676R, 676T: Protocol 3000 Commands

Kramer devices can be operated using Kramer Protocol 3000 commands sent via serial or Ethernet ports.

Understanding Protocol 3000

Protocol 3000 commands are a sequence of ASCII letters, structured according to the following.

- **Command format**:

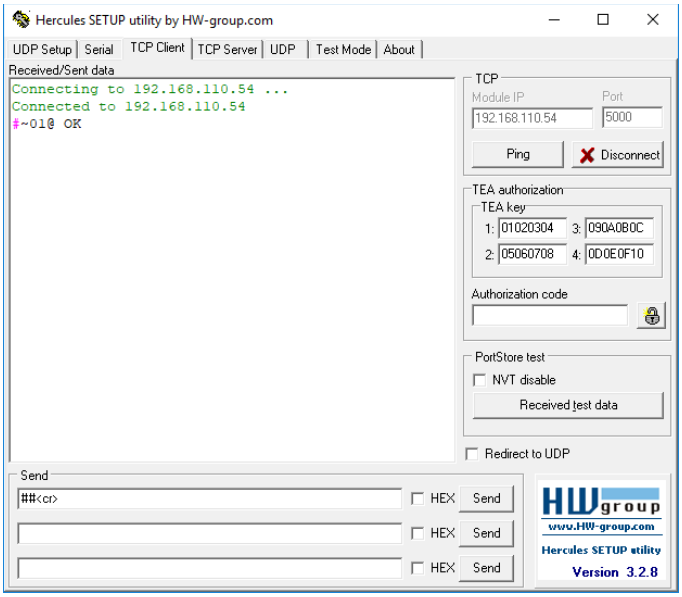
Prefix	Command Name	Constant (Space)	Parameter(s)	Suffix
#	Command	␣	Parameter	<CR>

- **Feedback format**:

Prefix	Device ID	Constant	Command Name	Parameter(s)	Suffix
~	nn	@	Command	Parameter	<CR><LF>

- **Command parameters** – Multiple parameters must be separated by a comma (,). In addition, multiple parameters can be grouped as a single parameter using brackets ([and]).
- **Command chain separator character** – Multiple commands can be chained in the same string. Each command is delimited by a pipe character (|).
- **Parameters attributes** – Parameters may contain multiple attributes. Attributes are indicated with pointy brackets (<…>) and must be separated by a period (.).

The command framing varies according to how you interface with **676R, 676T**. The following figure displays how the # command is framed using terminal communication software (such as Hercules):



Protocol 3000 Commands

Function	Description	Syntax	Parameters/Attributes	Example
#	Protocol handshaking. ⓘ Validates the Protocol 3000 connection and gets the machine number. Step-in master products use this command to identify the availability of a device.	**COMMAND** **#<CR>** **FEEDBACK** **~nn@ ok<CR><LF>**		**#<CR>**
AV-SW-TIMEOUT	Set auto switching timeout. For 676R only	**COMMAND** **#AV-SW-TIMEOUT switching_mode,time_out<CR>** **FEEDBACK** **~nn@AV-SW-TIMEOUT switching_mode,time_out<CR><LF>**	**switching_mode** – Switching mode 0 – Video signal lost 1 – New video signal detected 4 – Disable 5V on video output if no input signal detected 8 – CEC monitor standby command **time_out** – Timeout in seconds 0 - 80000	Set the auto switching timeout to 5 seconds in the event of 5V disable when no input signal is detected: **#AV-SW-TIMEOUT 4,5<CR>**
AV-SW-TIMEOUT?	Get auto switching timeout. For 676R only	**COMMAND** **#AV-SW-TIMEOUT? switching_mode<CR>** **FEEDBACK** **~nn@AV-SW-TIMEOUT switching_mode,time_out<CR><LF>**	**switching_mode** – Switching mode 0 – Video signal lost 1 – New video signal detected 4 – Disable 5V on video output if no input signal detected 8 – CEC monitor standby command **time_out** – Timeout in seconds 0 - 80000	Get the Disable 5V on video output if no input signal detected timeout: **#AV-SW-TIMEOUT? 4<CR>**
BUILD-DATE?	Get device build date.	**COMMAND** **#BUILD-DATE? <CR>** **FEEDBACK** **~nn@BUILD-DATE date,time<CR><LF>**	**date** – Format: YYYY/MM/DD where YYYY = Year MM = Month DD = Day **time** – Format: hh:mm:ss where hh = hours mm = minutes ss = seconds	Get the device build date: **#BUILD-DATE?<CR>**
CEC-SND	Send CEC command to port.	**COMMAND** **#CEC-SND port_index,sn_id,cmd_name,cec_len,cec_command<CR>** **FEEDBACK** **~nn@CEC-SND port_index,sn_id,cmd_name,cec_mode<CR><LF>**	**port_index** – CEC port transmitting the command (1 – number of ports) sn_id – serial number of command for flow control and response commands from device cmd_name – command name cec_len – 1–16 cec_command – CEC format command (in HEX format, no leading zeros, no '0x' prefix) cec_mode – CEC mode 0 – Sent 1 – Gateway disabled 2 – Inactive CEC-Master 3 – Busy 4 – Illegal Message Parameter 5 – Illegal CEC Address Parameter 6 – Illegal CEC Command 7 – Timeout 8 – Error	Send CEC command to port: **#CEC-SND 1,1,1,2,E004<CR>**

