MXA-MUTE

Command Strings

Shure MXA Network Mute Button command strings for third-party control systems, such as AMX, Crestron, or Extron. Includes all supported programming commands.

Version: 1.2 (2023-C)

Table of Contents

MXA-MUTE
Command Strings

Using a Third-Party Control System

This device can be controlled using a third-party control system with the appropriate command string.

Common applications:

- Mute
- LED color and behavior
- Loading presets
- Adjusting levels

The device is connected via Ethernet to a control system, such as AMX, Crestron or Extron.

- **Connection:** Ethernet (TCP/IP; select “Client” in the AMX/Crestron program)
- **Port:** 2202

If using static IP addresses, set the Shure Control and the Audio Network settings to Manual in Designer. Use the Control IP address for TCP/IP communication with Shure devices.

See below for all supported command strings. This list is updated with each firmware release.

Command String Conventions

When you make changes to a parameter, the device sends a REPORT string with information about what you changed. You don't need to constantly query parameters.

All messages are ASCII, including level and gain indicators.

This device uses 4 types of strings:

- **GET**
  - Finds the status of a parameter. The device responds with a REPORT string.
- **SET**
  - Changes the status of a parameter. The device responds with a REPORT string that shows the parameter's new value.
- **REP**
  - The device sends REPORT strings to show the status of parameters anytime a parameter changes.
- **SAMPLE**
  - Used for metering audio levels.

Get All

Parameter Name:	ALL
Command Types Supported:	GET, REP
Indexing:	n/a

Value(s):	Responds with `REP` for all device-specific properties and `ALL` channel-related properties.
Example(s):	< GET ALL >

Model

Parameter Name:	MODEL
Command Types Supported:	GET, REP
Indexing:	n/a
Value(s):	**model** is a 32 character quoted string. The value is padded with spaces to ensure that 32 characters are reported.
Example(s):	< GET MODEL > < REP MODEL model >

Serial Number

Parameter Name:	SERIAL_NUM
Command Types Supported:	GET, REP
Indexing:	n/a
Value(s):	**serial_num** is a 32 alphanumeric character string. Response is padded to ensure that 32 characters are always returned
Example(s):	< GET SERIAL_NUM > < REP SERIAL_NUM serial_num >

Firmware Version

Parameter Name:	FW_VER
Command Types Supported:	GET, REP
Indexing:	n/a

Value(s):	Where **ver** is an 18 character literal string: The value is 3 versions separated by a period. Each version shall be able to take on a value from 0 to 65535. **ver** has an "*****" if the firmware is invalid. **Example**: 65535.65535.65535
Example(s):	< GET FW_VER > < REP FW_VER ver >

Control MAC Address

Parameter Name:	CONTROL_MAC_ADDR
Command Types Supported:	GET, REP
Indexing:	n/a
Value(s):	**addr** is a 17 character literal string formatted as 6 octets, each separated by a colon. **Example:** 00:0E:DD:FF:F1:63
Example(s):	< GET CONTROL_MAC_ADDR > < REP CONTROL_MAC_ADDR addr > < REP ERR >

Device ID

Parameter Name:	DEVICE_ID
Command Types Supported:	GET, REP
Indexing:	n/a
Value(s):	Response is a text string. Most devices allow device ID to be up to 31 characters. Value is padded with spaces as needed to ensure that 31 characters are always reported

Example(s):	< GET DEVICE_ID > < REP DEVICE_ID string >

Identify Device (Flash LED)

Parameter Name:	FLASH
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**flash_state** takes on values ON OFF
Example(s):	< GET FLASH > < SET FLASH flash_state > < REP FLASH flash_state > < REP ERR >

Presets

Parameter Name:	PRESET
Command Types Supported:	GET, SET, REP
Indexing:	## is the preset number and takes on values 1-10.
Value(s):	n/a
Example(s):	< GET PRESET > < SET PRESET ## > < REP PRESET ## > < REP ERR >

Restore Default Settings

Parameter Name:	DEFAULT_SETTINGS

Command Types Supported:	SET, REP
Indexing:	n/a
Value(s):	## = 00 if restore is successful
Example(s):	< SET DEFAULT_SETTINGS > < REP DEFAULT_SETTINGS ## > < REP ERR >

View Preset Name

Parameter Name:	PRESET_NAME
Command Types Supported:	GET, REP
Indexing:	1-10: specific preset identifier
Value(s):	**name** is a literal string 25 alphanumeric characters long, special characters allowed except blank spaces, {} and < >. Note that if a preset is empty, **name** will say {empty}
Example(s):	< GET PRESET_NAME nn > < REP PRESET_NAME nn name > < REP ERR >

Reboot

Note: This command does not send acknowledgement.

