Answer:
```
A FAN 1
```

Get Power Supply Current Measurement (GET SUPPLYCURRENT)
Read power supply current measurement (in milliamperes [mA]).

Command:
```
GET SUPPLYCURRENT
```

Answer:
```
A SUPPLYCURRENT 350.8
```

Get Power Supply Power Measurement (GET SUPPLYPOWER)
Read power supply power usage measurement (in Watts [W]).

Command:
```
GET SUPPLYPOWER
```

Answer:
```
A SUPPLYPOWER 8.41
```

Get Error Description (GET ERRORTEXT)
Obtains error description for the specified error code.

Command:
```
GET ERRORTEXT 67
```

Answer:
```
A ERRORTEXT Invalid system configuration
```

System Control and Configuration

Get IP Address (GET IP)
Returns the current IP address of the Light Engine.

Command:
```
GET IP
```

Answer:

```
A IP 192.168.1.163
```

SET IP Address (SET IP)

Assigns static IP address to Light Engine. The change will take effect only after reboot. If the newly assigned address clashes with another device on the network, or if an invalid address is entered the unit can become unreachable via TCP.

If we provide only a single parameter (IP address), subnet mask and gateway will be cleared and the light engine will use system defaults.

We can optionally provide two additional parameters to this command: subnet mask and gateway.

Single parameter example

Command:

```
SET IP 192.168.1.180
```

To set dynamic IP assignment (via DHCP) use the following variation of the command:

```
SET IP DHCP
```

or

```
SET IP 0.0.0.0
```

Answer:

```
A IP
```

Multiple parameter example

Command:

```
SET IP 10.32.35.130 255.255.255.0 10.32.35.1
```

ip *subnet mask* *gateway*

Answer:

```
A IP
```

Set User Variable (SET USERVAR)

Set value of the user variable. Value can be any string or number. On startup/powerup user variable is assigned value "0".

Command:

```
SET USERVAR 1
```

Answer:

```
A USERVAR
```

Get User Variable (GET USERVAR)

Returns the current value of the user variable. If not assigned by SET command "0" will be returned by default.

Command:
```
GET USERVAR
```

Answer:
```
A USERVAR 0
```

Set Logging Level (SET LOGLVL)

Set current logging level. Valid logging levels are integers between 0 and 5. Value is persisted across sessions. Logging level determines how much information will be written into system log:

0 - None (system log turned off)
1 - Errors, System Notifications
2 - Warnings, Errors, System Notifications
3 - App Notifications, Warnings, Errors, System Notifications
4 - Debug mode (level 3 plus notification for every command)
5 - Complete trace (level 4 plus traces for all hardware (I2C, SPI, GPIO, AIN) communication)

Command:
```
SET LOGLVL 2
```

Answer:
```
A LOGLVL
```

Get Logging Level (GET LOGLVL)

Returns the current logging level.

Command:
```
GET LOGLVL
```

Answer:
```
A LOGLVL 2
```

Enable/Disable USB 5V Supply (SET USBPOWER)

Enable/disable 5V power supply for the USB interface.

Command:
`SET USBPOWER 1`
The above example turns on USB 5V supply (with argument value of "1"). To turn it off use "0" argument.

Answer:
`A USBPOWER`

Check USB 5V Supply State (GET USBPOWER)

Check if the USB 5V power supply is enabled.

Command:
`GET USBPOWER`

Answer:
`A USBPOWER 1`

Enable/Disable TTL Inputs (SET TTLENABLE)

Enable/disable TTL inputs (TTL inputs master switch).

Command:
`SET TTLENABLE 1`
The above example turns enables TTL inputs (with argument value of "1"). For disabling use "0" argument.

Answer:
`A TTLENABLE`

Check TTL Inputs Master Switch State (GET TTLENABLE)

Check if TTL inputs are enabled (TTL inputs master switch).

Command:
`GET TTLENABLE`

Answer:
`A TTLENABLE 1`

Set TTL Inputs Polarity (SET TTLPOL)

Set TTL inputs polarity (high or low).

Command:
`SET TTLPOL HIGH`
The above example sets TTL inputs high polarity (with argument value of "HIGH"). For low polarity use the "LOW" argument. The alternative syntax is to use POS and NEG as arguments instead (backward compatibility).

Answer:
`A TTLPOL`

Get TTL Inputs Polarity (GET TTLPOL)

Get TTL inputs polarity (high or low).

