**EVSP24VW**

**RS232 and TCP/IP Commands and Protocol**



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Command head  (2 byte) | Command index  (1 byte) | Command length  (1 byte) | Command body  (CMD\_LENGTH bytes) | Checksum  (1 byte) | Command tail  (2 byte) |
| CMD\_HEAD | CMD\_INDEX | CMD\_LENGTH | CMD\_BODY | CMD\_CHECKSUM | CMD\_TAIL |
| 7b 7b |  |  |  |  | 7d 7d |

**Note:**

1. **The command is begin with the command head 7b 7b in hex mode, and end with the command tail 7d 7d in hex mode;**
2. **Checksum: The checksum is the sum of command head, command index, command length, command body, command tail;**
3. **The command length is bytes of the command body;**
4. **The command total length is the command length +7 bytes;**
5. **All the command is in hex mode;**

Example: 7B 7B 01 02 01 01 F5 7D 7D

7b 7b: Command head

01: Command index(Change routing command)

02: Command length: the length of the command body

01 01: Command body

F5: Check-sum

7d 7d: Command tail

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

Code example to send the command:

#define CMD\_HEAD0 0x00

#define CMD\_HEAD1 0x01

#define CMD\_INDEX 0x02

#define CMD\_LENGHT 0x03

#define CMD\_BODYBEGIN 0x04

#define CMD\_HEAD\_DATA '{'//7b in hex mode

#define CMD\_TILE\_DATA '}'//7d in hex mode

//////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Function:

SendTheCmd

Description:

To send the command via RS232

Input:

CmdIndex: The command index to send;

pCmdBody: Point to the command body data to send;

CmdBodyLength: How many bytes of the command body;

Return:

None.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void SendTheCmd(unsigned char CmdIndex, unsigned char \*pCmdBody,unsigned char CmdBodyLength)

{

unsigned int i = 0;

unsigned int cmdtempchecksum = 0;

unsigned char cmdrealchecksum = 0;

uart2\_send\_char\_com(CMD\_HEAD\_DATA);

uart2\_send\_char\_com(CMD\_HEAD\_DATA);

uart2\_send\_char\_com(CmdIndex);

uart2\_send\_char\_com(CmdBodyLength);

cmdtempchecksum +=CMD\_HEAD\_DATA;

cmdtempchecksum +=CMD\_HEAD\_DATA;

cmdtempchecksum +=CmdIndex;

cmdtempchecksum +=CmdBodyLength;

for(i=0;i<CmdBodyLength;i++)

{

uart2\_send\_char\_com(\*(pCmdBody+i));

cmdtempchecksum +=\*(pCmdBody+i);

}

cmdtempchecksum +=CMD\_TILE\_DATA;

cmdtempchecksum +=CMD\_TILE\_DATA;

cmdrealchecksum = (unsigned char)cmdtempchecksum;

uart2\_send\_char\_com(cmdrealchecksum);

uart2\_send\_char\_com(CMD\_TILE\_DATA);

uart2\_send\_char\_com(CMD\_TILE\_DATA);

}

# Command for Primary User:

### 0x01: Switching Command:

If want to use UART to control the matrix to switch the input/output, please use this command.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | | | Check-sum  (1byte) | Command tail  (2byte) |
| 7b 7b | 01 | 03 | Device address | Input port | Output port |  | 7d 7d |
|  |  |  |

**NOTE:**

1. **The “Output port” byte represent the Output port index, 0~3 represent output port 1~4**
2. **The “Input port” byte represent the input port index, 0~1 represent input port 1~2**

Example:

Switch the input port 1 to output 1:

Command: 7B 7B 01 03 00 00 00 F4 7D 7D

Feedback: 7B 7B 11 05 00 00 01 01 01 09 7D 7D

Switch the input port 1 to output 2:

Command: 7B 7B 01 03 00 00 01 F5 7D 7D

Feedback: 7B 7B 11 05 00 00 00 01 01 08 7D 7D

Switch the input port 1 to all the output port: 7B 7B 01 03 FF 00 FF F2 7D 7D

Feedback: 7B 7B 11 05 00 00 00 00 00 06 7D 7D

### 0xD9: Loop Output Switching Command:

If want to use UART to control the matrix to switch the input/loop output, please use this command.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | | | Check-sum  (1byte) | Command tail  (2byte) |
| 7b 7b | D9 | 03 | Device address | Input port | Output port |  | 7d 7d |
|  |  |  |

Example:

Switch the input port 1 to loop output:

