IPCBox Commands

Version 1.06

IPCBox's API use TCP port 24.

1.06	add kvm command. Use for kvm roaming. add command 'config set device hpdrst {device_id or device_mac}'
1.05	add copy command in vwid. support to set bezel gap in vwid. add 'config set device kvmroaming RX1ID,X1,Y1:RX2ID,X2,Y2[:RX3ID,X3,Y3...] RX7ID'.
1.02	add vwid command.
1.01	update 'matrix aset' and 'matrix add' command description.

IPC System Commands

Command Format	Reply	Function	Note
config help	s, set, status, get, g	Get help of config	
config get help	dns, ipsetting2, version, session, hardver, ipsetting, eth1, rs-232, dev, devicelist, telnet, system, d, device, name, eth0	Get help of config get.	
config get name	AC-MXIP-CBOX	Get IPC's name	
config get version	1.10	Get version	
config get ipsetting	autoip or dhcp or static/192.168.99.1/255.255.255.0/192.168.99.254	Get LAN1's IP info.	
config get ipsetting2	autoip or dhcp or static/192.168.1.239/255.255.255.0/192.168.1.1	Get LAN2's IP info.	
config get devicelist	{ ["8278918939CB"] = { ch = "0000", dtype = "ast152x", id = "8278918939CB", ip = "169.254.8.17", mac = "8278918939CB" }, ["82BE33F3AE0C"] = { ch = "0000", dtype = "ast152x", id = "82BE33F3AE0C", ip = "169.254.6.242", mac = "82BE33F3AE0C" }, ["82FA3B00E809"] = { ch = "0000", dtype = "ast152x", id = "82FA3B00E809", ip = "169.254.5.62", mac = "82FA3B00E809" } }	Get device list	
config get telnet alias	on or off	Get telnet status	
config get rs-232 alias	on or off	Get rs-232 status	
config get system sshservice	on or off	Get ssh service status	
config get dns	8.8.8.8 8.8.4.4	Get dns	
config set help	webpass, ip4addr2, ip4addr, delete, telnetpasswd, passwd, system, d, webloginpasswd, restorefactory, dns, rs-232, eth1, reboot, telnet, eth0,	Get help of config set	

	device, dev, session		
config set ip4addr {autoip/dhcp/static:192.168.99.1:255.255.255.0:192.168.99.254}		Set LAN1's IP	
config set ip4addr2 {autoip/dhcp/static:192.168.100.1:255.255.255.0}		Set LAN2's IP	
config set webloginpasswd {username:password}		Set web login password.	
config set telnetpasswd {password} {username}		Set telnet {username}'s password	
config set delete telnetpasswd		Delete telnet's password	
config set restorefactory		Set IPC restore factory	
config set reboot		Reboot IPC	
config set telnet alias on/off		Set telnet on or off	
config set rs-232 alias on/off		Set rs-232 on or off	
config set system sshservice on/off		Set ssh service on or off	
config set dns {nameserverip1} [nameserverip2]		Set DNS	e. g. : config set dns 8.8.8.8 8.8.4.4
config set microusbmode {modeType}		Note: Only for AC-MXNET-SW1-ASM. Switch microUSB port between IPCBox and the Switch.	modeType = [1,2], 1:IPCBox, 2:Switch

IPC Device Commands

Command Format	Reply	Function	Note
config get device help	status, ip, info, param	Get help of config get device	
config get device status {device_id/device_mac}	s_attaching or s_srv_on	Get device status	Get all device status, use command: config get device status ALL
config get device info {device_id/device_mac}	Response as:(TX) { ch = "0110", dtype = "ast152x", id = "026FB1F2ABCC", ip = "169.254.11.80", ipmode = "autoip", is_host = 1, mac = "026FB1F2ABCC", online = 1585019377, state = "s_attaching"	Get device info	If RX info has not video, for 1G device, its default timing will be 3840x2160 30, and for 10G device, its default timing will be 3840x2160 60.