Command:
`GET TTLPOL`

Answer:
`A TTLPOL HIGH`

Get TTL Input Connector Pin (GET TTLPIN)

Get TTL input connector pin number (1..15) for a selected light channel. If the channel doesn't support TTL triggering, or the input connector pin is not defined server will respond with -1.

Command:
`GET TTLPIN 2`

Answer:
`A TTLPIN 11`

Get Multiple TTL Input Connector Pins (GET MULTTLPIN)

Get TTL input connector pin number (1..15) for all light channels. If a channel doesn't support TTL triggering, or the input connector pin is not defined server will respond with -1 (for that channel).

Command:
`GET MULTTLPIN`

Answer:

```
A MULTTLPIN 1 3 11 14 12
```

Set USB Port Mode (SET MODEUSB)

Sets communication mode for the USB port. "STD" for Standard and "LEGACY" for Legacy mode.

Command:
```
SET MODEUSB LEGACY
```

Answer:
```
A MODEUSB
```

Set RS232 Port Mode (SET MODECOM)

Sets communication mode for the RS232 port. "STD" for Standard and "LEGACY" for Legacy mode.

Command:
```
SET MODECOM STD
```

Answer:
```
A MODECOM
```

Set power reading crosstalk compensation (SET CROSSTALK)

Configure power regulator to compensate for power reading crosstalk.

Command:
```
SET CROSSTALK 1
```

Answer:
```
A CROSSTALK
```

Get power reading crosstalk compensation (GET CROSSTALK)

Check if the power regulator is configured to compensate for power reading crosstalk.

Command:
```
GET CROSSTALK
```

Answer:
```
A CROSSTALK 1
```

Reset governor lock (RESET GOVERNOR)

This command returns system to normal operation (after governor has locked the system due to excessive power usage).

Command:
`RESET GOVERNOR`

Answer:
`A GOVERNOR`

Set selected fiber / output (SET FBRSEL)

- Introduced in **v4.5.0**

Select active fiber / output. User can choose between outputs A (1) and B (0). Selected state is persisted between reboots.

Command:
`SET FBRSEL 1`

Answer:
`A FBRSEL`

Get selected fiber / output (GET FBRSEL)

- Introduced in **v4.5.0**

Check which fiber / output is currently active.

Command:
`GET FBRSEL`

Answer (output B is selected):
`A FBRSEL 0`

Reboot (REBOOT)

This command reboots the Light Engine control computer. After issuing this command and receiving answer, the system will be unresponsive during reboot process (about 20 sec).

Command:
`REBOOT`

Answer:

```
A REBOOT
```

Shutdown (SHUTDOWN)

This command shuts down the control computer. After issuing this command and receiving answer, the system will stop responding to commands until next power up.

Command:
```
SHUTDOWN
```

Answer:
```
A SHUTDOWN
```

Wake up system (WAKEUP)

Wake up system (TECs) from standby, and start TEC warm up. Command will return immediately (won't wait for TECs to warm up). If the system is not in standby state when the command is issued, it will return without affecting the engine state (on/off, intensities, etc.), instead it would only reset the standby countdown timer. Sending periodic WAKEUP commands will prevent the light engine from going into standby.

Command:
```
WAKEUP
```

Answer:
```
A WAKEUP
```

Switch Standby mode (SLEEP)

Change the system state to Standby. The light engine will unconditionally and immediately switch to Standby mode.

Command:
```
SLEEP
```

Answer:
```
A SLEEP
```

Light Output Control

Set Light State (ON/OFF) (SET CH)

Turns light ON or OFF for the specified channel.

Command:
`SET CH 2 1`
The above example sets channel 2 to ON

`SET CH 3 0`
The above example sets channel 3 to OFF

Answer:
`A CH`
if successful or
`E CH`
if error occurred

Get Light State (ON/OFF) (GET CH)

Obtain ON or OFF state for the specified channel.

Command:
`GET CH 2`
The above example requests state for channel 2

Answer:
`A CH 1`
or
`A CH 0`
if successful or
`E CH`
if error occurred

Get Light TTL State (ON/OFF) (GET CHTTL)

Obtain TTL ON or OFF state for the specified channel.

Command:
`GET CHTTL 2`
The above example requests state for channel 2

Answer:
`A CHTTL 1`
or
`A CHTTL 0`
if successful or
`E CH`
if error occurred

Get Actual Light State (ON/OFF) (GET CHACT)

Obtain actual ON or OFF state for the specified channel. This command checks both TTL inputs and channel state switches.

