

MIDI / DMX512 Control Box

User Manual for products delivered from December 2010 Status 17 May 2011

(The chaser function was improved. New "flash" feature. If you need manuals for elder product versions, please order by email: cinetix@t-online.de. If available, add date of delivery or the revision number)

Fields of application:

1. Control of a digital lighting bus connected at DMX OUT by means of MIDI channel messages as well as with System Exclusive commands

For any arbitrary DMX channel the MIDI / DMX Control Box is able to perform a complete fade process. Simply the final DMX level and the fade time are entered by command. The fade ramp is generated inside the MIDI / DMX Control Box. All DMX channels can be faded simultaneously. Based on this feature presets are loaded performing a smooth fade from the previous lighting scene to the new one. The fade time is adjustable between 0 and 31.9 seconds.

Up to 384 presets may be stored permanently and be reloaded by a PROGRAM CHANGE, NOTE ON or SysEx command. This way complete lighting scenes are set easily.

The DMX512 control cycle is transmitted in its whole length of 512 bytes/DMX channels. If needed, the DMX cycle can be shorted – which results in a higher rate of repetition.

2. Merge-capability: DMX levels set via MIDI interface can be merged with data received at DMX IN.

The data sequence composed of both is transmitted at DMX OUT. "Merge" means programmed switching the source of the transmitted signal for any DMX channel, not linear superposition of both.

The applied merge method can be chosen individually for any DMX-Kanal: transmit DMX level set by MIDI, retransmit the level received at DMX IN, transmit the less or the greater of both or transmit the level defined by the last changed one of both data sources ("last takes precedence").

Levels transmitted at DMX OUT may be controlled by the internal clock generator of the box or be synchronized externally with the signal at DMX IN.

3. Receive and evaluate data from a DMX512 bus fed into DMX IN.

Received data can be read out via the MIDI interface. Furthermore, the box can be configured in a way to report automatically MIDI messages when the received DMX level has changed more than a given threshold on selected or on all DMX channels.

4. Transmission of MIDI channel messages and complex strings released by special levels at DMX IN.

A broad spectrum of MIDI NOTE ON, CONTROL CHANGE, PROGRAM CHANGE, CHANNEL PRESSURE and PITCH CHANGE messages can be triggered via DMX IN.

Furthermore, up to 104 free programmable strings can be transmitted via MIDI OUT when DMX IN takes specific levels. Each string may contain up to 255 arbitrary bytes.

It is **NOT allowed** to use this instrument together with any safety critical application, where malfunction could result in personal injury or noticeable material damage !

Hardware:

Elements at the front panel:



IN is a standard MIDI input
OUT is a standard MIDI output
A MIDI-Through connector is not provided.

The power supply is designed for simple d.c. power supplies 9 - 12 Volt, min. 200 mA current output. A simple in-plug supply with "europlug" is included at delivery.

Generally there are only small demands concerning the power supply. Any regulated or unregulated DC supply can be used with output voltages between optimum 9 volt and max. 16 volt (which usually is the max. output of an unregulated 12V AC/DC adaptor). The DC input is designed for a concentric low voltage connector: external 5,0 - 5,5mm, internal 2,1mm. The **positive polarity** has to be connected with the **inner contact** ! Internally the DMX Control Box is **protected against wrong polarity**: if wrong, the box is not powered.

The code switch "Ch / Mode" is used to select different modes of operation of the MIDI/DMX512 Control Box:

switch pos	Function	MIDI I/O Baud	MIDI Channel
1	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	1 to 4
2	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	2 to 5
3	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	3 to 6
4	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	4 to 7
5	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	5 to 8
6	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	6 to 9
7	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	7 to 10
8	DMX receiver for VJ software analyzes&extracts slots 1 to 161 DMX transmitter/merger: channel messages and SysEx	MIDI:31250	MIDI OUT configurable MIDI IN: 8 - 11
9	DMX receiver for VJ software analyzes&extracts DMX channels 256 to 418 DMX transmitter/merger: channel messages and SysEx	MIDI:31250	MIDI OUT configurable MIDI IN: 9 - 12
A	DMX receiver for general transformation DMX to MIDI analyzes&extracts DMX channels 1 to 258 DMX transmitter/merger: channel messages and SysEx	MIDI:31250	MIDI OUT configurable MIDI IN: 10 - 13
B	DMX receiver for general transformation DMX to MIDI analyzes&extracts DMX channels 1 to 258 DMX transmitter/merger: channel messages and SysEx	MIDI:31250	MIDI OUT configurable MIDI IN: 11 - 14
C	DMX receiver for general transformation DMX to MIDI analyzes&extracts DMX channels 1 to 258 DMX transmitter/merger: channel messages and SysEx	MIDI:31250	MIDI OUT configurable MIDI IN:12 - 15
D	Transparent (bytes 1:1) bidirectional MIDI-transmission using RS-422 format via DMX IN/OUT	MIDI:31250	---
E	DMX receiver and transmitter plain ASCII text format without SysEx frame	auto detect default 19200	---
F	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	15,16
0	DMX Tx / Rx / Merger Channel Messages and SysEx	MIDI:31250	16

A **dual color LED** signalizes presence of power, the active mode of operation and the data flow via the MIDI and DMX interface

--- In the mode of **DMX merge and transmission using internal clock of DMX timing**, the **base color of the LED is green**. When MIDI commands are received, the led is flashing dark. While changes at DMX IN are messaged via MIDI OUT; the LED is flashing yellow-orange.

--- In the mode of **DMX merge and transmission synchronized by the signal at DMX IN**, the **base color of the LED is yellow-orange**. When MIDI commands are received, the led is flashing dark. While changes at DMX IN are messaged via MIDI OUT; the LED is flashing red.

--- In **DMX receive mode** the the base color of the LED is red. When MIDI commands are received, the led is flashing dark. While changes at DMX IN are messaged automatically via MIDI OUT; the LED is flashing yellow-orange.

--- During a reset, the LED is darkened for about one second.

Pinout of the DMX512 interface:

DMX IN:

5-pin male XLR connector (optionally equipped with 3 pin male XLR connector)

DMX IN	XLR Connector
Shield, Signal Ground	Pin 1
DMX- Receiver	Pin 2
DMX+ Receiver	Pin 3

In state of delivery the DMX input is terminated with a 120 Ohm resistor. If for special applications the termination shall be disabled: remove the 4 upper screws of the cabinet and lift off the top of it. Pull the jumper located near the DMX IN socket.

DMX OUT:

5-pin female XLR connector (optionally equipped with 3 pin female XLR connector)

DMX OUT	XLR Connector
Shield, Signal Ground	Pin 1
DMX- Transmitter	Pin 2
DMX+ Transmitter	Pin 3

The DMX input (receiver) as well as the DMX output (transmitter) each is galvanically isolated from the control circuitry by means of an optocoupler, both terminals are isolated against each other, too.

Important safety information:

The purpose of galvanic isolation of the DMX input and output is to eliminate DXM data errors due to moderate fault voltages which result from long loops of signal ground lines and protective earth wires within the complete lighting installation.

In conformance with the DMX standard, isolation is guaranteed only up to 42 Volt. It cannot provide an additional level of electrical safety against broken isolation and faulty safety measures of the connected lighting equipment!

The DMX standard demands that the complete bus, driven by the DMX transmitter should be connected with ground or "earth" at one single point.

In most cases this will be the protective earth of your mains line. This means: a galvanically isolated DMX transmitter particularly makes sense, if one or more DMX transmitters without galvanically isolated DMX input are connected to the bus.

In most cases, even with professional equipment using isolated DMX receivers, a completely galvanically isolated DMX bus seems to provide most reliable operation. In particular cases ("latch up" of the bus line) however, it may be necessary to eliminate the galvanic isolation of the DMX transmitter.

For this purpose a jumper is provided inside the box behind the aluminum heat sink. To change it, the box has to be opened: remove the 4 upper screws of the cabinet and lift off the top of it. In state of delivery it is connected towards the heat sink (=isolation present). To bridge the isolation, put it towards the rear panel.

Attention: The metal body of each DMX connector is connected with the metal cabinet of the MIDI / DMX Control Box and thus with the signal ground of the circuit board, too.

Use of DMX cables, whose shielding is connected with the metal body of of the DMX plug, cancels the galvanic isolation! The shielding of the DMX cable has to be connected exclusively with pin 1 of the DMX plug.

Installation of the MIDI / DMX512 Control Box

Safety Information: Because data transfer via DMX512 is regarded to be "unreliable" due to missing error checking, **the use of DMX control is explicitly FORBIDDEN together with all safety critical applications**, where malfunction could result in personal injury oder noticeable material damage !

First, connect the DMX equipment with each other and with the DMX Control Box. Power of all equipment should be OFF.

In most cases, the DMX Control Box will be used for the control of lighting equipment (i.e. as a DMX transmitter). Then put a 5 pin male plug into the 5 pin female socket "DMX OUT" at the rear panel.

Set the DMX address switches of all DMX dimmers etc. in a way which makes best sense for your specific project. If you are not sure and don't have experience with DMX, we recommend **for a first startup to use only one dimmer with lamp addressed to DMX channel no.1.**

If the box is intended to be used to **read data out from an external DMX bus**, then put a 5 pin female plug into the 5 pin male connector "DMX IN" at the rear panel. If DMX data are to be **received and transmitted simultaneously**, make both connections

Next, MIDI equipment is installed: As usual, MIDI OUT of your MIDI equipment is fed to MIDI IN of the MIDI / DMX Control Box and MIDI OUT of the box is connected with MIDI IN of your MIDI equipment.

Now the **code switch is set** to the desired mode of operation and MIDI channel:

- At switch positions **0 to C and F** it is possible to transmit and receive DMX signals and to merge (multiplex) the data from DMX IN together with MIDI generated states to be transmitted from DMX OUT. --
- Positions **8 and 9** are optimized for a special DMX to MIDI translation to control VJ software (VJ="video jockey", interactive control of a video projector show). Positions **A,B,C** extract a broad spectrum of MIDI messages for general use from the signal received at DMX IN. See part 3 of this manual.
- Position **D** is intended to enhance the radius of MIDI transmission by transfer of any byte from MIDI IN to DMX OUT with MIDI baud rate and RS-422 data format - and vice versa bidirectionally.
- At switch position **E** the box is **exclusively controlled by plain ASCII text syntax**, which is intended to simplify manual programming of presets, lighting scenes and automatically released character strings. At delivery, the **baud rate is set to 19200. Other standard baud rates are selected as follows:** After power is switched on or the code switch is turned, for about 1 second the device is in state of automatic baudrate recognition. Then repeat immediately any ASCII code via MIDI IN. Best reliability is achieved with upper case 'U' or lower case 'm'. The detected baud rate is automatically stored permanently and reactivated at next system start. So the detection procedure is only necessary when the baud rate shall be changed or is unknown. **To use it with a standard RS-232 port**, a simple adaptor can be built DIY or bought optionally. See "www.cinetix.de/interface/tiptrix/midi2rs.htm"

For a first operation, we recommend to put the code switch into position "1". This way, MIDI channel no. 1 is set as base channel.

To get access to all 512 DMX channels, now the MIDI / DMX Control Box is sensitive for signals on MIDI channels 2,3 and 4, too! If the code switch is set to a higher position, also the 3 next higher MIDI channels are occupied.

So on a sequencer **always a block of 4 MIDI channels has to be reserved for control of the MIDI / DMX Control Box. This does not apply for code switch positions F and 0:** then the DMX Control Box is exclusively sensitive on MIDI channels 15, 16 (Pos.F) or 16 (Pos.0). This may be advantageous when almost all MIDI channels are already used for other appliances.

After these preparations are finished, the power supply is connected and switched on.

Normally the LED at the front panel will be lit green now, i.e. **in default state the box operates as a DMX transmitter with internal DMX clock generation. All DMX OUT channels (DMX slang: "slots") should be set to level "0"**. If your MIDI equipment is able to monitor incoming MIDI messages, a PROGRAM CHANGE message will be prompted. Its data byte contains the firmware revision number.

Further handling for the first operation see "Quick start and basic commands" below.

MIDI Channel Message protocol of the MIDI/DMX512 Control Box

This chapter is only relevant to code switch settings 0 to C and F which activate the DMX transmitter, DMX receiver and DMX merger

--- This part of the manual is focused on the generation of DMX signals at MDX OUT by means of MIDI-channel messages. Nearly all functions can be controlled equivalently with MIDI System Exclusive messages. This aspect is lined out in part 2 of this manual.

--- At these code switch positions the DMX Control Box is operating simultaneously as DMX transmitter/merger and DMX receiver. In its function as DMX receiver a range of different MIDI messages can be extracted from the signal levels at DMX IN and be sent via MIDI OUT. For example, these can drive VJ ("Video Jockey") software or are useful for general control of MIDI equipment from an external lighting console. For detailed description of these transformations DMX to MIDI see part 3 of this manual.

The content of MIDI messages used here cannot follow the conventions which have been established in the world of musicians (e.g. note values and controller numbers), but has been optimized for tasks of lighting control. We have taken much care to design the codes in a way they can be reproduced with common MIDI equipment like sequencers and programmable keyboards. Nevertheless, conflicts are possible, when the MIDI / DMX Control Box is used together with musician applications on a shared MIDI data line. These conflicts can be solved by appropriate selection of MIDI channels or by using a MIDI interface with multiple sets of independent MIDI IN and MIDI OUT.

Next is described how the MIDI / DMX Control Box can be operated with simple commands for frequently used applications.

More below the complete set of MIDI commands is described in detail. The MIDI / DMX Control Box is a quite complex device with a lot of features. So, some basic knowledge of MIDI messages is advantageous to make best use of it. The internal structure of the MIDI / DMX Control Box is explained in appendix A.

Quick start and basic commands:

One of the most frequently used applications is **lighting control based on a MIDI sequencer**, which in most cases is available as software running on a PC. Depending on the specific model and personal liking, **NOTE ON or CONTROL CHANGE messages are used to program the lighting events**. Apart from the different type of command (MIDI status byte) the commands are constructed identically for both methods. **Both types of command can be mixed arbitrarily.**

The **status byte** (where the type of command and MIDI channel is coded) is followed - depending on the type of the MIDI message - by up to 2 **data bytes**. Contained in NOTE ON, NOTE OFF or POLY KEY PRESSURE messages, the **1st data byte** is the note value ("pitch", see table at the last page of the manual), the **2nd data byte** is the strength ("velocity") of the key stroke. With CONTROL CHANGE messages the 1st data byte is the controller number, the 2nd data byte is the corresponding controller value to be entered there

With NOTE ON messages it is possible to "play in" the timing structure of control messages using a keyboard in an interactive way. But when a note is finished (a key is released) most MIDI equipment sends another NOTE ON message with velocity=0 or a NOTE OFF message, which turns the lamp off again. This can be prevented with a special PROGRAM CHANGE command. When CONTROL CHANGE messages are used, many sequencer programs offer graphical programming of a fade ramp. In this case the very limited data capacity of a MIDI line has to be taken into account. You should decide to use one method at your taste.

To address any of the DMX channels (in DMX slang "slots") 1 to 127, control data have to be produced on the track of that MIDI channel which is selected at the code switch. (We recommend to take switch position "1" for first experiments and explore the other channels later).

The 1st data byte of the MIDI command defines the DMX channel to be addressed and manipulated.

The 2nd data byte of the MIDI command describes the light intensity to be set.

The 2nd MIDI data byte is multiplied by 2 inside the MIDI / DMX Control Box.

Following **exceptions** do apply:

When setting is done with a **CONTROL CHANGE** command, the 8-bit DMX level is additionally incremented, i.e. the next higher odd DMX level is set.

This behaviour can be **deactivated** with a PROGRAM CHANGE command, data byte=60)

only then is valid for NOTE ON as well as for CONTROL CHANGE commands:

if the 2nd MIDI data byte is = 127, then the DMX level is set to its maximum value 255.

Or described in the opposite way of thinking: **to set a certain DMX level (0 to 255)** with a simple MIDI command, **HALF OF the intended DMX level** has to be entered in the **2nd MIDI data byte**. **Odd DMX levels** are set with CONTROL CHANGE, **even DMX levels** are set with NOTE ON.

A simple example:

When the code switch is in position."1", DMX channel no. 1 is **set to DMX level decimal 64** (25% intensity) with this command: NOTE ON (status byte = hex90).1st byte = 1, 2nd data byte =dez32 (hex20).

DMX level 65, however, is achieved with CONTROL CHANGE: (status byte = hexB0). 1st data byte = 1, 2nd data byte =dez32 (hex20).

Some further annotations:

Some MIDI devices send a NOTE ON message with velocity 0 or a NOTE OFF message at the end of a note, which immediately would fade down this DMX channel when the box is in default configuration.

This behaviour is useful when a keyboard shall be used as a "light organ" or this kind of action is played from a sequencer.

In other applications, where NOTE ON messages are used to control steady lighting scenes, this behaviour is unpleasant. To solve the conflict, the sensitivity to NOTE ON messages with velocity=0 or to any NOTE OFF message can be switched off. Refer to the detailed description of the corresponding PROGRAM CHANGE command.

To write data into DMX channels 128 to 512 with these simple commands, the MIDI / DMX Control Box is sensitive at the next three MIDI channels too: (only valid for code switch positions 1 to C):

coded MIDI channel	1 st data byte	addresses DMX channel	calculation of 1 st data byte
as position of code switch	1 to 127	1 to 127	= DMX channel
next options are only available for code switch pos. 1 to C :			
as code switch + 1	0 to 127	128 to 255	= DMX channel minus 128
as code switch + 2	0 to 127	256 to 382	= DMX channel minus 256
as code switch + 3	0 to 127	384 to 511	= DMX channel minus 384
as code switch	0 special case !	512	= 0

So, if the code switch is in position 1, the MIDI / DMX Control Box is sensitive for the MIDI channels 2,3 and 4, too. This means: on a sequencer program you always have to reserve a

block of 4 MIDI channels to fully control the MIDI / DMX Control Box and the corresponding edit tracks have to be initialized.