	}
	Response as:(RX):
	{
	ch = "0000",
	ch_a = "0110",
	ch_p = "0000",
	ch_r = "0110",
	ch_s = "0110",
	ch_u = "0110",
	ch_v = "9001",
	description = "TV1 DES",
	dtype = "ast152x",
	id = "TV1",
	ip = "169.254.4.99",
	ipmode = "autoip",
	mac = "823E75329510",
	online = 6828,
	state = "s_srv_on",
	tx = {
	ch = "9001",
	description = "7G DES",
	dtype = "ast152x",
	id = "7G",
	ip = "169.254.7.14",
	ipmode = "autoip",
	is_host = 1,
	mac = "025F074EE46E",
	online = 6828,
	state = "s_attaching"
	},
	video = {
	frames_per_second = "60",
	height = "1080",
	width = "1920"
	}
	}		

config get device param {param_name} {device_id/device_mac}		Get device param	
config set device help	multicast, info, id, cec, reboot, ip, audio, restorefactory, rm, param	Get help of config set device.	
config set device reboot {device_id/device_mac}		Reboot device	
config set device param {param_name} {param_value} {device_id/device_mac}		Set device param	
config set device restorefactory {device_id/device_mac}		Set device{device_id} restore factory	
config set device id {new_id} {device_id/device_mac}		Set device id to new_id (alias name). Note: id can not contain a colon, can not be named ALL.	
config set device ip {autoip/dhcp/static:192.168.100.1:255.255.255.0:192.168.100.254}		Set device ip	

{device_id/device_mac}			
config set device rm {device_id/device_mac}		Remove device from device list	
config set device audio input type hdmi/analog/auto/auto_1/auto_2		Set device audio type. Note: host only.	
config set device audio volume {value} {device_id/device_mac}		Set device audio volume. Note: client only.	
config set device multicast on/off {device_id/device_mac}		Set device multicast on or off.	
config set device channel {ch_select} {device_id/device_mac}		Set device channel.	Only for TX of 1G device. Note: {ch_select}'s scope is 0001 to 9999. e. g. : config set device channel 3126 TX1
config set device name {name_may_include_space} {device_id/device_mac}		Set device name.	
config set device description {description_may_include_space} {device_id/device_mac}		Set device description.	
config set device video {width height fps} {device_id/device_mac}		Set RX device's timing. when 1G set pass-through, we need send: config set device video 0 0 0 {device_id/device_mac}	All RX support: pass-through: only 1G device support. 1280X720 50 1280X720 60 1920X1080 24 1920X1080 50 1920X1080 60 3840X2160 30 3840X2160 60 e. g. : config set device video 3840 2160 30 RX12
config set device edid {edidIndex} {device_id/device_mac}		Set TX device's EDID. Currently only supports 1G device. When using 1G device, its serial port's modeType need set to 2.	{edidIndex} = [0-15] 0: 1080P_2CH, 1: 1080P_6CH, 2: 1080P_3D_2CH, 3: 1080P_3D_6CH, 4: 4K30Hz_3D_2CH. 5: 4K30Hz_3D_6CH, 6: 4K30Hz_3D_8CH, 7: 1080P_2CH_HDR, 8: 1080P_6CH_HDR, 9: 1080P_3D_2CH_HDR, 10: 1080P_3D_6CH_HDR, 11: 4K30Hz_3D_2CH_HDR. 12: 4K30Hz_3D_6CH_HDR, 13: 4K30Hz_3D_8CH_HDR, 14: 1920X1200_2D_2CH_HDR. 15: User_EDID,