Command: 7B 7B D9 03 00 00 00 CC 7D 7D

Feedback: 7B 7B D9 02 00 00 CB 7D 7D

Switch the input port 2 to loop output:

Command: 7B 7B D9 03 00 01 00 CD 7D 7D

Feedback: 7B 7B D9 02 00 01 CC 7D 7D

### 0xD3: Audio Output Switching Command:

If want to use UART to control the matrix to switch the input/Audio output, please use this command.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | | | Check-sum  (1byte) | Command tail  (2byte) |
| 7b 7b | D3 | 03 | Device address | Input port | Output port |  | 7d 7d |
|  |  |  |

Example:

Switch the input port 1 to Audio output:

Command: 7B 7B D3 03 00 00 00 C6 7D 7D

Feedback: 7B 7B D4 06 00 00 01 00 00 00 CB 7D 7D

Switch the input port 2 to Audio output :

Command: 7B 7B D3 03 00 01 00 C7 7D 7D

Feedback: 7B 7B D4 06 00 01 01 00 00 00 CC 7D 7D

### 0xAA: Reset to factory default:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | | | Check-sum  (1byte) | Command tail  (2byte) |
| {{ | 0xAA | 0x03 | 0xff | 0x01 | 0x01 | 0x9e | }} |
| 0x7b 0x7b | 0xAA | 0x03 | 0xff | 0x01 | 0x01 | 0x9e | 0x7d 0x7d |

**Example：**

Command: 7B 7B AA 03 FF 01 01 9E 7D 7D

Feedback: 7B 7B 11 05 00 00 00 00 00 06 7D 7D

7B 7B D4 06 00 00 01 00 00 00 CB 7D 7D

# Video wall function command list

### 0x90: Set the video wall

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Set the video wall | Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | Check-sum  (1byte) | Command tail  (2byte) |
| {{ |  |  |  |  | }} |
| 7B 7B | 90 | **07** |  |  | 7D 7D |
| **Command body**: **Device Address** (1Byte)+ **Address**(1Byte) + **Ctrl**(1Byte) + Line(1Byte) + Column(1Byte) + P(1Byte) + Q(1Byte)  **Device Address: 0x00**  Address: The board address, to set or read. 0~15 means port 1~16  Ctrl  bit7:1-Read the video parameters; 0-Set the video parameters;  bit6:1-Send this command to the input port whose address is equal to the Address byte;  0-Send this command to the output port whose address is equal to the Address byte;  bit[5,0]:Reserved;  Line: Total number of rows of the video wall  Column: Total number of columns of the video wall  P:The sink row index that the current output 1 connected to;  Q: The sink column index that the current output 1 connected to;  Example;  Output 1 2 3 4 connect sink to build a 2x2 video wall:  Command:  7B 7B 90 07 00 00 00 02 02 01 01 8D 7D 7D  7B 7B 90 07 00 01 00 02 02 01 02 8F 7D 7D  7B 7B 90 07 00 02 00 02 02 02 01 90 7D 7D  7B 7B 90 07 00 03 00 02 02 02 02 92 7D 7D  Exit 2x2 video wall:  7B 7B 90 07 00 00 00 00 00 00 00 87 7D 7D  7B 7B 90 07 00 01 00 00 00 00 00 88 7D 7D  7B 7B 90 07 00 02 00 00 00 00 00 89 7D 7D  7B 7B 90 07 00 03 00 00 00 00 00 8A 7D 7D  Example;  Output 1 2 connect sink to build a 1x2 video wall:  Command:  7B 7B 90 07 00 00 00 01 02 01 01 8C 7D 7D  7B 7B 90 07 00 01 00 01 02 01 02 8E 7D 7D  Exit 1x2 video wall:  7B 7B 90 07 00 00 00 00 00 00 00 87 7D 7D  7B 7B 90 07 00 01 00 00 00 00 00 88 7D 7D | | | | | |

