**EVO-IP: Guide to ASCII Command Structure and Communication**

EVO-IP has evolved and has taken on additional functionality based on the feedback of our customers. For control box firmware 2.05.27 and later, there are ASCII commands that can be more easily used for custom programming and applications. Follow and refer to the information below. Please note that this requires a specific naming scheme for all EVO-IP transmitters and receivers, or EVOIPLITE devices, and not all functionality is listed able to be performed by EVOIPLITE, including CEC, OSD, and Video Wall.

**Table of Contents:**

1.1 How to Connect, Command Format, and List of Supported Commands pgs. 1-3

2.1 Matrix Command Definition pgs. 3-8

3.1 Video Wall Command Definition pgs. 9-11

4.1 OSD Command Definition pgs. 12-14

5.1 Schedule Command Definition pgs. 14-15

6.1 Event Command Definition pgs. 15-17

7.1 Access Command Definition pgs. 18-19

8.1 Setup Command Definition pgs. 19-25

#### 1.1 How to Connect

Use port no. **55688** to create a TCP connection and read/write data. Please write to this port to send commands in the following string format and read from this port to get the response after sending the request.

#### 1.2 General Command Format

1.2.1 Command format

**The following commands will be all described in string, but transmitted to the control box in ASCII.**

The command will have following prefix:

1. “C1” means command number 1
2. “R1” means command number 1 success and response normally
3. “E1” means command number 1 fail and response with error

The command after the prefix will be ‘[some messages]’ and content in the ‘[ ]’ is the command’s parameter. The details of each command will be listed as follows. All of these commands should be parsed to byte in ASCII and transmitted to the TCP port. The control box will return the response through the port as well.

The command to get information or setting from the control box will return the data in JSON format which is the same as the API found at <http://api.evo-ip.io>.

#### 1.3 Error response

When the control box encounters an error, it will send out this type of response and detail the error.

Format:

*E${command number}[${error message}]* Description:

The ‘E1’ means an error responding to command 1. And the message in the “[ ]” is the error message

Example:

*E1[can’t find the device named INPUT001]*

*E2[can’t find the videowall named VW02]*

#### 1.4 Supported command list

* Matrix
	1. [C1] Map an input to an output

○ [C2] Get the extender data

○ [C3] Get the extender current setting

○ [C4] Get the misc setting

○ [C5] Save current extender mapping to the preset

○ [C6] Load the preset as the current extender mapping

○ [C7] Switch the rx client source

○ [C8] Switch the CEC value of extender

○ [C9] Mute the extender HDMI

* Videowall
	1. [C10] Set videoWall config active or deactive

○ [C11] Add a new extender videowall config

○ [C12] Delete a videowall config

○ [C13] Enable/disable all extender's OSD config in a videowall

* OSD
	1. [C14] Set osd mapping to rx

○ [C15] Set the auto OSD config

○ [C16] Set the extender resolution

* Schedule
	1. [C17] Set the extender schedule
* Event
	1. [C18] Set the extender event

○ [C19] Set the event email

○ [C20] Set the event SMS

* Access
	1. [C21] Set the extender access config

○ [C22] Set the misc setting

* Setup
	1. [C23] Set extender LED flash

○ [C24] Set extender device name

○ [C25] Set upScale mode of extender

○ [C26] Set the video out of tx extender

○ [C27] Set extender reboot

○ [C28] Set extender factory default

○ [C29] Set extender hdmi reconnect

○ [C30] Set extender edid mode

○ [C31] Set extender rs232 baudRate ○ [C32] Set extender rotate

# Matrix Command Definition

#### 2.1 C1

###### 2.1.1 Request

C1 is used to map an INPUT to an OUTPUT.

Users need to name the transmitters to INPUT001, INPUT002, INPUT003…. And name the receivers to OUTPUT001, OUTPUT002…. With the prefix and 3 digits number, users can send the simple command to map the input to output easily.

If the control box can’t find the assigned input or output, an error message will be returned.