config set device cec {hexData} {device_id/device_mac}		Send CEC data to device. if hexData=poweron/poweroff, IPC will send all poweron/poweroff cec to the device such as: config set device cec poweron {device_id/device_mac} or: config set device cec poweroff {device_id/device_mac}	{hexData} is 0036 or 0004 or other hex data.
config set device rs232mode {modeType} {device_id/device_mac}		Set device's RS232 mode type. Only supports 1G device. After switching the mode, the device will automatically restart. Note: In mode 1, we can not control device light or RX's hdr , TX's edid, some status of video and audio on diagnostics page, also can not modify Custom Name on device LCD screen. And can not upgrade MCU by IPCBOX's WebGUI.	{modeType} = [1,2]. 1: rs232 transparent transmission mode, data is transmitted from TX(RX) to RX(TX). 2: rs232 guest mode, data is transmitted from IPCBox to RX or TX.
config set device rs232setting {xx yy zz aa bb} {device_id/device_mac}		Set device's RS232 setting. Currently only supports 1G device. When using 1G device, its serial port's modeType need set to 2.	xx is baudrate = 300 ~ 115200 yy is data bits = 7 or 8 zz is parity = 0: None, 1: Even, 2: Odd aa is stop bits = 1 bb is flow-control = 0: None e. g. : config set device rs232setting 57600 8 0 1 0 RX6.
config set device rs232 dataType {serial port data} {device_id/device_mac}		Send serial port data to device. When using 1G device, its serial port's modeType need set to 2.	dataType = [1,2], 1:ASCII, 2:HEX. {serial port data} can be terminator with \r or \n or \r\n, but need add double '\'. e.g.: 1, config set device rs232 1 DATA\\r RX7 2, config set device rs232 2 DATA\\n RX7 3, config set device rs232 1 DATA\\r\\n RX7 4, config set device rs232 2 DATA RX7 (DATA is serial port data, may include space.)
~~config set device rs232 \\r\\nset light on\\r\\n {device_id/device_mac}~~ New API: config set device light on {device_id/device_mac}		Set device's light on. When using 1G device, its serial port's modeType need set to 2.	
~~config set device rs232 \\r\\nset light off\\r\\n {device_id/device_mac}~~ New API:		Set device's light off. When using 1G device, its serial port's modeType need set to 2.	

config set device light off {device_id/device_mac}			
~~config set device rs232 \\r\\nset light flash\\r\\n {device_id/device_mac}~~ New API: config set device light flash {device_id/device_mac}		Set device's light flash. When using 1G device, its serial port's modeType need set to 2.	
~~config set device rs232 \\r\\nset alias {aliasName}\\r\\n {device_id/device_mac}~~		~~Set device's alias name. When using 1G device, its serial port's modeType need set to 2.~~	~~{aliasName)'s length need less than 12 characters.~~
~~config set device rs232 \\r\\nset displayIP {xxx.xxx.xxx.xxx}\\r\\n {device_id/device_mac}~~		~~Set device's display IP. When using 1G device, its serial port's modeType need set to 2.~~	~~{xxx.xxx.xxx.xxx} is the display IP.~~
config set device capture {device_id/device_mac}		Capture function	After send the command, there will be a new capture in the address. Such as: http://IPCBox'IP:81/capture.bmp?dev=device_mac
config set device osd on/off {device_id/device_mac}		Set osd on or off. If set to all device, we need send: config set device osd on ALL or config set device osd off ALL	
config get device status {device_id/device_mac}		Get devcie diagnostics status. If we need to get all device status, we can send: config get device status ALL	
config set device hdrmode XX {rx_device_id/rx_device_mac}		Set HDR on/off	[XX=0-1] (0=Disable,1=Enable)
config set device copyedid {RxID} {TxID }		Copy edid from RX	
config set device copyloopedid {TxID or TxMAC}		Copy edid from loop.	
config set device profile XX {TxID or TxMAC}		Only for 1G. Set bandwidth.	[XX=0,5] (0=auto,5=200M)
config set device stretch XX {RxID or RxMAC}		Set stretch.	[XX=1-2], 1=stretch out, 2=fit in.
config set device rotate XX {RxID or RxMAC}		Set rotate.	[XX=0,3,6 or 0,180,270], 0=rotate 0, 3=rotate 180, 6=rotate 270.