### 0x93: Vide wall parameter read and store

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Video wall parameter read and store** | Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | Check-sum  (1byte) | Command tail  (2byte) |
| {{ |  |  |  |  | }} |
| 7B 7B | 93 | 3+3+4\*N |  |  | 7D 7D |
| **Command body**: **Device Address(1Byte) + Address**(1Byte) + **Ctrl**(1Byte) + **Data**(3Byte+4\*N Byte, N is the quantity of the video wall panel)  **Device Address: 0x00**  **Address:** Not used;  **Ctrl:**  bit7:1-read the vide wall parameters, 0-Store the video wall parameters;  bit[6,0]:Reserved;  Data: video wall parameters, the first is TVWALL\_HEAD(3bytes),then is N\* TVWALL\_UNIT(4bytes),the data struct as follows:  typedef struct \_\_TVWALL\_HEAD\_\_  {  unsigned char u8NUM\_Valid;//valid video wall panel quantity(<=rows\*columns)  unsigned char u8NUM\_Line;//video wall rows quantity  unsigned char u8NUM\_Column;//video wall column quantity  }**TVWALL\_HEAD**;  typedef struct \_\_TVWALL\_UNIT\_\_  {  unsigned char bFlag;//bit0- bCombine; bit1-bSyncLock  unsigned char u8FirstCombineID;//M\*N fist panel ID  unsigned char u8OutputPort;//Output port  unsigned char u8AdjustValue;//frame with  }**TVWALL\_UNIT**;  Read Vide Wall:  (1) 2x2(Screen1-4):  Command:7B 7B 93 06 00 00 80 00 00 00 09 7D 7D  Feedback: 7B 7B 93 28 00 04 02 02 01 00 00 00 00 00 00 01 01 01 00 01 00 00 00 00 01 02 01 00 02 00 00 00 00 01 03 01 00 03 00 00 00 00 01 04 CB 7D 7D  (2)2x2(Screen1-3):  Command: 7B 7B 93 06 00 00 80 00 00 00 09 7D 7D  Feedback: 7B 7B 93 1F 00 03 02 02 00 00 00 00 00 00 00 01 01 00 00 01 00 00 00 00 01 02 00 00 02 00 00 00 00 01 03 B5 7D 7D  (3)2x2(Screen1-2):  Command: 7B 7B 93 06 00 00 80 00 00 00 09 7D 7D  Feedback: 7B 7B 93 16 00 02 02 02 00 00 00 00 00 00 00 01 01 00 00 01 00 00 00 00 01 02 A5 7D 7D  (4)2x2(Screen1):  Command: 7B 7B 93 06 00 00 80 00 00 00 09 7D 7D  Feedback: 7B 7B 93 0D 00 01 02 02 00 00 00 00 00 00 00 00 01 96 7D 7D  Set Vide Wall:  (1)2x2(Screen1-4):  7B 7B 93 2A 00 00 00 04 02 02 00 00 00 00 00 00 00 FF 01 00 00 01 00 00 00 00 01 02 00 00 02 00 00 00 00 01 03 00 00 03 00 00 00 00 01 04 C7 7D 7D  (2)2x2(Screen1-3):  7B 7B 93 21 00 00 00 03 02 02 00 00 00 00 00 00 00 FF 01 00 00 01 00 00 00 00 01 02 00 00 02 00 00 00 00 01 03 B5 7D 7D  (3)2x2(Screen1-2):  7B 7B 93 18 00 00 00 02 02 02 00 00 00 00 00 00 00 FF 01 00 00 01 00 00 00 00 01 02 A5 7D 7D  (4)2x2(Screen1):  7B 7B 93 0F 00 00 00 01 02 02 00 00 00 00 00 00 00 FF 01 97 7D 7D | | | | | |

# Input/Output port signal information get and set command list

### 0xB0: Read/Set the output port type:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Read/Set the output port type** | Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | Check-sum  (1byte) | Command tail  (2byte) |
| {{ |  |  |  |  | }} |
| 7B 7B | B0 | **04** |  |  | 7D 7D |
| **Command body: Device Address(1Byte) + Address(1Byte) + Ctrl(1Byte) + OuputType (1Byte)**  Device Address: 0x00  Address: The board address, to set or read. 0~3 means port 1~4  Ctrl:  bit7:1-Read the output port type; 0-Set the output port type;  bit6:1-Send this command to the input port whose address is equal to the Address byte;  0-Send this command to the output port whose address is equal to the Address byte;  bit5:1-Read/Set the port input type;  0-Read/Set the port output type  bit[4,0]:Reserved;  **OuputType:** Port type  **0x09: HDMI**  **0x0A: DVI**  **0x0B: HDMI with HDCP 1.4**  **0x0C: HDMI with HDCP 2.2**  Example:  Set the output port 2 **HDMI with HDCP 1.4**  Command: 7B 7B B0 04 00 01 00 0B B0 7D 7D  Feedback: 7B 7B B0 04 00 01 00 0B B0 7D 7D  Set the all output port **HDMI with HDCP 1.4**  Command: 7B 7B B0 04 FF FE 00 0B AC 7D 7D | | | | | |