Attention: data which are meant for other MIDI equipment which works on these channels may be misinterpreted.

Example: At code switch position "1" to set DMX slot No. 261 to full level (255) following MIDI message has to be sent: CONTROL CHANGE command at MIDI channel 3 (status byte = hexB2). First data byte = 5 (d.h. 261-256), second data byte = 127.

Differing from this rule, at code switch position F the MIDI / DMX Control Box reacts only on MIDI channels 15 and 16 and at position 0 only on MIDI channel 16. This is welcome in applications which use almost the complete set of MIDI channels. With simple commands (NOTE ON or CONTROL CHANGE), however, only DMX channels 1 to 255 respectively 1 to 127 are addressable then.

This can be worked around with somewhat more complex commands, which **allow to adjust any DMX channel 1 to 512 with full 8 bit accuracy.** See details at the description of the POLY KEY PRESSURE and the PITCH WHEEL CHANGE command below.

For precise entry of control data the list- (or event-) editor of the sequencer software is used. Before working with this editor don't forget to reserve and activate appropriate data tracks of sufficient length for all MIDI channels to be used.

In principle, NOTE ON events can be recorded on a sequencer with a MIDI keyboard. But because the intensity of a key touch ("Velocity") - which is sent as 2nd data byte - cannot be controlled precisely, always post-handling with the editor will be necessary. But at least the timing points can be entered well with a keyboard, following the rhythm of a music for example. Unfortunately most sequencer programs don't allow to manipulate the status byte in the list-(event-) editor, so NOTE ON messages cannot be turned into CONTROL CHANGE later.

If an event shall be entered by use of CONTROL CHANGE messages, first a new entry of this type has to be brought into the list. Most times this is done by clicking into appropriate lists or menus. When this is achieved, both data bytes can be edited in the list.

Simultaneously with editing the list, in most sequencer programs a marker is put into the timeline (or arrangement window), which can be dragged with the mouse to the correct time position. Manual editing of timing points is possible in the editor, too, but this is a more abstract way of work.

Instead of programming a control sequence manually, **the MIDI / DMX Control Box alternatively can be used as a recorder to get a scene which is generated on a DMX desk into the MIDI sequencer.** The MIDI / DMX Box extracts automatic messages in the same format as is used later when sending commands from the sequencer to the box.

To activate this function, two details have to be configured: first: which is the type of MIDI command to be recorded, next: the desired resolution or threshold for changes in the incoming DMX signal to release a message to the sequencer. At the sequencer appropriate recording tracks have to be prepared in advance. MIDI recording can be activated for all DMX channels or for a few selected ones. See description of the PITCH CHANGE command for details.

Good results will only be achieved with small installations under restricted and concentrated use of the DMX master console. Especially fade processes have to be used carefully not to overload the MIDI system with lots of data. In most cases some final editing will be necessary at the sequencer.

Please note that **due to the parameters of the MIDI system the speed of data throughput is quite limited.** Per second about 1000 MIDI messages can be transferred at maximum.

If a lot of dimmers are faded during automatic generation of NOTE ON or CONTROL CHANGE messages the MIDI line can easily become clogged. Delayed execution of commands or even loss of some data cannot be excluded then.

To avoid this bottleneck, **the MIDI / DMX Control Box is able to control a complete fade process automatically.** For this purpose, a nonzero value is written into the FADETIME register before sending a command to change the DMX level. Details see description of the POLYPHONIC KEY PRESSURE command.

Survey of all MIDI Channel Commands (MIDI Implementation Chart)

(appropriate only when code switch is in position 1 to C, F or 0 = MIDI channels 1-12,15,16)

With PROGRAM CHANGE 63 the functions of CONTROL CHANGE and POLY KEY PRESSURE commands (as they are described in this manual) may be exchanged.

This option provides better flexibility to work with different kind of MIDI control equipment.

Abbreviations:

DB means "data byte", DB1 means 1st data byte", DB2 means "2nd data byte", "CodeSw" means the basic MIDI channel which is selected by the code switch (=switch position).

When using PROGRAM CHANGE commands you should take into account, that most MIDI devices and software send the data byte value "0" when "program no.1" is selected!

In the table "DB" denotes the physically transferred data byte.

Note: some commands are changed with respect to previous versions !

MIDI message and coded MIDI channel	special data values	function /effect	p.
NOTE OFF	all, see detail description	DMX level --> 0	23
NOTE ON MIDI channel = CodeSw + next 3 channels	DB1 = DMX channel DB2 = DMX level ./ 2	set DMX channel and level (only 1 MIDI message / 7bit resolution) DMX level = DB2 * 2	12
CodeSw + 2 next channels	s. detailed description	load preset 0-383	22
POLY KEY PRESSURE MIDI channel = CodeSw	DB1 = 1 DB2 = 1/10 sec. 0-127	set FADETIME to 0-12.7 seconds fade time = DB2 ./10 seconds	14
notice PROGRAM CHANGE 63 !	DB1 = 2 DB2 = 1/10 sec. 0-127	set FADETIME to 10 - 22.7 seconds fade time = 10 + DB2 ./10 seconds	15
	DB1 = 3 DB2 = 1/10 sec. 0-119	set FADETIME to 20 - 31.9 seconds fade time = 20 + DB2 ./10 seconds n	15
	DB1 = 4 DB2 = 1/4 sec. 0-127	set FADETIME in the range 0 to 31.8 seconds with a single controller. Fade time = DB2 ./4 seconds	15
	DB1 = 7 DB2 = masterfader %	set masterfader in the range 0 – 127 % Note: DB1 was changed with respect to previous versions	15
	DB1 = 8 DB2 = masterfader-100	set masterfader in the range 100-200 % Note: DB1 was changed with respect to previous versions	15
	DB1 = 15 (hex F) DB2 = duration in 1/10s	release flash pulse: all DMX channels are set to 100% during a short time. See detailed description	15
	DB1 = 16 (hex 10) DB2 = step in 1/10 s	set step duration of the chaser. Is controlled by an internal timer. 0 = chaser OFF.	16
	DB1 = 17 (hex 11) DB2 = step in 1/8 notes	set step duration of the chaser. Is controlled by MIDI clock (1/8 note = 12 MIDI clock pulses)	16
	DB1 = 18 (hex 12) DB2 = count of scenes per cycle	set cycle length of the chaser and start it: display any preset (scene) 'step duration' long, then next preset is loaded. Repeat cycle after 'count' steps	16
	DB1 = 19 (hex 13) DB2 = start-128 (0-127)	set start scene (preset no.) of the chaser cycle. 128 is added internally. See detailed description	17
	DB1 = 20 (hex 14) DB2 = start-256 (0-127)	set start scene (preset no.) of the chaser cycle. 256 is added internally. See detailed description	17
	DB1 = 36,37 (hex24,25) DB2 = 0 to 127	poll (entry of DB2) levels of DMX OUT starting from DMX channel=SLOT (DB2=0: poll 128 channels)	18
	DB1 = 40-42(hex28-2A) DB2 = 0 to 127	poll (entry of DB2) levels of DMX IN starting from DMX channel=SLOT (DB2=0: poll 128 channels)	18
	DB1 = Mode 48-52 (hex 30 - 34) DB2 = threshold	(de)activate MIDI messages generated from input at DMX-IN for DMX channel addressed by "SLOT"	part 3 45/ 46
	DB1 = 58 - 62 (hex 3A-3E) DB2 = threshold	(de)activate MIDI messages generated from input at DMX-IN for all DMX channels equally	part 3 45

MIDI message and coded MIDI channel	special data values	function /effect	p.
POLY KEY PRESSURE MIDI channel = CodeSw	DB1 = 72 (hex 48)	fill a block of DB2 DMX channels starting from ch. SLOT+1 with the final level of DMX ch. "SLOT"	14
notice PROGRAM CHANGE 63 !	DB1 = 76 (hex 4C) DB2 = 0 to 127	set hue (color tone) for RGB lamp (3 consecutive DMX channels)	19
	DB1 = 77 (hex 4D) DB2 = 0 to 127	set color saturation for RGB lamp (3 consecutive DMX channels)	19
	DB1 = 78 (hex 4E) DB2 = 0 to 127	set brightness for RGB lamp (3 consecutive DMX channels)	19
	DB1 = 80 – 83 (hex 50 – 53)	address DMX channel using only one single MIDI channel. For loading data see "pitch change"	12
	DB1 = 84 (hex 54)	set DMX level (@SLOT) to DB2 *2 adjust DMX level with 8 bit resolution to even value	13
	DB1 = 85 (hex 55)	set DMX level (@SLOT)to DB2 *2 + 1 adjust DMX level with 8 bit resolution to odd value	13
	DB1 = 86 (hex 56)	set DMX level (@SLOT)to DB2 *2. First increase DMX channel (SLOT reg.) there adjust DMX level with 8 bit resolution to even value	13
	DB1 = 87 (hex 57)	set DMX level (@SLOT)to DB2 *2 + 1 First increase DMX channel (SLOT reg.) there adjust DMX level with 8 bit resolution to odd value	13
	DB1 = 88 (hex 58) DB2 = MIDI chan. 1-16	configure which MIDI channel will be inserted into MIDI messages produced by transformation DMX-Input -> MIDI messages (CodeSw pos 8 - C)	part 3 46
	DB1 = 96 (hex 60) DB2 = preset no.	load preset no. 0 – 127. The system configuration is exclusively loaded with preset no. 0 - 3	21
	DB1 = 97 (hex 61) DB2 = preset no. - 128	load preset no. 128 – 355	21
	DB1 = 98 (hex 62) DB2 = preset no. - 256	load preset no. 256 – 383	21
	DB1 = 99 (hex 63) DB2 = preset no.	load preset no. 0–127 partially. Preset DMX ch. 129-256 are loaded into DMX OUT channels 1-128	21
	DB1 = 100 (hex 64) DB2 = preset no.	load preset no. 0–127 partially. Preset DMX ch. 257-384 are loaded into DMX OUT channels 1-128	21
	DB1 = 101 (hex 65) DB2 = preset no.	load preset no. 0–127 partially. Preset DMX ch. 385-512 are loaded into DMX OUT channels 1-128	21
	DB1 = 102 (hex 66) DB2 = preset no. - 128	load preset no. 128–255 partially. Preset DMX ch. 129-256 are loaded into DMX OUT channels 1-128	21
	DB1 = 103 (hex 67) DB2 = preset no. - 128	load preset no. 128–255 partially. Preset DMX ch. 257-384 are loaded into DMX OUT channels 1-128	21
	DB1 = 104 (hex 68) DB2 = preset no. - 128	load preset no. 128–255 partially. Preset DMX ch. 385-512 are loaded into DMX OUT channels 1-128	21
	DB1 = 105 (hex 69) DB2 = preset no. - 256	load preset no. 256–383 partially. Preset DMX ch. 129-256 are loaded into DMX OUT channels 1-128	22
	DB1 = 106 (hex 6A) DB2 = preset no. - 256	load preset no. 256–383 partially. Preset DMX ch. 257-384 are loaded into DMX OUT channels 1-128	22
	DB1 = 107 (hex 6B) DB2 = preset no. - 256	load preset no. 256–383 partially. Preset DMX ch. 385-512 are loaded into DMX OUT channels 1-129	22
	DB1 = 108 (hex 6C) DB2 = preset no.	load preset no. 0–127 partially. Only DMX channels 1-128 are loaded, 129-512 remain unchanged	22
	DB1 = 109 (hex 6D) DB2 = preset no.	load preset no. 0–127 partially. Only DMX channels 129-256 are loaded, others remain unchanged	22
	DB1 = 110 (hex 6E) DB2 = preset no.	load preset no. 0–127 partially. Only DMX channels 257-384 are loaded, others remain unchanged	22
	DB1 = 111 (hex 6F) DB2 = preset no.	load preset no. 0–127 partially. Only DMX channels 385-512 are loaded, others remain unchanged	22

MIDI message and coded MIDI channel	special data values	function /effect	p.
POLY KEY PRESSURE MIDI channel = CodeSw	DB1 = 112 (hex 70) DB2 = preset no.	save preset no. 0 - 127. The system configuration is exclusively saved with preset no. 0 - 3	20
notice PROGRAM CHANGE 63 !	DB1 = 113 (hex 71) DB2 = preset no. - 128	save preset no. 128 - 255	20
	DB1 = 114 (hex 72) DB2 = preset no. - 256	save preset no. 256 - 383	20
	DB1 = 115 (hex 73) DB2 = preset no.	save preset no 0–127 partially. DMX OUT channels 1-128 are copied into preset channels 129-256	21
	DB1 = 116 (hex 74) DB2 = preset no.	save preset no 0–127 partially. DMX OUT channels 1-128 are copied into preset channels 257-384	21
	DB1 = 117 (hex 75) DB2 = preset no.	save preset no 0–127 partially. DMX OUT channels 1-128 are copied into preset channels 385-512	21
	DB1 = 118 (hex 76) DB2 = preset no. - 128	save preset no 128-255 partially. DMX OUT chann. 1-128 are copied into preset channels 129-256	21
	DB1 = 119 (hex 77) DB2 = preset no. - 128	save preset no 128-255 partially. DMX OUT chann. 1-128 are copied into preset channels 257-384	21
	DB1 = 120 (hex 78) DB2 = preset no. - 128	save preset no 128-255 partially. DMX OUT chann. 1-128 are copied into preset channels 385-512	21
	DB1 = 121 (hex 79) DB2 = preset no.- 256	save preset no 256-383 partially. DMX OUT chann. 1-128 are copied into preset channels 129-256	21
	DB1 = 122 (hex 7A) DB2 = preset no. - 256	save preset no 256-383 partially. DMX OUT chann. 1-128 are copied into preset channels 257-384	21
	DB1 = 123 (hex 7B) DB2 = preset no. - 256	save preset no 256-383 partially. DMX OUT chann. 1-128 are copied into preset channels 385-512	21
	DB1 = 126 (hex 7E) DB2 = only15(hex F)	Store actual lighting scene as a pattern which is loaded during the flash effect	21
	DB1 = 127 (hex 7F)	DB2 has the same meaning as the data byte of the corresponding PROGRAM CHANGE command.	
CONTROL CHANGE MIDI channel = CodeSw + next 3 channels	DB1 = DMX channel DB2 = DMX level ./ . 2	set DMX channel and level (only 1MIDI message / 7bit resolution) DMX level = DB2 * 2 +1	12
PROGRAM CHANGE MIDI channel = CodeSw	DB = 0	always send DMX level defined by MIDI command to DMX-OUT	17
	DB = 1	Merge: always send less of DMX IN and MIDI cmd	17
	DB = 2	Merge: always send greater of DMX IN and MIDI	17
	DB = 3	always send level from DMX IN to DMX OUT	17
	DB = 4	Merge: "last takes Precedence"	17
	DB = 8	decrease DMX level (minus 1) at channel "SLOT"	14
	DB = 9	increase DMX level (plus 1) at channel "SLOT"	14
	DB = 16 (hex 10)	forward chaser immediately by 1 step	14
	DB = 32 (hex 20)	copy level from DMX IN (@SLOT) to transmit buffer	14
	DB = 36 (hex 24)	block execution of MIDI commands	20
	DB = 37	stop command blocking and execute last received	20
	DB = 40 (hex 28)	deactivate automatic messages globally	part 3 44
	DB = 41	activate automatic messages globally format as SysEx/ASCII	part 3 47
	DB = 42 (hex 2A)	activate automatic messages globally format as MIDI channel messages (default)	part 3 47
	DB = 50 (hex 32)	deactivate DMX triggered msgs globally (default)	part 3 47
	DB = 51	activate DMX triggered msgs channel 500-512	part 3 47
	DB = 52	activate DMX triggered msgs channel 458-499	part 3 47
	DB = 53	activate DMX triggered msgs channel 458-512	part 3 47

MIDI message and coded MIDI channel	special data values	function /effect	
PROGRAM CHANGE MIDI channel = CodeSw	DB = 56 (hex 38)	No Running State allowed in commands	19
	DB = 57	accept Running State in commands (default *)	19
	DB = 58	No Running State in feedback msgs (default)	19
	DB = 59	send feedback messages with Running State	19
	DB = 60 (hex 3C)	CONTROL CHANGE behaves like NOTE ON Controller value 127 is transformed to DMX 255	19
	DB = 61	CONTROL CHANGE sets DMX level +1 (default)	19
	DB = 62 (hex 3E)	don 't exchange CONTROL CHANGE and POLY KEY PRESSURE (default)	19
	DB = 63	exchange CONTROL CHANGE and POLY KEY	19
	DB = 64 (hex 40)	set LOOP=SLOT	18
	DB = 72 (hex 48)	Stop all fade processes, freeze at momentary level	14
	DB = 80-84 (hex50-54)	global merge configuration,see detailedled description	17
	DB = 96 (hex 60)	NOTE ON 0-127 (keyboard) loads preset no 0-127	22
	DB = 97 (hex 61)	NOTE ON 0-127 loads preset no 128-255	22
	DB = 98 (hex 62)	NOTE ON 0-127 loads preset no 256-383	22
	DB = 99 (hex 63)	NOTE ON 0-127 loads preset no 0-127 partially See POLY KEY PRESSURE DB1=99	22
	DB = 100 (hex 64)	NOTE ON 0-127 loads preset no 0-127 teilweise Siehe POLY KEY PRESSURE DB1=100	22
	DB = 101 (hex 65)	NOTE ON 0-127 loads preset no 0-127 partially See POLY KEY PRESSURE DB1=101	22
	DB = 102 (hex 66)	NOTE ON 0-127 loads preset no 128-255 partially See POLY KEY PRESSURE DB1=102	22
	DB = 103 (hex 67)	NOTE ON 0-127 loads preset no 128-255 partially See POLY KEY PRESSURE DB1=103	22
	DB = 104 (hex 68)	NOTE ON 0-127 loads preset no 128-255 partially See POLY KEY PRESSURE DB1=104	22
	DB = 105 (hex 69)	NOTE ON 0-127 loads preset no 256-383 partially See POLY KEY PRESSURE DB1=105	23
	DB = 106 (hex 6A)	NOTE ON 0-127 loads preset no 256-383 partially See POLY KEY PRESSURE DB1=106	23
	DB = 107 (hex 6B)	NOTE ON 0-127 loads preset no 256-383 partially See POLY KEY PRESSURE DB1=107	23
	DB = 108 (hex 6C)	NOTE ON 0-127 loads preset no 0-127 partially See POLY KEY PRESSURE DB1=108	23
	DB = 109 (hex 6D)	NOTE ON 0-127 loads preset no 0-127 partially See POLY KEY PRESSURE DB1=109	23
	DB = 110 (hex 6E)	NOTE ON 0-127 lädt Preset Nr. 0-127 partially See POLY KEY PRESSURE DB1=110	23
	DB = 111 (hex 6F)	NOTE ON 0-127 loads preset no 0-127 partially See POLY KEY PRESSURE DB1=111	23
PROGRAM CHANGE MIDI channel = CodeSw	DB = 120 (hex78)	NOTE ON sets the level of a DMX channel Velocity=0 and NOTE OFF are accepted (default)	23
	DB = 121 (hex79)	NOTE ON sets the level of a DMX channel Velocity=0 is ignored	23
	DB = 122 (hex7A)	ask actual channel configuration (get SysEx msg)	23
	DB = 124 (hex7C)	DMX OUT and Merge w. internally generated clock	23
	DB = 125 (hex7D)	DMX OUT and Merge synchronized by DMX-IN	23
	DB = 126 (hex7E)	loop DMX IN to DMX OUT (receiver function only)	23
	DB = 127 (hex7F)	Clear All Memory	23
CHANNEL PRESSURE MIDI channel = CodeSw	DB= DMX level /.2	increase adressed DMX channel and set there DMX level = DB*2.	14