config set device hdcp XX {RxID or RxMAC}		Set hdcp.	[XX=0-4]{0=Auto,1=Bypass ,2=HDCP OFF, 3=HDCP1.4, 4=HDCP2.2}
config set device rs232pathdisable {RxID or RxMAC}		Disable rs232path.	
config set device irpathdisable {RxID or RxMAC}		Disable irpath.	
config set device rs232path {TxID or TxMAC} {RxID or RxMAC}		Set rs232path.	
config set device irpath {TxID or TxMAC} {RxID or RxMAC}		Set irpath.	
config set device ir XX {TxID or TxMAC}/{RxID or RxMAC}		Send ir data to Tx or Rx. Support ir formats: Pronto and Global Cache.	XX is IR data. It has tow formats. One is '0000 0068 0024 0000 0155 00b2 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016'. Such as: config set device ir 0000 0068 0024 0000 0155 00b2 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0016 0042 0015 0043 0015 0043 0015 0043 0015 0043 0015 0043 0015 0043 0016 0042 0016 0016 0016 0042 0016 0016 0015 0043 0016 0016 0015 0043 0016 0016 0016 0016 0015 0043 0016 0016 0016 0042 0015 0017 0015 0043 0016 0016 0015 0043 0016 062c 0156 0058 0016 07c5 82546745C4A5. The Global Cache format is '40000,1,1,344,180,20,23B BBBBBB,20,67,20,68CC,19, 68CCDBBCCCB,19,23BCDB FBCC D,20,1594,343,90,21,2006'.
config set device kvmroaming RX1ID,X1,Y1:RX2ID,X2,Y2[:RX3ID,X3,Y3...] RX7ID		Set kvm roaming, the last RX device RX7ID is the Primary device, it need connect the keyboard and mouse device in the front panel of the RX, its postion is (0, 0). Other RX (RX1, RX2, RX3, RX4...) are the secondary devices.	

config set device hpdrst {device_id or device_mac}		Reset hotplug of encoder or decoder.	

IPC Video Wall Commands

Command Format	Reply	Function	Note
vw help		Get help of vw	
vw list		Get video wall list	

vw get {vw_name}	Response as:
	{
	clients = {
	"8278918939CB",
	"82FA3B00E809",
	"8263075A9363",
	"823E75329510",
	},
	cols = 2,
	hosts = {
	"026FB1F2ABCC"
	},
	rotate = {
	0,
	180,
	0,
	0
	},
	rows = 2
	}
		Get video wall {vw_name}	The number of all clients can be less than or equal to rows * columns.

vw add {vw_name} {rows} {cols}		Add video wall, total rows is {rows}, total columns is {cols}	
vw gap {vw_name} {vw} {ow} {vh} {oh}		Set video wall bezel gap	
vw tx {vw_name} {tx1} ..		Set video wall tx1	Add source for video wall. When this command is executed, video wall will play this TX.
vw rx {vw_name} {rx1[:row:col[:rotate]]} ..		Set video wall {rx1} at row, col.	row>=1, col>=1, If the position row, col exist RX, new RX will replace old RX, and the old RX will be removed and displays an entire picture of TX. For 1G device, rotate =0 or 180, for 10G device rotate only equal 0.
vw osd [vw_name} {on/off}		Enable or disable osd	
vw rmtx {vw_name} {tx1} [tx2] ..		Delete video wall tx1	
vw rmrx {vw_name} {rx1} [rx2 rx3...]		Delete video wall rx1, rx2, rx3...	Removes one or multiple RX from video wall. If RX is removed, it displays an entire

			picture of TX.
vw rm {vw_name}		Delete video wall {vw_name}	
vw active {vw_name} [force]		Active video wall {vw_name}	

IPC Matrix Commands

Command Format	Reply	Function	Note
matrix help	active, list, set, aset, add, rm, get	Get help of matrix	

matrix list	{
	juzhen = {
	srcs = {
	["8278918939CB"] = "buzaixian",
	["8263075A9363"] = "zaixian"
	},
	type = "v"
	},
	m1 = {
	type = "z"
	},
	m2 = {
	srcs = {
	["82BE33F3AE0C"] = "02351CC64267"
	},
	type = "z"
	}
	}
		Get matrix list	

matrix get {name}	$ matrix get juzhen
	{
	srcs = {
	["8278918939CB"] = "buzaixian",
	["8263075A9363"] = "zaixian"
	},
	type = "v"
	}
		Get matrix {name}'s info.	

matrix add {name} {video/audio/usb/infrared/ serial/gpio/all}		Add new matrix {name}. parameters video is equal to v. (**video="v", usb="u", audio="a", infrared="r", serial="s", gpio="p", all="z", [""]="v"**). So, if we want to add a matrix named mx1, connect TX1's **video** and **audio** to RX1, we can use parameters	