Function	Description	Syntax	Parameters/Attributes	Example
CPEDID	Copy EDID data from the output to the input EEPROM. ⓘ Destination bitmap size depends on device properties (for 64 inputs it is a 64-bit word). Example: bitmap 0x0013 means inputs 1,2 and 5 are loaded with the new EDID. In certain products Safe_mode is an optional parameter. See the HELP command for its availability. For 676T only	COMMAND **#CPEDID** edid_io,src_id,edid_io,dest_bitmap**<CR>** or **#CPEDID** edid_io,src_id,edid_io,dest_bitmap,safe_mode**<CR>** FEEDBACK **~nn@CPEDID** edid_io,src_id,edid_io,dest_bitmap**<CR><LF>** **~nn@CPEDID** edid_io,src_id,edid_io,dest_bitmap,safe_mode**<CR><LF>**	**edid_io** – *EDID* source type (usually output) 0 – Input 1 – Output 2 – Default EDID 3 – Custom EDID **src_id** – Number of chosen source stage 0 – Default EDID source 1 – Output 1 2 – Output 2 **edid_io** – *EDID destination* type (usually input) 0 – Input 1 – Output 2 – Default EDID 3 – Custom EDID **dest_bitmap** – Bitmap representing destination IDs. Format: XXXX…X, where X is hex digit. The binary form of every hex digit represents corresponding destinations. 0 – indicates that EDID data is not copied to this destination. 1 – indicates that EDID data is copied to this destination. **safe_mode** – Safe mode 0 – device accepts the EDID as is without trying to adjust 1 – device tries to adjust the EDID (default value if no parameter is sent)	Copy the EDID data from the Output 1 (EDID source) to the Input: **#CPEDID** 1,1,0,0x1**<CR>** Copy the EDID data from the default EDID source to the Input: **#CPEDID** 2,0,0,0x1**<CR>**
DPSW-STATUS?	Get the DIP-switch state.	COMMAND **#DPSW-STATUS?** dip_id**<CR>** FEEDBACK **~nn@DPSW-STATUS** dip_id,status**<CR><LF>**	**dip_id** – 1 to 4 (number of DIP switches) **status** – Up/down 0 – Up 1 – Down	get the DIP-switch 2 status: **#DPSW-STATUS?** 2**<CR>**
FACTORY	Reset device to factory default configuration. ⓘ This command deletes all user data from the device. The deletion can take some time. Your device may require powering off and powering on for the changes to take effect.	COMMAND **#FACTORY<CR>** FEEDBACK **~nn@FACTORY** ok**<CR><LF>**		Reset the device to factory default configuration: **#FACTORY<CR>**
FPGA-VER?	Get current FPGA version.	COMMAND **#FPGA-VER?** fpga_id**<CR>** FEEDBACK **~nn@FPGA-VER** fpga_id,expected_ver,ver**<CR><LF>**	**fpga_id** – FPGA id **expected_ver** – Expected FPGA version for current firmware **ver** – Actual FPGA version	Get current FPGA version: **#FPGA-VER?** 1**<CR>**
GEDID	Get EDID support on certain input/output. ⓘ For old devices that do not support this command, ~nn@ERR 002<CR><LF> is received. For 676T only	COMMAND **#GEDID** io_mode,in_index**<CR>** FEEDBACK **~nn@GEDID** io_mode,in_index,size**<CR><LF>**	**io_mode** – Input/Output 0 – Input **in_index** – Number that indicates the specific input: 1 **size** – Size of data to be sent from device, 0 means no EDID support	Get EDID support information for input 1: **#GEDID** 0,1**<CR>**
HDCP-MOD?	Get HDCP mode. ⓘ Set HDCP working mode on the device input: HDCP supported - HDCP_ON [default]. HDCP not supported - HDCP OFF. HDCP support changes following detected sink - MIRROR OUTPUT. For 676T only	COMMAND **#HDCP-MOD?** in_index**<CR>** FEEDBACK **~nn@HDCP-MOD** in_index,mode**<CR><LF>**	**in_index** – Number that indicates the specific input: 1 **mode** – HDCP mode: 0 – HDCP Off 1 – HDCP On 2 – Follow Input 3 – HDCP defined according to the connected output	Get the input HDCP-MODE of IN 1 HDMI: **#HDCP-MOD?** 1**<CR>**
HDCP-STAT?	Get HDCP signal status. ⓘ io_mode =1 – get the HDCP signal status of the sink device connected to the specified output. io_mode =0 – get the HDCP signal status of the source device connected to the specified input. For 676T only	COMMAND **#HDCP-STAT?** io_mode,in_index**<CR>** FEEDBACK **~nn@HDCP-STAT** io_mode,in_index,status**<CR><LF>**	**io_mode** – Input/Output 0 – Input 1 – Output **io_index** – Number that indicates the specific number of inputs or outputs (based on io_mode): 1 **status** – Signal encryption status - valid values On/Off 0 – HDCP Off 1 – HDCP On	Get the output HDCP-STATUS of IN 1: **#HDCP-STAT?** 0,1**<CR>**