### 0xB1: Read/Set the output resolution

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Read/Set the output resolution** | Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | Check-sum  (1byte) | Command tail  (2byte) |
| {{ |  |  |  |  | }} |
| 7B 7B | B1 | **04** |  |  | 7D 7D |
| **Command body: Device Address(1Byte) + Address(1Byte) + Ctrl(1Byte) + OuputResolution (1Byte)**  **Device Address: 0x00**  **Address**: Address: The board address, to set or read. 0~3 means port 1~4  **Ctrl**:  bit7:1-Read the resolution; 0-Set the output resolution;  bit6:1-Send this command to the input port whose address is equal to the Address byte;  0-Send this command to the output port whose address is equal to the Address byte;  bit5:1-Read/Set the input resolution; 0- Read/Set the output resolution  bit[4,0]:Reserved;  **OuputResolution:** Output resolution  Input board’s input resolution:  Please use the 0xB6 command to read.  Input board’s output resolution table:  Output board’s input resolution table:  0x00:3840x2160@60HZ,  0x01:3840x2160@50Hz,  0x02:3840x2160@30Hz,  0x03:3840x2160@25Hz,  0x04:1920x1200@60Hz,  0x05:1920x1080p@60Hz,  0x06:1920x1080p@50Hz,  0x07:1600x1200@60Hz,  0x08:1400x1050@60Hz,  0x09:1366x768@60Hz,  0x0A:1360x768@60Hz,  0x0B:1280x1024@60Hz,  0x0C:1280x768@60Hz,  0x0D:1280x720pp@60Hz,  0x0E:1280x720p@50Hz,  0x0F:1024x768@60Hz,  Feedback 0xB7 command for resolution information  **Example:**  Set the output port 1 output resolution to 1280X720\_60:Command: 7B 7B B1 04 00 00 00 0D B2 7D 7D  Feedback: 7B 7B B1 04 00 00 00 0D B2 7D 7D  Set the all output port output resolution to 1280X720\_60:Command: 7B 7B B1 04 FF FE 00 0D AF 7D 7D | | | | | |

### 0xB2: Read/Set the port’s color setting:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Read/Set the port’s color setting | Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | Check-sum  (1byte) | Command tail  (2byte) |
| {{ |  |  |  |  | }} |
| 7B 7B | B2 | **04** |  |  | 7D 7D |
| **Command body: Device Address (1Byte) + Address (1Byte) + Ctrl(1Byte) + Brightness (1Byte)**  **Device Address: 0x00**  **Address**: The address of the port, which will react to this command;  **Ctrl**:  bit[7]:1-Read  0-Set  Bit6:1-Input port;  0-Output port;  bit[5,4]:  00:Increase 1  01:Decrease 1  10:Absolutely value set  11:Restore the factory default  bit[3,0]:  0000:Brightness  0001:Contrast  0010:Saturation  0011:Sharpness  0100:Tint  0101:R-Gain  0110:G-Gain  0111:B-Gain  1000:R-Offset  1001:G-Offset  1010:B-Offset  **Example:**  Command: 7B 7B B2 04 00 00 A0 00 46 7D 7D//Read the output port 1’s brightness  Feedback: 7B 7B B2 04 00 00 A0 32 78 7D 7D  Command: 7B 7B B2 04 00 00 A1 00 47 7D 7D//Read the output port 1’s contrast  Feedback: 7B 7B B2 04 00 00 A1 32 79 7D 7D  Command: 7B 7B B2 04 00 00 A2 00 48 7D 7D//Read the output port 1’s saturation  Feedback: 7B 7B B2 04 00 00 A2 32 7A 7D 7D  Command: 7B 7B B2 04 00 00 A3 00 49 7D 7D//Read the output port 1’s sharpness  Feedback: 7B 7B B2 04 00 00 A3 32 7B 7D 7D  Command:7B 7B B2 04 00 01 20 37 FE 7D 7D//Set the output port 2’s brightness 55  Feedback: 7B 7B B2 04 00 01 20 37 FE 7D 7D  Command:7B 7B B2 04 00 01 21 37 FF 7D 7D//Set the output port 2’s contrast 55  Feedback: 7B 7B B2 04 00 01 21 37 FF 7D 7D  Command:7B 7B B2 04 00 01 22 37 00 7D 7D//Set the output port 2’s saturation 55  Feedback: 7B 7B B2 04 00 01 22 37 00 7D 7D  Command:7B 7B B2 04 00 01 23 37 01 7D 7D//Set the output port 2’s sharpness 55  Feedback: 7B 7B B2 04 00 01 23 37 01 7D 7D  Command:7B 7B B2 04 00 01 30 00 D7 7D 7D//Reset the output port 2’s sharpness  Feedback: 7B 7B B2 04 00 01 30 00 D7 7D 7D  Command:7B 7B B2 04 00 01 31 00 D8 7D 7D//Reset the output port 2’s sharpness  Feedback: 7B 7B B2 04 00 01 31 00 D8 7D 7D  Command:7B 7B B2 04 00 01 32 00 D9 7D 7D//Reset the output port 2’s sharpness  Feedback: 7B 7B B2 04 00 01 32 00 D9 7D 7D  Command:7B 7B B2 04 00 01 33 00 DA 7D 7D//Reset the output port 2’s sharpness  Feedback: 7B 7B B2 04 00 01 33 00 DA 7D 7D | | | | | |