Command:
`GET CHACT 2`
The above example requests state for channel 2

Answer:
`A CHACT 1`
or
`A CHACT 0`
if successful or
`E CH`
if error occurred

Set Multiple Light States (ON/OFF) (SET MULCH)

Turns multiple lights ON or OFF with a single command. Channel ON or OFF settings (1 or 0) are provided in the order of channel IDs.

Command:
`SET MULCH 1 0 1 1`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

The example above would result in the following light status
VIOLET - **ON**
BLUE - **OFF**
GREEN - **ON**
RED - **ON**

Answer:
`A MULCH`
if successful or
`E MULCH`
if error occurred

Get Multiple Light States (ON/OFF) (GET MULCH)

Obtains multiple light channel states with a single command. Channel ON or OFF states (1 or 0) are provided in the order of channel IDs.

Command:
`GET MULCH`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

Answer:
`A MULCH 1 1 0 1`
if successful or
`E MULCH`
if error occurred

Get Multiple Light TTL States (ON/OFF) (GET MULCHTTL)

Obtains multiple light channel TTL states with a single command. Channel TTL ON or OFF states (1 or 0) are provided in the order of channel IDs.

Command:
`GET MULCHTTL`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

Answer:
`A MULCHTTL 1 1 0 1`
if successful or
`E MULCHTTL`
if error occurred

Get Multiple Light Actual States (ON/OFF) (GET MULCHACT)

Obtains multiple light channel actual states with a single command. This command checks both TTL inputs and channel state switches. Channel ON or OFF states (1 or 0) are provided in the order of channel IDs.

Command:
`GET MULCHACT`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

Answer:
`A MULCHACT 1 1 0 1`
if successful or
`E MULCHACT`
if error occurred

Get Maximum Intensity Level (GET MAXINT)

Obtain maximum valid intensity that can be applied to a light channel.

Command:
`GET MAXINT`

Answer:
`A MAXINT 1000`

Set Light Intensity (SET CHINT)

Sets intensity for a given channel. Allowed values range from 0 to maximum intensity. Maximum intensity can be obtained with the command GET MAXINT, and is typically 1000. This command is independent from ON/OFF commands. We can set intensity regardless of whether the specific light is turned ON or OFF, and this setting will not affect the light state.

For example if GREEN light is OFF and we set its intensity to 50, it will not turn ON. The next time ON command is sent the green light will turn ON with 50 intensity. However, if the light is already ON, CHINT command will instantaneously change its intensity.

If we set light intensity to "0", this will effectively disable the channel, i.e. the light will be always OFF, regardless of whether it was set to ON or OFF.

Command:
`SET CHINT 2 120`
The above example sets channel 2 intensity to 12%
`SET CHINT 3 800`
The above example sets channel 3 intensity to 80%

Answer:
`A CHINT`
if successful or
`E CHINT`
if error occurred

Get Light Intensity (GET CHINT)

Obtains current intensity setting for a given channel. Maximum value for intensities can be obtained with the command "GET MAXINT".

Command:
`GET CHINT 2`
The above example requests channel 2 intensity

Answer:
`A CHINT 370`
(for light intensity at 37%)
if successful or
`E CHINT`
if error occurred

Set Multiple Light Intensities (SET MULCHINT)

Sets multiple light channel intensities with a single command. Channel intensities (0 to 1000) are provided in the order of channel IDs. If power control is engaged some of the intensity values might be ignored.

Command:
`SET MULCHINT 100 900 400 850`

Assuming the following channel order (based on IDs):
VIOLET, BLUE, GREEN, RED

The example above would result in the following light intensities expressed in percents:
VIOLET = 10%
BLUE = 90%
GREEN = 40%
RED = 85%

Answer:
`A MULCHINT`
if successful or
`E MULCHINT`
if error occurred

Get Multiple Light Intensities (GET MULCHINT)

Obtains multiple light channel current intensities with a single command. Channel intensities (0 to 1000) are provided in the order of channel IDs.

Command:
`GET MULCHINT`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

Answer:
`A MULCHINT 100 310 500 900`
if successful or
`E MULCHINT`
if error occurred

Set Multiple Light States (ON/OFF) & Intensities (SET MULCHPROP)

Turns multiple lights ON or OFF, and sets multiple light channel intensities with a single command. Channel ON or OFF settings (1 or 0), as well as channel intensities (0 to 1000) are provided in the order of channel IDs. Channel state values precede (ON/OFF settings for all channels) light intensity values.

