8. Programmer's reference

Users can connect to the matrix through Ethernet, serial port or USB. After establishing connection, there is no difference between connection types (except some rare cases, which are uniquely noted).

Lightware matrix routers can be controlled with external devices which can communicate according to the router protocol. Lightware routers have a special protocol, but to interoperate with third party devices, a secondary protocol is also provided.

Please see section 5.8 on page 80 about remote operation and connection setup.

Renewed protocol

The MX-CPU2 processor board works with a similar but renewed protocol as the earlier generation matrix frames with 'CPU1'.

This icon indicates functions which are heavily modified in the MX-CPU2.

8.1. Changing protocols

The router is equipped with multiple router protocols. Different control interfaces can be set to use different protocols. E.g. the Ethernet interface can use the Lightware protocol while the Serial interface uses Protocol#2 at the same time.

The currently used protocol can be viewed or changed any time on the matrix front panel (see section 5.8.5 on page 83) or with protocol commands.

8.2. Protocol description

The protocol description hereinafter stands for **Lightware protocol**.

The matrices accept commands surrounded by curly brackets - { } - and responds data surrounded by round brackets - ( ) - only if a command was successfully executed. All input commands are converted to uppercase, but respond commands can contain upper and lower case letters as well.

Legend for control commands:

<in>	=	input number in 1 or 2 digit ASCII format (01,5,07,16 etc.)
<out>	=	output number in 1 or 2 digit ASCII format
<in/out>	=	input or output port number in 1 or 2 digit ASCII format.*
<in²>	=	input number in 2 digit ASCII format (01, 02, 10, 12 etc.)
<out²>	=	output number in 2 digit ASCII format
<in²/out²>	=	input or output number in 2 digit ASCII format*
<loc>	=	location number in 1, 2 or 3 digit ASCII format
<id>	=	id number in 1 or 2 digit ASCII format
<id²>	=	id number in 2 digit ASCII format
CrLf	=	Carriage return, Line feed (0x0D, 0x0A)
•	=	space character (0x20)
→	=	each command issued by the controller
←	=	each response received from the router

* The command has the same arguments on the input ports and the output port, as well.

ASCII table:

The most frequently used characters are highlighted.

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	00	[NUL]	32	20	[Space]	64	40	@	96	60	`
1	01	[SOH]	33	21	!	65	41	A	97	61	a
2	02	[STX]	34	22	"	66	42	B	98	62	b
3	03	[ETX]	35	23	#	67	43	C	99	63	c
4	04	[EOT]	36	24	$	68	44	D	100	64	d
5	05	[ENQ]	37	25	%	69	45	E	101	65	e
6	06	[ACK]	38	26	&	70	46	F	102	66	f
7	07	[BEL]	39	27	'	71	47	G	103	67	g
8	08	[BS]	40	28	(	72	48	H	104	68	h
9	09	[TAB]	41	29	)	73	49	I	105	69	i
10	0A	[LF]	42	2A	*	74	4A	J	106	6A	j
11	0B	[VT]	43	2B	+	75	4B	K	107	6B	k
12	0C	[FF]	44	2C	,	76	4C	L	108	6C	l
13	0D	[CR]	45	2D	-	77	4D	M	109	6D	m
14	0E	[SOH]	46	2E	.	78	4E	N	110	6E	n
15	0F	[SI]	47	2F	/	79	4F	O	111	6F	o
16	10	[DLE]	48	30	0	80	50	P	112	70	p
17	11	[DC1]	49	31	1	81	51	Q	113	71	q
18	12	[DC2]	50	32	2	82	52	R	114	72	r
19	13	[DC3]	51	33	3	83	53	S	115	73	s
20	14	[DC4]	52	34	4	84	54	T	116	74	t
21	15	[NAK]	53	35	5	85	55	U	117	75	u
22	16	[SYN]	54	36	6	86	56	V	118	76	v
23	17	[ETB]	55	37	7	87	57	W	119	77	w
24	18	[CAN]	56	38	8	88	58	X	120	78	x
25	19	[EM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUB]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESC]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FS]	60	3C	<	92	5C	\	124	7C	l
29	1D	[GS]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RS]	62	3E	>	94	5E	^	126	7E	~
31	1F	[US]	63	3F	?	95	5F	_	127	7F	[DEL]

8.3. Storage memories

The matrix stores many configuration settings and parameters and uses different memories. In some cases it is important to know which setting is stored in which memory.

Available storage memories:

8.3.1. Matrix frame memory (read only) // remain unchanged //

Settings

- Matrix router serial number
- HDCP options
- I/O slot limits

These settings are stored in matrix frame memory and remain after any board swap(ing) – even CPU2 – or any firmware upgrade.

8.3.2. CPU board memory, upgrade resistant // remain unchanged //

Settings

- I/O port and preset names
- EDID lists (F, U, D)
- EDID emulation table (E)
- Input port settings
- Output port settings
- Crosspoint settings
- Crosspoint presets
- Serial port settings
- IP settings
- Genlock settings
- Analog video timings
- Test input multiplexer (FR80)
- Protocol modes
- Remote alert send levels

These settings remain unchanged after any firmware upgrade.

8.3.3. CPU board memory, cleared by upgrade

Settings

- Basic error list

Info: As a matter of fact this error list is stores in a RAM and it is cleared after by every reset or power off. Since the firmware upgrade process ends with a reset so the log is lost, but the whole logged data is stored in the SD card, as well.

8.3.4. SD memory card on the CPU2 board // remain unchanged //

Settings

- Detailed error list

Error log helps the support team if there is any dysfunction. CPU2 board stores the error log with time stamps.

8.3.5. Input and output board memory (read only) // remain unchanged //

Settings

- Manufacturing parameters

An input or output board can stores manufacturing parameters which regards only that board which contains them.

8.4. Switching and control commands

8.4.1. MX-CPU2 Test input and Preview output

MX-FR80R and MX-FR80R

Used in the MX-FR80R or MX-FR65R router frame, the Preview output is directly connected to the 80th output port with a DVI splitter. Therefore this port always outputs the same signal as the 80th output, even if it has a different interface (TP, OPT, etc.).

The 80th input port of the crosspoint is multiplexed between the Test input port and the 8th port of the 10th input card. This switch is independent from the crosspoint state. The selected port (Test input or Input board #10) will be available as the 80th input on the crosspoint switch.

Other frames

All other frames use the Test input and Preview output just like any other ports. These ports are referred as the last port in the crosspoint.

Frame	Test input	Preview output
MX-FR9	in 9	out 9
MX-FR17	in 17	out 17
MX-FR33	in 33	out 33
MX-FR33R	in 33	out 33
MX-FR65R	in 80	out 80
MX-FR80R	multiplexed in 80	distributed out 80

8.4.2. Select 80th input port

Info: Available only for MX-FR80R and MX-FR65R.

Description: Configure the crosspoint's 80th port to use the Test input port or the 8th port of the 10th input card.

Format	Example (MX-FR80R)
Command {TI=<value>}	→ {TI=?}
Response (TI=<value>)CrLf	← (TI=1)CrLf

Explanation: Query Test input state. Test input is selected for the 80th input of crosspoint. The last port on the 10th input board is not used.

Legend: <value>: ? = Query 80th port multiplexer status.
0 = Set port multiplexer to use the 8th port of the 10th input card.
1 = Set port multiplexer to use the Test input port.

Info: The status of the multiplexer is not shown in other crosspoint commands. The crosspoint switching works independent from this setting.

8.4.3. Switch one input to one output

Description: Switch input <in> to output <out>.

Format	Example 1
Command {<in>@<out>}	→ {1@5}
Response (O<out²>●I<in²>)CrLf	← (O05 I01)CrLf

Explanation 1: Input 1 is switched to output 5.

Format	Example 2
Command {<in>@<out>}	→ {2@4}
Response (1LO<out²>)CrLf	← (1LO04)CrLf

Explanation 2: Input 2 to output 4 switch is not made because output 4 is locked.

Info: The response for this command does not show if the output is muted. To check the mute status a separate query has to be used like {VC}. See 8.4.7 on page 141.

Info: To achieve multiple switches executed together, see 8.4.5 on page 140.

8.4.4. Switch one input to all outputs

Description: Switch input <in> to all outputs.

Format	Example
Command {<in>@O}	→ {02@O}
Response (I<in²>●ALL)CrLf	← (I02 ALL)CrLf

Explanation: Input 2 is switched to all outputs.

Info: The response for this command does not show if any of the outputs are muted. To check the mute status a separate query has to be used.

Info: The response for this command does not show if there were some locked outputs which cannot be switched.

8.4.5. Batch switch outputs

Description: The router is capable of switching multiple outputs exactly at the same time. To do this, the normal switch commands have to be used. If the switch commands arrive to the router with less than 10 milliseconds delay, then the router collects the commands and changes the output connections together.

Required circumstances:

- Switch commands have this format: **{**<in>@<out>**}{**<in>@<out>**}**
- The delay between two '}' characters must be below 10 milliseconds
- No other command or junk character is allowed between switch commands
- Affected outputs must not be locked

If any of the above circumstances fail, then the commands will be processed separately and the output connections will change on by one.

Info: The delay timeout applies for the receiving time of characters. Please note that if LAN connection is used then the network may cause additional delays. This could result that batch switching does not occur.

The below example shows a command that resulted batch switching:

One by one commands	Batch commands
→ {02@01}CrLf ← (O01 I02)CrLf	→ {02@01}{05@04}CrLf ← (O01 I02)CrLf ← (O04 I05)CrLf
→ {05@04}CrLf ← (O04 I05)CrLf	

The below example shows a command that does not resulted batch switching, because another command get between:

One by one commands	Batch commands
→ {02@01}CrLf ← (O01 I02)CrLf	→ {02@01}{+06}{05@04}CrLf ← (O01 I02)CrLf
→ {+06}CrLf ← (0MT06)CrLf	← (0MT06)CrLf ← (O04 I05)CrLf
→ {05@04}CrLf ← (O04 I05)CrLf	

Info: The response itself does not show if batch switching happened or not. This assures that a third party controller does not get unknown responses.

8.4.6. View connection on the specified output

Info: Obsolete! Use {VC} instead. See 8.4.7 on page 141.

Description: View connection on output <out>.

Format	Example
Command {?<out>} Response (O<out²>●I<in²>)CrLf	→ {?05} ← (O05 I01)CrLf

Explanation: Viewing connection for output 5. The connected input is 1.

Info: If the output is locked, muted, or both locked and muted, the response format changes. If the output is muted you get a letter 'M', if locked a letter 'L' and if muted and locked at the same time 'U' before the 2 digit numbers (e. g. O05 IL01).

8.4.7. View connection on all outputs

Description: Viewing all outputs' connection results in different response length, because it depends on the frame size. The response below supposes a router having 17 outputs.

Info: The MX-CPU2 responds the connection of Preview Output port as well. The earlier 16x16 or 32x32 frames responded 16 and 32 outputs but with MX-CPU2 the response will be 17 and 33 correspondingly.