### 0xB6: Read the input port’s Signal format

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Read the input port’s input resolution** | Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | Check-sum  (1byte) | Command tail  (2byte) |
| {{ |  |  |  |  | }} |
| 7B 7B | B6 | **09** |  |  | 7D 7D |
| **Command body: Device Address (1Byte) + Address(1Byte) + Ctrl(1Byte) + Hori\_H + Hori\_L + Vert\_H + Vert\_L + bProgressive +Frequency**  **Device Address: 0x00**  **Address**: **The input port address**  **Ctrl**:  bit7:1-Read the input resolution;  0-Set the input resolution;//Not used  bit6:1-Send this command to the input port whose address is equal to the Address byte;  0-Send this command to the output port whose address is equal to the Address byte;  bit5:1-Read/Set the input resolution  0- Read/Set the output resolution  bit[4,0]:Reserved;  **OuputResolution:** Resolution,  Input port input resolution:  Hori\_H: Horizontal resolution high 8 bit; (1Byte);  Hori\_L: Horizontal resolution low 8 bit; (1Byte);  Vert\_H: Vertical resolution high 8 bit; (1Byte);  Vert\_L: Vertical resolution low 8 bit; (1Byte);  **bProgressive:1-progress; 0- interlace**  Frequency: Frame frequency  NOTE：  When HDMI is direct, use this command index to query the resolution.  Example:  Input port 1 input resolution:  Command: 7B 7B B6 09 00 00 E0 00 00 00 00 00 00 8F 7D 7D  Feedback: 7B 7B B6 0A 00 00 E0 0F 00 08 70 01 1E 00 36 7D 7D  (DVI@3840x2160p30)  Input port 2 input resolution:  Command: 7B 7B B6 09 00 01 E0 00 00 00 00 00 00 90 7D 7D  Feedback: 7B 7B B6 0A 00 01 E0 07 80 04 38 01 3C 01 92 7D 7D  (HDM@1920x1080p60) | | | | | |