================================================================ Format:

*C1[${3 digits input number without the prefix ‘INPUT’}-${3 digits output number without the prefix ‘OUTPUT’}]* Description:

“C1” means the command 1, the content between the ‘[]’ will be the command parameters. Please note there is “-” between 2 numbers, the first number is the input number, the second is the output number.

Example:

*C1[001-073]*

###### 2.1.2 Response

The response of C1 will have prefix “R1” to response the command Format:

*R1[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R1[applied successfully]*

###### 2.1.3 Error

The possible error message might be following

1. The INPUT001 is not found online
2. The OUTPUT073 is not found online

#### 2.2 C2

###### 2.2.1 Request

C2 is used get the extender’s data

Enter the extender’s name, case-sensitive, as a parameter to get the extender’s data.

The response will be the same content as the REST-API response in json format. ================================================================ Format:

*C2[${extender’s name}]* Description:

‘C2’ means command 2. Extender’s name is used to tell the control box which extender you want to query. Enter ‘all’ as the extender’s name with no extender named ‘all’ will return all extenders’ data.

Example:

*C1[INPUT001]*

###### 2.2.2 Response

The response of C2 will have prefix “R2” to response the command

Format:

*R2[${response message}]* Description:

The response will be an array in the json format as the API document describes.

Example:

Please refer to the [API Documentation](https://api.evo-ip.io/#/Matrix/GetExtenderLists)

###### 2.2.3 Error

The possible error message might be following

 1. INPUT001 is not online

#### 2.3 C3

###### 2.3.1 Request

C3 is used to get all current setting

================================================================ Format:

*C3*

Description:

‘C3’ returns all extender settings. Example:

*C3*

###### 2.3.2 Response

The response of C3 will have prefix “R3” to response the command

Format:

*R3[${response message}]* Description:

The response will be the content the same as the API Document describes.

Example:

Please refer to the [API Documentation](https://api.evo-ip.io/#/Matrix/GetExtenderSetting)

###### 2.3.3 Error

The possible error message might be following

 1. Temporarily unavailable

#### 2.4 C4

###### 2.4.1 Request

C4 is used to get the misc setting

================================================================ Format:

*C4*

Description:

This command needs no parameters to get the misc setting Example:

*C4*

###### 2.4.2 Response

The response of C4 will have prefix “R4” to response the command

Format:

*R4[${response message}]* Description:

The response will be the content the same as the API Document describes.

Example:

Please refer to the [API Documentation](https://api.evo-ip.io/#/Matrix/GetMiscSetting)

###### 2.4.3 Error

The possible error message might be following

 1. temporarily unavailable

#### 2.5 C5

###### 2.5.1 Request

C5 is used to save the current mapping to the preset

================================================================ Format:

*C5[${Preset number| 1-10}]* Description:

The parameter is the number of the Preset Example:

*C5[1]*

###### 2.5.2 Response

The response of C5 will have prefix “R5” to response the command

Format:

*R5[${response message}]* Description:

The response will be the ‘applied successfully’ Example:

R5[applied successfully]

###### 2.5.3 Error

The possible error message might be following

 1. temporarily unavailable

#### 2.6 C6

###### 2.6.1 Request

C6 is used to load the assigned Preset to the current mapping

================================================================ Format:

*C6[${Preset number| 1-10}]* Description:

The parameter is the number of the Preset Example:

*C6[1]*

###### 2.6.2 Response

The response of C6 will have prefix “R6” to response the command

Format:

*R6[${response message}]* Description:

The response will be the ‘applied successfully’ Example:

R6[applied successfully]

###### 2.6.3 Error

The possible error message might be following

 1. temporarily unavailable

#### 2.7 C7

###### 2.7.1 Request

C7 is used to set the rx client source

The extender’s name should be unique and case sensitive.

================================================================ Format:

*C7[${extender’s name}-${client source | ip/ local}]* Description:

The parameter is ‘ip’ or ‘local’ Example:

*C7[OUTPUT007-ip]*

*C7[smallTV-local]*

###### 2.7.2 Response

The response of C7 will have prefix “R7” to response the command

Format:

*R7[${response message}]* Description:

The response will be the ‘applied successfully’ Example:

R7[applied successfully]

###### 2.7.3 Error

The possible error message might be following

1. temporarily unavailable
2. the device is not online
3. the device is not a receiver

#### 2.8 C8

###### 2.8.1 Request

C8 is used to switch the CEC value of the extender.