		av. matrix add mx1 **av** matrix set mx1 TX1 RX1	
matrix set {name} {tx1 rx1 rx2 .. rxn[, tx2 rx..]}		Add tx1 points to rx1 and rx2, and more [tx2 point to rx...]	
matrix aset [[name]:[video/audio/usb/infrared/serial/gpio/all]] {tx1 rx1 rx2 .. rxn[, tx2 rx..]}		parameters video is equal to v. (**video="v", usb="u", audio="a", infrared="r", serial="s", gpio="p", all="z", [""]="v"**). So, if we want to connect TX1's **video** and **audio** to RX1, we can use parameters **av**. matrix aset :**av** TX1 RX1 RX2 .. RXn. Reset tx1 point to rx1 and rx2, or reset the devices in the matrix[name]. The parameters in **[]** is **not necessary**. The default type is **video**. Effective immediately.	e.g.: set infrared from TX1 to RX1, RX3, ... RXn matrix aset :infrared TX1 RX1 RX2 .. RXn e.g.: set usb from TX1 to RX1, RX3, ... RXn matrix aset :usb TX1 RX1 RX2 .. RXn
matrix rm {name}		Delete matrix{name}	
matrix active {name} [force]		Active matrix {name}, if add [force], IPC will force refresh all the devices of matrix {name}	

IPC Scene Commands

Command Format	Reply	Function	Note
scene help	active, list, set, rm, get	Get help of scene	

scene list	{
	scene1 = {
	matrix = {
	"juzhen"
	},
	vwall = {
	"vw22"
	}
	}
	}
		Get scene list	

scene get {name}	$ scene get scene1
	{
	matrix = {
	"juzhen"
	},
	vwall = {
	"vw22"
	}
	}
		Get scene {name}'s info	

scene set vw {name} {vw1 ...}	"OK"	Put {vw1} of video wall into {name} of scene.	$ scene set vw scene1 vw22 "OK"
scene set matrix {name} {mtx1 ...}	"OK"	Put {mtx1} of matrix into scene {name}.	$ scene set matrix scene1 juzhen "OK"
scene rm {name}	"OK"	Delete scene {name}	
scene active {name} [force]		Active scene {name}, if add [force], IPC will force refresh all the devices of scene{name}	

	Reply	Function	Note
scene set vw {name} {vw1 ...}	"OK"	Put {vw1} of video wall into {name} of scene.	$ scene set vw scene1 vw22 "OK"
scene set matrix {name} {mtx1 ...}	"OK"	Put {mtx1} of matrix into scene {name}.	$ scene set matrix scene1 juzhen "OK"
scene active {name} [force]		Active scene {name}, if add [force], IPC will force refresh all the devices of scene{name}	

IPC vwid Commands

vwid help

Function

Get help of vwid

Reply

vwid help

list, rm, add, layout, get

vwid list

Function

Get vwid list

Reply

vwid list

```
{
  videowall1 = {
    cols = 2,
    layouts = {
      vlayout1 = {
        cols = 2,
        layout = {},
        rows = 2
      },
      vlayout2 = {
        cols = 2,
        layout = {},
        rows = 2
      }
    },
    rows = 2
  },
  videowall2 = {
    cols = 2,
    layouts = {
      vlayout1 = {
        cols = 2,
        layout = {
          "1:1:TX1:RX1:1:1:1:1:0:2",
          "1:2:TX1:RX2:1:1:1:1:0:2",
          "2:1:TX1:RX3:2:2:1:1:180:2",
          "2:2:TX1:RX4:2:2:1:2:180:2",
          "3:1:TX1:RX5:2:2:2:1:0:2",
          "3:2:TX1:RX6:2:2:2:2:0:2"
        },
        rows = 3
```

```
        },
        vlayout2 = {
            cols = 2,
            layout = {},
            rows = 3
        }
    },
    rows = 3
  }
}
```

vwid get {videowall_name}

Function

Get vwid {videowall_name}'s info

Reply

```
vwid get videowall2
{
  cols = 2,
  layouts = {
    vlayout1 = {
      cols = 2,
      layout = {
        "1:1:TX1:RX1:1:1:1:1:0:2",
        "1:2:TX1:RX2:1:1:1:1:0:2",
        "2:1:TX1:RX3:2:2:1:1:180:2",
        "2:2:TX1:RX4:2:2:1:2:180:2",
        "3:1:TX1:RX5:2:2:2:1:0:2",
        "3:2:TX1:RX6:2:2:2:2:0:2"
      },
      rows = 3
    },
    vlayout2 = {
      cols = 2,
      layout = {},
      rows = 3
    }
  },
  rows = 3
}
```