Format	Example 1 (MX-FR17)
Command {VC} Response (ALL●<O1>●<O2>●<O3> ●<O4>●<O5>●<O6>●<O7> ●<O8>●<O9>●<O10> ●<O11>●<O121>●<O13> ●<O14>●<O15>●<O16> ●<O17>)CrLf	→ {VC} ← (ALL 02 02 02 05 05 05 08 08 08 08 08 08 08 08 08 08 08)CrLf

Legend 1: All <Ox> indexes show the corresponding output's connection state. If value <O5> equals 04 it means that output 5 is connected to input 4. All <Ox> indexes are two digit ASCII characters (01, 02, 04, etc.).

Explanation 1: Viewing connection for all outputs. Input 2 is connected to outputs 1, 2 and 3. Input 5 is connected to outputs 4, 5 and 6. Input 8 is connected to outputs 7 through 17.

Info: If an output is locked, muted, or both locked and muted, the response format changes. If outputs are muted you get a letter 'M', if locked a letter 'L' and if muted and locked at the same time 'U' before the 2 digit numbers.

Format		Example 2 (MX-FR17)
Command	{VC}	→ {VC}
Response	(ALL●<O1>●<O2>●<O3> ●<O4>●<O5>●<O6>●<O7> ●<O8>●<O9>●<O10> ●<O11>●<O121>●<O13> ●<O14>●<O15>●<O16> ●<O17>)CrLf	← (ALL M02 L02 U02 05 05 05 08 08 08 08 08 08 08 08 08 08)CrLf

Legend 2: Any <Ox> indexes can be a two digit number, or there can be a leading character showing the mute and/or lock state for the corresponding output.

Index	Legend	Explanation
<Ox>	<in²>	<Ox> is connected to <in²>, <Ox> neither muted nor locked.
<Ox>	M<in²>	<Ox> is connected to <in²>, <Ox> is muted, and unlocked.
<Ox>	L<in²>	<Ox> is connected to <in²>, <Ox> is not muted, but locked.
<Ox>	U<in²>	<Ox> is connected to <in²>, <Ox> is muted and locked.

Explanation 2: Viewing connection for all outputs. Input 2 is connected to outputs 1, 2 and 3. Output 1 is muted. Output 2 is locked. Output 3 is muted and locked. Input 5 is connected to outputs 4, 5 and 6. Input 8 is connected to outputs 7 through 16.

8.4.8. View mutes on all outputs

Description: Viewing all outputs' mute state results in different response length, depending on the frame size.

Format		Example (MX-FR17)
Command	{VM}	→ {VM}
Response	(MUT●<M1>●<M2>●<M3> ●<M4>●<M5>●<M6>●<M7> ●<M8>●<M9>●<M10> ●<M11>●<M12>●<M13> ●<M14>●<M15>●<M16> ●<M17>)CrLf	← (MUT 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0)CrLf

Legend: All <Mx> indexes are one digit numbers, showing the mute state for the corresponding output. If <Mx> equals 0 the output x is unmuted. If <Mx> equals 1, the output x is muted.

Explanation: Output 1, 3 and 4 are muted, the other outputs are not muted.

8.4.9. Mute specified output

Description: Mute output <out>. The output signal is turned off.

Format	Example
Command {#<out>}	→ {#03}
Response (1MT<out²>)CrLf	← (1MT03)CrLf

Explanation: Output 3 is muted. No signal is present on output 3 now.

Info: Muting does not change the crosspoint's state, but disables the output itself. This way the last connection can be easily restored with an unmute command.

Info: Switching a muted output does not unmute the output.

8.4.10. Unmute specified output

Description: Unmute output <out>.

Format	Example
Command {+<out>}	→ {+03}
Response (0MT<out²>)CrLf	← (0MT03)CrLf

Explanation: Output 3 is unmuted. Now output 3 is switched to the input it was connected to prior to the mute command.

Info: Unmuting an output makes the previous connection active as the crosspoint's state has not been changed with the muting command, only the output was disabled.

8.4.11. Disconnect any inputs from one output

Description: Switch an output to virtual unconnected input. No signal on output.

Format	Example
Command {0@<out>}	→ {0@3}
Response (O<out²>●I00)CrLf	← (O03 I00)CrLf

Explanation: Inputs are disconnected from output 5 (no input will be connected).

Info: The response for this command is (1LO<out²>) if the output is locked.

Info: Disconnecting acts similar to muting except that the previous connection cannot be restored with an unmute command. A disconnected output can still be muted or unmuted, however this has no real effect in this case.

Info: To make a disconnected output live again another input has to be switched to it.

8.4.12. Lock specified output

Description: Lock output <out>. Output's state cannot be changed until unlocking.

Format	Example
Command {#><out>}	→ {#>05}
Response (1LO<out²>)CrLf	← (1LO05)CrLf

Explanation: Output 5 is locked.

8.4.13. Unlock specified output

Description: Unlock output <out>. The connection on output can be changed.

Format	Example
Command {+<<out>}	→ {+<05}
Response (0LO<out²>)CrLf	← (0LO05)CrLf

Explanation: Output 5 is unlocked.

Info: The router issues the above response regardless of the previous state of the output (either it was locked or unlocked).

8.4.14. Save preset to the specified memory location

Description: Save current crosspoint configuration (output states) to preset <id>.

Format	Example
Command {$<id>}	→ {$4}
Response (SPR<id²>)CrLf	← (SPR04)CrLf

Explanation: Current crosspoint state is saved to preset 4, including the mute state of the outputs.

Info: Lock states are not saved. Lock state is assigned to the physical output of the router. Presets do not affect output locks.

Info: All frames have 32 preset memories.

8.4.15. Load preset from the specified location

Description: Load crosspoint configuration from preset <id>.

Format	Example
Command {%<id>}	→ {%4}
Response (LPR<id²>)CrLf	← (LPR04)CrLf

Explanation: Current crosspoint state is changed according to preset 4, including the mute state of the outputs.

Info: Locked outputs are left unchanged. Presets do not affect output locks.

8.4.16. Preview preset

Description: Preview stored connections in preset <id> without loading it. This results different response length, because it depends on the crosspoint size. The response below supposes a router having 17 outputs.

Format	Example (MX-FR17)
Command {VP#<id>=?} Response (VP#<id>=●<O1>●<O2> ●<O3>●<O4>●<O5>●<O6> ●<O7>●<O8>●<O9>●<O10> ●<O11>●<O121>●<O13> ●<O14>●<O15>●<O16> ●<O17>)CrLf	→ {VP#3=?} ← (VP#3= 02 M02 M01 02 02 01 01 01 01 01 01 01 01 01 01 01)CrLf

Legend: Any <Ox> indexes can be a two digit number, or there can be a leading character showing the mute state for the corresponding output.

Explanation: Viewing connections for preset 3. Input 2 is connected to outputs 1, 2, 4 and 5. Input 1 is connected to all other outputs. Outputs 2 and 3 are muted.

Index	Legend	Explanation
<Ox>	<in²>	<Ox> is connected to <in²>, <Ox> is not muted.
<Ox>	M<in²>	<Ox> is connected to <in²>, <Ox> is muted.

8.4.17. Renaming Presets / Inputs / Outputs

Description: Allows storing names for each preset / input / output. Any 15-byte long string is allowed (15 characters). The router accepts <id> for I/O names depending on the actual frame size. All router models have 32 presets memories.

Rename a preset

Format		Example
Command	{PNAME#<id>=<preset_name>}	→ {PNAME#1=First preset}
Response	(PNAME#<id>=<preset_name>)CrLf	← (PNAME#1=First preset)CrLf

Explanation: Preset 1 was named as "first preset".

Rename an input

Format		Example
Command	{INAME#<id>=<input_name>}	→ {INAME#3=Media_Player}
Response	(INAME#<id>=<input_name>)CrLf	← (INAME#3=Media_Player)CrLf

Explanation: Input 3 was named as "media player".

Rename an output

Format		Example
Command	{ONAME#<id>=<output_name>}	→ {ONAME#2=Monitor#2}
Response	(ONAME#<id>=<output_name>)CrLf	← (ONAME#2=Monitor#2)CrLf

Explanation: Output 2 was named as "monitor#2".

8.4.18. Query names of Presets / Inputs / Outputs

Description: Each preset / input / output name can be read from the router.

Read a preset's name

Format		Example
Command	{PNAME#<id>=?}	→ {PNAME#1=?}
Response	(PNAME#<id>=<preset_name>)CrLf	← (PNAME#1=First preset)CrLf

Explanation: Name for preset 1 is "first preset".

Read an input's name

Format		Example
Command	{INAME#<in>=?}	→ {INAME#3=?}
Response	(INAME#<in>=<input_name>)CrLf	← (INAME#3=Media_Player)CrLf

Explanation: Name for input 3 is "media player".

Read an output's name

Format		Example
Command	{ONAME#<out>=?}	→ {ONAME#2=?}
Response	(ONAME#<out>= <output_name>)CrLf	← (ONAME#2=Monitor#2)CrLf

Explanation: Name for output 2 is "monitor#2".

8.4.19. Set default names of Presets / Inputs / Outputs

Description: Renames **all** preset / input / output names to the default: Preset 1..32 / Input 1..17 / Output 1..17 respectively.

*Info: The <id> field is not relevant here, only has to be a valid one. The command will affect **ALL** Presets / Inputs / Outputs disregarding the actual number that was in the command.*

Reload default preset names

Format		Example
Command	{PNAME#<id>=!}	→ {PNAME#2=!}
Response	(PNAME#<id>= Preset●<id>)CrLf	← (PNAME#2=Preset 2)CrLf

Explanation: All preset names are set to default: "Preset 1", "Preset 2", and so on.

Reload default input names

Format		Example
Command	{INAME#<id>=!}	→ {INAME#4=!}
Response	(INAME#<id>= Input●<id>)CrLf	← (INAME#4=Input 4)CrLf

Explanation: All input names are set to default: "Input 1", "Input 2", and so on.

Reload default output names

Format		Example
Command	{ONAME#<id>=!}	→ {ONAME#3=!}
Response	(ONAME#<id>= Output●<id>)CrLf	← (ONAME#3=Output 3)CrLf

Explanation: All output names are set to default: "Output 1", "Output 2", and so on.

8.5. Communication setup commands

8.5.1. Query IP settings

Description: IP settings can be retrieved from the router with this command.

Format		Example
Command	{IP_CONFIG=?}	→ {IP_CONFIG=?}
Response	(IP_CONFIG=<id> ●<ip_address>●<port> ●<mask>●<gateway>)CrLf	← (IP_CONFIG=0 192.168.2.106 10001 255.255.000.000 192.168.002.001) CrLf

Legend:

Identifier	Description		Default value
<id>	0: fix IP	2: DHCP	0
<ip_address>	IP address		192.168.254.254
<port>	TCP/IP port		10001
<mask>	subnet mask		255.255.0.0
<gateway>	gateway address		0.0.0.0

Explanation: The router has a fix IP address 192.168.2.106 on the 255.255.0.0 subnet with a gateway on 192.168.2.1 and communicates over TCP port 10001.

8.5.2. Reload factory default IP settings

Description: This command sets the router to the factory default IP setup.