Please enable the CEC function in C22 before setting the extender’s CEC function

And the EVOIP-LITE does not support CEC function

================================================================ Format:

*C8[${extender’s name}-${CEC enabled | true/false}]* Description:

Example:

*C8[INPUT001-true]*

###### 2.8.2 Response

The response of C8 will have prefix “R8” to response the command

Format:

*R8[${response message}]* Description:

The response will be the ‘applied successfully’ Example:

R8[applied successfully]

###### 2.8.3 Error

The possible error message might be following

1. temporarily unavailable
2. the device is not online
3. the device is not a receiver
4. the global CEC function is disabled

#### 2.9 C9

###### 2.9.1 Request

C9 is used to mute the extender

The extender’s name should be unique to the control box.

================================================================ Format:

*C9[${extender name }-${mute | true/false}]* Description:

The parameters are the extender's name and a boolean to mute/unmute the extender. The extender’s name is case sensitive.

Example:

*C9[INPUT002-true]*

*C9[bigTV-false]*

###### 2.9.2 Response

The response of C9 will have prefix “R9” to response the command

Format:

*R9[${response message}]* Description:

The response will be the ‘applied successfully’ Example:

R9[applied successfully]

###### 2.9.3 Error

The possible error message might be following

1. temporarily unavailable
2. the device is not online

# Videowall Command Definition

#### 3.1 C10

###### 3.1.1 Request

C10 is used to activate/deactivate the videowall

Users need to name the videowall with the prefix VW and 2 digit number. For example,

VW01, VW02…

If the control box can’t find the assigned videowall name, an error will be returned.

================================================================ Format:

*C10[${2 digits videowall number}****-****${activate| true/false}]* Description:

The content in the “[]” will be the videowall number and the state user wants to set this videowall Example:

*C10[01:true]*

*C10[03:false]*

###### 3.1.2 Response

The response of C10 will have prefix “R10” to response the command

Format:

*R10[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R10[applied successfully]*

###### 3.1.3 Error

The possible error message might be following

 1. The videowall is not found

#### 3.2 C11

###### 3.2.1 Request

C11 is used to add a new videowall

User needs to enter a complete videowall configuration to add a new videowall. The parameters will be the json string of the field ‘params’ described in the [API Document](https://api.evo-ip.io/#/VideoWall/AddExtenderVideoWallConfig). And the amount limit of the videowall is 8 and the videowall name can not be duplicated.

If the configuration entered is invalid, the control box will return an error.

================================================================ Format:

*C11[${whole config of a new videowall in json format}]* Description:

The content is the json format of the whole videowall.

Example:

*C11[ {"active": false,"activeIndex": -1,"id": 1,"layout": {"columns": 3,"rows": 2},"name":*

*"VW7387","osd": false,"mapping": [{"index": 1,"rxName": "string","txName":*

*"string","column": 0,"row": 0,"groupId": 0,"adjustment": {"up": 0,"down": 0,"right":*

*0,"left": 0,"hScale": 0,"vScale": 0},"pos": {"alignment": "topLeft","offsetX": 0,"offsetY": 0,"height": 0,"width": 0}}]}]*

###### 3.2.2 Response

The response of C11 will have prefix “R11” to response the command

Format:

*R11[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R11[applied successfully]*

###### 3.2.3 Error

The possible error message might be following

1. Duplicated name
2. The amount limit of the videowall is 8
3. Invalid json format.
4. Invalid parameter
5. The name of the videowall is invalid, only characters or ‘\_’ or ‘-’ or numbers allowed and the length is 8.

#### 3.3 C12

###### 3.3.1 Request

C12 is used to delete an existed videowall

User needs to enter a videowall name to delete the videowall configuration.