vwid rm {videowall_name}

Function

Delete vwid {videowall_name}

Reply

vwid rm videowall1

"OK"

vwid add {videowall_name} {rows} {cols}

Function

Add vwid, name is videowall_name, total rows is {rows}, total columns is {cols}

Reply

vwid add videowall1 2 2

"OK"

vwid setrowcol {videowall_name} {rows} {cols}

Function

Set {rows} and {cols} of {videowall_name}

Reply

vwid setrowcol scene1 5 5

"OK"

vwid layout help

Function

Get help of vwid layout

Reply

vwid layout help

"active, list, set, add, osd, rm, get"

vwid layout add {videowall_name} {layout_name}

Function

Add a layout on {videowall_name}, its name is {layout_name}. Note: {videowall_name}'s rows and cols value will be automatically copied to {layout_name}'s rows and cols.

Reply

vwid layout add videowall2 vlayout2

"OK"

vwid layout setrowcol {videowall_name} {layout_name} {rows} {cols}

Function

Set layout {layout_name}'s row and column to {rows} and {cols} of {videowall_name}. Note: {videowall_name}'s rows and columns will be automatically modified to {rows} and {cols}.

Reply

vwid layout setrowcol scene2 layout3 3 2

"OK"

vwid layout list {videowall_name}

Function

Get layout list of vwid

Reply

```
vwid layout list videowall2
{
  vlayout1 = {
    cols = 2,
    layout = {
      "1:1:TX1:RX1:1:1:1:1:0:2",
      "1:2:TX1:RX2:1:1:1:1:0:2",
      "2:1:TX1:RX3:2:2:1:1:180:2",
      "2:2:TX1:RX4:2:2:1:2:180:2",
      "3:1:TX1:RX5:2:2:2:1:0:2",
      "3:2:TX1:RX6:2:2:2:2:0:2"
    },
    rows = 3
  },
  vlayout2 = {
```

```
    cols = 2,
    layout = {},
    rows = 3
  }
}
```

vwid layout get {videowall_name} {layout_name}

Function

Get {layout_name}'s info of {videowall_name}.

Reply

```
vwid layout get videowall2 vlayout1
{
  cols = 2,
  layout = {
    "1:1:TX1:RX1:1:1:1:1:0:2",
    "1:2:TX1:RX2:1:1:1:1:0:2",
    "2:1:TX1:RX3:2:2:1:1:180:2",
    "2:2:TX1:RX4:2:2:1:2:180:2",
    "3:1:TX1:RX5:2:2:2:1:0:2",
    "3:2:TX1:RX6:2:2:2:2:0:2"
  },
  rows = 3
}
```

vwid layout set {videowall_name} {layout_name} ...

Command

Format:

vwid layout set {videowall_name} {layout_name}
row:col:tx:rx:vwrows:vwcols:vwrow:vwcol:rotate:stretch:vw:ow:vh:oh
row:col:tx:rx:vwrows:vwcols:vwrow:vwcol:rotate:stretch:vw:ow:vh:oh ...

Function

Set **row:col:tx:rx:vwrows:vwcols:vwrow:vwcol:rotate:stretch:vw:ow:vh:oh** in {layout_name} of {videowall_name}.

row: the row in the layout, start with 1.

col: the column in the layout, start with 1.

tx: the encoder device id or mac.

rx: the decoder device id or mac.

vwrows: in internal video wall total rows.

vwcols: in internal video wall total columns.

vwrow: the device in the row of internal video wall.

vwcol: the device in the column of internal video wall.

rotate: the roata can be 0, 180 or 270.

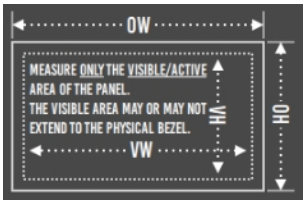
stretch: the stretch can be 1 or 2. 1 is stretch out, 2 is fit in.

vw: visible width of display.

ow: outer width of display.

vh: visible height of display.

oh: outer height of display.



Note: these params join with ":", can connect more than one value, each value is separated by spaces, vw:ow:vh:oh default value is 1:1:1:1.