MIDI message and coded MIDI channel	special data values	function /effect	p.
CHANNEL PRESSURE MIDI channel = CodeSw+1	DB= DMX level ./ 2	increase addressed DMX channel and set there DMX level = DB*2 +1.	14
MIDI channel = CodeSw+2	DB= DMX level ./ 2	increase addressed DMX channel and set there DMX level = DB*2. Running state temporarily ON	14
MIDI channel = CodeSw+3	DB= DMX level ./ 2	increase addressed DMX channel and set there DMX level = DB*2+1. Running state temporarily ON	14
PITCH CHANGE MIDI channel = CodeSw	DB1 = less than 64 ? DB2 = DMX level ./ 2	set DMX level at position SLOT (8bit resolution) If DB1 >= 64, then DMX level = DB2*2 + 1	13
SYSTEM EXCLUSIVE	channel independent	encloses all ASCII commands for MIDI	part2
MIDI RESET		Reset, loads preset no.0	24

Commands where is annotated "CodeSw + next 3 channels" are able to address all 512 DMX slots (valid only for code switch positions 1 to C) The relationship between the MIDI channel in the command and its effect on addressing a DMX channel is described in the table below

Express setting of DMX channel (= "SLOT") with a single MIDI message:
NOTE ON or CONTROL CHANGE

The 1st data byte of the MIDI command defines the DMX channel to be addressed and manipulated.

The 2nd data byte of the MIDI command describes the light intensity to be set.

The 2nd MIDI data byte is multiplied by 2 inside the MIDI / DMX Control Box.

Following **exceptions** do apply:

When setting is done with a **CONTROL CHANGE** command, the 8-bit DMX level is additionally incremented, i.e. the next higher odd DMX level is set.

This behaviour can be **deactivated** with a PROGRAM CHANGE command, data byte=60)

only then is valid for NOTE ON as well as for CONTROL CHANGE commands:

if the 2nd MIDI data byte is = 127, then the DMX level is set to its maximum value 255.

This simple method works only for DMX channels 1 to 127. To address DMX channels 128 to 512, the MIDI / DMX Control Box expects control commands on a higher MIDI channel corresponding with following table (only valid for code switch positions 1 to C):

coded MIDI channel	1 st data byte	addresses DMX channel	calculation of 1 st data byte
as position of code switch	1 to 127	1 to 127	= DMX channel
next options are only available for code switch pos. 1 to C :			
as code switch + 1	0 to 127	128 to 255	= DMX channel minus 128
as code switch + 2	0 to 127	256 to 382	= DMX channel minus 256
as code switch + 3	0 to 127	384 to 511	= DMX channel minus 384
as code switch	0 special case !	512	= 0

Special reaction on NOTE ON messages with velocity=0 and on NOTE OFF messages see PROGRAM CHANGE commands 120 and 121.

Address the active DMX channel (= "SLOT" register) with:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

This alternative command version is used, **when all MIDI messages shall be sent on the single MIDI channel which is selected by the code switch.**

The coarse range is selected with the first data byte, which has to be chosen according to this table:

1st data byte	addresses DMX chan.= SLOT	2nd data byte	calculation of 1st data byte
80 (hex50)	1 to 127	1 to 127	= DMX channel
81 (hex51)	128 to 255	0 to 127	= DMX channel minus 128
82 (hex52)	256 to 382	0 to 127	= DMX channel minus 256
83 (hex53)	384 to 511	0 to 127	= DMX channel minus 384
80 (hex50)	512	0 (special case!)	= 0

Comment: This command does not directly trigger any action. But the updated content of the SLOT register will be executed together with subsequent commands. In DMX slang a DMX channel is called a "SLOT" (physically it denotes a time slot in the DMX transmit cycle, therefore this strange name)

Set DMX level with 8 bit resolution at DMX channel = SLOT with:

Method 1:

PITCH WHEEL CHANGE

MIDI channel as position of code switch

1st data byte: if equal or greater than 64, "1" is added to the DMX level
if less than 64 (hex 40), nothing is added to the DMX level

2nd data byte: desired DMX level divided by 2

inside the MIDI / DMX Control Box, this data is multiplied by 2 before it is written into the DMX transmit buffer

Comment: This coding scheme looks strange at first glance. But it is compatible with the standard MIDI method to put the 7 "most significant bits" of 14bit data into the second data byte zu. So it can be used together with simple MIDI equipment, which has only 7 bit capability of PITCH WHEEL CHANGE operation.

Method 2:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 84 (hex54) adjusts to an **even** DMX level

2nd data byte: desired DMX level divided by 2
i.e vice versa: DMX level= 2nd data byte * 2

1st data byte = 85 (hex55) adjusts to **next odd** DMX level

2nd data byte: desired DMX level divided by 2
i.e vice versa: DMX level= 2nd data byte * 2 **plus 1**

1st data byte = 86 (hex56) **first increases the addressed DMX channel**
and adjusts this one to an **even** DMX level

2nd data byte: desired DMX level divided by 2
i.e vice versa: DMX level= 2nd data byte * 2

1st data byte = 87 (hex57) **first increases the addressed DMX channel**
and adjusts this one to **next odd** DMX level

2nd data byte: desired DMX level divided by 2
i.e vice versa: DMX level= 2nd data byte * 2 **plus 1**

Increase addressed DMX channel ("SLOT") and set DMX level with:

CHANNEL PRESSURE (equivalent to the ASCII "comma" command)

MIDI channel as code switch: DMX level = data byte*2

MIDI channel as code switch +1: DMX level = data byte*2 +1

MIDI channel as code switch +2: DMX level = data byte*2, activates Running State

MIDI channel as code switch +3: DMX level = data byte*2 +1, activates Running State

Comment: with this method a block of consecutive DMX channels can be set with different levels in a quite compact way. To maintain the start channel, the first DMX channel is set with NOTE ON or CONTROL CHANGE, all following ones with a CHANNEL PRESSURE command.

If the command is sent on the MIDI channel which is selected by the code switch or the next higher one, running state in the command sequence is accepted in correspondence with the general system setup. However, if the 2nd or 3rd higher MIDI channel is coded into the status byte, running state is temporarily accepted until reception of the next status byte which differs from this rule.

Example: Assume code switch in position 2 (base channel status byte Low Nibble =1). DMX channels 1 to 5 shall be set to DMX levels 10,20,30,40,50 and the channels 6,7,8 shall be set to levels 55,65,75. Following sequence of decimal bytes has to be transmitted: 145,1,5,211,10,15,20,25,212,27,32,210,37. With the last status byte 210 the acceptance of running state is reset to the previous system setup.

Fill block of DMX channels from "SLOT+1" with the final level of DMX channel "SLOT" with:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 72 (hex48)

2nd data byte: block length (1 to 127)

Comment: Every DMX channel in the commanded range is faded or switched from its present level to the final state of the DMX channel which is preselected by the SLOT register. The fade duration is given by the actual setting of the FADETIME register.

Simple modifications of the DMX level with:

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 8 decrement (subtract 1 from) the level of DMX channel "SLOT"

data byte = 9 increment (add 1 to) the level of DMX channel "SLOT"

Comment: With these commands the disadvantage of lower accuracy of 7 bit MIDI data can be compensated. Furthermore it is useful to perform extremely slow fade transitions.

data byte = 16 (hex10) forward chaser immediately by 1 step

data byte = 32 (hex 20) the present content of the **receive buffer is completely copied into the transmit buffer** (and appropriate merge configuration provided: transmitted via DMX-OUT immediately)

Also see the corresponding ASCII/SysEx command 'J'

Comment: use this command to store a complete lighting scene from an external console as a preset.

data byte = 72 (hex48) **Stop all fade processes immediately.**

Freeze all DMX levels at their present state.

Set FADETIME with:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = **1**, then

2nd data byte = (0 - 127) fade time in 1/10 second units (0 -12,7 seconds).

1st data byte = **2**, then

2nd data byte = (0 - 127) fade time in 1/10 second units
plus 10 seconds (setting range 10,0 - 22,7 seconds).

1st data byte = **3**, then

2nd data byte = (0 - 127) fade time in 1/10 second units
plus 20 seconds (setting range 20,0 - 31,9 seconds)..

1st data byte = **4**, then

2nd data byte = (0 - 127) fade time in **quarter seconds** (0 - 31,8 sec.)

This variant makes it possible to handle the complete range of fading time with a single MIDI controller.

remaining quarter seconds are internally rounded up to the next higher step of 1/10 seconds, i.e. 0.25 to 0.3 and 0.75 to 0.8.

Comment: The actual value of FADETIME is copied into the respective resource when the fade process is started. Immediately after FADETIME can be modified without retroactivity on running fade processes. Any number of fade processes can be active simultaneously.

In earlier **firmware versions (< 83)** the FADETIME was set differently:
POLY KEY PRESSURE, 1st data byte 0 to 31:fade time in seconds,
2nd data byte 0 to 9: additional 1/10 seconds

Set the MASTERFADER with:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = **7**

then 2nd data byte = 0-127 (hex 7F) masterfader setting in %

1st data byte = **8**

then 2nd data byte = 0-100 (hex 64) masterfader setting 100-200%
(internally 100 is added to the data byte)

Comment: from December 2010 (revision number 90) the 1st data byte was changed to obtain better compatibility with MIDI Volume Control. At earlier firmware versions the 1st data byte was 8 resp.9

Also see comment at the equivalent ASCII/SysEx command 'M'

Release "flash" effect with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE) (new from Dec 2010 / rev.no 90)

MIDI channel as position of code switch

1st data byte = **15** (hex F)

2nd data byte = flash duration in 1/10 s units (0,1 to 12,7 seconds)

all DMX channels are pulsed to 100%

= 0 terminate flash immediately, independently from a previously entered duration

Comment: with this command a special preset (=special lighting scene) is transmitted at all 512 channels of the DMX bus during a time period given by the 2nd data byte. After this timing period, all previous DMX levels are restored.

At delivery the flash lighting scene is prestored in a way that all DMX channels are pushed to 100% level. Used together with installation of complex lamp fixtures, this may result in undesirable complications (start stroboscope effect, for example).

To avoid this, a user prepared lighting scene may be stored permanently in a special memory segment with a POLY KEY PRESSURE command 1st data byte 126 (hex 7E), 2nd data byte =15 (hex F). This lighting scene should be designed in a way to avoid these side effects - or even a very individual flash pattern may be created.

Set chaser cycle length and start it with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE) (new from Dec 2010 / rev.no 90)

MIDI channel as position of code switch

1st data byte = 18 (hex12)

2nd data byte = 2 - 127 (hex 7F) set chaser cycle length 2-127

or = 0: switches the chaser **OFF**

Comment: before the chaser can be started, the **step duration** (POLY KEY PRESSURE, 1st data byte= 16 or 17) as well as the **start scene** (POLY KEY PRESSURE 1st data byte= 19 or 20) has to be adjusted – **Details see description of these commands.** For an easy start, start at scene 128 and step duration 20 (= 2 seconds) is preset as default.

All settings of the chaser are stored in presets 0 to 3 and reactivated when any of these presets is loaded. This applies especially for preset no.0, which is loaded when the DMX Generator is started.

The actual settings of fade time and master fader are taken over by the chaser.

The chaser feature works as follows: a sequence of lighting scenes is loaded in a cyclic manner to DMX channels 1 to 128. To maintain this, the DMX levels which are stored in presets no. 128 to 383 for **DMX channels 385 to 512 are exclusively copied to the DMX transmit buffer channels 1 to 128** and transmitted on the DMX bus (**see commands for partial saving and loading of presets below !**)

Example: if the chaser cycle is set to 4 and the chaser start is set to 128, then presets no. 128,129,130,131 are loaded partially, then preset no. 128 again and so on. Any other DMX channels 129 to 512 are not influenced by the chaser and may be operated independently.

Set chaser step duration with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 16 (hex 10)

the duration of chaser steps is controlled by an internal timer

2nd data byte = (1 - 127) chaser step duration in 1/10 second units

or = 0 switches the chaser **OFF**

at delivery, a default step duration of 2 seconds is active to provide an easy start

Alternative:

1st data byte = 17 (hex 11)

then the chaser is stepped by MIDI CLOCK signals at MIDI IN:

12 MIDI CLOCK messages = ca 1/8 note = 1 chaser step unit

2nd data byte = (1 - 127) step duration in 1/8 notes

or = 0: switches the chaser **OFF**

Comment: After the duration of any chaser step is over, the chaser automatically loads a part of the next preset in the cycle: **stored DMX levels from DMX channels 385 to 512 are copied to and transmitted at the DMX bus channels 1 to 128.** As soon as <cycle length> presets are loaded in sequence, the procedure is repeated from <chaser start>.

Set chaser start preset (lighting scene) with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 19 (hex13) ,

2nd data byte: **offset = start preset minus 128** (0 - 127)

alternatively the chaser cycle may start with a preset in the range 256-383:

1st data byte = 20 (hex14)

2nd data byte: **offset = start preset minus 256** (0 - 127)

Comment: due to limited coding possibilities with MIDI channel messages, the number of the start preset of the chaser cycle cannot be entered directly. Instead, the **difference to preset no.128 is entered as the 'offset'**.

If start preset no.128 (offset=0) is chosen (default setting at delivery) the chaser cycle always starts with loading preset no. 128. **With an offset unequal to 0 the chaser cycle starts with preset no. (128 + offset).** If the chaser would access a preset beyond 383, the sequence continues with loading preset no. 128 etc..

By use of this method, a big number of individual chaser effects may be created and run easily and flexible. This arrangement is optimized for lighting installations, which use a maximum of 384 DMX channels in normal operation and whose chaser effects are concentrated on DMX channels 1 to 128. Experience has shown that this is the case for stage lighting of most music bands and small theaters.

Set the merge method (transmission from DMX OUT) for DMX channel ="SLOT"
with:

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 0 always transmit the MIDI activated level from the transmit buffer

data byte = 1 transmit the less of transmit buffer and DMX IN

data byte = 2 transmit the greater of transmit buffer and DMX IN

data byte = 3 always forward the level from DMX IN to the transmit buffer

data byte = 4 transmit the last changed one of MIDI activated transmit buffer
or DMX IN ("last takes precedence")

Set the merge method (transmission from DMX OUT) equally for DMX channels
with:

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 80 (hex 50) transmit the MIDI activated level from the transmit buffer

data byte = 81 (hex 51) transmit the less of transmit buffer and DMX IN

data byte = 82 (hex 52) transmit the greater of transmit buffer and DMX IN

data byte = 83 (hex 53) always forward the level from DMX IN to the transmit buffer

data byte = 84 (hex 54) transmit the last changed one of MIDI activated transmit

buffer or DMX IN receive buffer ("last takes precedence")

Poll DMX transmit buffer at DMX channel ="SLOT" and subsequent ones with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 36 (hex24) or 37 (hex 25)

2nd data byte (0 - 127) = number of polled DMX channels (1 -128)

2nd data byte=0 polls 128 channels

Comment: If the **1 st data byte = 36**, a sequence of **NOTE ON** and **CONTROL CHANGE** messages is returned, **their format is identical to the short format used to set a DMX level**. When this method is used, every single MIDI message contains the corresponding DMX channel, but possibly different MIDI channels are contained in a block of messages.

However, if the **1 st data byte = 37**: first a **POLY KEY PRESSURE** message is returned which contains **the number of the first DMX channel to be read out**, followed by a **POLY KEY PRESSURE** with the 1st data byte = 84 or 85 and the DMX level of this channel. Next follows a **sequence of POLY KEY PRESSURE** messages with **1st data byte = 86 or 87**, which report the respective DMX levels channel by channel in rising order. This message is more complex but needs only 1 MIDI channel.

Both message types may be recorded as MIDI file and be reloaded at a later time.

Poll DMX receive buffer at DMX channel ="SLOT" and subsequent ones with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 40 (hex28) response:sequence of MIDI NOTE ON and/or CONTROL CHANGE messages as "express setting" described above

1st data byte = 41 (hex 29) response: first a POLY KEY PRESSURE (no. of the 1st DMX channel) followed by POLY KEY PRESSURE messages (successive DMX levels).