Command:
`SET MULCHPROP 1 0 1 1 250 0 124 55`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

The example above would result in the following light status & intensity
VIOLET ON / INTENSITY 25%
BLUE OFF / INTENSITY 0%
GREEN ON / INTENSITY 12.4%
RED ON / INTENSITY 5.5%

Answer:
`A MULCHPROP`
if successful or
`E MULCHPROP`
if error occurred

Set Multiple Selected Light States (ON/OFF) & Intensities (SET MULCHPROPALT)

Turns multiple lights ON or OFF, and sets multiple non-consecutive light channel intensities with a single command. Channel ON or OFF settings (1 or 0), as well as channel intensities (0 to 1000) are provided along with the channel index. Channel state values precede (ON/OFF settings for all channels) light intensity values.

Command:
`SET MULCHPROPALT 0 1 250 3 0 55 2 1 124`

Assuming the following channel order (based on IDs): VIOLET, BLUE, GREEN, RED

The example above would result in the following light status & intensity:
VIOLET ON / INTENSITY 25%
GREEN ON / INTENSITY 12.4%
RED OFF / INTENSITY 5.5%

Answer:
`A MULCHPROPALT`
if successful or
`E MULCHPROPALT`
if error occurred

Get Light Power Level (GET CHPWR)

Obtains estimated power level for a given channel. If the crosstalk compensation is active, the power level is adjusted to eliminate crosstalk.

Command:
`GET CHPWR 2`
The above example requests channel 2 power

Answer:
`A CHPWR 337`
(returns power level of 337)
if successful or
`E CHPWR`
if error occurred

Get Multiple Light Power Levels (GET MULCHPWR)

Obtains multiple light channel estimated power levels with a single command. Power levels are provided in the order of channel IDs.

Command:
`GET MULCHPWR`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

Answer:
`A MULCHPWR 0 310 0 0`

if successful or
`E MULCHPWR`
if error occurred

Get Light Estimated Power Output (GET CHPWRWATTS)

Obtains estimated power output for a given channel (in mW). Power estimation is based on the power output calibration factor, power sensor reading, power sensor exposure, power sensor gain and crosstalk level. Estimation model assumes linear dependency.

Command:
`GET CHPWRWATTS 2`
The above example requests channel 2 power output

Answer:
`A CHPWRWATTS 158.5`
if successful or
`E CHPWRWATTS`
if error occurred

Get Multiple Light Estimated Power Outputs (GET MULCHPWRWATTS)

Obtains multiple light channel estimated power output (in mW) with a single command. Power estimation is based on the power output calibration factor, power sensor reading, power sensor exposure, power sensor gain and crosstalk level. Estimation model assumes linear dependency. Power outputs are provided in the order of channel IDs.

Command:
`GET MULCHPWRWATTS`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

Answer:
`A MULCHPWRWATTS 0 158.5 256.8 0`
if successful or
`E MULCHPWRWATTS`
if error occurred

Set Light Power Regulation Lock State (SET PWRLOCK)

Activate/deactivate power regulation for a specified light channel. Power regulator adjusts light intensity in order to keep power level at the referent value.

Command:
`SET PWRLOCK 3 1`
The above example turns on power regulation for channel 3, with the second argument value of "1". To turn it off use "0" as the second argument.

Answer:
`A PWRLOCK`

Get Light Power Regulation Lock State (GET PWRLOCK)

Returns the power regulator state (active = 1 / inactive = 0) for a selected light channel.

Command:
`GET PWRLOCK 3`
The above example requests power regulation lock state for channel 3

Answer:
`A PWRLOCK 1`

Set Multiple Light Power Regulation Lock States (SET MULPWRLOCK)

Activate/deactivate power regulation for all channels (per channel control) in a single command. Power regulator adjusts light intensity in order to keep power level at the referent value.

Command:
`SET MULPWRLOCK 1 0 1 1`

Assuming the following channel order (based on IDs)
VIOLET, BLUE, GREEN, RED

The example above would result in the following power regulation lock states
VIOLET - **ON**
BLUE - **OFF**
GREEN - **ON**
RED - **ON**

Answer:
`A MULPWRLOCK`

Get Multiple Light Power Regulation Lock States (GET MULPWRLOCK)

Returns the power regulator state (active = 1 / inactive = 0) for all light channels in a single command.

Command:
`GET MULPWRLOCK`

Answer:
`A MULPWRLOCK 1 0 1 1`

Set Light Power Reference (SET PWRREF)

Set power reference value (in mW) for the selected light channel. Power reference is independent of the regulation state for the specified channel. We can set power reference regardless of whether the specific light is regulated or not. Power reference value must be a positive number within the power range of the specified channel. If we pass zero or negative value as the power reference, power regulator will ignore the specified light channel.