Reply

vwid layout set videowall2 vlayout1 1:1:TX1:RX1:1:1:1:1:0:2:1:1:1:1 1:2:TX1:RX2:1:1:1:1:0:2 2:1:TX1:RX3:2:2:1:1:180:2 2:2:TX1:RX4:2:2:1:2:180:2 3:1:TX1:RX5:2:2:2:1:0:2

vwid layout osd {videowall_name} {layout_name} {on/off}

Function

Show or hide osd on {layout_name} of {videowall_name}

Reply

vwid layout osd videowall2 vlayout1 on

or

vwid layout osd videowall2 vlayout1 off

vwid layout rm {videowall_name} {layout_name}

Function

Remove {layout_name} of {videowall_name}.

Reply

vwid layout rm videowall1 vlayout2

vwid layout copy {videowall_name} {layout_name} {new_layout_name}

Function

Copy {layout_name} of {videowall_name} to new layout named {new_layout_name}.

Reply

vwid layout copy videowall2 vlayout1 vlayout1_new

"OK"

vwid layout active {videowall_name} {layout_name}

Function

Active {layout_name} of {videowall_name}.

Reply

vwid layout active videowall2 vlayout1

"OK"

IPC kvm Commands

Use for kvm roaming.

kvm help

Function

Get help of kvm

Reply

kvm help

osd, active, list, setrowcol, set, add, rm, get

kvm list

Function

Get kvm list

Reply

kvm list

```
{
   kvm1 = {
      cols = 2,
      layout = {
         "1:1:tx1:rx1:0",
         "1:2:tx2:rx2:0",
         "2:1:tx3:rx3:0",
         "2:2:tx4:rx4:1",
         "3:1:tx5:rx5:0",
         "3:2:tx6:rx6:0"
      },
      rows = 3
   },
   kvm2 = {
      cols = 3,
      layout = {
         "1:1:tx1:rx1:0",
         "1:2:tx2:rx2:0",
         "1:3:tx3:rx3:0",
         "2:1:tx4:rx4:0",
         "2:2:tx5:rx5:1",
         "2:3:tx6:rx6:0",
         "3:1:tx7:rx7:0",
         "3:2:tx8:rx8:0",
         "3:3:tx9:rx9:0"
      },
      rows = 3
   }
}
```

kvm get {kvm_name}

Function

Get info of {kvm_name}

Reply

kvm get kvm1

```
{
   cols = 2,
   layout = {
      "1:1:tx1:rx1:0",
      "1:2:tx1:rx2:0",
```

```
      "2:1:tx1:rx1:0",
      "2:2:tx1:rx1:1",
      "3:1:tx1:rx1:0",
      "3:2:tx1:rx2:0"
   },
   rows = 3
}
```


kvm add {kvm_name} {rows} {cols}

Function

Add {kvm_name}, its row and col is {rows} and {cols}.

Reply

kvm add kvm1 3 2

"OK"

kvm rm {kvm_name}

Function

Romove kvm {kvm_name}.

Reply

kvm rm kvm1

OK

kvm set {kvm_name} row:col:tx:rx:isprimary

row:col:tx:rxisprimary ...

Function

Set kvm {kvm_name}'s info in row and col.

row: The row in kvm's rows.

col: The col in kvm's cols.

tx: encoder.

rx: decoder.

isprimary: Its value is 1 or 0. 1 means the rx is primary, 0 meas the rx is secondary. **Note, only one primary in one kvm.**

Reply

kvm set kvm1 1:1:tx1:rx1:0 1:2:tx2:rx2:0 2:1:tx3:rx3:0 2:2:tx4:rx4:1 3:1:tx5:rx5:0 3:2:tx6:rx6:0

"OK"

kvm osd {kvm_name} {on/off}

Function

Set {kvm_name}'s osd on or off.

Reply

kvm osd kvm1 on

"OK"

kvm osd kvm1 off

"OK"

kvm setrowcol {kvm_name} {rows} {cols}

Function

Modify {kvm_name}'s rows and cols to {rows} and {cols}.

Reply

kvm setrowcol kvm1 3 2

"OK"

kvm active {kvm_name}

Function

Active {kvm_name}. **Note, this command will make decoder's video, audio, usb be switched to corresponding encoder.**

Reply

kvm active kvm1

"OK"