Format		Example
Command	{IP_CONFIG=!}	→ {IP_CONFIG=!}
Response	(Changing●IP● configuration…)CrLf (DONE!)CrLf *or* (FAILED!)CrLf	← (Changing IP configuration…)CrLf ← (DONE!)CrLf

Parameters after successful command execution:

Parameter	Value
IP address	192.168.254.254
port number	10001
Subnet mask	255.255.0.0
Gateway	0.0.0.0

Info: This command can be used on all control interfaces (LAN, RS-232 and USB) but the '(DONE!)' response cannot be seen on LAN because the connection is dropped just after the '(Changing IP configuration…)' response.

Info: Factory default setting can be reloaded by the front panel buttons (section 5.8.3 on page 82) or on the front panel LCD menu as well.

8.5.3. Enable DHCP IP setting

Description: After sending this command the router will inquire IP address with DHCP.

Format		Example
Command	{IP_CONFIG=D}	→ {IP_CONFIG=D}
Response	(Changing●IP● configuration…)CrLf (DONE!)CrLf *or* (FAILED!)CrLf	← (Changing IP configuration…)CrLf (DONE!)CrLf

Parameters after successful command execution:

Parameter	Value
IP address	Acquired with DHCP
port number	unchanged
Subnet mask	Acquired with DHCP
Gateway	Acquired with DHCP

Info: This command can be used on all control interfaces (LAN, RS-232 and USB) but the '(DONE!)' response cannot be seen on LAN because the connection is dropped just after the '(Changing IP configuration…)' response.

Info: DHCP setting can be reloaded by the front panel buttons as well (section 5.8.3 on page 82) or on the front panel LCD menu as well.

8.5.4. Query RS-232 baud rate

Description: The RS-232 baud rate can be checked. It works via LAN and RS-232 as well, but if RS-232 is used the command has to be sent with the appropriate baud rate.

Format	Example
Command {RS232BAUD=?} Response (RS232BAUD=<rate>)CrLf	→ {RS232BAUD=?} ← (RS232BAUD=57600)CrLf

Explanation: The router communicates with 57600 baud on the RS-232 port.

Info: RS-232 Baud rate can be checked and set on the front panel LCD menu as well.

8.5.5. Change RS-232 baud rate

Description: The RS-232 baud rate can be set. If RS-232 connection is used, the command has to be sent with the earlier baud rate but the response comes with the new baud rate.

Format	Example
Command {RS232BAUD=<rate>} Response (RS232BAUD=<rate>)CrLf	→ {RS232BAUD=9600} ← (RS232BAUD=9600)CrLf

Explanation: The router RS-232 port is set to 9600 baud.

Possible settings:

<rate>	Baud rate	
9600	9600 baud	
19200	19 200 baud	
38400	38 400 baud	
57600	57 600 baud	(default)
115200	115 200 baud	

Info: RS-232 Baud rate can be checked and set on the front panel LCD menu as well.

8.5.6. Query control protocol

Description: Matrix routers can be controlled with different control protocols. This command queries the active protocol for the used control interface.

Info: Be aware that different control interfaces can be set to use different protocols. E.g. the Ethernet interface can use the Lightware protocol while the Serial interface uses Protocol#2 at the same time.

Info: The response shows only the active protocol for the interface that was used to send the command!

Format	Example
Command {P_?} Response (CURRENT●PROTOCOL●=●#<protocol>)CrLf	→ {P_?} ← (CURRENT PROTOCOL = #1)CrLf

Explanation: The matrix communicates with Lightware protocol.

Possible settings:

<protocol>	Control protocol	
1	Lightware	(default)
2	Protocol #2	
3	LW simple	(not for use)

Info: Control protocol for each interface can be checked by the front panel buttons (section 5.8.5 on page 83) or on the front panel LCD menu as well.

8.5.7. Change control protocol

Description: Matrix routers can be controlled with different control protocols. This command sets the active protocol only for the currently used control interface.

Info: The setting applies only for the interface that was used to send the command!

Info: The USB interface always uses the Lightware protocol, this cannot be changed.

Format		Example
Command	{P_<protocol>}	→ {P_1}
Response	(PROTOCOL●#<protocol>● SELECTED!)CrLf	← (PROTOCOL #1 SELECTED!)CrLf

Explanation: The matrix communicates with Lightware protocol.

Possible settings:

<protocol>	Control protocol	
1	Lightware	(default)
2	Protocol #2	
3	LW simple	(not for use)

Info: Be aware that different control interfaces can be set to use different protocols. E.g. the Ethernet interface can use the Lightware protocol while the Serial interface uses Protocol#2 at the same time.

Info: Control protocol for each interface can be checked by the front panel buttons (section 5.8.5 on page 83) or on the front panel LCD menu as well.

8.5.8. Configure remote alerts

Description: The matrix logs different levels of errors. Configure which level of errors has to be sent out as an alarm message.

Format		Example
Command	{ELEVELSEND#<p>=<0>;<1>;<2>;<3>;<4>}	→ {ELEVELSEND#0=0;0;1;1;1}
Response	(ELEVELSEND#<p>=<0>;<1>;<2>;<3>;<4>)CrLf	← (ELEVELSEND#0=0;0;1;1;1)CrLf

Explanation: The matrix will send an immediate message on all control interfaces when a 'matter', 'error' or 'fatal' level error occurs.

Legend:

<p>:	Adjusted control interface	0 = all 1 = RS-232 2 = LAN 3 = USB
<0>:	'Notice' level events	0 = no immediate message send 1 = immediate message
<1>:	'Warning' level events	0 = no immediate message send 1 = immediate message
<2>:	'Matter' level events	0 = no immediate message send 1 = immediate message
<3>:	'Error' level events	0 = no immediate message send 1 = immediate message
<4>:	'Fatal' level events	0 = no immediate message send 1 = immediate message

See section 8.6.10 on page 153 for more information about error levels.

8.6. Router Status commands

8.6.1. View product type

Description: The router responds its name.

Format	Example
Command {i} Response (<PRODUCT_TYPE>)CrLf	→ {i} ← (I: MX-FR17)CrLf

Explanation: The connected device is a MX-FR17.

Legend: <PRODUCT_TYPE> shows the router model:

Possible responses	crosspoint size
(I:MX-FR9)CrLf	9 x 9
(I:MX-FR17)CrLf	17 x 17
(I:MX-FR33)CrLf	33 x 33
(I:MX-FR33L)CrLf	33 x 33
(I:MX-FR33R)CrLf	33 x 33
(I:MX-FR80R)CrLf	80 x 80

Info: Please note that MX-FR65R gives (I:MX-FR80R) response.

8.6.2. View serial number

Description: The router responds its 8-digit serial number.

Format	Example
Command {s} Response (SN:<SERIAL_N>)CrLf	→ {s} ← (SN:11270142)CrLf

Info: Old devices may have only the last 4 numbers written onto the back of the router.

8.6.3. View firmware version of the CPU

Description: View the CPU firmware revision. To view other controller's firmware version see {FC} command.

Format	Example
Command {f} Response (FW:<FW_VER><s>)CrLf	→ {f} ← (FW:3.3.1r)CrLf

Legend: <FW_VER> is the firmware version. It is followed by <s> string which may indicate special versions. <s>=r indicates standard version.

8.6.4. View crosspoint size

Description: Shows the physical crosspoint size.

Format	Example
Command {getsize} Response (SIZE=<size>)CrLf	→ {getsize} ← (SIZE=17x17)CrLf

Explanation: The router reports that it has a 17x17 crosspoint.

Legend: <size> can be 17x17, 33x33 or 80x80.

8.6.5. View I/O slot limits

Description: Check the number of I/O boards limited by factory.

Format		Example (MX-FR17)
Command	{MAXSLOTS=?}	→ {maxslots=?}
Response	(MAXSLOTS= IB:<num1>,OB:<num2>)CrLf	← (MAXSLOTS=IB:01,OB:01)CrLf

Explanation: The router is limited for one input board and one output board.

Legend: <num1> and <num2> are two digit numbers showing the maximum number of allowed input and output boards correspondingly.

8.6.6. View installed I/O boards

Description: Shows the hardware name and revision of the installed cards. The number of responses varies regarding the frame size (number of slots).

Format		Example
Command	{is}	→ {is}
Response	(SL#●0●<MB_DESC>)CrLf (SL#●1●<OB_DESC>)CrLf (SL#●2●<OB_DESC>)CrLf ... (SL#●51●<IB_DESC>)CrLf (SL#●52●<IB_DESC>)CrLf … (SL●END)CrLf	← (SL# 0 MX-DVI-MB80 SCH_1.1 PCB_1.1)CrLf ← (SL# 1 MX-DVID-OB SCH_2.0 PCB_2.0)CrLf ← (SL# 2 empty)CrLf … ← (SL# 51 MX-DVID-IB SCH_2.0 PCB_2.0)CrLf ← (SL# 52 MX-DVII-HDCP-IB SCH_2.1 PCB_2.1)CrLf … ← (SL END)CrLf

Explanation: The router reports that it has two output and two input slots. There are two input cards and one output card installed, and one output slot is empty.

Legend: Slot 0 represents the motherboard. Slots from 1 to 50 are showing the output boards. Slots from 51 to 100 are showing the input boards.

Legend	Explanation
SL# 0 …	This "slot" represents the motherboard.
SL# 1-50 …	Slots from 1 to 50 are showing the output boards.
SL# 51-100 …	Slots from 51 to 100 are showing the input boards.
SL END	This message indicates the end of the list.

Info: The matrix responds an 'empty' board descriptor for empty physical slots.

8.6.7. View firmware for all controllers'

Description: Shows the firmware versions of all installed controllers. The number of responses depends on the router configuration.

Format		Example (MX-FR17)
Command	{FC}	→ {fc}
Response	(CF●<DESC>)CrLf	← (CF MX-CPU2 FW:3.1.6v SCH_2.2)CrLf
	(CF●<DESC>)CrLf	← (CF MX-CP FW:1.0.8 @ 0x10)CrLf
	…	…
	(CF END)CrLf	← (CF END) CrLf

Explanation: The matrix has an MX-CPU2 processor. There is one control panel in the frame.

8.6.8. View LAN versions

Description: Shows information about the LAN interface.

Format		Example (MX-FR17)
Command	{LAN_VER=?}	→ {LAN_VER=?}
Response	(MAC_ADDR=<mac>)CrLf	← (MAC_ADDR=00-20-4A-C7-AC-C0
	(WEB_VER=<ver1>)CrLf	←)CrLf
	(SERVER_VER=<ver2>)CrLf	← (WEB_VER=1.4.0)CrLf (SERVER_VER=3.0.6)CrLf

Explanation: MAC address, webcontent and webserver versions are shown.

Legend:
- <mac>: MAC address of LAN controller in the matrix.
- <ver1>: Version of built-in website user interface (webcontent).
- <ver2>: Version of LAN controller firmware (webserver).

8.6.9. View router's health

Description: Queries health status. Response varies depending on the frame type.

MX-FR17, MXFR33 and MX-FR33L

Format		Example (MX-FR17)
Command	{ST}	→ {st}
Response	(ST●<DESC>)CrLf	← (ST CPU 3.32V 5.03V 3.05V 5.03V 12.11V 31.6C)CrLf
	…	← (ST FAN#1 1530RPM)CrLf
	(ST●<DESC>)CrLf	← (ST FAN#2 1530RPM)CrLf

Explanation: Internal voltages, temperature and fan speeds shown.