### 0xB7: Set/Read signal detail information

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Set/Read signal detail information | Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | Check-sum  (1byte) | Command tail  (2byte) |
| {{ |  |  |  |  | }} |
| 7B 7B | B7 | **08** |  |  | 7D 7D |
| **Command body format 1: Device Address(1Byte) + Address(1Byte) + Ctrl(1Byte) + DataValid + InType + OutType + InResolution + OutResolution**  **Command body format 2: Device Address (1Byte) + Address(1Byte) + Ctrl(1Byte) + Hori\_H + Hori\_L + Vert\_H + Vert\_L+ Frequency**  **Device Address: 0x00**  **Address**: The address of the port, which will react to this command;  **Ctrl**:  bit7:1-Read;  0-Set;  bit6:1-Input port;  0-Output port;//This command only send to the output port  bit5:0-Set the command body to format 1,1-Set the command body to format 2  bit[4,0]:Reserved;  **Command body format 1:**  **DataValid**:bit[7-4]:Reserved  bit3:0-InType Invalid, will ignore, 1-InType valid  bit2:0-OutType Invalid, will ignore,, 1-OutType valid  bit1:0-InResolution Invalid, will ignore,1-InResolution valid  bit0:0-OutResolution Invalid, will ignore,1-OutResolution valid  **InType**: The input signal type  **OutType**: Output signal type  **0x09: HDMI**  **0x0A: DVI**  **0x0B: HDMI with HDCP 1.4**  **0x0C: HDMI with HDCP 2.2**  **InResolution**: Input signal resolution  **OutResolution**: Output signal resolution  1. Input port input resolution: Not support  2. Input port’s output resolution/Output port’s input resolution:  0x00:3840x2160@60HZ,  0x01:3840x2160@50Hz,  0x02:3840x2160@30Hz,  0x03:3840x2160@25Hz,  0x04:1920x1200@60Hz,  0x05:1920x1080p@60Hz,  0x06:1920x1080p@50Hz,  0x07:1600x1200@60Hz,  0x08:1400x1050@60Hz,  0x09:1366x768@60Hz,  0x0A:1360x768@60Hz,  0x0B:1280x1024@60Hz,  0x0C:1280x768@60Hz,  0x0D:1280x720pp@60Hz,  0x0E:1280x720p@50Hz,  0x0F:1024x768@60Hz,  Please use the format 2 of this command to get the resolution information  **Command body format 2:*(Only used for the input port’s input resolution)***  Hori\_H: Horizontal resolution high 8 bit; (1Byte);  Hori\_L: Horizontal resolution low 8 bit; (1Byte);  Vert\_H: Vertical resolution high 8 bit; (1Byte);  Vert\_L: Vertical resolution low 8 bit; (1Byte);  **P\_Frequency bit8: bProgressive -> 1-Progress; 0- Interlace**  **P\_Frequency bit[7:0]:** Frequency;  Example:  Output port 1 resolution:  Command: 7B 7B B7 08 00 00 80 00 00 00 00 00 2F 7D 7D  Feedback: 7B 7B B7 09 00 00 80 1F 09 0B 02 0D 00 72 7D 7D  (IN:HDMI@3840x2160p30; OUT:UHD-HDDMI-1.4@ 1280x720p60)  Output port 2 resolution:  Command: 7B 7B B7 08 00 01 80 00 00 00 00 00 30 7D 7D  Feedback: 7B 7B B7 09 00 01 80 1F 09 0C 02 02 00 69 7D 7D  (IN:HDMI@3840x2160p30; OUT:UHD-HDDMI-2.2@3840x2160p30)  Output port 3 resolution:  Command: 7B 7B B7 08 00 02 80 00 00 00 00 00 31 7D 7D  Feedback: 7B 7B B7 09 00 02 80 1F 09 0B 02 0D 00 74 7D 7D  (IN:HDMI@3840x2160p30; OUT:UHD-HDDMI-1.4@ 1280x720p60)  Output port 4 resolution:  Command: 7B 7B B7 08 00 03 80 00 00 00 00 00 32 7D 7D  Feedback: 7B 7B B7 09 00 03 80 1F 09 0C 02 02 00 6B 7D 7D  (IN:HDMI@3840x2160p30; OUT:UHD-HDDMI-2.2@3840x2160p30) | | | | | |