If the videowall name entered does not exist, the control box will return error

================================================================ Format:

*C12[${videowall name}]* Description:

The content is the name of the videowall Example:

*C12[vw01]*

###### 3.3.2 Response

The response of C12 will have prefix “R12” to response the command

Format:

*R12[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R12[applied successfully]*

###### 3.3.3 Error

The possible error message might be following

 1. This videowall doesn’t exist

#### 3.4 C13

###### 3.4.1 Request

C13 is used to enable/disable the OSD of all extenders inside a videowall

User needs to enter the videowall’s name and enable/disable to operate

If the videowall name entered does not exist, the control box will return error

================================================================ Format:

*C13[${videowall name}-${enable | true/false}]* Description:

The content will be the videowall’s name and a boolean value to enable/disable

OSD

Example:

*C13[VW01-true]*

*C13[VW02-false]*

###### 3.4.2 Response

The response of C13 will have prefix “R13” to response the command

Format:

*R13[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R13[applied successfully]*

###### 3.4.3 Error

The possible error message might be following

 1. This videowall doesn’t exist

# OSD Command Definition

#### 4.1 C14

###### 4.1.1 Request

C14 is used to map an OSD to a receiver

User needs to name the OSD to OSD001, OSD002…. and name the receiver as OUTPUT001, OUTPUT002… to map the OSD to the receiver. Just like the command C1.

================================================================ Format:

*C14[${3 digits OSD number without the prefix ‘OSD’}****-****${3 digits output number without the prefix ‘OUTPUT’’}]* Description:

The content between the ‘[]’ will be the command parameters. Please note there is “:” between 2 numbers, the first number is the OSDnumber, the second is the output number.

Example:

*C14[001:073]*

###### 4.1.2 Response

The response of C14 will have prefix “R14” to response the command

Format:

*R14[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R14[applied successfully]*

###### 4.1.3 Error

The possible error message might be following

1. The OSD001 doesn’t exist
2. The OUTPUT 073 doesn’t exist

#### 4.2 C15

###### 4.2.1 Request

C15 is used to apply the auto OSD setting

User needs to enter a complete OSD auto configuration to apply. The parameters will be the json string of the field ‘params’ described in the [API Document](https://api.evo-ip.io/#/OSD/SetExtenderAutoOSDConfig).

================================================================ Format:

*C15[${whole config of an OSD auto config in json format}]* Description:

The content is the config of OSD auto config Example:

*C15[{"setting": {"backgroudColor": 0,"textColor": "#ff0000","timeout":*

*5,"transparency": 0},"status": {"accessQRCode": false,"groupId": false,"groupIpAddress": false,"groupName": false,"rxIpAddress": false,"rxMacAddress": false,"rxName": false}}]*

###### 4.2.2 Response

The response of C15 will have prefix “R15” to response the command

Format:

*R15[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R15[applied successfully]*

###### 4.2.3 Error

The possible error message might be following

 1. Invalid json format

#### 4.3 C16

###### 4.3.1 Request

C16 is used to change the receiver’s resolution

User needs to enter the receiver’s name and the string described in the [API Document](https://api.evo-ip.io/#/OSD/SetExtenderResolutionCommand) to change the receiver’s resolution

================================================================ Format:

*C16[${receiver's name}-${valid string described in the API Document}]* Description:

The string is described in the model section of the API Example:

*C16[OUTPU001-1920x1080@60]*

###### 4.3.2 Response

The response of C16 will have prefix “R16” to response the command

Format:

*R16[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R16[applied successfully]*

###### 4.3.3 Error

The possible error message might be following

1. Invalid resolution
2. The device is not online

#### 4.4 C33

###### 4.4.1 Request

C33 is used to mute/unmute the receiver’s OSD

================================================================ Format:

*C33[${receiver's name}-${mute, true|false}]* Description:

Example:

*C33[OUTPU001-true]*

###### 4.4.2 Response

The response of C33 will have prefix “R33” to response the command

Format:

*R33[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R33[applied successfully]*

###### 4.4.3 Error

The possible error message might be following

1. The device is not online
2. This device is not a receiver

# Schedule Command Definition

#### 5.1 C17

###### 5.1.1 Request

C17 is used to add/delete the schedule configuration

User needs to enter the field ‘params’ described in the [API Document](https://api.evo-ip.io/#/Schedule/SetExtenderScheduleConfig) without ‘rawIndex’ to apply the schedule configuration.