1st data byte = 42 (hex 2A) response: first a MIDI NOTE ON or CONTROL CHANGE followed by CHANNEL PRESSURE messages (successive DMX levels).

2nd data byte (0 - 127) = number of polled DMX channels (1 -128)

2nd data byte=0 polls 128 channels

Set length of DMX cycle to the actual value of "SLOT" with

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 64 (hex40)

Comment: The minimum length of the DMX cycle (= number of DMX channels transmitted between two DMX reset pulses) is 24.

Set hue (color tone) of a RGB lamp with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 76 (hex 4C): command specifier

2nd data byte = hue (color tone) 0 - 127

This will approximately result in following colors. Intermediate hue values will result in intermediate colors:
2nd data byte= 0:red, 22:yellow, 43:green, 64:cyan, 85:blue, 106:magenta, 127:red again.

In correspondence with the model of the driven lamp and setting of saturation and brightness the resulting color tone may differ somewhat. **See detailed comment at ASCII command H.**

Set color saturation of a RGB lamp with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 77 (hex 4D): command specifier

2nd databyte = color saturation 0 - 127 (is internally multiplied by 2, **see ASCII command W**)

Set brightness of a RGB lamp with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 78 (hex 4E): command specifier

2nd databyte = brightness 0 - 127 (is internally multiplied by 2, **see ASCII command "**)

Switch acceptance of Running State in commands with:

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 56 (hex38) Running State in commands **forbidden** (default when RevNo.<77)

= 57 (hex39) Running State in commands is **accepted** (default now)

Comment: "Running State" means that subsequent equal status bytes are not repeated.

Switch Running State behaviour in messages sent at MIDI OUT with:

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 58 (hex3A) **No** Running State in messages (default)

= 59 (hex3B) Running State is **activated**

CONTROL CHANGE sets odd DMX levels: activate and deactivate with:

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 60 (hex3C) DMX level = MIDI data byte *2

data byte = 61 (hex3D) DMX level = MIDI data byte*2+1 (default)

Comment: when this effect is deactivated, CONTROL CHANGE commands have the same effect as NOTE ON commands. As a special case with NOTE ON as well as with CONTROL CHANGE the MIDI-data byte 127 is transformed into DMX level 255.

Exchange CONTROL CHANGE and POLY KEY PRESSURE functionally with:

PROGRAM CHANGE

MIDI channel as position of code switch

databyte = 62 (hex3E) CONTROL CHANGE and POLY KEY PRESSURE have the same meaning as described in this manual (default)
= 63 (hex3F) **all functions of CONTROL CHANGE and POLY KEY PRESSURE are exchanged with respect to the description in this manual**

Comment: With this option, the command set may be better fitted to the particular control task or features of the given control equipment.

The default setting as described in the manual is most appropriate when the control equipment (sequencer for example) provides good possibilities to work with POLY KEY PRESSURE messages and/or simultaneous fade ramps shall be (graphically) executed on a sequencer with CONTROL CHANGE messages. Recommended for recording and control with sequencer.

The alternative setup with exchanged functions shows advantages if many settings are made preferably with a controller pad or with a programmable keyboard - especially when the POLY KEY PRESSURE messages, which are less frequently used in normal MIDI operation, are not or badly supported. Recommended for control by keyboard or controller pad and evaluation of lighting scenes (presets).

Block command execution with:

PROGRAM CHANGE

MIDI channel as position of code switch

databyte = 36 (hex 24)

Comment: this command is especially useful for example, when lighting scenes (presets, see below) to be loaded or saved are selected with a controller pad or similar. The blocking feature prevents that all intermediate values, which are emitted during motion of the controller, are executed. When programming of lighting scenes is made with a controller pad, we recommend that a pair of two key buttons is reserved for the block and unblock function.

Terminate blocking of commands and execute last received command immediately with:

PROGRAM CHANGE

MIDI channel as position of code switch

databyte = 37 (hex 25)

Comment: While blocking of commands is active, the MIDI / DMX Control Box remembers the last received MIDI channel message. This is executed immediately after the blocking is terminated. This feature is helpful to set controllers etc. fast and precisely without having troubles with meanwhile emitted values.

Save, store presets (= lighting scenes) with:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 112 (hex70): basic value 0 = save preset no. 0-127

113 (hex71): basic value 128 = save preset no. 128-255

114 (hex72): basic value 256 = save preset no. 256-383

2nd data byte = 0 to 127 (hex7F): add 0 - 127 to basic value

Comment: After the MIDI/DMX Control Box is switched on, the code switch is rotated or a Empfang MIDI SYSTEM RESET message is received, generally preset no.0 is loaded. Together with presets no. 0

bis 3 the system configuration is stored in addition to the actual lighting scene. This includes especially: format of DMX levels reported via MIDI OUT, running state behaviour, reaction on CONTROL CHANGE commands, LOOP, chaser loop. Together with presets no. 4 to 383 only the lighting scene is saved.

Special cases: store preset partially (new from May 2010 / revision number 83)

1st data byte = 115 to 117: basic value 0, i.e. partially store preset no 0 - 127
115 (hex73): store DMX channels 1-128 from DMX OUT to preset channels 129 - 256
160 (hex74): store DMX channels 1-128 from preset to DMX OUT channels 257 - 384
117 (hex75): store DMX channels 1-128 from preset to DMX OUT channels 385 - 512

1st data byte = 118 to 120: basic value 128, i.e. partially store preset no 128 - 255
118 (hex76): store DMX channels 1-128 from DMX OUT to preset channels 129 - 256
119 (hex77): store DMX channels 1-128 from DMX OUT to preset channels 257 - 384
120 (hex78): store DMX channels 1-128 from DMX OUT to preset channels 385 - 512

1st data byte = 121 to 123: basic value 256, i.e. partially store preset no 256 - 383
121 (hex79): store DMX channels 1-128 from DMX OUT to preset channels 129 - 256
122 (hex7A): store DMX channels 1-128 from DMX OUT to preset channels 257 - 384
123 (hex7B): store DMX channels 1-128 from DMX OUT to preset channels 385 - 512

2nd data byte in all cases = 0 to 127 (hex7F) = preset no. minus basic value

any other levels in the preset remain unchanged.

This feature is intended as a counterpart to the corresponding partial load command to produce and test "long" presets with small lamp configurations.

Attention: with this special feature exclusively DMX levels are re-stored, in no case the channel specific details of the system configuration. Especially when use is made of RGB triplets special care is necessary to keep the markings of these triples aligned with the shifted channel numbers.

Save flash pattern with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE) (new from Dec 2010 / rev.no 26)

MIDI channel as position of code switch

1st data byte = 126 (hex 7E)

2nd data byte = 15 (hex F) any values else are ignored

Comment: For detailed description, see flash start command above

Load presets (= lighting scenes) with:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 96 (hex60): basic value 0 = load preset 0-127

97 (hex61): basic value 128 = load preset 128-255

98 (hex62): basic value 256 = load preset 256-383

2nd data byte = 0 to 127 (hex7F): add 0 - 127 to basic value

Special cases: load preset partially (new from May 2010 / revision number 83)

1st data byte = 99 to 101: basic value 0, i.e. partially load preset no 0 - 127
99 (hex63): load DMX channels 129 - 256 from preset to DMX OUT channels 1 - 128
100 (hex64): load DMX channels 257 - 384 from preset to DMX OUT channels 1 - 128
101 (hex65): load DMX channels 385 - 512 from preset to DMX OUT channels 1 - 128

1st data byte = 102 to 104: basic value 128, i.e. partially load preset no 128 - 255
102 (hex66): load DMX channels 129 - 256 from preset to DMX OUT channels 1 - 128
103 (hex67): load DMX channels 257 - 384 from preset to DMX OUT channels 1 - 128
104 (hex68): load DMX channels 385 - 512 from preset to DMX OUT channels 1 - 128

1st data byte = 105 to 107: basic value 256, i.e. partially load preset no 256 - 383
105 (hex69): load DMX channels 129 - 256 from preset to DMX OUT channels 1 - 128
106 (hex6A): load DMX channels 257 - 384 from preset to DMX OUT channels 1 - 128
107 (hex6B): load DMX channels 385 - 512 from preset to DMX OUT channels 1 - 128

1st data byte = 108 to 111: basic value 0, i.e. partially load preset no 0 - 127
108 (hex6C): load DMX channels 1 - 128 from preset to DMX OUT channels 1 - 128
109 (hex6D): load DMX channels 129 - 256 from preset to DMX OUT channels 129 - 256
110 (hex6E): load DMX channels 257 - 384 from preset to DMX OUT channels 257 - 384
111 (hex6F): load DMX channels 385 - 512 from preset to DMX OUT channels 385 - 512

2nd data byte in all cases = 0 to 127 (hex7F) = preset no. minus basic value

any other levels in the preset remain unchanged.

This feature is intended to get more freedom with small installations.

Attention: with this special feature exclusively DMX levels are loaded, in no case the channel specific details of the system configuration. Especially when use is made of RGB triplets special care is necessary to keep the markings of these triples aligned with the shifted channel numbers.

Alternatively load presets (= lighting scenes) with :

NOTE ON

1st data byte (0 -127): preset number to be loaded **minus basic value**
corresponding with the range setting which was made before with a
PROGRAM CHANGE command

2nd data byte: not relevant, except when velocity = 0, then no preset is loaded

Comment: this method is useful to change lighting scenes quickly during "live operation", but helpful too for simple programming of a lighting track with a sequencer. As preparatory work an appropriate set of presets has to be produced. To manage this best, the number of keys of the keyboard which is used for this task has to be considered and possibly the pitch has to be transposed. If organized well, this way lighting design of a bigger repertoire may be performed quite efficient.

This mode of operation has to be activated before with a PROGRAM CHANGE command, data byte 96 to 111 (hex 60 to hex 6F). It remains active until it is changed with new PROGRAM CHANGE command. The default setting for NOTE ON is "short command" to simply set a DMX channel and level with one MIDI message.

Change mode of operation with :

PROGRAM CHANGE

MIDI channel as position of code switch

data byte 96 to 111 (hex 60 to 6F) **NOTE ON commands load presets (see above)**

listed more specifically:

data byte = 96 (hex 60): **basic value 0**, i.e. NOTE ON loads preset no. 0-127

97 (hex 61): **basic value 128**, i.e. NOTE ON loads preset no. 128-255

98 (hex 62): **basic value 256**, i.e. NOTE ON loads preset no. 256-383

and additional **special functions:** load presets partially

data byte = 99 ... 101: basic value 0, i.e. loads preset no. 0 - 127 partially

99 (hex63): NOTE ON loads DMX channels 129 - 256 from preset to DMX OUT chan. 1 - 128

100 (hex64): NOTE ON loads DMX channels 257 - 384 from preset to DMX OUT chan. 1 - 128

101 (hex65): NOTE ON loads DMX channels 385 - 512 from preset to DMX OUT chan. 1 - 128

data byte = 102 ... 104: basic value 128, i.e. loads preset no. 128 - 255 partially

102 (hex66): NOTE ON loads DMX channels 129 - 256 from preset to DMX OUT chan. 1 - 128

103 (hex67): NOTE ON loads DMX channels 257 - 384 from preset to DMX OUT chan. 1 - 128

104 (hex68): NOTE ON loads DMX channels 385 - 512 from preset to DMX OUT chan. 1 - 128

data byte = 105 ... 107: basic value 256, i.e. loads preset no. 256 - 383 partially
105 (hex69): NOTE ON loads DMX channels 129 - 256 from preset to DMX OUT chan. 1 - 128
106 (hex6A): NOTE ON loads DMX channels 257 - 384 from preset to DMX OUT chan. 1 - 128
107 (hex6B): NOTE ON loads DMX channels 385 - 512 from preset to DMX OUT chan. 1 - 128

data byte = 108 ... 111: basic value 0, i.e. loads preset no. 0 - 127 partially
108 (hex6C): NOTE ON loads DMX channels 1 - 128 from preset to DMX OUT channels 1 - 128
109 (hex6D): NOTE ON loads DMX channels 129 - 256 from preset to DMX OUT ch. 129 - 256
110 (hex6E): NOTE ON loads DMX channels 257 - 384 from preset to DMX OUT ch. 257 - 384
111 (hex6F): NOTE ON loads DMX channels 385 - 512 from preset to DMX OUT ch. 385 - 512

data byte = 120 (hex 78) **NOTE ON messages set DMX channel and level.**
velocity = 0 is accepted and sets the DMX level to 0 (=default)
Any NOTE OFF message (with arbitrary velocity) sets the DMX level to 0

data byte = 121 (hex 79) **NOTE ON messages set DMX channel and level.**
NOTE ON messages with 2nd data byte (velocity) = 0 are ignored
and all NOTE OFF messages are ignored
but: NOTE ON messages with velocity = 1 set the DMX level to 0.

data byte = 122 (hex 7A) ask configuration of DMX channel= "SLOT"
The response comes as a SysEx message with content equal to ASCII command "Q".

data byte = 124 (hex 7C) activates merge operation with **internal DMX clock**

data byte = 125 (hex 7D) activates merge operation with **external synchronisation** by the signal at DMX IN

data byte = 126 (hex 7E) activates **DMX receive** operation.
The **signal is looped through** from DMX IN to DMX OUT.
It can be polled and automatic messages can be activated

data byte = 127 (hex 7F) **clears** all buffers and working registers
This implies: the actual mode of operation is reset to the default setup.
However: presets are not modified

Comment: Many MIDI sequencers and other MIDI control equipment demand that when formatting a PROGRAM CHANGE message the data value (program number) has to be entered one higher than physically transmitted as data byte.(Example: user entered program #1 is transmitted as hex C0 00)
The MIDI/DMX Control Box evaluates PROGRAM CHANGE messages always according to the physically received data bytes.

Some MIDI devices automatically send together with every PROGRAM CHANGE message a BANK SELECT message (=CONTROL CHANGE to controller no.20.). If this conflict is noticed in you installation, we recommend not to address any DMX receiver to DMX channel no. 20.

As an alternative, all commands which can be performed with a PROGRAM CHANGE can be sent as a POLY KEY PRESSURE command, whose 1st data byte is =127. Then its 2nd data byte is equal to the data byte of the corresponding PROGRAM CHANGE command.

Reset all DMX channels to "0" and configuration to default with :

PROGRAM CHANGE

MIDI channel as configured

Datenbyte = 127 (hex7F)

Action memory: the merger transmits exclusively from the transmit buffer.
All automatic messages are switched OFF.

Transmit buffer: All DMX levels of the transmit buffer are reset to "0".
Number base = "decimal". Masterfader = 100% The chaser is switched OFF.
LOOP = 512, FADETIME = 0.0. Presets are not deleted or changed otherwise.

Data format MIDI reset:

data byte = 255 (hexFF) MIDI System Reset

When this command is received, the LED of the Control Box is darkened for about 1 second and it is restarted in the mode of operation given by the code switch and then loading preset no.0.

2nd part of the manual "MIDI/DMX Control Box":

System Exclusive Commands enclosing ASCII text

General structure of commands:

<SysEx header> CmdType Parameter [CmdType Parameter] <EOX>

The System Exclusive command frame:

The SysEx command frame embeds ASCII text messages as a kind of brace to make them compatible with the MIDI format.

The **SysEx header** always is the same 4 byte sequence, which contains the Cinetix SysEx manufacturer ID:

hexadecimal: **F0 00 20 5D**

or decimal: **240 0 32 93**

EOX is a single byte, which has the value **hexF7=decimal 247** generally throughout MIDI.

Into this System Exclusive command frame different ASCII text commands can be packed or enclosed.

--- **First the SysEx header opens the ASCII command interpreter.**

--- **Then every ASCII command is executed as soon as the necessary number of ASCII characters is entered** and interpreted. Input of any number is automatically finished when the maximum number of digits - context dependent - is entered. Numbers smaller than maximum are finished by starting the next command (input of a command letter) or when an adequate number or leading zeroes is put ahead or the SysEx packet is finished with EOX.

--- **The EOX-message causes** immediate execution of the last entered command and **closes the ASCII command interpreter. From now on incoming data are evaluated as MIDI channel messages** (until a next SysEx header).

Within a SystemExclusive command frame or packet an arbitrary number of ASCII commands can be contained, but the text length per SysEx packet should not exceed 80 characters.

Now the ASCII commands are described as such:

Every control command and every state message starts with a single **characteristic letter**. If necessary, it is followed by a **number as parameter**.

Spaces are not interpreted, i.e. can be used to make the commands more readable

All characters are interpreted case independent. Feedback, however, is always written in capitals.

Invalid commands are responded by an **error message**, which is a question mark "?" embedded into a SysEx frame.

Elementary format of the most frequently used operations:

The **number of the DMX channel** (1 - 512) to be addressed by subsequent commands is set with the command "**S**" followed by the number of the DMX channel as decimal number of one to three digits. This is stored in the internal SLOT register (see appendix A). No action else is directly triggered by this command.

Next enter the **level (lighting intensity, "value") for this DMX channel** with the command "V" followed by the desired value as a decimal number 0 to 255 (must have three digits or be directly followed by another command or must be terminated with <return>).

Example: S1V45 <return>
is equivalent to **S001V045**

If you want to **start a complete fade process** with one command, first set the fadetime with command "T" (parameter in seconds, optionally tenths of seconds separated by a period). It works on all following commands which set the DMX level until a new fadetime is entered. It may be changed immediately after a level command without retroactive effect on given commands. **Fadetime = 0 switches directly to the final level.** Max. fadetime is 31.9 sec.

Example: T3.5S1V45<return>

If you want to set values of a block of consecutive DMX channels, use the **',' (comma)** command to pre-increment the DMX address before setting a new value:

Example: S1V45,0,255,36....<return>
modifies DMX channels 1,2,3,4, and fades with the previously set fadetime.