### 0xB8: EDID Control

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Read the output EDID or set the input EDID | Command head (2byte) | Command index  (1byte) | Command length  (1byte) | Command body (CMD\_LENGTH byte) | Check-sum  (1byte) | Command tail  (2byte) |
| CMD\_HEAD | CMD\_INDEX | CMD\_LENGHT | CMD\_BODY | CMD\_CHECKSUM | CMD\_TAIL |
| {{ |  |  |  |  | }} |
| 7B 7B | B8 | **0x14** |  |  | 7D 7D |
| **Command body：Device Address(1Byte) + Address(1Byte) + Ctrl(1Byte) + Offset (1Byte) + EDID\_Content (16Bytes)**  **Device Address: 0x00**  **Address**: destination address  **Ctrl**：bit7：1：read  0：set  bit6：1-input board；  0-output board；  bit[5,0]：reserved  **Offset：the offset of the EDID content of the total EDID**  **EDID Content：**the EDID content in this command  NOTE:   1. Only can set the EDID of the input board 2. Only can read the EDID of the output board 3. Read EDID data from output port.   For example, GET (Read) EDID data from output port 1 (Read 16 times)  Send: 7B 7B B8 14 00 00 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3C 7D 7D  Receive: 7B 7B B8 14 00 00 80 00 00 FF FF FF FF FF FF 00 41 0C FA 08 2B 19 00 00 C9 7D 7D  Send: 7B 7B B8 14 00 00 80 10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 4C 7D 7D  Receive: 7B 7B B8 14 00 00 80 10 11 1B 01 03 80 5F 36 78 2A 58 3D A5 55 53 9C 25 D6 7D 7D  Send: 7B 7B B8 14 00 00 80 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 5C 7D 7D  Receive: 7B 7B B8 14 00 00 80 20 0C 47 4A BF EF 00 D1 C0 B3 00 95 00 81 80 81 40 42 7D 7D  Send: 7B 7B B8 14 00 00 80 30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 6C 7D 7D  Receive: 7B 7B B8 14 00 00 80 30 81 C0 01 01 01 01 A3 66 00 A0 F0 70 1F 80 30 20 A9 7D 7D  Send: 7B 7B B8 14 00 00 80 40 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 7C 7D 7D  Receive: 7B 7B B8 14 00 00 80 40 35 00 B9 1F 32 00 00 1A 56 5E 00 A0 A0 A0 29 50 E2 7D 7D  Send: 7B 7B B8 14 00 00 80 50 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 8C 7D 7D  Receive: 7B 7B B8 14 00 00 80 50 30 20 35 00 B9 1F 32 00 00 1E 00 00 00 FD 00 17 4D 7D 7D  Send: 7B 7B B8 14 00 00 80 60 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 9C 7D 7D  Receive: 7B 7B B8 14 00 00 80 60 50 1E 63 1E 00 0A 20 20 20 20 20 20 00 00 00 FC 51 7D 7D  Send: 7B 7B B8 14 00 00 80 70 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 AC 7D 7D  Receive: 7B 7B B8 14 00 00 80 70 00 50 48 4C 20 42 44 4D 34 33 35 30 0A 20 01 1C 96 7D 7D  Send: 7B 7B B8 14 00 00 80 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 BC 7D 7D  Receive: 7B 7B B8 14 00 00 80 80 02 03 26 F1 4B 10 1F 05 14 04 13 03 12 02 11 01 AB 7D 7D  Send: 7B 7B B8 14 00 00 80 90 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 CC 7D 7D  Receive: 7B 7B B8 14 00 00 80 90 23 09 07 07 83 01 00 00 6D 03 0C 00 20 00 38 3C 9A 7D 7D  Send: 7B 7B B8 14 00 00 80 A0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 DC 7D 7D  Receive: 7B 7B B8 14 00 00 80 A0 20 00 60 01 02 03 01 1D 00 72 51 D0 1E 20 6E 28 E7 7D 7D  Send: 7B 7B B8 14 00 00 80 B0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 EC 7D 7D  Receive: 7B 7B B8 14 00 00 80 B0 55 00 B9 1F 32 00 00 1E 8C 0A D0 8A 20 E0 2D 10 96 7D 7D  Send: 7B 7B B8 14 00 00 80 C0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FC 7D 7D  Receive: 7B 7B B8 14 00 00 80 C0 10 3E 96 00 B9 1F 32 00 00 18 8C 0A D0 90 20 40 58 7D 7D  Send: 7B 7B B8 14 00 00 80 D0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0C 7D 7D  Receive: 7B 7B B8 14 00 00 80 D0 31 20 0C 40 55 00 B9 1F 32 00 00 18 4D 6C 80 A0 F9 7D 7D  Send: 7B 7B B8 14 00 00 80 E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 1C 7D 7D  Receive: 7B 7B B8 14 00 00 80 E0 70 70 3E 80 30 20 3A 00 B9 1F 32 00 00 1A 00 00 68 7D 7D  Send: 7B 7B B8 14 00 00 80 F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 2C 7D 7D  Receive: 7B 7B B8 14 00 00 80 F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F9 25 7D 7D  NOTE:  EDID totally has 256 bytes, so 16 bytes will be read per time (16 times).  