================================================================ Format:

*C17[add-${The parameter described in the API Document}]*

*C17[delete-${The schedule’s index to delete}]* Description:

This command is used to add a new schedule. Or delete an existing schedule with the index.

Example:

*C17[add-{"deviceType": "TX","deviceName": "txA","startDate":*

*1520837143,"endDate": 1520837143,"dayOfTheWeek": 2,"hour": 8,"minute": 21,"action": {"name": "reboot","key": "","duration": 20}}]* C17[delete-1]

###### 5.1.2 Response

The response of C17 will have prefix “R17” to response the command

Format:

*R17[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R17[applied successfully]*

###### 5.1.3 Error

The possible error message might be following

1. The amount limit of the schedules is 80
2. The extender doesn’t exist
3. This schedule is duplicated
4. The schedule index doesn’t exist

# Event Command Definition

#### 6.1 C18

###### 6.1.1 Request

C18 is used to add/delete extender’s event configuration

User needs to enter the field ‘params’ described in the [API Document](https://api.evo-ip.io/#/Event/SetExtenderEventsConfig) to apply the event configuration.

================================================================ Format:

*C18[add-${The parameter described in the API Document}]*

*C18[delete-${extenderName}-${eventName videoLost|connectionLost}]* Description:

Users can only map 1 action to 1 event at the same time, otherwise will get error Example:

*C18[add-{"deviceType": "TX","deviceName": "txB","event": "videoLost","actions":*

*[{"duration": 0,"key": "new\_group\_0;new\_group\_1","name": "email"}]}]*

*C18[delete-txB-videoLost]*

###### 6.1.2 Response

The response of C18 will have prefix “R18” to response the command

Format:

*R18[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R18[applied successfully]*

###### 6.1.3 Error

The possible error message might be following

1. The limit of the event setting is 80
2. Invalid json format
3. Invalid parameter
4. Duplicated event on the same extender
5. The extender does not exist

#### 6.2 C19

###### 6.2.1 Request

C19 is used to add/edit/delete the email groups

User needs to enter a param object

*{"groupName": "group1","receivers":"test123@gmail.com;walter123@gmail.com"}* to apply the email operation.

Detail is described in the [API Document](https://api.evo-ip.io/#/Event/SetExtenderEmailConfig)

================================================================ Format:

*C19[add}-${Param object}]*

*C19[edit-${idx to edit}-${Param object}]*

*C19[delete-${groupName to delete}]* Description:

Users need to enter valid email addresses to the email groups, and the emails should be separated by the ‘;’ if there are multiple receivers.

Example:

*C19[add-{"groupName":*

*"group1","receivers":"test123@gmail.com;walter123@gmail.com"}] C19[edit-3-{"groupName":*

*"group1","receivers":"test123@gmail.com;walter123@gmail.com"}] C19[delete-group1]*

###### 6.2.2 Response

The response of C19 will have prefix “R19” to response the command

Format:

*R19[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R19[applied successfully]*

###### 6.2.3 Error

The possible error message might be following

1. Invalid email address
2. Invalid json format

#### 6.3 C20

###### 6.3.1 Request

C20 is used to add/edit/delete the SMS groups

User needs to enter a param object

*{"groupName": "group1","receivers":"8860972201633;8860912345678"}* to apply the sms operation.

Detail is described in the [API Document](https://api.evo-ip.io/#/Event/SetExtenderSmsConfig)

================================================================ Format:

*C19[add}-${Param object}]*

*C19[edit-${idx to edit}-${Param object}]*

*C19[delete-${groupName to delete}]* Description:

Users need to enter valid sms phone numbers to the smsgroups, and the number should be separated by the ‘;’ if there are multiple receivers.