The signal at DMX IN may be evaluated by different methods. The most simple but not most flexible is **polling DMX levels with the command R:**

Example: S3R100 returns the actual DMX IN -levels of channels 3 to 103 .

More elegant is **activation of "automatic messages" with command N or / ("slash"):**

Example: after being activated with **S1N8** a message is transmitted automatically via MIDI OUT, as soon as the DMX level of channel no. 1 has changed at least 8 in relationship to the previous message. After command **/8** a message is released automatically, when any DMX channel changed by 8 or more.

The command "Q" does not change any values but it informs about the actual settings of the DMX channel which was addressed with the "S" command before. Typical example:

Mi: CH=1 OUT=13(U) TX=27 MF=50% RX=34 MSG=012/0 CS=128/0/20 L=512 T=3.2

Comment: OUT is the presently transmitted DMX level of the channel addressed by CH= SLOT. TX describes the content of the transmit buffer of this DMX channel. Both differ when the global masterfader (item MF) differs from 100%. RX is the level actually received at DMX IN. MSG reflects the settings for messages about the level at DMX IN: 1st digit: setting by command 'X', 2nd digit: setting by command 'Y' (global settings). 3rd digit reflects channel the channel specific setting of the MIDI-Format for automatic messages (code sw.0-7,F) or activation (code sw. 8 - C). The value following the slash is the threshold for "automatic"messages. CS describes start, length and timing of the chaser cycle. L shows the number of channels transmitted per DMX cycle and T reports the fade time set by a previous command.

General structure of commands:

CommandCode Parameter [CmdCode Parameter] <CARRIAGE RETURN=hexD>

Every command causes its parameter to be written into the corresponding internal register. Depending on the type of command, when all information is entered the intended action is started using the actual content of ALL OF THE REGISTERS. For this reason, the order of entering data may influence the result. In general **the order FADETIME, SLOT, DMX level should be followed**, but most times not all of the registers have to be updated.

Parameters (i.e. numeric values) for DMX channel (SLOT), FADETIME and LOOP always have to be entered in **decimal format** and are fed back in decimal.

Parameters for DMX levels may be entered as a **decimal number, as a hex number or percentage scale** and are messaged in this active number base.

Any DMX level is stored internally as one byte (8 valid bits) i.e. can take values between 0 and 255.

Short reference of all ASCII commands

Sn	address DMX channel (write SLOT register) for subsequent action (n=1 - 512)	p.28
Vn	set DMX level at DMX channel=SLOT (n=0 - 255)	p.28
,n	(comma) increment SLOT first then set level at new DMX channel (n=0 - 255)	p.28
=n	fill block of n DMX channels starting from SLOT+1 with level of SLOT (n=1-512)	p.29
+	increase transmit buffer level DMX channel=SLOT by one	p.29
-	decrease transmit buffer level DMX channel=SLOT by one	p.29
^n	add n to transmit buffer level DMX channel=SLOT (n=0 - 255)	p.29
_n	subtract n from transmit buffer level DMX channel=SLOT (n=0 - 255)	p.29
\$	from now DMX level in HEX (only V- , comma- , ^ , _- , R- , Z- , Q- command)	p.29
&	from now DMX level DECIMAL (only V , comma , ^ , _ , R , Z and Q-command)	p.30
%	from now DMX-level in PERCENT (only V , comma , R , Z and Q- command)	p.30
Ts.t	set FADETIME s=seconds t=tenths	p.30
!	stop all fade processes and freeze them at actual DMX level	p.30
Jn	copy DMX receive buffer starting from channel=SLOT into the transmit buffer	p.30
Zn	read n bytes starting at DMX channel=SLOT from transmit buffer to MIDI OUT	p.31
Rn	read n bytes starting at DMX channel=SLOT from receive buffer to MIDI OUT	p.31
U	always transmit DMX OUT at channel=SLOT from the transmit buffer (n=0)	p.31
K	transmit less buffer level at channel=SLOT (n=1)	p.32
G	transmit greater buffer level at channel=SLOT (n=2)	p.32
o	always transmit DMX OUT at channel=SLOT from the receive buffer (n=3)	p.32
P	transmit last changed buffer level at channel=SLOT (n=4)	p.32
*n	set merge method (n=0,1,2,3,4 see above) globally for all DMX channels	p.32
Nt	configure automatic message at channel=SLOT with threshold t (t=0-127)	p.33
/t	configure automatic messages for all DMX channels equally with threshold t	p.34
Yn	switch automatic messages globally ON/OFF and select format(n=0 - 4)	p.34
Xn	switch messaging of DMX triggered strings ON/OFF (n=0,1,2,3)	p.34
: <memory name> <string>	" enter and store user programmable string	p.36
; <memory name>	check string, message it for test via MIDI OUT	p.37
Mn	set the masterfader: n=0 to 200 (in percent, see detailed description below)	p.37
in	set length of chaser cycle (n=2 to 127) and start the chaser	p.37
>t	set duration of chaser step in 1/10 s units	p.38
(n	enter start scene (preset no.) of chaser cycle . See detailed description	p.38
)	forward chaser immediately for one step	p.38
<t	release a lighting flash: all DMX channels are pulsed to 100% for t * 1/10s	p.38
H	set hue (spectral color) for RGB lamp	p.38
W	set color saturation for RGB lamp	p.39
"	set brightness (luminance) for RGB lamp	p.39
Ln	set length of DMX Cycle (n=24 - 512)	p.39
Q	show content of all DMX registers at DMX channel=SLOT via MIDI OUT	p.40
[run DMX transmitter/merger with internal timing clock	p.40
\	run DMX transmitter/merger synchronised by DMX IN	p.40
]	loop DMX receiver from DMX IN to DMX OUT	p.40
~n	save transmit buffer and configuration as preset no. n	p.41
@n	load preset Nr. n into buffers and update configuration	p.41
{]	"download", nonvolatile memory (i.e. backup all presets to PC) via MIDI OUT	p.42
{ ["upload" saved backup (all presets) via MIDI OUT into nonvolatile memory	p.42
 	reset all buffers and configuration to default (delivery) state	p.43
'	(Apostroph) return firmware version number via RS-232	p.43

Detailed description of all ASCII commands:

All **ASCII commands have to be packed into a SysEx frame (except rotary switch position E)**, details see above. Here only the ASCII part is described.

Every control command and every state message starts with a single characteristic letter. If a command expects a parameter, it is listed here in acute angular brackets <...>. The brackets are used to mark parameters within the manual, but they are not part of the data transfer.

Addressing the DMX channel ("slot") to be operated with following commands:

S <channel number>

The **parameter addresses a DMX channel**, on which many of the subsequently described commands have an effect. Internally the parameter value is stored in the SLOT register.

In DMX slang sometimes the word "slot" is used as synonym for DMXchannel because during DMX transmission every DMX channel is represented by a specific time slot in the transmission cycle.

Parameter: slot number (range 1 to 512) is the number of the DMX channel to be manipulated with subsequent commands

Comment: No action is started immediately. But the register content will be used for subsequently given commands.

Example: S123 writes 123 into register SLOT

Transmit buffer manipulation:

V <level>

Write parameter into the transmit buffer of DMX channel = "SLOT".

Parameter: level (range 0 to 255) is the value (lamp intensity, e.g.) which will be transmitted at the DMX channel addressed by SLOT.

Comment: Changes the transmitted DMX packet sequence in accordance with previously entered values in the SLOT and FADETIME register. It depends on the entry of the action memory of the addressed DMX channel if this new level is transmitted actually.

If FADETIME is equal to zero, the value of the addressed DMX channel is immediately set to <level >

If FADETIME is nonzero, a fade process is started, which begins at the actual value of the addressed DMX channel and finishes, when the value of the addressed DMX slot is equal to <level>.

Example: V34 sets the DMX level to 34 at the DMX channel which is actually addressed by SLOT (i.e. selected before with the "S" command). The parameter is interpreted in the active number base (decimal=default, hex or percent).

, (comma) <level>

First this command increases the SLOT register automatically, then it writes the parameter into the transmit buffer for the new DMX channel = "SLOT".

Parameter: level (range 0 to 255) is the value or intensity which will be transmitted at the DMX slot addressed by the new, incremented SLOT.

Comment: except the fact that the SLOT register is pre-incremented, the ',' (comma) command does the same as the V command.

= <block length>

This command writes the final level of the DMX channel addressed by SLOT into the number of <block length> DMX channels starting from (SLOT+1). Starting from the actual level of each of these channels a new fade to this final level is started. The fade time is given by the actual content of the FADETIME register.

Parameter: <block length> (1 to 512) is the number of DMX channels into which the same level is copied. Independent of the value of <block length> DMX channel no.512 is not exceeded.

+ (no parameter)

Increase (add 1 to) the level of the DMX channel addressed by SLOT

Comment: The byte cannot be made greater than decimal 255. If it is already equal to 255, the + command is ignored.

Any fade process currently active on this time slot is cancelled immediately.

- (minus, no parameter)

Decrease (subtract 1 from) the level of the DMX channel addressed by SLOT

Comment: The byte cannot be made less than 0. If it is already zero, the - command is ignored.

Any fade process currently active on this time slot is cancelled immediately.

^ <summand>

Adds <summand> in the transmit buffer to the actual level of the DMX channel addressed by SLOT (and starts a new fade process)

Comment: The final level cannot be made greater than decimal 255. If the addition would cause an overflow, the level is limited to 255.

Has essentially the same function as the V command. However, instead of an absolute DMX level the sum of the actual level plus <summand> is entered as the final level of a new started fade or switch process.

_ <subtrahend>

Subtracts <subtrahend> in the transmit buffer from the actual level of the DMX channel addressed by SLOT (and starts a new fade process)

Comment: The final level cannot be made less than 0. If the subtraction would cause a borrow, the level is limited to 0.

Has essentially the same function as the V command. However, instead of an absolute DMX level the difference of the actual level minus <subtrahend> is entered as the final level of a new started fade or switch process.

\$ (no parameter)

Set number base for input/output of VALUE as **hexadecimal**

Comment: All following parameter values of the commands V, ',' (comma), ^ and _ are interpreted as hexadecimal numbers (0 to FF). This behaviour remains active until a different number base is set with

another command. Because the number base is stored in presets no. 0 to 9, loading of a preset may change the active number base.

All messaged DMX level values are coded as hexadecimal numbers with a prefix "\$".

& (no parameter)

Set number base for input/output of DMX levels as **decimal**

Comment: All following parameter values of the commands V, ',' (comma), ^ and _ are interpreted as decimal numbers (0 to 255). This behaviour remains active until a different number base is set with another command. Because the number base is stored in presets no. 0 to 9, loading of a preset may change the active number base.

All messaged DMX level values are coded as decimal numbers without specifier symbol.

% (no parameter)

Set number base for input/output of DMX levels as **percent**.

Comment: All following parameter values of the commands V, ',' (comma), ^ and _ are interpreted as hexadecimal numbers (0 to 100). This behaviour remains active until a different number base is set with another command. Because the number base is stored in presets no. 0 to 9, loading of a preset may change the active number base.

All messaged DMX level values are coded in percentage scale with a postfix "%".

The use of the **percentage scale is significantly less accurate as the decimal or hex scale**. Internally the MIDI/DMX Control Box always operates with the full scale 0-255 while input/output is rounded as integer. Consequently the repeated use of the percentage scale may cause remarkable differences from a precise calculation.

T <seconds.tenths>

Enter parameter into **FADETIME**. No action is started directly.

Parameter: Fadetime is always entered in seconds. **Optionally** - separated by a period - tenths of seconds can be added. Maximum fadetime is 31 seconds plus 9 tenths of a second.

Example: **T13.4** sets **FADETIME** to 13.4 seconds

! (no parameter)

All fade processes are stopped immediately and all DMX channels are freezed on their present

J <block size>

Copies a block of <block size> actual DMX levels from the **receive** buffer into the **transmit** buffer starting from the DMX channel which is currently addressed by the **SLOT** register (**S** command)

Comment: At elder firmware versions (revision number <83) this command did only copy the **single** DMX level of the addressed DMX channel. In the actual firmware version it combines the features of the commands **J** and **H** used in previous old versions.

Example: S1J512 copies the complete receive buffer into the transmit buffer. This is a useful feature to save to save an externally created lighting scene which is received at DMX IN as a preset of the MIDI/DMX Control Box, using the ~ command. Replaces command H of earlier firmware versions.

Poll the transmit buffer:

Z <number of bytes>

Poll <number of bytes> of the DMX transmit buffer starting from the DMX level = SLOT and send them via MIDI OUT.

Parameter: number of polled bytes (1 to max. 128)

Syntax of the resulting state message:

s <1st channel no.> v [\$]DMX level[%] [,[\$]DMX level[%]] <CR = dec13 =hexD>

Comment and example see below at the R command

Readout (poll) the receive buffer:

R <number of bytes>

Read out (retrieve) a block of <number of bytes> starting from DMX channel = SLOT and message it to the host PC in ASCII text format

Parameter: number of bytes to be read (1 to max. 128)

Structure of a read out message responding to the R command:

S <1st channel no.> v [\$]DMX level[%] [,[\$]DMX level[%]] <CR =dec13 =hexD>

DMX levels reported as decimal numbers are sent without a specifier symbol

DMX levels reported in hexadecimal are sent with prefix '\$'

DMX levels reported in percent are sent with post added '%'

It is impossible to read data out of DMX packets with nonzero start byte.

Comment: The first byte is read from the DMX channel actually addressed by SLOT. The printout of every polled DMX level is followed by a comma, the end of the message is marked with a <CARRIAGE RETURN =hexD>. This format is suitable for recording, it corresponds with the command sequence to set the same level configuration at DMX OUT.

Automatic messages (see N command) are an alternative to polled messages.

Example: S100R8 reads a block of 8 bytes starting from DMX channel no.100

Set the method to merge MIDI entered commands with levels at DMX IN:

Throughout this manual, "merge" means switching or multiplexing the source of the data transmitted at DMX OUT channel by channel between transmit buffer and receive buffer. It does not mean linear superposition of both values!

Any of the commands described in this part remains active until it is revised by another command. Note that merge methods and threshold values are stored for every channel in any of the presets no. 0 to 9. So loading of one of these presets can change the merge method and automatic messages. Loading presets 10 to 299 does not affect the merge method.

U (no parameter)

Transmit from the transmit buffer at the DMX channel addressed by SLOT

Comment: This method of merging transmit buffer and receive buffer is valid until the next command is given that changes this feature.

Example: S99U enables transmission of the actual value of the transmit buffer at DMX channel 99

K (no parameter)

Send the the less of transmit buffer and receive buffer at the DMX channel addressed by SLOT

Comment: This method of merging transmit buffer and receive buffer is valid until the next command is given that changes this feature.

Example: S99K enables transmission of the less of the actual values of the transmit buffer and receive buffer at DMX channel no.99

G (no parameter)

Send the the greater of transmit buffer and receive buffer at the DMX channel addressed by SLOT

Comment: This method of merging transmit buffer and receive buffer is valid until the next command is given that changes this feature.

O (no parameter)

Transmit from the receive buffer at the DMX channel addressed by SLOT

Comment: This method of merging transmit buffer and receive buffer is valid until the next command is given that changes this feature.

Example: S99o enables transmission of the actual value of the receive buffer at DMX channel 99

P (no parameter)

Send either transmit buffer or receive buffer which has changed last at the DMX channel addressed by SLOT

Comment: This method of merging transmit buffer and receive buffer is valid until the next command is given that changes this feature.

An attempt to reload the existing value to the transmit buffer is handled as "change".

Example: S99P enables transmission of that one of the actual values of the transmit buffer and receive buffer at DMX channel no.99 **which has changed last** (to any value).

* <method>

Set the merge method for all DMX channels identically:

Parameter: method

0= send all channels from the transmit buffer

1= for all channels send the the less of transmit buffer and receive buffer

2= for all channels send the the greater of transmit buffer and receive buffer

3= send all slots from the receive buffer

4= for all slots either send transmit buffer or receive buffer which has changed last

Example: *3 causes DMX transmission exclusively from the transmit buffer.

Trigger data output from MIDI OUT by signal change at DMX IN

Specific procedures for different kinds of task are available:

1.) "**Automatic Messages**" are transmitted as soon as the DMX IN signal level at an activated DMX channel changes for a certain amount. Then a specially formatted character string is transmitted via MIDI OUT. This string has essentially the same format as the command, which would have to be sent to reproduce the corresponding signal level at DMX OUT.

This way "Automatic Messages" are especially useful for recording and later playback as well as for analytic evaluation of specific signal levels at DMX IN. As an essential feature of this kind of message **the trigger level can be adjusted arbitrarily** per DMX channel. This way the extracted amount of data can be fitted optimal for the respective application.

2.) Up to 108 arbitrarily "**preprogrammed**" strings can be stored nonvolatile in the MIDI / DMX Control Box by the user. These are transmitted automatically after certain level changes at the DMX channels 500 to 512 via MIDI OUT. This way external equipment with a serial port may be controlled by means of a DMX bus to a limited extent.

3.) New added is a similar freely applicable feature, which "spontaneously" uses the levels of the DMX channels 458 to 499: **A level change at DMX channel 458** (trigger slope from 0 to 100%) **forces transmission of exactly the present levels of DMX channels 460 to 499 as a binary string** via MIDI OUT. The number of transmitted characters (1 to 40) is defined by the present DMX level at channel 459 ("counted string").

N <threshold>

activate automatic messages at the DMX channel currently addressed by SLOT and set the threshold level of the messages

Parameter n: threshold - describes how much the **received** DMX signal at the DMX channel addressed by SLOT must change with respect to the previous message until an automatic message is released:

n=0: deactivate automatic messages for this slot

n=1: every change of received data releases an automatic message

With all threshold settings else (2 through 127=hex7F,49%), a message is sent when the current DMX value of the respective DMX slot differs at least the threshold from the value reported or polled before. For many typical applications a good compromise between sensitivity and amount of data is a threshold of 8.