B. SET (Write) EDID data to input port  For example, SET (Write) EDID data to input port 1 (Write 16 times)  Send:7B 7B B8 14 00 00 40 00 00 FF FF FF FF FF FF 00 41 0C FA 08 2B 19 00 00 89 7D 7D  Receive: 7B 7B B8 14 00 00 40 00 00 FF FF FF FF FF FF 00 41 0C FA 08 2B 19 00 00 89 7D 7D  Send:7B 7B B8 14 00 00 40 10 11 1B 01 03 80 5F 36 78 2A 58 3D A5 55 53 9C 25 96 7D 7D  Receive: 7B 7B B8 14 00 00 40 10 11 1B 01 03 80 5F 36 78 2A 58 3D A5 55 53 9C 25 96 7D 7D  Send:7B 7B B8 14 00 00 40 20 0C 47 4A BF EF 00 D1 C0 B3 00 95 00 81 80 81 40 02 7D 7D  Receive: 7B 7B B8 14 00 00 40 20 0C 47 4A BF EF 00 D1 C0 B3 00 95 00 81 80 81 40 02 7D 7D  Send:7B 7B B8 14 00 00 40 30 81 C0 01 01 01 01 A3 66 00 A0 F0 70 1F 80 30 20 69 7D 7D  Receive: 7B 7B B8 14 00 00 40 30 81 C0 01 01 01 01 A3 66 00 A0 F0 70 1F 80 30 20 69 7D 7D  Send:7B 7B B8 14 00 00 40 40 35 00 B9 1F 32 00 00 1A 56 5E 00 A0 A0 A0 29 50 A2 7D 7D  Receive: 7B 7B B8 14 00 00 40 40 35 00 B9 1F 32 00 00 1A 56 5E 00 A0 A0 A0 29 50 A2 7D 7D  Send:7B 7B B8 14 00 00 40 50 30 20 35 00 B9 1F 32 00 00 1E 00 00 00 FD 00 17 0D 7D 7D  Receive: 7B 7B B8 14 00 00 40 50 30 20 35 00 B9 1F 32 00 00 1E 00 00 00 FD 00 17 0D 7D 7D  Send:7B 7B B8 14 00 00 40 60 50 1E 63 1E 00 0A 20 20 20 20 20 20 00 00 00 FC 11 7D 7D  Receive: 7B 7B B8 14 00 00 40 60 50 1E 63 1E 00 0A 20 20 20 20 20 20 00 00 00 FC 11 7D 7D  Send:7B 7B B8 14 00 00 40 70 00 50 48 4C 20 42 44 4D 34 33 35 30 0A 20 01 1C 56 7D 7D  Receive: 7B 7B B8 14 00 00 40 70 00 50 48 4C 20 42 44 4D 34 33 35 30 0A 20 01 1C 56 7D 7D  Send:7B 7B B8 14 00 00 40 80 02 03 26 F1 4B 10 1F 05 14 04 13 03 12 02 11 01 6B 7D 7D  Receive: 7B 7B B8 14 00 00 40 80 02 03 26 F1 4B 10 1F 05 14 04 13 03 12 02 11 01 6B 7D 7D  Send:7B 7B B8 14 00 00 40 90 23 09 07 07 83 01 00 00 6D 03 0C 00 20 00 38 3C 5A 7D 7D  Receive: 7B 7B B8 14 00 00 40 90 23 09 07 07 83 01 00 00 6D 03 0C 00 20 00 38 3C 5A 7D 7D  Send:7B 7B B8 14 00 00 40 A0 20 00 60 01 02 03 01 1D 00 72 51 D0 1E 20 6E 28 A7 7D 7D  Receive: 7B 7B B8 14 00 00 40 A0 20 00 60 01 02 03 01 1D 00 72 51 D0 1E 20 6E 28 A7 7D 7D  Send:7B 7B B8 14 00 00 40 B0 55 00 B9 1F 32 00 00 1E 8C 0A D0 8A 20 E0 2D 10 56 7D 7D  Receive: 7B 7B B8 14 00 00 40 B0 55 00 B9 1F 32 00 00 1E 8C 0A D0 8A 20 E0 2D 10 56 7D 7D  Send:7B 7B B8 14 00 00 40 C0 10 3E 96 00 B9 1F 32 00 00 18 8C 0A D0 90 20 40 18 7D 7D  Receive: 7B 7B B8 14 00 00 40 C0 10 3E 96 00 B9 1F 32 00 00 18 8C 0A D0 90 20 40 18 7D 7D  Send:7B 7B B8 14 00 00 40 D0 31 20 0C 40 55 00 B9 1F 32 00 00 18 4D 6C 80 A0 B9 7D 7D  Receive: 7B 7B B8 14 00 00 40 D0 31 20 0C 40 55 00 B9 1F 32 00 00 18 4D 6C 80 A0 B9 7D 7D  Send:7B 7B B8 14 00 00 40 E0 70 70 3E 80 30 20 3A 00 B9 1F 32 00 00 1A 00 00 28 7D 7D  Receive: 7B 7B B8 14 00 00 40 E0 70 70 3E 80 30 20 3A 00 B9 1F 32 00 00 1A 00 00 28 7D 7D  Send:7B 7B B8 14 00 00 40 F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F9 E5 7D 7D  Receive: 7B 7B B8 14 00 00 40 F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F9 E5 7D 7D  NOTE:   1. EDID totally have 256 bytes, so we need to write 16 times and 16 bytes will be write per time. | | | | | |