Example command for setting power reference for channel 2:

Command:
`SET PWRREF 2 800`

Answer:
`A PWRREF`

Get Light Power Reference (GET PWRREF)

Returns the light channel power reference (in mW). Value -1 is returned if the power reference hasn't been defined for a specified light channel.

Command:
`GET PWRREF 2`
The above example requests power reference for channel 2

Answer:
`A PWRREF 800.0`

Set Multiple Light Power References (SET MULPWRREF)

Set power reference values (in mW) for all light channels in a single command. Power reference is independent of the channel regulation state. We can set power reference regardless of whether the specific light is regulated or not. For each light channel power reference value must be a positive number within the power range for the specific

channel. If we pass zero or negative value as the power reference, power regulator will ignore the specific light channel.

Example command for setting reference for a three channel system (Green, Red, Blue). It is important to observe correct ordering of channels. If in doubt, use GET CHMAP command to obtain correct channel order.

Command:
```
SET MULPWRREF 800 -1 750
```

Answer:
```
A MULPWRREF
```

Get Multiple Light Power References (GET MULPWRREF)

Returns power reference values (in mW) for all light channels in a single command. Value -1 is returned if the power reference hasn't been defined for a specific light channel.

Command:
```
GET MULPWRREF
```

Answer:
```
A MULPWRREF 800.0 -1 750.0
```

Get Average Power During Power Regulation (GET PWRAVG)

- Obsolete from **v4.3.0** / Reintroduced in **v4.5.4**

Returns the average power (in mW) for each channel during power regulation. If the channel is not regulated return value will be -1.

Command:
```
GET PWRAVG
```

Answer:
```
A PWRAVG 125.0 -1 64.5
```

Get Standard Deviation During Power Regulation (GET PWRDEV)

- Obsolete from **v4.3.0** / Reintroduced in **v4.5.4**

Returns the standard deviation for each channel during power regulation. If the channel is not regulated return value will be -1.

Command:

```
GET PWRDEV
```

Answer:
```
A PWRDEV 1.2 -1 3.1
```

Get Maximum Error Value During Power Regulation (GET PWRMAXERR)

- Obsolete from **v4.3.0** / Reintroduced in **v4.5.4**

Returns the maximum absolute error value (in mW) for each channel during power regulation. If the channel is not regulated return value will be -1.

Command:
```
GET PWRMAXERR
```

Answer:
```
A PWRMAXERR 3.2 -1 0.4
```

Set PID Mode (SET PIDMODE)

Activate / deactivate PID intensity mode. In PID intensity mode power output is controlled via channel intensity commands. PID intensity mode is not available for all light engine types. PID mode setting is persisted across light engine sessions.

Command:
```
SET PIDMODE 1
```

Answer:
```
A PIDMODE
```

Get PID Mode (GET PIDMODE)

Checks if PID intensity mode is active.

Command:
```
GET PIDMODE
```

Answer:
```
A PIDMODE 1
```

Multi-channel Modes

Multi-channel modes allow the user to control multiple sources with a single intensity input. Each input intensity step maps into an array of DAC (Digital to Analog Converter) outputs, one for each channel. DAC values are fed directly to the light source, determining the amount of electrical current, and subsequently the output power.

The DAC output for each channel is determined by the corresponding Lookup Table, which is defined by the user and can express an arbitrary The user can configure any number of modes, each consisting of a different set of channels. The LUT contents can express any relationship between intensity and DAC (output power). In this way the mode can be calibrated to conform to various constraints, e.g. CRI or Color Temperature specifications.

Create Multi-channel Mode (CREATE MULTIMODE)

Create a new Multi-channel mode. Provide the new mode name and a list of channels (each one defined by its zero-based index) that constitute the mode. Channels that are missing from the list will be turned off and not respond to intensity changes.

Upon creation the mode will be initialized with default LUTs, one for each specified channel. The default LUTs have the entire DAC range mapped to linearly increase with intensity.

Mode name is case insensitive, i.e. it will automatically converted to uppercase.

Command:

```
CREATE MULTIMODE [name] [channel] [channel] ...
```

Example:

```
CREATE MULTIMODE TEST1 0 1 4 5
A MULTIMODE
```

Delete Multi-channel Mode (DELETE MULTIMODE)

Deletes the specified Multi-channel mode and associated LUTs.