MX-FR33R

Format		Example (MX-FR33R)
Command	{ST}	→ {st}
Response	(ST●<DESC>)CrLf	← (ST CPU 3.31V 5.03V 3.08V 5.03V 11.74V 29.9C)
	…	← (ST FAN#1 1500RPM)
	(ST●<DESC>)CrLf	← (ST FAN#2 1440RPM)

Explanation: Internal voltages, temperature and fan speeds shown.

MX-FR80R

Format		Example (MX-FR80R)
Command	{ST}	→ {st}
Response	(ST●<DESC>)CrLf … (ST●<DESC>)CrLf	← (ST CPU N/A 5.03V 3.27V N/A 12.27V 31.2C)CrLf (ST FAN#1 1500RPM 1470RPM 1500RPM 1470RPM 1500RPM 4770RPM)CrLf (ST FAN#2 1500RPM 1500RPM 1410RPM 1470RPM 1470RPM 4560RPM)CrLf (ST PS#1 Not powered!)CrLf (ST PS#2 12.14V 14.32A 10200RPM)CrLf (ST PS#3 Not seated!)CrLf (ST MB_TOP 1.21V 1.76V 3.31V 3.29V 5.02V 5.06V 12.17V 35.48C 36.63C 36.15C)CrLf (ST MB_BOT 1.19V 1.77V 3.30V 3.31V 5.04V 5.02V 12.13V 34.70C 24.10C 23.35C)CrLf (ST XP_TOP 98 89)CrLf (ST XP_BOT 89 89)CrLf

Explanation: Internal voltages, temperature and fan speeds shown.

8.6.10. View error list

Description: Shows the basic error list since last boot up.

Format		Example (MX-FR17)
Command	{elist=?}	→ {elist=?}
Response	(ELIST#<num>● <elevel>●<code>● <param>●<occ>)CrLf … (ELIST#<num>● <elevel>●<code>● <param>●<occ>)CrLf	← (ELIST#1 Notice BOOT p:2 o:1)CrLf ← (ELIST#2 Notice SERIAL p:0 o:1)CrLf ← (ELIST#3 Notice CARDINIT p:81 o:1) CrLf ← (ELIST#4 Notice CARDINIT p:2 o:1)CrLf ← (ELIST#5 Notice READY p:0 o:1)CrLf

Explanation: There are no errors only standard notices that occur on boot up.

Legend: <num>: line number

<elevel>: NOTICE = Not an error. Initialization information.
WARNING = Possible problem without influencing normal operation.
MATTER = Problem that may lead to further errors.
ERROR = Serious error. Must report to support.
FATAL = Fatal error. Normal operation is not possible.

<code>: Short name for type of log entry.

<param>:Technical parameter.

<occ>: Occurrence number for this type of log entry.

Info: The error list can contain NOTICEs and WARNINGs under normal operation. These entries do not mean that there is any problem with the matrix!

8.7. System commands

8.7.1. Reload factory defaults

Description: Factory default settings can be reloaded for different functions separately. Multiple functions can be entered.

Format	Example
Command {FACTORY=<f1>;<f2>;…;fx}	→ {FACTORY=XPOINT;IOCARDS; EDIDS}
Response (FACTORY●<f1>…)CrLf	← (FACTORY XPOINT…)CrLf
(FACTORY●<f2>…)CrLf	← (FACTORY IOCARDS…)CrLf
…	← (FACTORY EDIDS…)CrLf
(FACTORY●<fx>…)CrLf	

Explanation: Factory default settings reloaded for crosspoint and I/O card configurations and emulated EDIDs.

Legend: <f1>, <f2> are the names of the functions which have to be reset to factory default. Any number of <fx> can be entered, separated by semicolons.

<fx>	Restores factory settings to	Additional response
GENERAL	Control protocols, Front panel state, Alarm message levels	none
IOCARDS	All I/O settings for boards currently in the frame	none
XPOINT	Crosspoint table and configuration (All outputs to in1, unmute, unlock)	none
PRESETS	Crosspoint presets (All output to in1, unmuted), and preset names	(PNAME#1=Preset1) (I1 ALL) (SPR01)…(SPR32)
IONAMES	Input and output names	(INAME#1=Input1) (ONAME#1=Output1)
GENLOCK	All genlock parameters	none
EDIDS	Emulated EDIDs (F49 is default)	none
EDIDMEM	Clear User and Dynamic EDIDs	(DE_OK) (DE_OK)

Info: The response may contain additional messages as the router makes the configurations. These responses can be omitted.

Info: After resetting the needed parameters, the matrix restarts.

8.7.2. Clear HDCP key cache

Description: The matrix stores the HDCP keys from the connected devices. These cached keys can be cleared with this command.

Format	Example
Command {:HDCPRESET}	→ {:HDCPRESET}
Response (DONE)CrLf	← (DONE)CrLf

Explanation: HDCP key cache is cleared.

Info: This function is useful when too many keys were cached and a connected source device cannot accept so many keys.

8.7.3. Set CPU time

Description: The matrix router has a built-in real time clock on the MX-CPU2 processor board. This command allows setting the correct time.

Format		Example (MX-FR17)
Command	{SETTIME=<date>●<time>● UTC+<zone>}	→ {SETTIME= 15.10.2012. 16:52:34 UTC+0100}
Response	(<date>●<time>● UTC+<zone>)CrLf	← (15.10.2012. 16:52:34 UTC+0100)CrLf

Explanation: The matrix router's processor stores the new time.

Legend: <date> Date in DD.MM.YYYY. format.

<time> Time in HH:MM:SS format.

<zone> Time zone related to UTC (Universal coordinated time) in HHMM format.

Info: The UTC, and therefore processor time does not observe daylight saving. For example the Central European time is UTC+1 during winter and UTC+2 during summer.

Info: The CPU time is used mainly for timestamp in the error log.

Info: The MX-CPU2 board has a CR2032 button battery which supplies power to the clock when the matrix is not powered on.

8.7.4. Query CPU time

Description: This command allows reading the CPU time.

Format		Example (MX-FR17)
Command	{GETTIME}	→ {GETTIME}
Response	(<date>●<time>● UTC+<zone>)CrLf	← (15.10.2012. 16:52:34 UTC+0100)CrLf

Explanation: The matrix router responds the current CPU time.

Legend: See above.

8.7.5. Restart matrix router

Description: The matrix router can be restarted without unplugging power.

Format		Example (MX-FR17)
Command	{RST}	→ {RST}
Response	(Booting…)CrLf (<name>●Ready!)CrLf	← (Booting…)CrLf (MX-FR17 Ready!)CrLf

Explanation: The matrix reboots and sends a message when it is ready.

Legend: <name> is the type of the matrix.

Info: The response can be seen only if the connection to the router is still alive.

8.7.6. Switch matrix router to standby

Description: This command works only in the MX-FR80R and MX-FR65R. The frame can be switched to standby without unplugging power. The CPU can still communicate.

Format		Example (MX-FR17)
Command	{PWR_<state>}	→ {PWR_OFF}
Response	(Powered <state>)CrLf	← (Powered off)CrLf

Explanation: The switches to standby mode.

Legend: <state> can be OFF or ON.

Info: The I/O boards do not get any power when in standby mode. The CPU will still work and respond only for status commands.

8.8. EDID router commands

The EDID router manipulates the EDID memory, which has memory locations that are assigned to specific input or output ports. Please read section 5.3 on page 54 about EDID memory structure.

8.8.1. Change EDID on input

Description: Copy EDID from memory location <loc2> to input port <loc1>.

Format	Example
Command {<loc1>:<loc2>} Response (E_SW_OK)CrLf …delay… (E_S_C) CrLf	→ {E5:F10} ← (E_SW_OK)CrLf …delay… ← (E_S_C) CrLf

Explanation: Factory EDID #10 is copied to input 5.

Legend: <loc1> has to be 'Exx'.

<loc2> can be 'Fxx' or 'Uxx' or 'Dxx'.

Info: If <loc2> is 'Fxx' or 'Uxx' then static EDID routing occurs. In this case the router will keep the same EDID on the input until it is changed with another command.

Info: If <loc2> is 'Dxx' then dynamic EDID routing occurs. In this case the router will follow the EDID changes on the output. Every time a different EDID is recognized on the output, it is copied instantly to the input.

Info: The router sends (E_S_C) only if the new EDID is different from the earlier one.

8.8.2. Change EDID on all inputs

Description: Copy EDID from memory location <loc2> to all inputs. Location <loc2> should be 'Fxx' or 'Uxx' for static routing and 'Dxx' for dynamic routing.

Format	Example
Command {EA:<loc2>} Response (E_SW_OK)CrLf …delay… (E_S_C) CrLf	→ {EA:U2} ← (E_SW_OK)CrLf …delay… ← (E_S_C) CrLf

Explanation: User EDID #2 is copied to all inputs.

Info: This operation can take several seconds depending on the frame size.

8.8.3. Save EDID to user memory (Learn EDID)

Description: Learn EDID from <loc2> to <loc1>.

Format	Example
Command {<loc1>:<loc2>} Response (E_SW_OK)CrLf (E_S_C) CrLf	→ {U4:D3} ← (E_SW_OK)CrLf ← (E_S_C) CrLf

Explanation: EDID from output 3 is saved to user EDID #4.

Legend: <loc1> has to be 'Uxx'.

<loc2> can be 'Fxx' or 'Uxx' or 'Dxx' or 'Exx'.

8.8.4. View emulated EDIDs on all inputs

Description: Shows the currently emulated EDIDs for each input. The response length depends on the frame size (number of inputs). The value at the given index (<in1>..<inN>) shows which EDID is used on that particular input.

Format		Example 1 (MX-FR17)
Command	{VEDID}	→ {VEDID}
Response	(VEDID●<IN1>●<IN2>● <IN3>●<IN4>●<IN5>● <IN6>●<IN7> ●<IN8>● <IN9>●<IN10>●<IN11>● <IN12>●<IN13>●<IN14>● <IN15>●<IN16>● <IN17>)CrLf	← (VEDID F049 F049 F049 F049 F049 F049 F049 F049 U002 U002 U002 U002 F049 F049 F049 F049 D004)CrLf

Legend: All <INx> indexes show a <loc> which was copied to that input port.

Explanation: F049 (Factory preset EDID #49) is emulated on all inputs except 9-12 and 17. U002 (User saved EDID #2) is emulated on inputs 9-12. EDID from output 4 is dynamically emulated on input 17.

8.8.5. Watch EDID validity table

Description: Shows EDID validity table, which contains information about the EDID memory states.

Format		Example
Command	{WV<type>}	→ {WV*}
Response	(EV<type>● <VALIDITY_TABLE>)CrLf	← (EVU 00000000000000000000000000000000000000000000000000000000)CrLf ← (EVD 10001001000000000)CrLf ← (EVE 13111111111111111)CrLf

Explanation: There is one '3' on the second position of the emulated EDID table. This means that the emulated EDID on input 2 is changed since the last EDID query on that port.