Parameter Name:	REBOOT
Command Types Supported:	SET
Indexing:	n/a
Value(s):	n/a
Example(s):	< SET REBOOT >

Get Error Events

Parameter Name:	LAST_ERROR_EVENT

Command Types Supported:	GET, REP
Indexing:	n/a
Value(s):	Sends the last error logged on the device, as represented by **{str}. {str}** is up to 128 characters long.
Example(s):	< GET LAST_ERROR_EVENT > < REP LAST_ERROR_EVENT {str} > < REP ERR >

Mute LED State

Parameter Name:	DEV_MUTE_STATUS_LED_STATE
Command Types Supported:	GET, REP
Indexing:	n/a
Value(s):	**sts** is current mute LED state that takes on these values: ON = MUTED OFF = UNMUTED
Example(s):	< GET DEV_MUTE_STATUS_LED_STATE > < REP DEV_MUTE_STATUS_LED_STATE sts > < REP ERR >

LED Brightness

Parameter Name:	LED_BRIGHTNESS
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**level** is the desired brightness level and takes on values: 0: Disabled 1: 20% 2: 40% 3: 60% 4: 80% 5: 100%

Example(s):	< GET LED_BRIGHTNESS > < SET LED_BRIGHTNESS level > < REP LED_BRIGHTNESS level > < REP ERR >

LED Mute Indication

Parameter Name:	LED_COLOR_UNMUTED
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**color:** RED, ORANGE, GOLD, YELLOW, YELLOWGREEN, GREEN, TURQUOISE, POWDERBLUE, CYAN, SKYBLUE, BLUE, PURPLE, LIGHTPURPLE, VIOLET, ORCHID, PINK, WHITE
Example(s):	< GET LED_COLOR_UNMUTED > < SET LED_COLOR_UNMUTED color > < REP LED_COLOR_UNMUTED color > < REP ERR >

LED Color Muted

Parameter Name:	LED_COLOR_MUTED
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**color:** RED, ORANGE, GOLD, YELLOW, YELLOWGREEN, GREEN, TURQUOISE, POWDERBLUE, CYAN, SKYBLUE, BLUE, PURPLE, LIGHTPURPLE, VIOLET, ORCHID, PINK, WHITE
Example(s):	< GET LED_COLOR_MUTED > < SET LED_COLOR_MUTED color > < REP LED_COLOR_MUTED color > < REP ERR >

LED State Muted

Parameter Name:	LED_STATE_MUTED

Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**state**: ON, FLASHING, OFF
Example(s):	< GET LED_STATE_MUTED > < SET LED_STATE_MUTED state > < REP LED_STATE_MUTED state > < REP ERR >

LED State Unmuted

Parameter Name:	LED_STATE_UNMUTED
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**state**: ON, FLASHING, OFF
Example(s):	< GET LED_STATE_UNMUTED > < SET LED_STATE_UNMUTED state > < REP LED_STATE_UNMUTED state > < REP ERR >

Device LED In State

Parameter Name:	DEV_LED_IN_STATE
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**sts** indicates device's LED-In state: 1. OFF = Mute 2. ON = Unmute
Example(s):	< GET DEV_LED_IN_STATE > < SET DEV_LED_IN_STATE sts > < REP DEV_LED_IN_STATE sts > < REP ERR >

Mute Button Status

Parameter Name:	MUTE_BUTTON_STATUS
Command Types Supported:	GET, REP
Indexing:	n/a
Value(s):	**sts** is current mute button press status and takes on values: ON, OFF, or UNKNOWN
Example(s):	< GET MUTE_BUTTON_STATUS > < REP MUTE_BUTTON_STATUS sts > < REP ERR >

Device Switch Out State

Parameter Name:	EXT_SWITCH_OUT_STATE
Command Types Supported:	GET, REP
Indexing:	n/a
Value(s):	**sts** indicates device's switch out state: 1. Off = Mute 2. On = Unmute
Example(s):	< GET EXT_SWITCH_OUT_STATE > < REP EXT_SWITCH_OUT_STATE sts > < REP ERR >

Mute Control Function

Parameter Name:	MUTE_CONTROL_FUNC
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**sts** indicates device's mute control setting: ENABLED DISABLED

Example(s):	< GET MUTE_CONTROL_FUNC > < SET MUTE_CONTROL_FUNC sts > < REP MUTE_CONTROL_FUNC sts > < REP ERR >

Mute Control Mode

Parameter Name:	MUTE_CONTROL_MODE
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**mode** is: TOG: Toggle PTT: Push-to-talk PTM: Push-to-mute
Example(s):	< GET MUTE_CONTROL_MODE > < SET MUTE_CONTROL_MODE mode > < REP MUTE_CONTROL_MODE mode > < REP ERR >

Default Toggle State

Parameter Name:	DEFAULT_TOGGLE_STATE
Command Types Supported:	GET, SET, REP
Indexing:	n/a
Value(s):	**state** is: Muted Unmuted
Example(s):	< GET DEFAULT_TOGGLE_STATE > < SET DEFAULT_TOGGLE_STATE state > < REP DEFAULT_TOGGLE_STATE state > < REP ERR >

Using PuTTY and Other Telnet Clients

For all Telnet clients (including PuTTY), set Telnet negotiation to disabled or passive mode. Active Telnet negotiation is not supported by MXA devices.

If using PuTTY to enter commands for MXA devices, the first command you send may return an error. To fix, enter the command again and it should work normally.