Example:

*C19[add-{"groupName":*

*"group1","receivers":"8860972201633;8860912345678"}] C19[edit-3-{"groupName":*

*"group1","receivers":"8860972201633;8860912345678"}] C19[delete-group1]*

###### 6.3.2 Response

The response of C20 will have prefix “R20” to response the command

Format:

*R20[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R20[applied successfully]*

###### 6.3.3 Error

The possible error message might be following

1. Invalid phone number
2. Invalid json format

# Access Command Definition

#### 7.1 C21

###### 7.1.1 Request

C21 is used to revise the access configuration

The access configuration is automatically generated based on the amount of the receiver. Users just need to edit it.

================================================================ Format:

*C21[${The parameter described in the API Document}]* Description:

The ‘deviceName’ means the receiver you want to revise access configuration.

Example:

*C21[{"account": "t8z85x","deviceName": "RX\_82231C88A77B","deviceType": "RX","password": "cr8i5l","rules": ["bigTV"]}]*

###### 7.1.2 Response

The response of C21 will have prefix “R21” to response the command

Format:

*R21[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R21[applied successfully]*

###### 7.1.3 Error

The possible error message might be following

1. Invalid json format
2. The transmitter doesn’t exist
3. The receiver doesn’t exist

#### 7.2 C22

###### 7.2.1 Request

C22 is used to set the misc setting

User needs to enter the field ‘params’ described in the [API Document](https://api.evo-ip.io/#/Access/SetMiscSetting) to apply the access configuration.

================================================================ Format:

*C22[${The parameter described in the API Document}]* Description:

Example:

*C22[{"cecEnabled": false,"qrCodeEnabled": false,"stopUpdate": false, “controllerPort”:*

*8080, “userReboot”: {“enabled”: false, “dayOfWeek”: “0”, “hour”: 0, “minute”: 0}}]*

###### 7.2.2 Response

The response of C22 will have prefix “R22” to response the command

Format:

*R22[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R22[applied successfully]*

###### 7.2.3 Error

The possible error message might be following

 1. Invalid json format

# Setup Command Definition

## 8.1 C23

##### 8.1.1 Request

C23 is used to set the extender LED flash and it will stop flashing automatically

================================================================ Format:

*C23[${extender’s name}]* Description:

Enter an extender’s name to start flashing the extender’s LED. The extender’s name is case sensitive.

Example:

*C23[INPUT001]*

*C23[bigTV]*

##### 8.1.2 Response

The response of C23 will have prefix “R23” to response the command

Format:

*R23[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R23[applied successfully]*

##### 8.1.3 Error

The possible error message might be following

1. the device is not online
2. temporarily unavailable

## 8.2 C24

##### 8.2.1 Request

C24 is used to change the extender’s name

================================================================ Format:

*C24[${extender’s macAddress}-${device name}]* Description:

Enter an extender’s macAddress and the new name to change the device name Example:

*C24[11:22:33:44:55:66-newTV]*

*C24[AA:BB:CC:DD:EE:FF-television]*

##### 8.2.2 Response

The response of C24 will have prefix “R24” to response the command

Format:

*R24[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R24[applied successfully]*

##### 8.2.3 Error

The possible error message might be following

1. the device is not online
2. temporarily unavailable
3. Lowercase macAddress is not acceptable.

## 8.3 C25

##### 8.3.1 Request

C25 is used to change the receiver’s upscale mode. The possible value of the upscale is described in the [API Document](https://api.evo-ip.io/#/Setup/SetExtenderUpScaleCommand)

================================================================ Format:

*C25[${Receiver’s name}-${upscale mode]* Description:

The value of the upscale is case sensitive.

Example:

*C25[tv1-passThrough]*

*C25[tv2-1080p60]*

##### 8.3.2 Response

The response of C25 will have prefix “R25” to response the command

Format:

*R25[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R25[applied successfully]*

##### 8.3.3 Error

The possible error message might be following

1. The receiver is not online
2. Invalid upscale value
3. the device is not a receiver

## 8.4 C26

##### 8.4.1 Request

C26 is used to change the transmitter’s videoout. The possible value of the upscale is described in the [API Document](https://api.evo-ip.io/#/Setup/SetExtenderVideoOutCommand)

================================================================ Format:

*C26[${transmitter’s name}-${videoout value]* Description:

Please note that the value is different when it comes to EVOIP and EVOIPLite Example:

*C26[appleTV-0]*

##### 8.4.2 Response

The response of C26 will have prefix “R26” to response the command

Format:

*R26[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R26[applied successfully]*

##### 8.4.3 Error

The possible error message might be following

1. The transmitter is not online
2. Invalid parameter
3. the device is not a transmitter

## 8.5 C27

##### 8.5.1 Request

C27 is used to reboot the extender.