Legend:

<type>		Response length
F	Factory preset EDIDs	
U	User saved EDIDs	maximum 50
D	Dynamic EDIDs	according to frame size
E	Emulated EDIDs	according to frame size
*	'U', 'D' and 'E' EDIDs	

Each number represents the EDID validity state for the corresponding memory location.

Value	Description
'0'	invalid EDID
'1'	valid EDID
'2'	deleted EDID
'3'	changed EDID

Info: If a changed EDID is queried by the {WH} command (see the next section), its value returns to '1'. The status of a deleted EDID returns to '0' after query.

8.8.6. View EDID header

Description: Shows basic information about EDIDs in the memory.

Format	Example
Command {WH<loc>} Response (EH#<loc>● <EDID_HEADER>)CrLf	→ {WHD14} ← (EH#D14 NEC 1280x1024@60 LCD1970NXp)CrLf

Explanation: Shows the EDID from memory location D14 which is the EDID from the Last attached monitor on output 14.

Legend: Depending on <loc> the query can be for one EDID, all EDID in the block.

<loc>	Result	Response
Fxx	Factory EDID query	header for one EDID
Uxx	User EDID query	
Dxx	Dynamic EDID query	
Exx	Emulated EDID query	
F*	All Factory preset EDIDs	headers for all Factory EDIDs
U*	All User saved EDIDs	headers for 50 user EDIDs
D*	All Dynamic EDIDs	headers from all outputs (frame size)
E*	All Emulated EDIDs	headers from all inputs (frame size)

<EDID_HEADER> consists of 3 fields separated by spaces:

PNPID code The three letter abbreviation of the manufacturer

Preferred resolution The resolution and refresh rate stored in the preferred detailed timing block.

Name The name of display device stored in product descriptor.

The <EDID_HEADER> is '-' for invalid EDIDs.

8.8.7. Download EDID content from the router

Description: EDID hex bytes can be read directly. The router will issue the whole content of the EDID present on memory location <loc> (256 bytes).

Format	Example
Command {WE<loc>} Response (EB#<loc>●<B1> ●<B2>●..●<B256>)CrLf	→ {WEF1>} ← (EB#F1 00 FF FF FF FF FF FF 00 32 F2 00 00 00 .. .. .. 00 00) CrLf

Legend: <B1>..<B256> are space separated hex characters represented in ASCII format.

Explanation: Full EDID from memory location F1 is downloaded.

8.8.8. Upload EDID content to the router

Description: EDID hex bytes can be written directly to the user programmable memory locations.

Sequence:

Step 1. Prepare the router to accept EDID bytes to the specified location <loc> with command *{WL#<loc>}*

Step 2. Router responds that it is ready to accept EDID bytes with *(E_L_S)CrLf*

Step 3. Send 1 block of EDID (1 block consist of 8 bytes of hex data represented in ASCII format) with command *{WB#<num>●<B1>●<B2>●<B3>●<B4>●<B5>●<B6>●<B7>●<B8>}*

Step 4. The router acknowledges with response *(EL#<num>)*

Step 5. Repeat steps 3 and 4 to send the remaining 31 blocks of EDID (32 altogether)

Step 6. After the last acknowledge, the router indicates that the EDID status changed by sending *(E_S_C) CrLf*

Format		Example
Command	{WL#<loc>}	→ {WL#U3}
Response	(E_L_S)CrLf	← (E_L_S) CrLf
Command	{WB#1●<B1>●<B2>●<B3>●<B4>●<B5>●<B6>●<B7>●<B8>}	→ {WB#1 00 FF FF FF FF FF FF 00}
Response	(EL#<num>)CrLf	← (EL#1) CrLf
Command	{WB#2●<B9>●<B10>●<B11>●<B12>●<B13>●<B14>●<B15>●<B16>}	→ {WB#2 38 A3 8E 66 01 01 01 01}
Response	(EL#<num>) CrLf	← (EL#2) CrLf
	⋮	⋮
Command	{WB#32●<B249>●<B250>●<B251>●<B252>●<B253>●<B254>●<B255>●<B256>}	→ {WB#32 36 59 42 0A 20 20 00 96}
Response	(EL#<num>) CrLf	← (EL#32) CrLf
Response	(E_S_C) CrLf	← (E_S_C) CrLf

Legend: <num> represents the sequential number of every 8 byte part of EDID. <num> is between 1 and 32. <B1>..<B256> are the bytes of EDID.

Explanation: Full EDID uploaded to memory location U3.

8.8.9. Delete EDID from memory

Description: Clear EDID from memory location <loc>.

Format		Example
Command	{DE<loc>}	→ {DEU*}
Response	(DE_OK)CrLf (E_S_C)CrLf	← (DE_OK)CrLf (E_S_C)CrLf

Explanation: All user EDIDs are cleared from memory.

Legend: Depending on <loc>, one EDID, or all EDIDs in a block can be cleared.

<loc>	Result
Fxx	Not valid! Factory EDID cannot be deleted. No response.
Uxx	Specified User EDID is deleted.
Dxx	Specified Dynamic EDID is deleted. It will be empty until a new monitor is connected.
Exx	Specified Emulated EDID cleared. By default F49 EDID is copied to it.
F*	Not valid! Factory EDID cannot be deleted. No response.
U*	All User EDIDs are deleted.
D*	All Dynamic EDIDs are deleted. They will be empty until a new monitor is connected.
E*	All Emulated EDIDs are cleared. By default F49 EDID is copied to them.

8.9. Port status commands

8.9.1. Input port status

Description: Shows the actual status of the input ports. The response length changes regarding the frame size. The meaning of the values changes regarding the input board types as the boards have different functions and capabilities.

Format	Example (MX-FR17)
Command {:ISD} Response (ISD●<INPUT_D>)CrLf	→ {:ISD} ← (ISD 1133377700110000000000000000000007)CrLf

Explanation: The first input board is a HDMI board. Input 1 and 2 have a connected source but no signal. Inputs 3-5 have DVI signals and inputs 6-8 have HDMI signals. The second input board is a DVI board. Input 11 and 12 have DVI signals. The Test Input port has a HDMI signal.

Legend: <INPUT_D> may contain 9, 17, 33 or 80 hexadecimal numbers. Each number represents the state for the corresponding input port. (The response characters are grayed in eights for easier reading only.)

Port state sensing availability is differs with input board types. The meaning of the responded number depends on the actual board type for that port. The binary representation of the responded hexadecimal numbers is shown below.

Board type	3. bit (MSB)	2. bit	1. bit	0. bit (LSB)
MX-DVID-IB	0	0	0	clock detect
MX-DVI-TP-IB	0	0	0	clock detect
MX-DVI-TP-IB+	0	0	0	clock detect
MX-DVI-OPT-IB	0	0	0	laser + clock
MX-DVIDL-IB	0	0	0	clock detect
MX-DVIDL-OPT-IB	0	0	0	laser + clock
MX-HDMI-IB	0	HDMI mode	signal detect	source 5V
MX-DVI-HDCP-IB	0	HDMI mode	signal detect	source 5V
MX-HDMI-TP-IB	0	HDMI mode	signal detect	source 5V
MXD-HDMI-TP-IB	0	HDMI mode	signal detect	source 5V
MX-HDMI-OPT-IB	HDCP active	HDMI mode	TX detect	clock detect
MX-DVII-HDCP-IB	analog signal	HDCP active	digital signal	source 5V
MXD-UMX-IB	analog signal	HDCP active	digital signal	source 5V
MX-3GSDI-IB	video detect	audio detect	type: 01=SD, 10=HD, 11=3G	
MX-TPS-IB, -S, -A	HDCP active	HDMI mode	clock detect	TPS link pres.
MX-HDMI-3D-IB, -S, -A	HDCP active	HDMI mode	clock detect	source 5V

- Source 5V: The connected source sends 5V
- Clock detect: TMDS clock is present
- Laser + Clock: Laser detected and TMDS clock is present
- Signal Detect: Video signal is present (TMDS stream can be recognized)
- HDMI mode: Incoming signal is HDMI
- HDCP active: Incoming signal is encrypted
- TX detect: Communication with optical transmitter is OK
- Analog signal: Video signal is present on analog input
- Digital signal: Video signal is present on digital input

Info: Both Clock Detect or Signal Detect can be used to determine if there is an incoming signal.

8.9.2. Output port status

Description: Shows the actual status of the output ports. The response length changes regarding the frame size. The meaning of the values changes regarding the output board types as the boards have different functions and capabilities.

Format	Example (MX-FR17)
Command {:OSD} Response (ISD●<OUTPUT_D>)CrLf	→ {:OSD} ← (OSD 01000000101100000000000000000000)CrLf

Explanation: There are four DVI sinks connected on ports 2, 9, 11 and 12. No other output port is connected.

Legend: <OUTPUT_D> may contain 9, 17, 33 or 80 hexadecimal numbers. Each number represents the state for the corresponding output port. (The response characters are grayed in eights for easier reading only.)

Port state sensing availability is differs with input board types. The meaning of the responded number depends on the actual board type for that port. The binary representation of the responded hexadecimal numbers is shown below.

Board type	3. bit (MSB)	2. bit	1. bit	0. bit (LSB)
MX-DVID-OB	0	0	0	receiver sense
MX-DVI-TP-OB	0	0	0	0
MX-DVI-TP-OB+	0	0	0	hotplug detect
MX-DVI-OPT-OB	0	0	0	0
MX-DVI-OPT-OB-R	0	0	0	0
MX-DVIDL-OB	0	0	0	hotplug detect
MX-DVIDL-OPT-OB	0	0	0	0
MX-HDMI-OB	0	0	0	receiver sense
MX-DVI-HDCP-OB	0	0	0	receiver sense
MX-HDMI-TP-OB	0	0	0	1 (fixed)
MXD-HDMI-TP-OB	0	0	0	receiver sense
MX-HDMI-OPT-OB	HDMI mode	HDCP active	RX detect	laser enable
MX-TPS-OB; -A; -S	0	0	0	TPS link pres.
MX-HDMI-3D-OB; -A; -S	0	0	0	receiver sense

- Receiver Sense: TMDS termination present in connected device
- Hotplug Detect: Hotplug signal is presented by the connected device
- Laser Enable: Optical transmitter is active on output
- RX detect: Communication with optical receiver is OK
- HDMI mode: Incoming signal is HDMI
- HDCP active: Incoming signal is encrypted

Info: Both Receiver Sense or Hotplug Detect can be used to determine if there is an attached monitor or other sink device.

8.9.3. All port status

Description: Shows the actual status of all input and output ports.

Format	Example (MX-FR17)
Command {PS} Response (PS● <INPUT_D>, <OUTPUT_D>)CrLf	→ {PS} ← (PS 113337770011000000000000000000007,01000000101100000000000 000000000)CrLf

Legend: <INPUT_D> and <OUTPUT_D> is the same as for {:ISD} and {:OSD} commands. The input and output state tables are separated with a comma ",". See 8.9.1 on page 160 and 8.9.2 on page 161 for detailed legend.

8.10. Genlock related commands

8.10.1. Set the genlock type

Description: The most fitting genlock type can be chosen for the synchronized crosspoint switching.