Function	Description	Syntax	Parameters/Attributes	Example
HELP	Get command list or help for specific command.	**COMMAND** **#HELP<CR>** **#HELP** cmd_name**<CR>** **FEEDBACK** 1. Multi-line: **~nn@Device** cmd_name, cmd_name…**<CR><LF>** To get help for command use: HELP (COMMAND_NAME)<CR><LF> **~nn@HELP** cmd_name**:<CR><LF>** **description<CR><LF>** **USAGE:usage<CR><LF>**	**cmd_name** – Name of a specific command	Get the command list: **#HELP<CR>** To get help for AV-SW-TIMEOUT: HELP **av-sw-timeout**<CR>
LDEDID	Write EDID data from external application to device. When the unit receives the LDEDID command it replies with READY and enters the special EDID packet wait mode. In this mode the unit can receive only packets and not regular protocol commands. If the unit does not receive correct packets for 30 seconds or is interrupted for more than 30 seconds before receiving all packets, it sends timeout error **~nn@LDEDID err01**<CR><LF> and returns to the regular protocol mode. If the unit received data that is not a correct packet, it sends the corresponding error and returns to the regular protocol mode. For 676T only	**COMMAND** Multi-step syntax **FEEDBACK** Step 1: **#LDEDID** edid_io,dest_bitmask,edid_size,safe_mode**<CR>** Response 1: **~nn@LDEDID** edid_io,dest_bitmask,edid_size,safe_mode ready**<CR><LF>** or **~nn@LDEDID** errnn**<CR><LF>** Step 2: If ready was received, send EDID_DATA Response 2: **~nn@LDEDID** edid_io,dest_bitmask,edid_size,safe_mode ok**<CR><LF>** or **~nn@LDEDID** errnn**<CR><LF>**	**edid_io** – EDID destination type (usually input) 0 – Input 1 – Output 2 – Default EDID 3 – Custom EDID **dest_bitmask** – Bitmap representing destination IDs. Format: 0x********, where * is ASCII presentation of hex digit. The binary presentation of this number is a bit mask for destinations. Setting '1' means EDID data has to be copied to this destination **edid_size** – EDID data size **safe_mode** – Safe mode 0 – Device accepts the EDID as is without trying to adjust 1 – Device tries to adjust the EDID **edid_data** – Data in protocol packets **Using the Packet Protocol** Send a command: LDRV, LOAD, IROUT, LDEDID Receive Ready or ERR### If Ready: **a**. Send a packet, **b**. Receive OK on the last packet, **c**. Receive OK for the command Packet structure: **Packet ID** (1, 2, 3…) (2 bytes in length) **Length** (data length + 2 for CRC) – (2 bytes in length) **Data** (data length -2 bytes) **CRC** – 2 bytes 01 02 03 04 05 Packet ID Length Data CRC 5. Response: **~nnnn_ok**<CR><LF> (Where ***NNNN*** is the received packet ID in ASCII hex digits.)	Write the EDID data from an external application to the HDMI In 1 input without adjustment attempts: **#LDEDID 0,0x1,2340,0<CR>** Write the EDID data from an external application to HDMI In 1 and PC In inputs with adjustment attempts: **#LDEDID 0,0x5,2340,1<CR>**
MODEL?	Get device model. This command identifies equipment connected to **676R, 676T** and notifies of identity changes to the connected equipment. The Matrix saves this data in memory to answer REMOTE-INFO requests.	**COMMAND** **#MODEL? <CR>** **FEEDBACK** **~nn@MODEL** model_name**<CR><LF>**	**model_name** – String of up to 19 printable ASCII chars	Get the device model: **#MODEL? <CR>**
PROT-VER?	Get device protocol version.	**COMMAND** **#PROT-VER? <CR>** **FEEDBACK** **~nn@PROT-VER** 3000:version**<CR><LF>**	**version** – XX.XX where X is a decimal digit	Get the device protocol version: **#PROT-VER? <CR>**
RESET	Reset device. To avoid locking the port due to a USB bug in Windows, disconnect USB connections immediately after running this command. If the port was locked, disconnect and reconnect the cable to reopen the port.	**COMMAND** **#RESET<CR>** **FEEDBACK** **~nn@RESET** ok**<CR><LF>**		Reset the device: **#RESET<CR>**
SN?	Get device serial number.	**COMMAND** **#SN? <CR>** **FEEDBACK** **~nn@SN** serial_num**<CR><LF>**	**serial_num** – 14 decimal digits, factory assigned	Get the device serial number: **#SN? <CR>**