================================================================ Format:

*C27[${extender’s name}]* Description:

Set the extender’s name to reboot.

Example:

*C27[appleTV]*

##### 8.5.2 Response

The response of C27 will have prefix “R27” to response the command

Format:

*R27[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R27[applied successfully]*

##### 8.5.3 Error

The possible error message might be following

1. the device is not online
2. temporarily unavailable

## 8.6 C28

##### 8.6.1 Request

C28 is used to factory default the extender.

================================================================ Format:

*C28[${extender’s name}]* Description:

Set the extender's name to factory default the extender. Please note that the device name will also be back to default.

Example:

*C28[appleTV]*

##### 8.6.2 Response

The response of C28 will have prefix “R28” to response the command

Format:

*R28[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R28[applied successfully]*

##### 8.6.3 Error

The possible error message might be following

1. the device is not online
2. temporarily unavailable

## 8.7 C29

##### 8.7.1 Request

C29 is used to reconnect the device’s HDMI

================================================================ Format:

*C29[${extender’s name}]* Description:

Set the extender's name to reconnect the device’s HDMI port Example:

*C29[appleTV]*

##### 8.7.2 Response

The response of C29 will have prefix “R29” to response the command

Format:

*R29[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R29[applied successfully]*

##### 8.7.3 Error

The possible error message might be following

1. the device is not online
2. temporarily unavailable

## 8.8 C30

##### 8.8.1 Request

C30 is used to set the transmitter EDID mode. The parameter is described in the [API Document](https://api.evo-ip.io/#/Setup/SetExtenderEDIDCommand)

================================================================ Format:

*C30[${The params in json format}]* Description:

Example:

*C30[{"macAddress": "28:60:12:45:12:A4","edidMode":*

*0,"learningEDIDFrom":"11:22:33:12:54:A3"}]*

##### 8.8.2 Response

The response of C30 will have prefix “R30” to response the command

Format:

*R30[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R30[applied successfully]*

##### 8.8.3 Error

The possible error message might be following

1. This transmitter is not online
2. temporarily unavailable
3. the device is not a transmitter

## 8.9 C31

##### 8.9.1 Request

C31 is used to set the extender’s baud rate. The parameter is described in the [API](https://api.evo-ip.io/#/Setup/SetExtenderBaudRateCommand)

[Document](https://api.evo-ip.io/#/Setup/SetExtenderBaudRateCommand)

================================================================ Format:

*C31[${extender’s name}-${baud rate}]* Description:

Please note that EVOIPLite does not support the baud rate 300, 600,1200 Example:

*C31[INPUT001-300]*

*C31[INPUTLite002-115200]*

##### 8.9.2 Response

The response of C31 will have prefix “R31” to response the command

Format:

*R31[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R31[applied successfully]*

##### 8.9.3 Error

The possible error message might be following

1. the device is not online
2. temporarily unavailable
3. Not supported baud rate

## 8.10 C32

##### 8.10.1 Request

C32 is used to rotate the receiver. The parameter is described in the [API Document](https://api.evo-ip.io/#/Setup/SetExtenderRotateCommand)

================================================================ Format:

*C32[${receiver’s name}-${rotate value}]* Description:

Please note the value is 0, 180, 270 Example:

*C32[OUTPUT001-180]*

##### 8.10.2 Response

The response of C32 will have prefix “R32” to response the command

Format:

*R32[${response message}]* Description:

The response will be ‘applied successfully’.

Example:

*R32[applied successfully]*

### 8.10.3 Error

The possible error message might be following

1. the device is not online
2. temporarily unavailable