Format		Example
Command	{GENLOCK=?}	→ {genlock=?}
Response	(GENLOCK=<gls><term>;<ttld>;<ttlp>;<gvid>;<row>;)CrLf	← (GENLOCK=1;1;2;1;720x480P60;1;)CrLf

Explanation: The crosspoint switching is locked for the preview output of CPU2 card with 4,35 µs delay.

Legend:

<gls> **Genlock source [0 .. 3]:**
0 = Test input on CPU card
1 = Preview output on CPU CPU card
2 = Analog bi-level-sync or tri-level-sync
3 = TTL signal on the BNC

<term>: **75 ohm termination (if the <gls> = 2 or 3) [0, 1]:**
(if the <gls> = 2 or 3)
0 = 75 ohm termination is disabled
1 = 75 ohm termination is enable

<ttld> **TTL delay [0 .. 255]:**
(if the <gls> = 3)
t = 3.75 µs + X · 0,3 µs

<ttlp> **TTL polarity [0, 1]:**
(if the <gls> = 3)
0 = trigger on falling edge (from high to low)
1 = trigger on rising edge (from low to high)

<gvid> **Video format for genlock [string]:**
(calculated value, cannot modify)
empty string :
If there is no genlock signal on the selected source
Hres and Scan (e.g. 1080P) :
if the <gls> = 2
Hres x Vres Scan and Freq (e.g. 1920x1080P60) :
If the <gls> = 0 or 1

<row> **Currently used row of the genlock table [1 .. 255]:**
(calculated value, cannot modify)
If it is 255, there is no proper resolution and the switching takes effect immediately after vsync signal

8.10.2. Query the genlock type

Description: Checks the type of the genlock.

Format		Example
Command	{GENLOCK=?}	→ {genlock=?}
Response	(GENLOCK=<gls><term>;<ttld>;<ttlp>;<gvid>;<row>;)CrLf	← (GENLOCK=1;1;2;1;720x480P60;1;)CrLf

Explanation: The response is the same as the set command's one.

Legend: Please read section 8.10.1 (Set the genlock type) on page 162.

About the genlock formats

Genlock settings (e.g. timings) can be determined automatically. The matrix is able to select the corresponding resolution - based on the incoming video – from a lookup table and use the belonging parameters for the switching.

Lightware matrices contain two lookup tables with 10 – 10 rows. The first table is for the analog (from row nr. 0 to row nr. 9) and the second one (from row nr. 10 to row nr. 19) is for the digital video formats. The first 7 rows contain a resolution and its settings the others are empty by factory defaults. These are predefined values but all of them are variable by the user.

The tables can be reached and set by the GLFORMAT command.

8.10.3. Set the genlock tables

Description: The switching conditions of the genlock can be set manually.

Format		Example
Command	{GLFORMAT#<trow>=<fid>; <oln>;<oen>;<eln>;<een>;<hsd>}	→ {GLFORMAT#1=480P; 10;1;0;0;30;}
Response	(GLFORMAT#<trow>=<fid>; <oln>;<oen>;<eln>;<een>;<hsd>;)CrLf	← (GLFORMAT#1=480P; 10;1;0;0;30;)CrLf

Explanation: The values of the first row of the first table (analog) can be seen.

Legend:

<trow> **Row of the format table [1 ..19]:**
Row number (1,3,4 etc.)

<fid>: **Format identifier [string]:**
Hres and Scan (e.g. 1080P) for analog resolutions:
Hres x Vres Scan and Freq (e.g. 1920x1080P60) for digital resolutions

<oln> **Odd frame switching line number [0 .. 500]:**
The line number of the odd frame where the switching starts

<oen> **Odd frame switching enable [0, 1]:**
0 = disabled
1 = enabled

<eln> **Even frame switching line number [0 .. 500]**
The line number of the even frame where the switching starts

<een> **Even frame switching enable [0, 1]:**
0 = disabled
1 = enabled

<hsd> **Delay after Hsync [0 .. 65536]:**
t = X · 0,15 µs

8.10.4. Query the genlock format

Description: Checks the settings of the previously set genlock format options.

Format		Example
Command	{GLFORMAT#<trow>=?}	→ {glformat#1=?}
Response	(GLFORMAT#<trow>=<fid>; <oln>;<oen>;<eln>;<een>;<hsd>)CrLf	← (GLFORMAT#1=480P; 10;1;0;0;30;)CrLf

Explanation: The values of the first row of the first table (analog) can be seen.

Legend: Please read section 8.10.3 (Set the genlock tables) on page 163.

8.10.5. Reset genlock tables to factory default

Description: The rows of the genlock format tables can be reloaded with the factory default values.

Format		Example
Command	{GLFORMAT#<trow>=FACTORY}	→ {glformat#1=factory}
Response	(GLFORMAT#<trow>=<fid>; <oln>;<oen>;<eln>;<een>;<hsd>)CrLf	← (GLFORMAT#1=480P; 10;1;0;0;30;)CrLf

Explanation: The first row of the first table is reset to factory default values.

Legend: Please read section 8.10.3 (Set the genlock tables) on page 163.

8.11. TPS I/O board specific commands

8.11.1. Set the parameters of a TPS input or output port

Description: Set the values of a TPS port. The formats and the usage are the same in case of input and output ports.

Format		Example
Command	{:TPS#<in/out>@<S>I/O= <mod>;<eth>}	→ {:tps#1@so=H;1}
Response	(TPS#<in/out>@<S/A>I/O= <mod>;<eth>;<cmod>;<pwr>;<upl>; <qual>;<err>;<len>;<rid>;<tmp>)CrLf	← (TPS#1@SO=H;1;H;0;0;1716161 7;23222020;0;0384;38;)CrLf

Explanation: The last measured and status values are in the response.

Legend:

<in>/<out>: **Input or output port number:**
Input or output number in 1 or 2 digit ASCII format (01,3,04 etc.)

<I/O>: **Input or output port type:**
I = input
O = output

<mod> **Mode of the selected TPS port:**
"**H**": HDBaseT
"**L**": Longreach
"**A**": Automatic (Default)
"**1**": Low power 1, RS-232 only
"**2**": Low power 2, RS-232+ETH

<eth> **Ethernet status of the selected port [0 .. 15]:**
0: Ethernet is disabled
1: Ethernet is enabled (Default)
2 – **15**: Reserved for future use

The following parameters cannot be set, they only appear in response.

<cmod> **The current mode of the selected TPS port:**
"**E**": Ethernet fallback
"**1**": Low power 1, RS-232 only
"**2**": Low power 2, RS-232+ETH
"**L**": Longreach
"**H**": HDBaseT
"**0**"(zero): Link is not present
"**-**"(hyphen): The chip is under external process
(e.g. during boot, fw upgrade)

<pwr> **Remote power source (+12V) of the board [0,1]:**
(belong to the board; all ports give the same value on a board)
"**0**"(zero): Not connected
"**1**" Connected

<upl> **Ethernet uplink status [0,1] of the board:**

(belong to the board; all ports give the same value on a board)
"**0**"(zero): Ethernet is inactive
"**1**": Ethernet is active

Info: This parameter belongs to the Ethernet port only. The uplink status of a TPS input or output port is not available.

<qual> **Quality of the link[HEX number]**
Measured values give information about the quality of the link.

<err> **Maximum measured error[HEX number]**
Measured values give information about the quality of the cable.

<len> **Length of the cable**
Measured length of the link cable. If the cable is shorter than 20m the response is 0. If the cable is longer than 110m the response is 1000. If there is no data the response is 0, as well.

<rid> **Remote device identifier[HEX number]:**
Lightware devices have a unique ID for identifying each other.

<tmp> **Temperature [integer]:**
Measured temperature of the board in Celsius.

8.11.2. Query TPS parameters of a TPS input or output port

Description: Check the values of a TPS port. The formats and the usage are the same in case of input and output ports.

Format		Example	
Command	{:TPS#<in/out>@<S>I/O=?}	→	{:tps#1@so=?}
Response	(TPS#<in/out>@<S>I/O= <mod>;<eth>;<cmod>;<pwr>;<upl>; <qual>;<err>;<len>;<rid>;<tmp>)CrLf	←	(TPS#1@SO=H;1;H;0;0;17161617;23222020;0;0384;38;)CrLf

Explanation: HDBaseT mode and enabled Ethernet are set on the 1st output port.

Legend: See 8.11.2 on page 165 for detailed legend.

8.12. RICOD related commands

8.12.1. Set RICOD MASTER command on input

Description: Sets the RICOD command for the selected input port.

Format		Example
Command	{:RICOD#<in>@<S/A>I=<A1><A2>;<B>;<C>;}	→ {:RICOD#1@SI=10;2;1;}
Response	(RICOD#<in>@<S>I=<A1><A2>;<B>;<C>;)CrLf	← (RICOD#1@SI=10;2;1;)CrLf

Explanation: RICOD control is enabled on the first input port, which unlocks the remote device and selects the second video input and the first audio input port on it.

Legend:

<in>: **Input port number:**
Input number in 1 or 2 digit ASCII format (01, 3, 04 etc.)

<S/A>: **Affected ports:**
S = single selected input
A = all inputs

<A1> **RICOD enable parameter on the input:**
1: The RICOD function is enabled to this input port on the local device (where the command was given).
0: The RICOD function is disabled to this input port on the local device (where the command was given).

<A2> **Remote lock enable parameter:**
1: The front panel buttons (in case of transmitters) or the output (in case of video matrix) is locked on the remote device.
0: The front panel buttons (in case of transmitters) or the output (in case of video matrix) is unlocked on the remote device.

Info: *The remote lock function takes effect only if RICOD is enabled by <A1>.*

<B> **The selected video input:**
"**-**" (hyphen) = There is no video switch command.
"**A**" = Automatic input select must be performed on the remote device if available. (i.e. Autoselect function)
"**0**" (zero) = The output needs to be muted if available. (switch to zero)
"1" , "2" , .. , "80" (number between 1 and 80) - Use the given input number if available.

<C> **The selected audio input:**
"**-**" (hyphen) = There is no video switch command.
"**A**" = Automatic input select must be performed on the remote device if available. (i.e. Autoselect function)
"**0**" (zero) = The output needs to be muted if available. (switch to zero)
"1","2",..,"80" (number between 1 and 80) - Use the given input number if available.

Info: *If the first character of <A1><A2> is zero, then no command is sent, the RICOD function is disabled on this input.*

8.12.2. Query the set RICOD MASTER command on input

Description: Checks the status of the previously set RICOD command for the selected input port.

Format		Example
Command	{:RICOD#<in>@<S/A>I=?}	→ {:ricod#1@si=?}
Response	(RICOD#<in>@<S>I=<A1><A2>;<B>;<C>;)CrLf	← (RICOD#1@SI=11;1;1;)CrLf

Explanation: RICOD command was enabled on the first input port, which locks the remote device and selects the first video and audio input port on it.

Legend: Please read section 8.12.1 (Set RICOD MASTER command on input) on page 166.

8.12.3. Set RICOD SLAVE status on output

Description: Enables or disables the reception of RICOD commands over the selected output port of the local device.