Function	Description	Syntax	Parameters/Attributes	Example
UPGRADE	Perform firmware upgrade. ⓘ Not necessary for some devices. Firmware usually uploads to a device via a command like LDFW. Reset the device to complete the process.	**COMMAND** **#UPGRADE<CR>** **FEEDBACK** **~nn@UPGRADE** ok**<CR><LF>**		Perform firmware upgrade: **#UPGRADE<CR>**
VERSION?	Get firmware version number.	**COMMAND** **#VERSION?** **<CR>** **FEEDBACK** **~nn@VERSION** firmware_version**<CR><LF>**	**firmware_version** – XX.XX.XXXX where the digit groups are: major.minor.build version	Get the device firmware version number: **#VERSION?** **<CR>**
X-GW-PORT-ACTIVE	Set the gateway activation state per port. ⓘ This is an Extended Protocol 3000 command.	**COMMAND** **#X-GW-PORT-ACTIVE** <direction_type>.<port_format>.<port_index>,state**<CR>** **FEEDBACK** **~nn@X-GW-PORT-ACTIVE** <direction_type>.<port_format>.<port_index>,state**<CR><LF>**	The following attributes comprise the port ID: ▪ **<direction_type>** – Direction of the port: ○ IN ○ OUT ○ BOTH ▪ **<port_format>** – Type of signal on the port: ○ HDMI ○ RS-232 ▪ **<port_index>** – The port number as printed on the front or rear panel **state** – Global gateway activation state: ○ OFF – disabled ○ ON – enabled	Activate HDMI input #2 on the gateway: **#X-GW-PORT-ACTIVE** in.hdmi.2.video.1,on**<CR>**
X-GW-PORT-ACTIVE?	Get the gateway activation state per port. ⓘ This is an Extended Protocol 3000 command.	**COMMAND** **#X-GW-PORT-ACTIVE?** <direction_type>.<port_format>.<port_index>**<CR>** **FEEDBACK** **~nn@X-GW-PORT-ACTIVE** <direction_type>.<port_format>.<port_index>,state**<CR><LF>**	The following attributes comprise the port ID: ▪ **<direction_type>** – Direction of the port: ○ IN ○ OUT ○ BOTH ▪ **<port_format>** – Type of signal on the port: ○ HDMI ○ RS-232 ▪ **<port_index>** – The port number as printed on the front or rear panel **state** – Global gateway activation state: ○ OFF – disabled ○ ON – enabled	Get the gateway activation state for HDMI input #2 on the gateway: **#X-GW-PORT-ACTIVE?** in.hdmi.2.video.1**<CR>**