Command:

```
DELETE MULTIMODE [name]
```

Example:

```
DELETE MULTIMODE TEST1
A MULTIMODE
```

List all available Multi-channel Modes (GET MULTIMODES)

Returns a list of all available Multi-channel modes.

Command:
`GET MULTIMODES`

Example:
```
GET MULTIMODES
A MULTIMODES TEST1 TEST2 TEST3
```

Set Multi-channel Mode (SET MULTIMODE)

Applies the specified Multi-channel mode. The light engine will remain in this mode until the mode is changed again. The mode state is persistent, i.e. retained after the power cycle.

The light engine always maintains one special mode: STANDARD. This is the normal, non-modal state of the light engine, where each channel has its own intensity setting. We can always exit the current mode and return to standard light engine functionality by applying the STANDARD mode.

When it is in one of the Multi-channel modes, the light engine behaves as a single-channel light engine, with that single virtual channel being a composite of multiple channels. We can still separately control all channels with appropriate light engine commands, both the ones that belong to a mode and the ones that are not included. By sending individual commands to mode channels, we can disrupt integrity of a mode, i.e. the output won't correspond to what was prescribed by the mode lookup tables for a given mode intensity.

Command:
`SET MULTIMODE [name]`

Example:
```
SET MULTIMODE TEST1
A MULTIMODE
```

Get Multi-channel Mode (GET MULTIMODE)

Returns the current Multi-channel mode. The command will return STANDARD if we are in the normal operating mode (independent channels).

Command:
`GET MULTIMODE`

Example:

```
GET MULTIMODE
A MULTIMODE TEST1
```

Get channels corresponding to a given Multi-channel Mode (GET MULTIMODECHANNELS)

Returns a list of channels (channel indices) comprising the given mode.

Command:

```
GET MULTIMODECHANNELS [name]
```

Example:

```
GET MULTIMODECHANNELS TEST1
A MULTIMODECHANNELS 0 1 4 5
```

Set Multi-channel Mode Intensity (SET MULTIMODEINT)

Set the current Multi-channel mode intensity. This command works the same way as Set Light Intensity (SET CHINT) does for a single channel, except that the intensity will be mapped into the DACs (using LUTs) for multiple channels simultaneously.

Command:

```
SET MULTIMODEINT [intensity]
```

Example:

```
SET MULTIMODEINT 550
A MULTIMODEINT
```

Get Multi-channel Mode Intensity (GET MULTIMODEINT)

Returns the current Multi-channel mode intensity.

Command:

```
GET MULTIMODEINT
```

Example:

```
GET MULTIMODEINT
A MULTIMODEINT 550
```

Set Multi-channel Mode State: On/Off (SET MULTIMODESW)

Set the current Multi-channel mode switch on (1) or off (0). This command works the same as the Set Light State (ON/OFF) (SET CH), i.e. it turns the mode light output on or off.

Command:
`SET MULTIMODESW [ 1 or 0]`

Example:
```
SET MULTIMODESW 1
A MULTIMODESW
```

Get Multi-channel Mode State: On/Off (GET MULTIMODESW)

Get the on/off state of the current Multi-channel mode.

Command:
`GET MULTIMODESW`

Example:
```
GET MULTIMODSW
A MULTIMODESW 1
```

Set single channel intensity in Multi-channel Mode (SET MULTIMODESINGLEINT)

Set the single channel intensity while the engine is in multi-channel mode. This command works the same way as Set Light Intensity (SET CHINT), which is unavailable in multi-channel mode.

Command:
`SET MULTIMODESINGLEINT 2 120`
The above example sets channel 2 intensity to 12%

Answer:
`A MULTIMODESINGLEINT`

Set single channel state (ON/OFF) in Multi-channel Mode (SET MULTIMODESINGLESW)

Set the single channel switch ON (1) or OFF (0) while the engine is in multi-channel mode. This command works the same as the Set Light State (ON/OFF) (SET CH), which is unavailable in multi-channel mode.

Command:
`SET MULTIMODESINGLESW 2 1`
The above example sets channel 2 to ON

Answer:
`A MULTIMODESINGLESW`

Set Multi-channel Mode Power Regulation Lock State (SET MULTIMODEPWRLOCK)

- Introduced in **v4.5.5**

Activate/deactivate power regulation for the active multi-channel mode. Power regulator adjusts light intensity in order to keep power level at the referent value.