Special option: automatic messages in MIDI compatible **binary format:**

are activated with the command Y2. The binary coded messages are better readable with some kind of control equipment. See part 2 of this manual. Command Y1 switches back to output in ASCII format (default).

Structure of an automatically generated ASCII message:

S <slot> v [\$] DMX level [%] <CARRIAGE RETURN=hexD>

Subsequently queued automatic messages are separated with a <CARRIAGE RETURN=hexD>.

Example: S27N8 from now on, every change of the received signal at slot no.27, which differs more than 8 from the previous message, releases an automatic message.

Structure of an automatically generated binary message (after command Y1):

same format as MIDI compatible commands for setting a DMX level, see pages 27/28.

/ <threshold>

activates automatic messages for all DMX channels and sets the same threshold level for all channels

Parameter: threshold - describes how much the **received** DMX signal at a given channel must have changed with respect to the previous message until an automatic message is released.

Comment: All details are identical with the N command. Please note that the data capacity of the MIDI interface is quite limited. So, this command should only be used, when changes at the DMX512 bus take place under controlled conditions. Else messages may be corrupted or lost.

Y <0 , 1 , 2 , 3 , 4>

switch automatic messages globally ON and OFF and select format

Y0 switch automatic messages globally **OFF**

Y1 switch automatic messages globally **ON** as **MIDI channel messages** (see p.24)

Y2 automatic messages globally **ON** in **ASCII** format **without timestamp** (default)

Y3 automatic messages globally **ON** in **ASCII** format **with relative timestamp**

format: r <thze> S <DMX channel no> v [\$\$]level[%] <CR>

<thze> is the time difference in 1/25 second units since the pervious automatic message

Y3 automatic messages globally **ON** in **ASCII** format **with absolute timestamp**

format: t <thze> S <DMX channel no> v [\$\$]level[%] <CR>

<thze> is the time difference in 1/25 second units.

Comment: Settings of these parameters are stored in any of the presets no 0 to 3.

When switched off globally with Y0, all automatic messages are kept stored. Only their output is suppressed. When switched on again with Y1, Y2,Y3 or Y4, the previously stored messages and trigger levels reactivated, but their output format may change.

Comment on timestamps : Followwing 9999 the count restarts in cyclic repetition from 0. So at maximum time differences up to 6.7 minutes can be distinguished.

Automatic messages with timestamp are not applicable for recording and later 1:1 playback via DMX OUT, but they may be helpful for manual analysis of lighting sequences at DMX IN.

X <0 ,1, 2 , 3>

Switch emission of preprogrammed strings globally ON and OFF

X0 Switch emission of strings globally **OFF** n

X1 Switch emission of preprogrammed strings (DMX channel 500-512) globally **ON**

X2 Switch emission "spontaneous" of strings (DMX channel 458-499) globally **ON**

X3 Switch emission of strings (DMX channel 458-512) globally **ON**

Important safety information:

Data transmission via DMX512 is technically regarded to be "unreliable". For this reason **it is STRICTLY FORBIDDEN to use the techniqe described here together with all safety critical applications, where malfunction could result in personal injury oder noticeable material damage !**

Comment: Wen switched off globally with command Xo, all preprogrammed strings to be called with appropriate levels at DMX channels 500-512 are kept stored. Only their output is suppressed. This setting

is stored in any of the presets no 0 to 9. When switched on again, the previous configuration is reactivated exactly and completely.

New added is the similar freely useable feature, which exploits "spontaneous" levels of the DMX cannels 458 to 499: if the level of DMX channel no. 458 changes (trigger edge from 0 to 100%) exactly the actual levels of the DMX channels 460 to 499 are sent as binary string via MIDI OUT. The number of transmitted bytes (1 to 40) is defined by the DMX level at channel 459 ("counted string"). This way any kind and number of strings can be released via MIDI OUT "spontaneously" at runtime. Because this feature makes only sense when driven by a prezise working computer controlled DMX desk, it is deactivated by default and under normal operation. So the DMX channels 458 to 499 may be used for normal operations without limitation. Furthermore free use is possible even in activated state as long as channel no 458 performs no triggering edge 0 to 100%level.

When specific data are received at DMX IN

send preprogrammed strings via MIDI OUT:

This allows messaging of user preprogrammed strings (text or binary) via MIDI OUT, if specific MDX levels are received at DMX IN on channels 500 to 512. How to program these strings using a PC keyboard is subsequently described. Maximum length of a string is 255 bytes.

This function is quite useful, for instance, to switch mixers or other media devices with MIDI or serial input from a separate DMX desk. Even coarse control of brightness and sound level is possible.

Table of "string names" in relationship with DMX channels and DMX levels which trigger messaging of strings:

DMX channel:

DMX level	500	501	502	503	504	505	506	507	508	509	510	511	512
hex 08	000	010	020	030	040	050	060	070	080	090	100	110	120
hex 28	002	012	022	032	042	052	062	072	082	092	102	112	122
hex 48	004	014	024	034	044	054	064	074	084	094	104	114	124
hex 68	006	016	026	036	046	056	066	076	086	096	106	116	126
hex 88	008	018	028	038	048	058	068	078	088	098	108	118	128
hex A8	00A	01A	02A	03A	04A	05A	06A	07A	08A	09A	10A	11A	12A
hex C8	00C	01C	02C	03C	04C	05C	06C	07C	08C	09C	10C	11C	12C
hex E8	00E	01E	02E	03E	04E	05E	06E	07E	08E	09E	10E	11E	12E

In the table above, every DMX "event" which is able to trigger a string message is assigned to a "string name", which is mnemonically oriented at the related DMX parameters.

A specific string is exclusively messaged, if

- 1.) at the specific DMX channel 500 to 512 **exactly** the specific one of the **sensitive DMX levels** as listed in the table above -Werte is recognized **new but stable** during 2 subsequent poll cycles. These poll cycles are not synchronized with the incoming DMX signal, i.e. the received DMX level has to be constant long enough (at least 0,2 seconds).
- 2.) a string has to be programmed by the user for the addressed string name. Unprogrammed or deleted strings are stored with length 0 and are not messaged.
- 3.) string messaging has to be globally activated (command "X1").

Any string is only messaged once. An example: if the level of DMX channel 500 is pulled from hexA8 to hexA9 and back again to hexA8, the string named 00A is not messaged again. For a repeated message, another string must be released at the same DMX channel before. There is a trick to message a certain string repeatedly: "message" an unprogrammed string name in between. This complicated looking procedure serves to avoid unintended messages when corrupted DMX data are received.

Messaging of strings controlled by DMX IN is internally handled with low priority and independently of all other modes of operation. Automatic messages take precedence but cannot slice a string which is in state of transmission.

For clarity we recommend to switch off automatic messages when string messaging is active.

String messaging is **globally switched ON** with the command **X1** and globally **switched OFF** with the command **X0** (programmed strings are not deleted). This setting is stored in the presets no. 0 to 9. Especially by saving preset no. 0 can be determined if this feature is active when the box is powered on. Default setting is OFF.

How to enter a string and store it:

: <string name> <string> "

The command code is a colon followed by the string name (3 places of ASCII text according to table above) of the DMX event which shall be assigned to the message. Letters A,C,E may be entered case independent.

Next the sting itself is entered as a sequence of ASCII text. It is terminated and permanently stored by a final quotation mark (ASCII code hex22). The quotation mark does not become part of the string, of course.

Example: :12eHello World"

"Printable" characters (i.e. all which can be entered directly with a standard keyboard - ASCII code range hex20 to hex7E) **are typed directly.**

To **enter any other byte value** (control characters, country specific letters), first type a backslash \ (ASCII code hex 5C), next type the ASCII code of that byte as **hex number of 2 digits** in ASCII text representation. A "new line" command would be entered as text sequence \0d\0a. No leading and trailing spaces should be typed because they will appear in the stored string.

The third "special key" is backspace (ASCII code hex08), which is used to delete a part of the entered string.

So, if any of the following characters shall become part of the string instead of being used to format the string, it has to be entered by use of the backslash method:

enter **backslash** \ as part of the string: type \5c

enter **quotation mark** " as part of the string: type \22

enter **backspace** as part of the string: type \08

Mouse functions, cursor keys as well as special function keys F1 to F12 are not recognised or evaluated.

Every typed character is echoed via MIDI OUT. "Printable" characters are echoed 1:1, any else is echoed by backslash plus two hex digits. If a printable character was entered in backslash style it will be echoed as in "printed" style. As an example: \40 would cause the echo @.

As soon as a quotation mark is added to the entered string, it is stored permanently within the Control Box. The string can be deleted or overwritten at any time (up to 10.000 repetitions). No "editing" is possible.

A string is **deleted** by entering : **<string name>**" , i.e. simply type a quotation mark directly after the string name. The string is empty then and has length 0. This configuration is identical with that of a new unprogrammed memory.

Check a user programmed string and view it via MIDI OUT:

; **<string name>**

The command code is a semicolon followed by the string name according to the table above.

First the number of stored characters is messaged, next the string itself in screen-readable format identical to the echo when programming strings as described above. "Non printable" characters are shown as a combination of a backslash followed by two hex digits. "Printable" characters are shown in "printed" style even if they were entered with backslash.

Attention: This test message is a readable "interpretation" of the memory programmed before. Every byte is stored binary. **When the strings are triggered by signals at DMX IN they are messaged in "raw binary", of course.**

M **<percent>**

Enter parameter for the masterfader. All DMX levels are modulated immediately

Parameter: The masterfader is always entered in decimal percentage scale (without postponed % sign and independent of the number base for DMX levels).

Default =100, maximum = 200, minimum = 0.

Comment: The masterfader works like a digital signal processor when the **transmit buffer is written into the DMX transmitter hardware**. It is useful for global adjustment of lighting scenes. **It does not change or influence any internal data of the DMX Control Box**. Data looped from DMX-IN to DMX-OUT are **NOT** modified.

The **actually transmitted level of every DMX channel** is the transmit buffer value multiplied by the masterfader factor, i.e. up to 200%. Due to internal fast integer arithmetics, the transmitted level may be slightly lower than exactly calculated (intermediate fractionals lost). Changes of the parameter are applied immediately, not influenced by FADETIME. The masterfader parameter is not stored in presets.

i **<cycle length>**

Set the length of the chaser cycle (n=2 to 127) and start the chaser

<cycle length> = 0 switches the chaser OFF (new from Dec.2010 / revision no. 90)

Comment: before the chaser can be started, the **step duration** (command >) as well as the **start scene** (command () has to be adjusted – **Details see description of these commands**.

For an easy start, start at scene 128 and step duration 20 (= 2 seconds) is preset as default.

All settings of the chaser are stored in presets 0 to 3 and reactivated when any of these presets is loaded. This applies especially for preset no.0, which is loaded when the DMX Generator is started.

The actual settings of fade time and master fader are taken over by the chaser.

The chaser feature works as follows: a sequence of lighting scenes is loaded in a cyclic manner to DMX channels 1 to 128. To obtain this, the DMX levels which are stored in presets no. 128 to 383 for **DMX channels 385 to 512 are exclusively copied to the DMX transmit buffer channels 1 to 128** and transmitted on the DMX bus (**see commands for partial saving and loading of presets below !**)

Example: if the chaser cycle is set to 4 and the the chaser start is set to 128, then presets no. 128,129,130,131 are loaded partially, then preset no. 128 again and so on. Any other DXM channels 129 to 512 are not influenced by the chaser and may be operated independently.

> <chaser step duration>

Set duration of chaser step in 1/10 s units

Comment: After the duration of any chaser step is over, the chaser automatically loads a part of the next preset in the cycle: **stored DMX levels from DMX channels 385 to 512 are copied to and transmitted at the DMX bus channels 1 to 128.** After <cycle length> presets were loaded in sequence, the procedure is repeated form <chaser start> .

(<chaser start>

Enter start scene (preset no.) of the chaser cycle

Accepted start values: 128 to 219. Default at delivery: 128

If the chaser would access a preset beyond 383, the sequence continues with loading preset no. 128 etc..

Comment: By use of this method, a big number of individual chaser effects may be created and run easily and flexible. This arrangement is optimized for lighting installations, which use a maximum of 384 DMX channels in normal operation and whose chaser effects are concentrated on DMX channels 1 to 128. Experience has shown that this is the case for stage lighting of most music bands and small theaters.

) (no parameter)

Forward the chaser immediately (asynchronous) for one step

< <flash duration>

(new from Dec. 2010 / revision no. 88)

Release a lighting flash: all DMX channels are pulsed to 100% for $t * 1/10s$

Comment: with this command a special preset (=special lighting scene) is transmitted at all 512 channels of the DMX bus during a time period given by <flash duration>. After this timing period, all previous DMX levels are restored.

At delivery the flash lighting scene is prestored in a way that all DMX channels are pushed to 100% level. Used together with installation of complex lamp fixtures, this may result in undesirable complications (start stroboscope effect, for example).

To avoid this, a user prepared lighting scene may be stored permanently in a special memory area with the command ~FF. This lighting scene should be designed in a way to avoid these side effects - or a very individual flash pattern may be created. Note: this memory area may be rewritten up to 10.000 times. We recommend not to create new flash configurations dynamically during runtime.

H <hue>

Sets the spectral color (hue) for a group of 3 subsequent DMX channels (RGB lamp)

Comment: The hue may be entered in the range 0 to 255. This will approximately result in following colors. Intermediate hue values will result in intermediate colors:

H0:red, H43:yellow, H85:green, H128:cyan, H170:blue, H213:magenta, H255:red again.

In correspondence with the model of the driven lamp and setting of saturation and brightness the resulting color tone may differ somewhat.

The H command influences the actually addressed DMX channel (actual entry to the SLOT register, for example set with command S) and the two next higher neighbours. It is provided that the RGB setting of

the respective lamp is done on these 3 successive DMX channels. All features else of a complex lamp ("fixture") may be used independently.

When the H command is applied to a RGB triplet for a first time, previously set (global) values of color saturation (W command) and brightness (" command) are taken over. At the same time this triplet of DMX channels is marked for the RGB mode. From then on hue, saturation and brightness can be set independently, where the command has always to be addressed to the first DMX channel of the triplet. But as soon as a "normal" command to set the DMX level is sent to the first DMX channel of the triplet (especially V, comma ^ , _ , = and ! command) the RGB mode is resolved and can only be reactivated with a new H command.

A previously set fade time also works on a RGB triplet marked with the H command. Apart from possibly the first fade operation the fade process then is not simply performed channel by channel to the new final levels, but the 3 grouped channels are faded through the spectral color space.

Attention: The marker for the RGB feature is only saved in presets no. 0 to 3 as part of the system configuration. If presets are loaded under an incompatible system configuration, this may result in unexpected DMX levels at the corresponding DMX channels

W <saturation>

Sets the color saturation for a group of 3 subsequent DMX channels (RGB lamp).

Comment: The parameter of <saturation> may take values between 0 and 255. The maximum value 255 sets a pure spectral color, at lower parameter values other color components are partially added which results in a pastel light. When the saturation is set to 0, independently of the hue setting a white or grey light is composed.

The parameter entered with this command is applied immediately, if the SLOT register points to a marked RGB triplet. Else it is stored globally and applied together with a primary use of the H command (when it marks a RGB triplets). See comments at command H.

" <brightness>

Sets the resulting brightness for a group of 3 subsequent DMX channels (RGB lamp).

Comment: the <brightness> parameter may take values between 0 and 255. The maximum value 255 sets maximum light intensity, the value 0 switches the light intensity off. Fading down is performed linear, without taking the gamma characteristics of the driven lamp into account. Specially when high performance LEDs are driven most times very strong changes of light intensity are observed at low brightness. So in this range small parameter steps may result heavy changes of the RGB composition. The parameter entered with this command is applied immediately, if the SLOT register points to a marked RGB triplet. Else it is stored globally and applied together with a primary use of the H command (when it marks a RGB triplets).

L <length>

Set the length of the DMX loop.

Parameter: length (range 24 to 512) is the number of user controlled channels to be transmitted per DMX packet

Comment: During **transmission with internal clock**, LOOP sets the number of channels transmitted in every DMX packet. When at later time any DMX channel above this value is addressed or written, the DMX cycle is automatically lengthened. With a new L command the active cycle can be reduced at any time. Due to the regulations of the DMX512 standard the cycle cannot be made shorter than 24 channels.

During transmission with external synchronisation LOOP is automatically fitted to the received DMX signal. In this case, the L command cannot be applied.

During **receive and loop through operation**, the value of LOOP is not important.

Example: L512 sets the length of the transmitted DMX packet to 512

Q (no parameter)

Returns actual settings of all registers of the DMX channel addressed by SLOT. The response is sent as readable SysEx/ASCII text

Example of a typical message:

Mi: CH=1 OUT=13(U) TX=27 MF=50% RX=34 MSG=012/0 CS=128/0/20 L=512 T=3.2

Comment about example:

OUT is the presently transmitted DMX level of the channel addressed by CH= SLOT.
TX describes the content of the transmit buffer of this DMX channel. Both differ when the global masterfader (item MF) differs from 100%. RX is the level actually received at DMX IN. MSG reflects the settings for messages about the level at DMX IN: 1st digit: setting by command 'X', 2nd digit: setting by command 'Y' (global settings). 3rd digit reflects channel the channel specific setting of the MIDI-Format for automatic messages (code sw.0-7,F) or activation (code sw. 8 - C). The value following the slash is the threshold for "automatic"messages. CS describes the actual setting of the chaser in the order: start scene, cycle length, step duration. L shows the number of channels transmitted per DMX cycle andT reports the fade time set by a previous command.

[(no parameter)

Start merge operation with internally generated DMX timing clock:

Comment: The merge operation with internal clock should be used when only or essentially DMX data uploaded by the user are to be transmitted at DMX OUT and only few data received at DMX are to be inserted. Timing of DMX OUT is not synchronous with the received DMX packets. This may result in subtle inconsistencies of received and retransmitted DMX packets - especially at fast faders, chasers and strobes - or if pauses between transmitting slots are considerably longer than two stop bits. The base color of the control LED is green.