Result and Error Codes

Syntax

In case of an error, the device responds with an error message. The error message syntax:

- **~NN@ERR XXX<CR><LF>** – when general error, no specific command
- **~NN@CMD ERR XXX<CR><LF>** – for specific command
- **NN** – machine number of device, default = 01
- **XXX** – error code

Error Codes

Error Name	Error Code	Description
P3K_NO_ERROR	0	No error
ERR_PROTOCOL_SYNTAX	1	Protocol syntax
ERR_COMMAND_NOT_AVAILABLE	2	Command not available
ERR_PARAMETER_OUT_OF_RANGE	3	Parameter out of range
ERR_UNAUTHORIZED_ACCESS	4	Unauthorized access
ERR_INTERNAL_FW_ERROR	5	Internal FW error
ERR_BUSY	6	Protocol busy
ERR_WRONG_CRC	7	Wrong CRC
ERR_TIMEDOUT	8	Timeout
ERR_RESERVED	9	(Reserved)
ERR_FW_NOT_ENOUGH_SPACE	10	Not enough space for data (firmware, FPGA…)
ERR_FS_NOT_ENOUGH_SPACE	11	Not enough space – file system
ERR_FS_FILE_NOT_EXISTS	12	File does not exist
ERR_FS_FILE_CANT_CREATED	13	File can’t be created
ERR_FS_FILE_CANT_OPEN	14	File can’t open
ERR_FEATURE_NOT_SUPPORTED	15	Feature is not supported
ERR_RESERVED_2	16	(Reserved)
ERR_RESERVED_3	17	(Reserved)
ERR_RESERVED_4	18	(Reserved)
ERR_RESERVED_5	19	(Reserved)
ERR_RESERVED_6	20	(Reserved)
ERR_PACKET_CRC	21	Packet CRC error
ERR_PACKET_MISSED	22	Packet number isn't expected (missing packet)
ERR_PACKET_SIZE	23	Packet size is wrong
ERR_RESERVED_7	24	(Reserved)
ERR_RESERVED_8	25	(Reserved)
ERR_RESERVED_9	26	(Reserved)
ERR_RESERVED_10	27	(Reserved)
ERR_RESERVED_11	28	(Reserved)
ERR_RESERVED_12	29	(Reserved)
ERR_EDID_CORRUPTED	30	EDID corrupted
ERR_NON_LISTED	31	Device specific errors
ERR_SAME_CRC	32	File has the same CRC – no changed
ERR_WRONG_MODE	33	Wrong operation mode
ERR_NOT_CONFIGURED	34	Device/chip was not initialized