Command:
`SET MULTIMODEPWRLOCK 1`
The above example turns on power regulation, with argument value of “1”. To turn it off us “0” as the argument

Answer:
`A MULTIMODEPWRLOCK`

Get Multi-channel Mode Power Regulation Lock State (GET MULTIMODEPWRLOCK)

- Introduced in **v4.5.5**

Returns the power regulator state (active = 1 / inactive = 0) for the active multi-channel mode.

Command:
`GET MULTIMODEPWRLOCK`

Answer:
`A MULTIMODEPWRLOCK 1`

Set Multi-channel Mode Power Reference (SET MULTIMODEPWRREF)

- Introduced in **v4.5.5**

Set power reference value (relative to theoretical setpoint) for the active multi-channel mode. Power reference is independent of the regulation state. We can set power reference regardless of whether the multi-channel mode is regulated or not. Power reference value must be a positive number lower than 1.0. If we pass zero or negative value as the power reference, power regulator will disable regulation.

Command:
`SET MULTIMODEPWRREF 0.7`

Answer:
`A MULTIMODEPWRREF`

Get Multi-channel Mode Power Reference (GET MULTIMODEPWRREF)

- Introduced in **v4.5.5**

Returns the active multi-channel mode power reference (relative to theoretical setpoint). Value -1 is returned if the power reference hasn't been defined.

Command:
`GET MULTIMODEPWRREF`

Answer:
`A MULTIMODEPWRREF 0.7`

Set Multi-channel Mode LUT (SET MULTIMODELUT)

For a given mode [mode name] this command sets the DAC value [DAC value] for the specified channel [channel index] and specified intensity [intensity].
Command affects intensity LUTs for the primary output (output A if dual outputs are available).

Command:
`SET MULTIMODELUT [mode name] [channel index] [intensity] [DAC value]`

- [**mode name**] multi-channel mode
- [**channel index**] index of the channel (must be one of the channels selected when the mode was created)
- [**intensity**] the intensity step that this command applies to (LUT index), valid values are 0 - 1000
- [**DAC value**] the DAC value corresponding to the above intensity level, valid values are 0 - MAXDAC

MAXDAC value can be obtained with the GET MAXDAC command.

This command is in effect as soon as it is executed, but the data is not saved in the light engine until the LUT data is saved with SAVE MULTIMODELUT command. If we fail to call SAVE MULTIMODELUT command after we are done with setting up mode LUTs, the LUT data will not survive the power cycle.

Example:
```
SET MULTIMODELUT TEST1 0 21 435
A MULTIMODELUT
```

Get Multi-channel Mode LUT (GET MULTIMODELUT)

Return the DAC value for given [mode name] [channel index] and [intensity].

Command:

```
GET MULTIMODELUT [mode name] [channel index] [intensity]
```

- [**mode name**] multi-channel mode
- [**channel index**] index of the channel (must be one of the channels selected when the mode was created)
- [**intensity**] the intensity step that this command applies to (LUT index), valid values are 0 - 1000

Example:

```
GET MULTIMODELUT TEST1 0 21
A MULTIMODELUT 435
```

Set Multi-channel Mode LUT for the second output (SET MULTIMODELUTB)

- Introduced in **v4.5.3**

For a given mode [mode name] the command sets the DAC value [DAC value] for the specified channel [channel index] and specified intensity [intensity].
Command affects intensity LUTs for the second output (output B). For models without dual outputs this command will return error 61.

Command:

```
SET MULTIMODELUTB [mode name] [channel index] [intensity] [DAC value]
```

- [**mode name**] multi-channel mode
- [**channel index**] index of the channel (must be one of the channels selected when the mode was created)
- [**intensity**] the intensity step that this command applies to (LUT index), valid values are 0 - 1000
- [**DAC value**] the DAC value corresponding to the above intensity level, valid values are 0 - MAXDAC

MAXDAC value can be obtained with the GET MAXDAC command

This command is in effect as soon as it is executed, but the data is not saved in the light engine until the LUT data is saved with SAVE MULTIMODELUT command. If we fail to call SAVE MULTIMODELUT command after we are done with setting up mode LUTs, the LUT data will not survive the power cycle.

Example:
```
SET MULTIMODELUTB TEST1 0 21 435
A MULTIMODELUTB
```

Get Multi-channel Mode LUT for the second output (GET MULTIMODELUTB)

- Introduced in **v4.5.3**

Return the DAC value for given [mode name] [channel index] and [intensity]. Command queries intensity LUTs for the second output (output B). For models without dual outputs this command will return error 61.