Format		Example
Command	{:RICOD_SLEN#<out>@<S/A>O=<num>}	→ {:RICOD_SLEN#1@SO=1}
Response	(RICOD_SLEN#<out>@<S>O=<num>)CrLf	← (RICOD_SLEN#1@SO=1)CrLf

Explanation: RICOD functionality is enabled on the first output port on the local device.

Legend:

<out> **Output port number:**
Output number in 1 or 2 digit ASCII format (01, 3, 04 etc.)

<S/A>: **Affected ports:**
S = single selected output
A = all outputs

<num> **RICOD enable parameter on output:**
1: The RICOD functionality is enabled on the given output port. If a RICOD command is detected it will be executed. (If it is possible.)
0: The RICOD functionality is disabled on the given output port. Any incoming RICOD command will be rejected.

8.12.4. Query RICOD SLAVE status on output

Description: Checks the previously set RICOD status for the selected output port of the local device.

Format		Example
Command	{:RICOD_SLEN#<out>@<S/A>O=?}	→ {:RICOD_SLEN#1@SO=?}
Response	(RICOD_SLEN#<out>@<S>O=<num>)CrLf	← (RICOD_SLEN#1@SO=1)CrLf

Explanation: RICOD functionality is enabled on the first output port on the local device.

Legend: Please read section 8.12.3 (Set RICOD SLAVE status on output) on page 167.

8.13. RS-232 over fiber commands

Maximum 64 byte data can be sent and 54 byte data can be received at once.

Info: It is also important that the control interfaces on the router (USB, IP and RS232) have 57600 bit/sec maximum bandwidth, so heavy traffic should be avoided.

Warning: If the endpoint (the controlled device) sends more than 54 bytes at once without at least 100ms break between the data packets then the sent data could be lost.

The data rate can be 9600, 14400, 19200, 38400, 57600 baud. There is one stop bit and no parity bit.

In case of HDMI-OPT-TX100R / HDMI-OPT-TX200R and MX-HDMI-OPT-IB connection, the baud rate can be set on the HDMI-OPT transmitter unit via a rotary switch.

In case of MX-HDMI-OPT-OB and HDMI-OPT-RX100R / HDMI-OPT-RX200R connection, the baud rate can be set on the router via protocol command or from the LCD menu. The baud rate can be set independently on each port.

There are two methods for sending data: ASCII and binary modes.

8.13.1. Sending data in text format from the router to the endpoint

Description: Sends the data from the matrix's input or output port in text format which is after the equal sign.

Format	Example
Command {:S#<in^2/out^2>@<S>I/O=<ascii text>}	→ {:s#17@so=Blind text\r\n}
Response No response by the matrix	No response by the matrix

Explanation: 'Blind text' with <CrLf> is sent out on the 17th output.

Legend: <in^2>/<out^2>: **Input or output port number:**
Input or output number in 1 or 2 digit ASCII format (01,3,04 etc.)
<I/O>: **Input or output port type:**
I = input
O = output
<ascii text> **ASCII text:**
Text to be sent

* The matrix does not response. If the remote controlled device responses the matrix is able to receive and show it.

Escape characters

The text may contain any characters except { and }. As these characters are used by command framing, there is no way to send them as pure ascii characters.

The text may contain escape sequences. The following escape sequences are supported: \r (carriage return); \n (new line); \t (bs); \x [hex code]

For example, the controller wants to send the next string to the second output:

{power on}\r\n

The command is the following:

{:S#2@SO=\x7Bpower on\x7D\r\n}

It is possible to send the real characters (new line-carriage return) instead of \r\n, but curly brackets must be escaped. Other characters can also be escaped if it is preferred:

{:S#1@SO=\x7Bpower\x20on\x7D\r\n}

8.13.2. Sending data in binary format from the router to the endpoint

Description: Sends the data from the matrix's input or output port in binary format which is after the equal sign.

Format	Example
Command {:B#<in^2/out^2>@<S>I/O=<hex string>}	→ {:b#17@so=0d0aad}
Response No response by the matrix	No response by the matrix

Explanation: '0D 0A AD is sent out on the 17th output.

Legend: <in^2>/<out^2>: **Input or output port number:**
Input or output number in 1 or 2 digit ASCII format (01,3,04 etc.)
<I/O>: **Input or output port type:**
I = input
O = output
<hex string> **HEX string:**
String to be sent

* The matrix does not response. If the remote controlled device responses the matrix is able to receive and show it.

Every 2 characters represent a hexadecimal code. The maximum length of the data is 64 characters, so the max length of the string is 128 char. With this method it is possible to send any special characters.

Receiving data from the far endpoint

Every port can operate either in ASCII or Binary mode. The mode can be set up with the :SERIAL command, see the next chapter. Depending on the selected mode, different messages are sent by the router when it receives data from the far endpoint.

These messages arrive from the router asynchronously without any query command. (The router sends out the message immediately when it receives the data). Therefore it may happen that this message inserted between another command from the controller and the response from the router. The controller must be able to handle this case. For example a simple switch:

- Controller: {2@3} //sending a switch command
- Router: (S#I3=Powered on) //asynchrony serial message
- Router: (O03 I02) //response to the switch command

ASCII

When the router receives a message, the next message will be sent to the controllers:

(S#In=[received text]) or

(S#On=[received text])

where n is the port number, I refers input, O stands for the output ports.

The received data is represented as plain ASCII text and the maximum length of it is 54 byte. The (and) characters are frame delimiters, so they cannot be inside a message. Therefore all (characters will be replaced to \x28, while all) will be replaced to \x29 escape sequences, while \ will be escaped as \\. No other characters will be escaped.

If the programmer of the controller doesn't want to parse escape sequences (it is in fact just a sprintf(...) function call), binary mode should be used.

Example:

Far endpoint sends: Simple math: 2\[3-(2+8)]. Solve it!

The router sends: (S#I1=Simple math: 2\\[3-\x282+8\x29]. Solve it!)

Binary

(B#In=[received text as binary data, e.g. 736F6D657468696E67]) or

(B#On=[received text as binary data, eg. 736F6D657468696E67])

where n is the port number, I refers input, O stands for the output ports.

The received text is translated to binary form. The maximum length of the received text is 54 byte, so the length of the hex data can be up to 108 characters.

Example

Far endpoint, connected to input port 1 is sending data, the router sends:
(B#I1=736F6D657468696E67)

8.13.3. Set up serial pass-through parameters

Description: The properties of the serial pass-through can be modified with this command on the input and the output side.

Format		Example
Command	{:SERIAL#<in^2/out^2>@<S>I/O=<a>;<b>;<c>;<d>}	→ {:serial#24@so=1;9600;1;1}
Response	(SERIAL#<in^2/out^2>@<S>I/O=<a>;<b>;<c>;<d>)	← (SERIAL#24@SO=1;9600;1;1)

Explanation: Serial pass-through sending and receiving is enabled on the 24th output port with 9600 baud.

Legend: <in^2>/<out^2>: **Input or output port number:**
Input or output number in 1 or 2 digit ASCII format (01,3,04 etc.)

<I/O>: **Input or output port type:**
I = input
O = output

<a> **Receiving is enabled:**
0: Incoming data is rejected.
1: The incoming data is sent to the controllers in ASCII mode. (default)
2: The incoming data is sent to the controllers in HEX mode.

<b> **Actual baud rate:**
9600 (default)
14400
19200
38400
57600

<c> **Serial pass-through enable:**
1: Serial pass-through capabilities are enabled on this port.
0: Serial pass-through capabilities are disabled on this port.

The following parameter cannot be set, they only appear in response.

<d> **Serial pass-through capable device is present:**
1: Serial link is active
0: There is no active serial link

8.13.4. Query serial pass-through parameters

Description: The properties of the serial pass-through can be queried with this command on the input and the output side.

Format		Example
Command	{:SERIAL#<in^2/out^2>@<S>I/O=?}	→ {:serial#24@so=?}
Response	(SERIAL#<in^2/out^2>@<S>I/O=<a>;<b>;<c>;<d>)	← (SERIAL#24@SO=1;9600;1;1)

Explanation: Serial pass-through sending and receiving is enabled on the 24th output port with 9600 baud.

Legend: Please read section 8.13.3 (Set up serial pass-through parameters) on page 170.

8.14. RS-232 over TPS commands

Maximum 128 byte data can be sent and 64 byte data can be received at once.

Info: It is also important that the control interfaces on the router (USB, IP and RS232) have 57600 bit/sec maximum bandwidth, so heavy traffic should be avoided.

Warning: If the endpoint (the controlled device) sends more than 54 bytes at once without at least 100ms break between the data packets then the sent data could be lost.

The data rate can be 9600, 14400, 19200, 38400, 57600 baud. The number of the stop bit(s) and the parity can be set up as well.

If the TPS link operation is HDBaseT (and not long reach) mode and if there is no video signal transmission then the link can only operate on 9600 baud data rate.

It is important to be aware that the communication parameters are not detected automatically, so the right values must be set for both the input and the output boards.

There are two methods for sending data: ASCII and binary modes.

8.14.1. Sending data in text format from the router to the endpoint

Description: Sends the data from the matrix's input or output port in text format which is after the equal sign.

Format	Example
Command {:S#<in^2/out^2>@<S>I/O=<ascii text>}	→ {:s#9@so=Blind text\r\n}
Response No response by the matrix	No response by the matrix

Explanation: 'Blind text' with <CrLf> is sent out on the 9th output.

Legend: <in^2>/<out^2>: **Input or output port number:**
Input or output number in 1 or 2 digit ASCII format (01,3,04 etc.)
<I/O>: **Input or output port type:**
I = input
O = output
<ascii text> **ASCII text:**
Text to be sent

* The matrix does not response. If the remote controlled device responses the matrix is able to receive and show it.

Escape characters

The text may contain any characters except { and }. As these characters are used by command framing, there is no way to send them as pure ascii characters.

The text may contain escape sequences. The following escape sequences are supported: \r (carriage return); \n (new line); \t (bs); \x [hex code]

For example, the controller wants to send the next string to the second output:

{power on}\r\n

The command is the following:

{:S#2@SO=\x7Bpower on\x7D\r\n}

It is possible to send the real characters (new line-carriage return) instead of \r\n, but curly brackets must be escaped. Other characters can also be escaped if it is preferred:

{:S#1@SO=\x7Bpower\x20on\x7D\r\n}

8.14.2. Sending data in binary format from the router to the endpoint

Description: Sends the data from the matrix's input or output port in binary format which is after the equal sign.

Format	Example
Command {:B#<in^2/out^2>@<S>I/O=<hex string>}	→ {:b#9@so=0d0aad}
Response No response by the matrix	No response by the matrix

Explanation: '0D 0A AD is sent out on the 9th output.

Legend: <in^2>/<out^2>: **Input or output port number:**
Input or output number in 1 or 2 digit ASCII format (01,3,04 etc.)
<I/O>: **Input or output port type:**
I = input
O = output
<hex string> **HEX string:**
String to be sent

* The matrix does not response. If the remote controlled device responses the matrix is able to receive and show it.

Every 2 characters represent a hexadecimal code. The maximum length of the data is 64 characters, so the max length of the string is 128 char. With this method it is possible to send any special characters.

Receiving data from the far endpoint

Every port can operate either in ASCII or Binary mode. The mode can be set up with the :SERIAL command, see the next chapter. Depending on the selected mode, different messages are sent by the router when it receives data from the far endpoint.