Note that received DMX packets with nonzero startbytes are ignored and not fed into the receive buffer. Instead the data of the last received packet with zero start byte ist kept transmitting.

**** (no parameter)

Start merged operation with DMX clock externally synchronized with the signal received at DMX IN:

Comment: The merge operation with internal clock should be used when essentially DMX data received at DMX IN are to be transmitted at DMX OUT and only few data uploaded by the user are to be inserted. If the signal at DMX IN fails, the transmission at DMX OUT is corrupted, too. **So, this method is not useable for exclusive transmit operation when no signal is fed into DMX IN.** But the external synchronisation has the advantage to retransmit received DMX signals without timing distortion and it is able to forward DMX packets with nonzero startbytes. Problems may arise when the received DMX signal has timing parameters at the lowest edge or below the DMX standard. The base color of the control LED is yellow-orange.

Note that no user data are merged into received DMX packets with nonzero startbytes. These are forwarded as received. It is impossible to read data out of such packets, too.

] (no parameter)

Loop received DMX data directly to DMX OUT by hardware.

Comment: In this mode of operation no user programmed data can be transmitted. It is provided to read out data from a signal at DMX IN and no falsification of this signal retransmitted at DM OUT can be tolerated. The base color of the control LED is red.

~ <preset#>

save current content of the transmit buffer preset (=lighting scene) number <preset#>.

Parameter: preset# (range 0 to 383)

Comment: The setting of LOOP and the number base for DMX levels i.e. the "system configuration" – is only saved in presets no. 0 to 3. Thus, in presets no. 0 to 3 preferably different system configurations are stored for quick change. **The parameter value of FADETIME is exclusively stored in preset no. 0** (new from firmware revision no. 73). **Because this preset is automatically loaded when the Control Box is powered on, this way a "soft start" may be configured.**

With all presets else only the actual lighting scene of the transmit buffer is saved, so these may be reloaded universally from different system configurations.

The masterfader parameter is not stored in presets.

Example: ~23 saves the actual state of the DMX Control Box as preset no. 23

Special feature: save transmit buffer partially (new from May 2010 / revision number 83)
an optionally added hex digit (case independent) expands the command:

~A<preset#> stores DMX OUT channels 1-128 to preset channels 129 - 256

~B<preset#> stores DMX OUT channels 1-128 to preset channels 257 - 384

~C<preset#> stores DMX OUT channels 1-128 to preset channels 385 - 512

~D<preset#> exclusively stores DMX OUT channels 1-128 to preset ch. 1 - 128

any other levels in the preset remain unchanged. This feature is intended as a counterpart to the corresponding @ command to produce and test "long" presets with small lamp configurations.

Attention: with this special feature exclusively DMX levels are re-stored, in no case the channel specific details of the system configuration. Especially when use is made of RGB triplets special care is necessary to keep the markings of these triples aligned with the shifted channel numbers.

Special feature: save flash pattern permanently (new from Dec 2010 / revision number 90)

~FF saves the actual lighting scene in a special memory area,

which is loaded temporarily to DMX-channels 1 to 512 with the "flash" command.

Comment: this memory area may be rewritten up to 10.000 times. We recommend not to create new flash configurations dynamically during runtime.

Further details about the flash function see command < above

@ <preset#>

Recalls and activates preset (= lighting scene) number <preset#>

Parameter: preset# (range 0 to 383)

Comment: When one of the presets no 0 to 3 is loaded, the system is reconfigured in correspondence with the stored parameters. When any preset else is loaded, only the content of the transmit buffer is reloaded (lighting scene). At delivery all presets of the DMX Control Box are formatted to load the default state of the device. For details see command "|"

After switching power on or after a reset automatically preset no. 0 is loaded.

When FADETIME is set different from 0, the actual lighting scene is faded over into the loading one with this time constant. The parameter of FADETIME is not changed when a preset is loaded.

Exception (new from firmware revision no. 73): when preset no. 0 is loaded (switching the device on, for example), the value of FADETIME which is stored in this preset is applied.

Example: @34 loads preset no. 34 and makes it active

Special feature: load preset partially

(new from May 2010 / revision number 83)

an optionally added hex digit (case independent) expands the command:

@A<preset#> loads DMX channels 129 - 256 from preset to ch. 1 - 128 DMX OUT

@B<preset#> loads DMX channels 257 - 384 from preset to ch. 1 - 128 DMX OUT

@C<preset#> loads DMX channels 385 - 512 from preset to ch. 1 - 128 DMX OUT

@D<preset#> loads DMX channels 1 - 128 from preset to ch. 1 - 128 DMX OUT

@E<preset#> loads DMX channels 129 - 256 from preset to ch. 129 - 256 DMX OUT

@F<preset#> loads DMX channels 257 - 512 from preset to ch. 257 - 512 DMX OUT

any other DMX OUT levels remain unchanged.

Cases A,B,C are intended to get effectively more presets at small installations

Cases D,E,F are intended to handle multi-room installations more comfortable

Attention: with this special feature exclusively DMX levels are loaded, in no case the channel specific details of the system configuration. Especially when use is made of RGB triplets special care is necessary to keep the markings of these triples aligned with the shifted channel numbers. Refer to p.10

{] (no parameter)

Initiates "download" of the complete nonvolatile memory (all presets) to a PC via RS-232 using the "XMODEM CRC" protocol. Only available at code switch position E !

Comment: This command allows to make a backup of possibly expensively created presets or exchange different sets of presets. Together with presets no 0 to 3 the system configuration stored in these presets is downloaded, too.

To avoid accidental start of this command, a sequence of both command bytes has to be entered.

On the computer which is used for the backup software has to be installed and started which is able to perform the transfer als a sequence of defined data packets using the XMODEM CRC protocol. Applicable transfer software is for example "Hyperterminal", which is combined with a terminal software to perform the normal operation of the MIDI/DMX Control Box, too.

First the command sequence {] has to be entered. Then a message is sent back to the controlling PC that the XMODEM CRC 'Receive' function should be started. Now the user opens a corresponding dialog at the XMODEM software where the name of the backup file is entered. The MIDI/DMX Control Box remains waiting for max 100 seconds while these steps are performed on the PC. When this is finished the PC initiates and controls the download. The download takes about 2 minutes.

The command syntax has changed 17 May 2011 (revision number 93 and higher).

For elder firmware the command is { } (no parameter)

{ [(no parameter)

Initiates "upload" of a previously stored backup to the nonvolatile memory (all presets) from a PC via RS-232 using the "XMODEM CRC" protocol. Only available at switch pos. E !

Comment: On the controlling PC a software has to be started, which is able to transfer the backup as a sequence of defined data packets using the "XMODEM CRC" protocol.

To avoid accidental start of this command, a sequence of both command bytes has to be entered.

It is exclusively possible to upload a backup from a MIDI/DMX Control Box, not from other similar Cinetix products. Under certain conditions after a hardware update (microcontroller module) a backup made with a previous version may be uploaded. Contact Cinetix service to check if this is possible in a specific case.

First the command sequence { [has to be entered. Then a message is sent back to the controlling PC that the XMODEM CRC 'Send' function should be started. Further guide see comment of the complementary download command {].

The command syntax has changed 17 May 2011 (revision number 93 and higher).

For elder firmware the command is { { (no parameter)

| (no parameter)

"clear all memory": all buffers and modes of operation are reset to default

Action memory: the merger transmits exclusively from the transmit buffer.
All automatic messages are switched OFF.

Transmit buffer: All DMX levels of the transmit buffer are reset to "0" gesetzt.
Number base = "decimal". Masterfader = 100% The Chaser is switched OFF.
LOOP = 512, FADETIME = 0.0. Presets are not deleted or changed otherwise.

' (apostrophe, no parameter)

returns the letter "R", followed by a version number of two decimal places

Comment: Special feature to operate the MIDI/DMX Control Box with the software "DMX Control" when the rotary switch at the front panel is in position 'E' and a special MIDI to RS-232 interface is connected. . Will be recognized as "RS-232/DMX Control Box" then. Not used during normal operation.

} <pulse length> } <pulse length>

Special command to modify the duration of the DMX reset pulse

Comment: This command is intended to solve extremely seldom problem cases and **should not be used without recommendation of our service department.**

After the brace a byte is entered which is composed of two hex nibbles 0 .. F, each as a text character.

The same command must be repeated immediately after with the same byte parameter, no space nor new line are allowed to be put between. The parameter is stored permanently independent of presets and is reloaded during every system start.

1st hex digit (high nibble): Duration of the DMX reset pulse: ca. 90us base value + (nibble value * 8 us). This way, the range of the DMX reset pulse may be varied between about 90us and 210 us.

2nd hex digit (low nibble): Duration of the mark pulse between DMX reset and start byte: ca. 10us + nibble value. So, the setup range is about 10us to 25us.

Exception: The command }00}00 resets the DMX timing permanently to its default values.

3rd part of the manual "MIDI/DMX Control Box": special modes of operation

Generation of MIDI messages by signal change at DMX IN

Formatting and extraction of the MIDI messages is possible under different criteria:

1.) MIDI- or SysEx/ASCII Messages in **the same format as the corresponding control command** (only available when code switch is at positions 0 to 7 and F).

This format is best useable for **recording on a MIDI sequencer** for later reproduction of the received lighting scenes via MIDI OUT of the DMX Control Box. A characteristic feature of this message type is the **freely adjustable trigger threshold**, by means of which the resulting quantity of data may be fitted to the respective task. In state of delivery all messages of this kind are deactivated and can be activated channel by channel or globally. This type of automatic messages is specified here as **"recording"** to differentiate them from the message formats sent at code switch positions 8 to C:

2.) MIDI messages specially **formatted for control of other MIDI applications** by means of a DMX signal (only available when code switch is at positions 8 to C).

Example: control of an audio mixer as a subsidiary job of the lighting control.

Care was taken to offer **generation of a broad spectrum of MIDI messages**. In state of delivery all available messages (don't occupy all DMX channels) are activated. They can be deactivated channel by channel or globally.

In models with revision no. < 66 this feature could only be used as DMX receiver. Newer versions allow to use the Control Box simultaneously as DMX transmitter and send DMX levels to DMX OUT controlled by MIDI or merged /multiplexed with the received signal from DMX IN.

Both types of these messages are called **"automatic"** in this manual, in contrast to the **"preprogrammed"** ones released by input on DMX channels 500 to 512 and the **"spontaneous"** which are released by special signals on DMX channels 458 to 499. These are described below.

3.) A limited number of freely **"preprogrammed" strings** can be stored permanently in the MIDI / DMX Control Box and be sent via MIDI OUT released by special DMX level changes on DMX channels 500 to 512. These may be System Exclusive messages for example or such ones which are fully out of the MIDI context.

These strings can only be **programmed with the ASCII command ":"** (colon) and checked with the command **";"** (semicolon).

Details see **description of these commands in part2** of the manual. This can be performed with MIDI baudrate and commands embedded in SysEx frames or - at most times more comfortable - with the code switch in position E with normal ASCII text, 9600 Baud. An adaptor for level shifting RS-232 / MIDI can be made easily DIY or supplied optionally.

4.) Another freely usable feature exploits **"spontaneous"** special level values at DMX channels 458 to 499: **A level change at DMX channel 458** (trigger edge from 0 to 100%) **causes emission of exactly the actual levels of DMX channels 460 to 499 as binary string** via MIDI. The count of emitted bytes (1 to max. 40) is determined by the level at DMX channel 459 ("counted string").

This way arbitrary and an unlimited number of strings can be sent "spontaneously" at runtime via MIDI OUT. Because this option practically can be only used when driven by a precise digitally controlled DMX signal, it is deactivated normally. Then DMX channels 458 to 499 can be used without restriction for "normal" lighting control. Even when the effect is activated, DMX channels 459 to 499 can be used normally, as long as there is no trigger edge (0 to 100%) at channel 458.

Following restriction has to be remarked: a MIDI data line has a **relativ low data capacity** (about max. 1000 MIDI messages per second). With the tools described here, easily more

MIDI data can be generated. **So it is highly recommended, to concentrate the extraction of MIDI data on as few as possible representative DMX channels.**

Activation and format of "automatic" messages

Activate "recording" messages for DMX channel ="SLOT" with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as code switch, **pay attention to comment**

1st data byte: select MIDI format of "recording" messages

This selection is only active when the code switch is in one of the positions 0 to 7 or F. Setting options at code switch positions 8 to C see below.

Following **options are available:**

1st data byte = **48** (hex 30) send a mix of NOTE ON and CONTROL CHANGE messages (default) This format is identical to the one described above under "quick start".

EVEN DMX levels are sent in 7 bit format as NOTE ON messages

ODD DMX levels are sent in 7 bit format as CONTROL CHANGE messages

1st data byte = **49** (hex 31) send a CONTROL CHANGE message

The DMX level is reported in 7 bit format.

Optimum for quick and easy post-editable recording.

1st data byte = **50** (hex 32) send a NOTE ON message

The DMX level is reported in 7 bit format. **When suppression of NOTE OFF is activated (PROGRAM CHANGE / data byte = 121,) the 2nd data byte (velocity) will be set to 1, if the DMX level is equal to 0 or 1.**

1st data byte = **51** (hex 33) send a pair of messages contained of

POLY KEY PRESSURE and PITCH CHANGE message

The modified DMX channel is transferred as a POLY KEY PRESSURE message, the corresponding DMX level is transferred with 8 bits of resolution as a PITCH CHANGE message. Both are sent in the same format as the command described above for transmission of DMX levels via DMX OUT. This data format is most useful, when only one MIDI channel is available for the DMX Control Box.

2nd data byte: desired **threshold (0 to 127)**

threshold = 0 causes deactivation of this message

Comment: This configuration can be made at all code switch positions except D and E. **But the messages activated this way are executed exclusively when the code switch is in one of the positions 0 to 7 or F.** About transmission of automatic messages at code switch positions 8 to C see below.

If the activation of these messages was stored in one of the presets 0 to 3, they are reactivated when the preset is loaded and sent on the MIDI channel which is presently selected with the code switch (if applicable: on additional MIDI channels, too), independent of which MIDI channel was selected when the preset was saved. But if MIDI channel 15 or 16 is selected while the preset is reloaded, all messages are suppressed which would have been sent on MIDI channels >16.

Activate "recording" messages for all DMX channels with:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as code switch, **see comment about POLY KEY PRESSURE DB1=48-51**

Same activation and annotations as described above for an individual DMX channel **but the**

1st data byte is 58,59,60,61 (hex 3A-3D) to activate the message types described above under 48,49,50,51 equally for all DMX channels.

Activation of "general purpose" and "VJ" MIDI messages for DMX channel = "SLOT" with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as code switch, **pay attention to comment**

1st data byte always = 52 (hex 34)

2nd data byte = 0: **deactivate** messages on this DMXchannel

= 1: **activate** messages on this DMXchannel

Comment: This configuration can be made at all code switch positions except D and E. **But the messages activated this way are executed exclusively when the code switch is in one of the positions 8 to C.** About transmission of automatic messages at code switch positions 0 to 7 and F see above.

By default, all messages of this type are activated, they can be **deactivated** individually per DMX channel or globally. If the activation of these messages was stored in one of the presets 0 to 3, they are reactivated when the preset is loaded. **Independent of the code switch position, always that MIDI channel is inserted into these messages, which has been selected with the POLY KEY PRESSURE command** (1st data byte=88) described below (default= MIDI channel 1).

Activation of "general purpose" and "VJ" MIDI messages for all DMX channels with :

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as code switch, **but see comment about POLY KEY PRESSURE DB1=52**

1st data byte always = 62 (hex 3E)

2nd data byte = 0: **deactivate** messages for all DMXchannels

= 1: **activate** messages for all DMXchannels

Select MIDI channel for "general purpose" and "VJ" MIDI messages with:

POLY KEY PRESSURE (=POLYPHONIC AFTERTOUCHE)

MIDI channel as position of code switch

1st data byte = 88 (hex58)

2nd data byte = 1-16 MIDI channel to be inserted into automatically generated messages (default = 1)

Comment: with this command the MIDI channel has to be inserted in the range 1 to 16 (hex10), like it is used to be counted in manuals or in literature. In the status bytes of the MIDI messages generated this way it is inserted one less 0-15 (hex F) - like common use with MIDI.

This command is only relevant together with code switch positions 8 to C.

In automatic messages, which are generated at code switch positions 0 to 7 or F, always the same MIDI channel is inserted as selected with the code switch. In automatic messages which refer to DMX channels >127, in correspondence with the selected MIDI format a higher MIDI channel may be inserted.

This setting is stored together with presets 0 to 3 and reactivated when one of these presets is loaded.

Switch "automatic" messages globally ON and OFF with:

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 40 (hex28) switch automatic messages globally **OFF**

data byte = 41 (hex29)

switch automatic messages globally **ON**,

format **MIDI channel messages (default)**

the automatically sent MIDI messages are formatted corresponding to the last made configuration with POLY KEY PRESSURE

DB1= 48, 49, 50, 51, 52 or 58, 59, 60, 61, 62 details see above

data byte = 42 (hex2A) switch automatic messages globally **ON**,

message format **SysEx/ASCII**

When globally switched off, all automatic messages are kept stored individually per DMX channel. Only their output is suppressed. This setting is stored in any of the presets no 0 to 3. When switched on again, the previous configuration including threshold levels is reactivated - but the message format may differ.

Switch emission of "preprogrammed" and "spontaneous" strings globally ON and OFF with:

PROGRAM CHANGE

MIDI channel as position of code switch

data byte = 50 (hex32) switch emission of strings globally **OFF**

= 51 (hex33) switch emission of strings (DMX channel 500-512)
globally **ON**

= 52 (hex34) switch emission of strings (DMX channel 458-499)
globally **ON**

= 53 (hex35) switch emission of strings (DMX channel 458-512)
globally **ON**

Comment: The commands 51 and 52, which activate only a subset, implicitly deactivate emission of the other string types.