Command:
```
GET MULTIMODELUTB [mode name] [channel index] [intensity]
```

- [**mode name**] multi-channel mode
- [**channel index**] index of the channel (must be one of the channels selected when the mode was created)
- [**intensity**] the intensity step that this command applies to (LUT index), valid values are 0 - 1000

Example:
```
GET MULTIMODELUTB TEST1 0 21
A MULTIMODELUTB 435
```

Save Multi-channel Mode LUT (SAVE MULTIMODELUT)

Saves all mode LUTs. After saving LUTs they become persistent and will be still in effect after powercycle.

Command:
```
SAVE MULTIMODELUT [mode name]
```

Example:
```
SAVE MULTIMODELUT TEST1
A MULTIMODELUT
```

Get Maximum DAC value (GET MAXDAC)

Get light channel DAC limit (DAC level for max light intensity).

Command:
```
GET MAXDAC [channel index]
```

Example:
```
GET MAXDAC 2
A MAXDAC 3080
```

Get Channel DAC value (GET CHDAC)

Get light channel DAC value for the current intensity. This applies to the intrinsic (standard mode) relationship between intensity and DAC. Not to be confused with mode LUTs and DAC values.

Command:
```
GET CHDAC [channel index]
```

Example:
```
GET CHDAC 2
A CHDAC 1020
```

Error Codes

CODE	DESCRIPTION
0	OK (no error)
41	Invalid I2C bus
42	Invalid I2C slave (device) address
43	I2C bus write error
44	I2C bus read error
45	SPI bus write error
46	SPI bus read error
47	GPIO set state error
48	GPIO get state error
49	Analog input sampling error
51	Invalid light channel index
52	Invalid command format (syntax)

53	Unknown command
55	Invalid command argument (invalid argument value or type)
56	Hardware component unavailable / Hardware configuration error
57x	Channel locked, because one of the following occurred:
572	Fan malfunction
571	Max temperature was exceeded
573	Interlock activated (573)
574	Power supply current limit exceeded (574)
58	System is busy (long running operation)
59	Set intensity command failed because one of the channels is under PID
60	Interlock active
61	Feature unavailable
62	Power supply is overloaded
63	Predictive power limiter blocked the command due to projected power
64	TEC is warming up or failed
65	Temperature and humidity exceed the allowed range (dewpoint)
66	Permanent storage error (eMMC)
67	Invalid system configuration
68	Invalid app configuration
69	Invalid serial interface configuration (both ports in legacy mode)
70	Unauthorized access
71	Power level exceeds the power limit (power reference clipped)
72	Power regulation unavailable for multiple channels on the same power
73	Light engine no longer supports the specified command
74	TEC warming up
75	Unable to reach stable DAC level during MAXDAC search
76	PID mode unavailable when engine is in factory mode

77	PID mode unavailable for this light engine type
78	Shutter closed
79	Command disabled in the current operating mode

Channel status codes

CODE	DESCRIPTION
0	Channel OK
51	Invalid channel index
56	Invalid hardware configuration
57x	Channel locked, because one of the following occurred:
572	Fan malfunction
571	Max temperature was exceeded
573	Interlock activated
574	Power supply current limit exceeded
58	Channel busy (long running operation)
60	Interlock active
64	TEC loss of lock
65	TEC temperature out of range (dew point reached)

REST Interface

Light Engine REST interface follows the following format:

```
http://[SERVERIP]/service/?command=[COMMANDTEXT]
```

SERVERIP - IP address
COMMANDTEXT - command text, as defined in above in command reference

Engine responds with JSON encoded string containing two fields "`status`" and "`message`". Status will be always empty (its value is reserved for future use and should be ignored), and the message will contain the answer as defined in the command reference above.

Here are a couple of examples that we can simply type in your browser URL field to test how Light Engine responds to commands. The same can be done from any program capable of addressing URL interface.

Example: GET IP
Trivial example of confirming the IP address

```
http://192.168.3.9/service/?command=GET IP
```

response:
```
{ "status": "", "message": "A IP 192.168.3.9" }
```

Example: GET CHMAP
Obtain channel names as shown and ordered in the GUI
```
http://192.168.3.9/service/?command=GET CHMAP
```
response:
```
{ "status": "", "message": "A CHMAP 6-CYAN 5-TEAL 1-YELLOW 4-RED 3-GREEN 7-BLUE 8-VIOLET 2-NIR" }
```

When we type commands as shown above in the browser URL field, browser will automatically replace each space by "`%20`" sequence. That is required for standard "UTF-8" URL encoding. Web browser does that automatically for us, but inside the program we might have to do UTF-8 encoding explicitly, depending on the programming environment.