These messages arrive from the router asynchronously without any query command. (The router sends out the message immediately when it receives the data). Therefore it may happen that this message inserted between another command from the controller and the response from the router. The controller must be able to handle this case. For example a simple switch:

- Controller: {2@3} //sending a switch command
- Router: (S#I3=Powered on) //asynchrony serial message
- Router: (O03 I02) //response to the switch command

ASCII

When the router receives a message, the next message will be sent to the controllers:

(S#In=[received text]) or

(S#On=[received text])

where n is the port number, I refers input, O stands for the output ports.

The received data is represented as plain ASCII text and the maximum length of it is 54 byte. The (and) characters are frame delimiters, so they cannot be inside a message. Therefore all (characters will be replaced to \x28, while all) will be replaced to \x29 escape sequences, while \ will be escaped as \\. No other characters will be escaped.

If the programmer of the controller doesn't want to parse escape sequences (it is in fact just a sprintf(...) function call), binary mode should be used.

Example:

Far endpoint sends: Simple math: 2\[3-(2+8)]. Solve it!

The router sends: (S#I1=Simple math: 2\\[3-\x282+8\x29]. Solve it!)

Binary

(B#In=[received text as binary data, e.g. 736F6D657468696E67]) or

(B#On=[received text as binary data, eg. 736F6D657468696E67])

where n is the port number, I refers input, O stands for the output ports.

The received text is translated to binary form. The maximum length of the received text is 54 byte, so the length of the hex data can be up to 108 characters.

Example

Far endpoint, connected to input port 1 is sending data, the router sends:

(B#I1=736F6D657468696E67)

8.14.3. Set up serial pass-through parameters

Description: The properties of the serial pass-through can be modified with this command on the input and the output side.

Format		Example
Command	{:SERIAL#<in^2/out^2>@<S>I/O=<a>;<b>;<c>}	→ {:serial#9@so=1;9600;8n1}
Response	(SERIAL#<in^2/out^2>@<S>I/O=<a>;<b>;<c>)	← (SERIAL#9@SO=1;9600;8N1)

Explanation: Serial pass-through sending and receiving is enabled on the 9th output port with 9600 baud.

Legend: <in^2>/<out^2>: **Input or output port number:**
Input or output number in 1 or 2 digit ASCII format (01,3,04 etc.)

<I/O>: **Input or output port type:**
I = input
O = output

<a> **Receiving mode:**
0: Disabled. All incoming data is ignored - Sending data is still possible of course.
1: The incoming data is sent to the controllers in ASCII mode. (default)
2: The incoming data is sent to the controllers in HEX mode.

<b> **Actual baud rate:**
9600
14400
19200
38400
57600 (default)
115200

<c> **Port setting:**
(according to the standardized formats e.g. 8N1)
/8 bit, No parity, 1 stop bit – this is the default setting/
First character: number of data bits: 5, 6, 7 or 8 (default)
Second character: parity bit. Possible values are:
N: No parity (default)
O: Odd parity
E: Even parity
M: Fixed high (Mark)
S: Fixed low (Space)
Third character: number of stop bits: 1 (default) or 2

8.14.4. Query serial pass-through parameters

Description: The properties of the serial pass-through can be queried with this command on the input and the output side.

Format		Example
Command	{:SERIAL#<in^2/out^2>@<S>I/O=?}	→ {:serial#9@so=?}
Response	(SERIAL#<in^2/out^2>@<S>I/O=<a>;<b>;<c>)	← (SERIAL#9@SO=1;9600;8N1)

Explanation: Serial pass-through sending and receiving is enabled on the 9th output port with 9600 baud.

Legend: Please read section 8.14.3 (Set up serial pass-through parameters) on page 173.

8.15. Router initiated commands

8.15.1. EDID status changed

Description: This is sent after any command which changed the EDID table (EDID copy, EDID switch), or after a new EDID source e.g. a new display device is connected to the router.

Format	Example
Command various	a new monitor is connected to an output
Response (E_S_C) CrLf	← (E_S_C) CrLf

Explanation: When a new monitor is connected to an output port, its EDID is read. The message from the router shows that an EDID has changed.

Info The router stores the last attached display device's EDID connected to the output. After disconnecting this device its EDID is still present at the router's memory, therefore no status change message is issued by the router if a display device having the same EDID is connected to that output. (The same display device is connected again, or another display device (same brand) from the same manufacturer).

Info: To keep your application in sync with the router it is recommended to issue a watch validity ({wvd}, {wvu}, {wve}) commands after receiving an EDID status changed response, and read all location indicating '2' or '3' in the table, as the change of these EDIDs triggered the (E_S_C) message.

8.15.2. Port status changed

Description: This message is sent when any value changes in the response for the {PS} command. The message means that an input or output port's state has changed e.g. a source or display device is connected or disconnected. See XXX for more information.

Format	Example
Command none	an input port looses signal
Response (PSC) CrLf	← (PSC)CrLf

Explanation: An input port (which had signal present before) detects no signal. The router sends a message to indicate port status change.

Info: The (PSC) message can be omitted by third party controller, or it can be used to trigger a {PS} command. In the latter case, the controller can be up to date with the port status without continuous queries.

8.15.3. Error responses

Invalid input number

Description: Given input number exceeds the maximum number of inputs or equals zero.

Response	(ERR01)CrLf

Invalid output number

Description: Given output number exceeds the installed number of outputs or equals zero.

Response	(ERR02)CrLf

Invalid value

Description: Given value exceeds the maximum allowed value can be sent.

Response	(ERR03)CrLf

Invalid preset number

Description: Given preset number exceeds the maximum allowed preset number.

Response	(ERR04)CrLf

Info: The maximum preset number is limited to 32 for all routers.

9. Error handling

The MX-CPU2 can detect and log many system events. Every log entry gets a time stamp based on the CPU real time clock. These events are categorized by levels.

Level	Description
Notice	Not an error. Initialization information.
Warning	Possible problem without influencing normal operation.
Matter	Problem that may lead to further errors.
Error	Serious error. Must report to support.
Fatal	Fatal error. Normal operation is not possible.

The matrix router saves error logs on the built-in micro SD memory card. These log files can be downloaded and viewed with the controller software.

The error log entries have an error level, time, error code, error parameter, processor task identifier, occurrences and extra information.

The device creates a new error log file every time it is started except if there is already a log file created for that day. The software allows to select only months and days which have a log.

The matrix can indicate if an error occurred in several ways:

- Show alert on the front panel LCD
- Send protocol messages when errors occur. The levels for which this immediate message is sent out can be changed by protocol command.
- Indicate with ALERT LED and SMPTE alarm output on the MX-CPU2 board. If the Alarm LED was triggered it remains lit until the frame is rebooted.

The default levels which trigger an alarm for the specific method are shown below:

Level	Name	LCD alert	LED, SMPTE	RS-232, LAN
0	NOTICE			
1	WARNING			
2	MATTER		yes	yes
3	ERROR	yes	yes	yes
4	FATAL	yes	yes	yes

Info: This log can contain NOTICEs and WARNINGs under normal operation. These entries do not mean that there is any problem with the matrix!

11. Troubleshooting

11.1. General problems

Check the router

Check whether the router is properly powered and whether CPU LIVE LED is blinking. Try performing a reset through the controller software, or unplug and reconnect the router's power cable.

11.2. Serial connection problems

Check the protocol

Check whether the proper protocol is selected (see section 8.1 about changing protocols on page 136). Select Protocol #1 in order to use the matrix with the controller software.

Check the cable

Check whether your serial cable is properly connected. A straight thru male-female cable is needed for connecting the matrix to a computer's COM port.

Check software settings

The router communicates by default with **57600** Baud, **8** data bit, **N**o parity, and **1** stop bit. The baud rate can be changed and checked on the front panel LCD menu.

Check if the correct COM port is selected in the computer for the connection.

11.3. TCP/IP connection problems

Check the LAN cable type

If you connect the router directly to your computer, you must use a cross-link cable. If the matrix is connected to an Ethernet hub, switch or router, you have to use a straight patch LAN cable.

Check the network connection

The computer and the router have to be in the same network. If your computer has multiple network connections (for example WiFi and LAN connections are used simultaneously), check which network the router is connected to. The appropriate Ethernet interface has to be selected in the Find dialog box of the Matrix Controller software (see section 6.2 Establishing the connection on page 85).

Check the IP settings

If you connect the router directly to your computer, you have to set the router's IP address manually, since in this case there is no DHCP server that could assign an address to the matrix.

If the IP address is set manually, check if there is an IP address conflict. If there is a DHCP server on the network, try to set the matrix to DHCP mode. See section 5.8.3 about how to reset the IP address with the front panel buttons on page 82.

The matrix accepts LAN connection on the 10001 TCP port. This can be changed or checked on the front panel LCD menu.

Check whether your computer's firewall blocks the selected port.

Check the protocol

Check whether the proper protocol is selected (see section 8.1 about changing protocols on page 136). Select Protocol #1 in order to use the matrix with the controller software.

12. Appendix

12.1. Maximum cable length

The maximum cable lengths at the inputs of the MX-…-TP-IB are shown below:

Resolution	Cat5e UTP	Cat5e FTP	CAT6 UTP	CAT6 FTP	CAT6 SFTP	CAT7 SFTP
640x480@60	60 m	60 m	65 m	70 m	70 m	80 m
800x600@60	60 m	60 m	65 m	65 m	65 m	75 m
1024x768@60	55 m	55 m	60 m	60 m	60 m	75 m
1280x720p60	55 m	55 m	60 m	60 m	60 m	70 m
1280x1024@60	50 m	50 m	55 m	60 m	60 m	65 m
1400x1050@60	45 m	45 m	45 m	55 m	55 m	60 m
1600x1200@60	30 m	35 m	35 m	45 m	45 m	50 m
1920x1080p60	30 m	35 m	35 m	45 m	45 m	50 m
1920x1200p60	30 m	35 m	35 m	45 m	45 m	50 m

12.2. Control over fiber compatible devices

Extenders:	Modular matrix frames	Modular matrix boards
HDMI-OPT-TX100R	MX-FR9	MX-HDMI-OPT-IB
HDMI-OPT-RX100R	MX-FR17	MX-HDMI-OPT-OB
HDMI-OPT-TX200R	MX-FR33L	
HDMI-OPT-RX200R	MX-FR33R	
HDMI-3D-OPT-RX150RA	MX-FR65R	
	MX-FR80R	

12.3. RICOD capable devices

The RICOD functionality is currently implemented in the following devices:

- MX-FR frames with CPU2, firmware version 3.3.5r and above
- UMX4x4 Pro, firmware version 1.2.4r and above
- UMX-TP-TX100R, firmware version 1.1.6r and above
- WP-UMX-TP-TX100, firmware version 1.1.0r and above
- FP-UMX-TP-TX100, firmware version 1.1.0r and above

** The MX-FR series modular matrix frames with MX-CPU2 processor boards (with firmware version 3.3.5r and above) support reduced RICOD capabilities: only MASTER mode is available and only by protocol commands. RICOD SLAVE mode and the front panel operation are not implemented.*