When globally switched off, all preprogrammed strings which are released by changes of DMX channels 500-512 are kept stored. Only their emission due to special levels at DMX IN is suppressed until they are activated again.

Important safety information:

Data transmission via DMX512 is technically regarded to be "unreliable". For this reason **it is STRICTLY FORBIDDEN to use the technique described here together with all safety critical applications, where malfunction could result in personal injury or noticeable material damage !**

Control of VJ-software using DMX512

This section is only relevant at code switch positions 8 and 9.

The range of DMX channels which are able to release MIDI messages is selected with the code switch. **The inserted MIDI channel is selected freely with PROGRAM CHANGE command 88**, default is channel 1

code switch position	DMX channel	generates NOTE ON NOTE OFF switch	DMX channel	generates CONTROL CHANGE (DMX-Val/2)	DMX ch. generates PROG-CH (DMX-Val/2)	DMX ch generates CH-PRESS (DMX-Val/2)	DMX channel generates PITCH CHANGE MSB=DMX-Val/2
8	1 - 127	1 - 127	128 - 159	0 - 31	160	160	162
9	256 - 383	0 - 127	384 - 415	0 - 31	416	417	418

As soon as one of these modes of operation is selected, the present DMX levels are analyzed and stored als initial registry. Therefore the messages described above are sent on the basis of the DMX bus levels found at start. No MIDI messages are sent before the DMX input changes significantly. If no DMX signal is present at the beginning, for all DMX channels the level zero respectively OFF is registered. In exceptional cases noise at the DMX bus may cause registration of nonzero levels.

The **first 128 bytes** (offset 0 to 127 with respect to the DMX base channel) are evaluated as follows:

If the data level of any DMX channel within this block exceeds the value 23, a MIDI NOTE ON message with velocity (2nd data byte) = 64 (hex40) is sent at MIDI OUT -

The first data byte (which represents the note pitch) reflects the relative offset of the changed DMX channel within the selected block. The second data byte is always coded with standard velocity = 64. This NOTE ON message is sent only once, if the state was NOTE OFF before and the level exceeds 23 rising from lower values.

If the data level of any DMX channel within this block runs below the value 20, a MIDI NOTE OFF message with velocity (2nd data byte) = 0 is sent at MIDI OUT

The first data byte (which represents the note pitch) reflects the relative offset of the changed DMX channel within the selected block. The second data byte is always coded with zero velocity.

This NOTE OFF message is sent only once, if the state was NOTE ON before and the level sinks below 20 descending from higher values.

The **hysteresis** decribed above serves to avoid permanent transmission of insignificant data due to a noisy DMX signal, which may easily be generated, when an analog potentiometer is positionned between two digital values. This trigger behaviour was chosen to get a clear effect even with simple analog lighting consoles: NOTE ON with velocity=64 is released by slightly pulling up the potentiometer and velocity=0 is released by pulling it completely down without special care to be absolutely at zero.

The **next 32 bytes** of the selected DMX range (offset 128 to 159 with respect to the DMX base slot) are evaluated as follows:

Every signifant change of the DMX signal releases a CONTROL CHANGE message,

The **controller number (first MIDI data byte)** is equal to the relative offset of the changed DMX channel within the selected block, how to calculate see table above. At code switch position "8" for example DMX channel no 130 would release a CONTROL CHANGE message for controller no.2.

The **controller value (second data byte)** is equal to the DMX level actually received at this DMX channel divided by 2 (i.e. the 8 bit data range of DMX projected onto the 7 bit data range of MIDI).

If the DMX-Byte with relative **offset of 160** with respect to the DMX base channel changes significantly a **PROGRAM CHANGE message is released at MIDI OUT.**

The **program number** (only one data byte) is equal to the DMX level actually received at this DMX channel divided by 2 (i.e. the 8 bit data range of DMX projected onto the 7 bit data range of MIDI).

If the DMX-Byte with relative **offset of 161** with respect to the DMX base channel changes significantly a **CHANNEL PRESSURE message is released at MIDI OUT.**

The **pressure value** (only one data byte) is equal to the DMX level actually received at this DMX channel divided by 2 (i.e. the 8 bit data range of DMX projected onto the 7 bit data range of MIDI).

If the DMX-Byte with relative **offset of 162** with respect to the DMX base channel changes significantly a **PITCHWHEEL CHANGE message is released at MIDI OUT**.

The **first data byte** (LSB of the pitchwheel setting) is always transmitted as zero.

The **2nd data byte** (MSB of the pitchwheel setting) is equal to the DMX level actually received at this DMX channel divided by 2 (i.e. the 8 bit data range of DMX projected onto the 7 bit data range of MIDI).

General Purpose extraction of MIDI messages from DMX512

This section is only relevant when the code switch is in position A,B or C.

DMX channel	generates POLY KEY PRESSURE (DMX/2)	DMX channel	generates NOTE ON (DMX/2)	DM channel	generates CONTROL CHANGE (DMX/2)	DMXchannel generates PROG-CH (DMX/2)	DMXchannel generates CH-PRESS (DMX/2)	DMXchannel generates PITCH CH MSB=DMX/2
1 - 127	1 - 127	128 - 255	1 - 127	256 - 383	0 - 127	384	385	386

When the code switch is in position A, no NOTE ON and no POLY KEY PRESSURE messages are generated, i.e. extraction of MIDI messages takes only place when DMX channels 256 to 386 are changed.

When the code switch is in position B, no POLY KEY PRESSURE messages are generated, i.e. extraction of MIDI messages takes only place when DMX channels 128 to 386 are changed.

When the code switch is in position C all messages listed in the table are generated, i.e. extraction of MIDI messages takes place when DMX channels 1 to 386 are changed.

All MIDI channels else are unrestricted available for "normal" lighting control. But at code switch positions 8 to C no extraction of automatic messages in "recording" format is possible.

Every significant change of the signal at DMX IN releases a single MIDI message of one of the types described below. **All MIDI messages have that MIDI channel inserted, which is selected with PROGRAM CHANGE command 88 (default 1).**

POLY KEY PRESSURE messages generated from DMX channels 1 to 127

The **note pitch (first MIDI data byte)** is equal to the actual DMX channel.

The **velocity (2nd data byte)** is equal to the actually received level at this DMX channel divided by 2

NOTE ON messages generated from DMX channels 128 to 255

The **note pitch (first MIDI data byte)** is equal to the actual DMX channel **minus 128**.

The **velocity (2nd data byte)** is equal to the actually received level at this DMX channel divided by 2

If these NOTE ON messages are used for control of equipment like MIDI synthesizers, it has to be taken into account that another NOTE ON message with velocity=0 has to be sent to finish the sound of this note.

CONTROL CHANGE messages generated from DMX channels 256 to 383

The **controller number (first MIDI data byte)** is equal to the actual DMX channel **minus 256**.

The **controller value (2nd data byte)** is equal to the actually received level at this DMX channel divided by 2

The **PROGRAM CHANGE** and the **CHANNEL PRESSURE** message is evaluated as follows:

The **programm number** respectively the **pressure value** (only one data byte) is equal to the DMX level actually received at this DMX channel divided by 2.

The **PITCHWHEEL CHANGE** message is evaluated as follows:

The **first data byte** (LSB of the pitchwheel setting) is always transmitted as zero.

The **2nd data byte** (MSB of the pitchwheel setting) is equal to the DMX level actually received at this DMX channel divided by 2.

Transmission of MIDI signals via DMX cables over long distances up to 1000 m

This feature is only available when the code switch is in position D.

In this mode of operation, all data received at MIDI IN are retransmitted unmodified 1:1 with MIDI baud rate (31250 bit/second) from the DMX OUT connector. Simultaneously all signals received with MIDI baudrate at DMX IN are retransmitted unmodified from MIDI OUT.

Data transfer at DMX IN and DMX OUT is according to the RS-485 format like DMX (more exactly according to RS-422 format, but which is identical in this kind of application). **This way 2 MIDI/DMX Control Boxes make a secure bidirectional extension of MIDI-connections.** The use of standard DMX connectors and cables can be an advantage in stage building.

Data transfer is signalled with a dual colour LED: In idle state it is shining yellow orange. During data flow from MIDI IN to DMX OUT the red part of the LED is off, i.e. it is shining green. During data flow from DMX IN to MIDI OUT the green part of the LED is off, i.e. it is shining red. While both directions are busy, the LED is off.

Appendix A:

Concept and data format of DMX512 transmission:

To understand the function of the MIDI / DMX Control Box and its commands, you should roughly know the concept of internal memories for received and transmitted DMX signals.

For each of the 512 DMX channels three specific memory cells are provided: transmit buffer, receive buffer and action memory.

The received DMX signal is permanently written in realtime into the receive buffer.

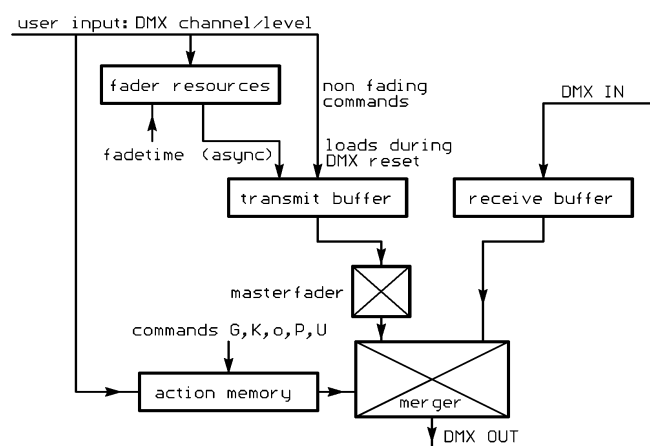
DMX level data entered by the user are preprocessed if necessary (e.g. a fade process initiated) and then written into the transmit buffer.

The expression "transmit buffer" does not mean that its content is transmitted unconditionally:

Before transmission of any next DMX byte the merger routine looks into the action memory if this byte shall be taken from the receive buffer or from the transmit buffer.

The **masterfader** modulates the DMX data bytes as a simple signal processor when they are transferred from the **transmit buffer** into the DMX transmitter hardware. This way a complete lighting scene may be "stretched" globally in a simple way.

Internally stored DMX data are not manipulated by the masterfader. The masterfader does not influence any data which are retransmitted from the receive buffer.



The control philosophy of the MIDI / DMX Generator is based on the cooperation of three registers:

- **SLOT register** (written with the first data byte of a NOTE ON or CONTROL CHANGE message or with a POLY KEY PRESSURE message).
The **SLOT register** stores the address (= physical time slot 1 to 512) within the DMX packet, to which write and read commands are currently applied.
- **DMX level register** (written with the second data byte of a NOTE ON or CONTROL CHANGE message or with a PITCH CHANGE message).
- **FADETIME Register** (written with a POLY KEY PRESSURE message).

The **SLOT register** stores the actually addressed DMX channel (= physical time slot 1 to 512 within the DMX cycle), which shall be written or read subsequently by MIDI channel message or SysEx commands.

For any DMX channel a complete fade process can be performed. The fadetime to be used is set in the **FADETIME register**. This fade process is completely controlled by the firmware of the device. Fade jobs are automatically assigned to internal resources. The actual value of FADETIME is copied into the respective fade process when the fade process is started. Straight after then FADETIME can be changed for the next command without influence on running fade processes.

Last not least, a **LOOP register** is implemented, which stores the actual actual number of DMX channels to be transmitted under user control in each DMX packet.

Every command causes its parameter to be written into the corresponding one of the registers listed above. Depending on the type of command, when all information is entered the intended action is started using the actual content of ALL OF THESE REGISTERS. For this reason, the order of entering data may influence the result. In general **the order FADETIME, SLOT, DMX level should be followed**, but most times not all of the registers have to be updated.

After power-up or reset, SLOT is initialized with "1", DMX levels and FADETIME are set to "0", the MASTERFADER is initialized with 100% and LOOP is started with "512" or value read from preset no.0.

Entries to the action memory - which controls the merge behaviour - are controlled with PROGRAM CHANGE 0,1,2,3,4 or with the equivalent SyEx/ASCII commands G,K,o,P,U.

Appendix B:

Data format DMX512

The "DMX512" standard assumes that a bus master transmits DMX data packets on a DMX data bus in a permanent cyclic repetition. This bus line is looped from the transmitter to the next receiver and from there to the next receiver and so on in a linear bus topology. **Every DMX receiver connected at the bus is permanently supplied with actual control data for all receivers**, no matter if these data have been changed in the meantime.

Every **DMX packet transmission cycle starts with a special reset sequence and a start byte** as header. Subsequently all data bytes are transmitted in 8 bit serial format to all potential receivers. This way every transmitted DMX byte is assigned with a specific time slot within every DMX packet and it can be addressed with the number of this time slot. Therefore the "addresses" or "channels" in a DMX data packet are called "slots". Per DMX slot one data byte is transmitted. The value of this data byte (range 0 to 255) describes the intensity of a lamp or the position of a "moving head". **Throughout this manual we use the expression "DMX level" or "DMX value" to describe this byte value.**

Addressing of receivers is defined by the time position ("slot") of their data bytes within the cyclic DMX package. By means of a code switch or something similar a number between 1 and 512 can be selected at every DMX receiver. The receiver scans the count of slots during every DMX packet and starts to read out the slot bytes starting from the slot number selected with its switch. The number of evaluated bytes depends on the specific function of the receiver. If, for instance, the switch of a certain receiver is set to 28, this receiver will read the 28th, 29th, 30th ... byte following after the start byte out of the DMX data stream during every DMX cycle. These data

are not removed out of the DMX data stream. But all receivers with a different setting of their code switch do ignore them. Depending on the setting of their code switches, several receivers can get the same data - intentionally or by mistake.

Due to its design the "DMX512" standard does not offer inherent error checking and error correction of transferred data and does not supply a usefully standardized feedback channel from receiver to transmitter. While slowly changing actuators like bulb lamps or servo motors are driven, single faulty bytes almost have no effect on the visual behaviour, because they are corrected in the next cyclic transmitted DMX packet with high probability. **Very critical however is to release singular switching events via the DMX-bus.**

For these reasons the the use of DMX control is explicitly FORBIDDEN together with all safety critical applications, where malfunction could result in personal injury oder noticeable material damage !

To install a DMX512 bus, preferably a special 2 wire twisted and shielded "RS-485" data cable is recommended, which unfortunately is quite expensive and only available from special providers for industrial control. CAT5 ethernet cable has been proved to be a good material for DMX installation, too. For solid installations we have made good experience with shielded ISDN cable, which is traded in germany as type JY(St)Y. The version with 0,8mm wire diameter should be used if available. Using bus lengths up to 100 m, a high quality 2 wire microphone cable will do.

Long bus lines have to be terminated at the most distant receiver. Termination means: DMX+ and DMX- are connected with a resistor of 120 Ohm.

Normally the bus line is "looped through" from receiver to receiver. Most DMX equipment is supplied with a DMX IN and a DMX OUT connector, both are physically wired together inside. For this kind of installation you should have available a 5-pin male XLR-connector prepared with a built-in resistor, which is plugged into the DMX OUT of the last DMX receiver in the chain.

Inside the MIDI / DMX Control Box the DMX IN signals are electronically refreshed before retransmission. This means, even when the DMX data are "looped through", the **MIDI / DMX Control Box has to be regarded as terminal device at the bus line. The termination resistor is already built in, no external termination is necessary!** If in special cases this termination is not wanted, a jumper is provided inside the box (close to the DMX IN socket) which has to be pulled then. To open the box: remove 4 screws and pull off the upper part of the metal cabinet.

Appendix C

Structure of MIDI channel messages (super compact crash course)

The first byte is the **status byte, only this one has bit 7 set.**

It is composed of 2 hex nibbles. The most significant (high) nibble describes the message type, the least significant (low) nibble contains the MIDI channel. The MIDI channel is always physically coded one less than written in any MIDI manual or literature. Putting both together: (MIDI channel-1) has to be added to the message type status body.

For MIDI channel no. 1 the different types of MIDI messages show following status byte (=status body):

NOTE OFF:hex80,dec128

NOTE ON:hex90,dec144

POLY KEY PRESSURE(=POLY KEY AFTERTOUCH):hexA0,dec160

CONTROL CHANGE:hexB0,dec176

PROGRAM CHANGE:hexC0,dec192

CHANNEL PRESSURE:hexD0,dec208

PITCH CHANGE:hexE0,dec224

The status byte of PROGRAM CHANGE and CHANNEL PRESSURE is followed by exactly 1 data byte

The status byte of all other message types is followed by exactly 2 data bytes

In all data bytes the most significant **bit7 is cleared**, the other bits are content specific.

Determination of MIDI note values from common note names:

	C	C#	D	D#	E	F	F#	G	G#	A	B	H
-2	0	1	2	3	4	5	6	7	8	9	10	11
-1	12	13	14	15	16	17	18	19	20	21	22	23
	24	25	26	27	28	29	30	31	32	33	34	35
1	36	37	38	39	40	41	42	43	44	45	46	47
2	48	49	50	51	52	53	54	55	56	57	58	59
3	60	61	62	63	64	65	66	67	68	69	70	71
4	72	73	74	75	76	77	78	79	80	81	82	83
5	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107
7	108	109	110	111	112	113	114	115	116	117	118	119
8	120	121	122	123	124	125	126	127	-	-	-	-

This scheme is used by several MIDI sequencers. But deviating names of the different octaves are in use, too. Following unique relationship may be helpful: the concert pitch a (440Hz) -- in the table above "A3" - is always corresponding with MIDI note value 69 (hex 45).



Cinetix Medien und Interface GmbH
 Gemuendenerstr. 27 D-60599 Frankfurt Germany
 Phone: +49-69-68 51 05 Fax +49-69-68 600 409
<http://www.cinetix.de/interface/english/>

* Right of technical modifications reserved.

* This description is for information only, no product specifications are assured in juridical sense.

* Trademarks and product names cited in this text are property of their respective owners.