STM32G431 based DMX controller with USB Interface

 25 January 2025

This is a DIY construction manual for a simple and compact DMX512 controller based on the **STM32G431KB (or STM32G441KB)** microcontroller (32 pin LQFP).

DMX commands and power supply are provided via USB. **It is seen by a host PC as a MIDI interface or as a virtual COM port**. So DMX control is possible by MIDI messages as well as by ASCII text based commands. The USB connection is "full speed USB2.0" grade.

Furthermore, a special version of the **CForth interpreter** is provided for this hardware

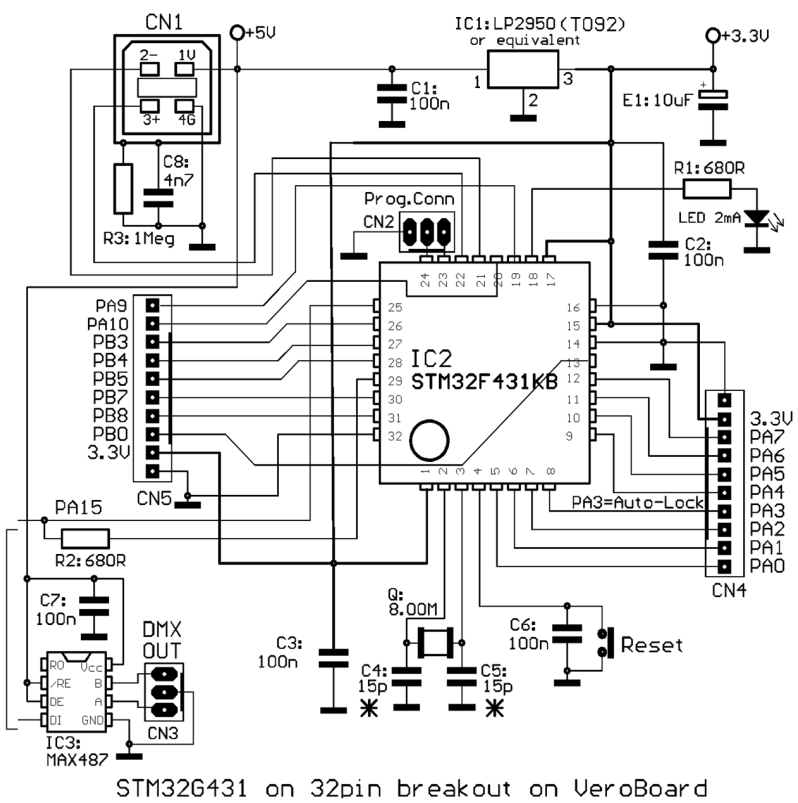
It is **NOT allowed to use this device together with any safety critical applications**, where malfunction could result in personal injury oder noticeable material damage !
All information about this project is provided 'as is' – without any warranty or responsibility

Hardware

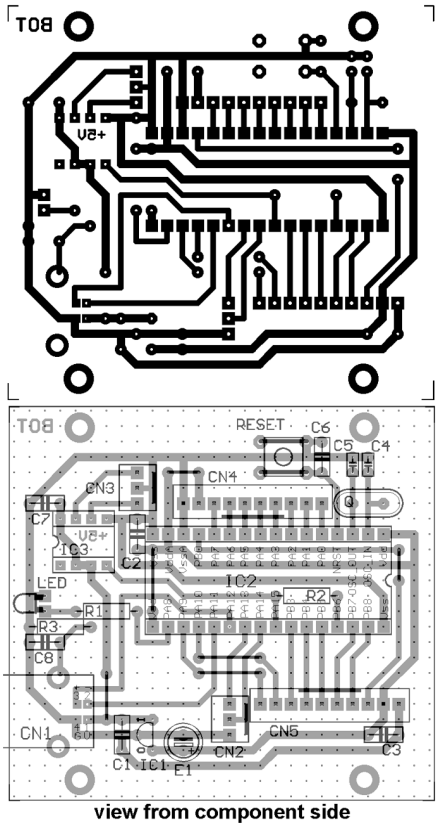
Because it is not possible to re-start the STM32 microcontroller USART immediately after the DMX reset pulse, the USART Tx is kept running in idle state during DMX reset. It is fed via a resistor to the DMX driver MAX487, which is pulled low by another Open Drain output of the microcontroller during DMX reset. Dirty solution, but it works.
The total supply current without DMX load is ca 30 mA, with terminated DMX bus up to 60 mA.

For easy reproduction with simple tools, **Veroboard compatible layout is proposed**. It can be built without etching a PCB with a small number of jumper wires, but a TIF file to etch a 1 layer PCB is provided.
The board is designed to fit into a "Bopla Unimas U85" cabinet. With slight deplacement of mounting holes, it does fit into "EUROBOX" case (Reichelt) = "Donau Elektronik Universal Gehäuse" 523132(Conrad)

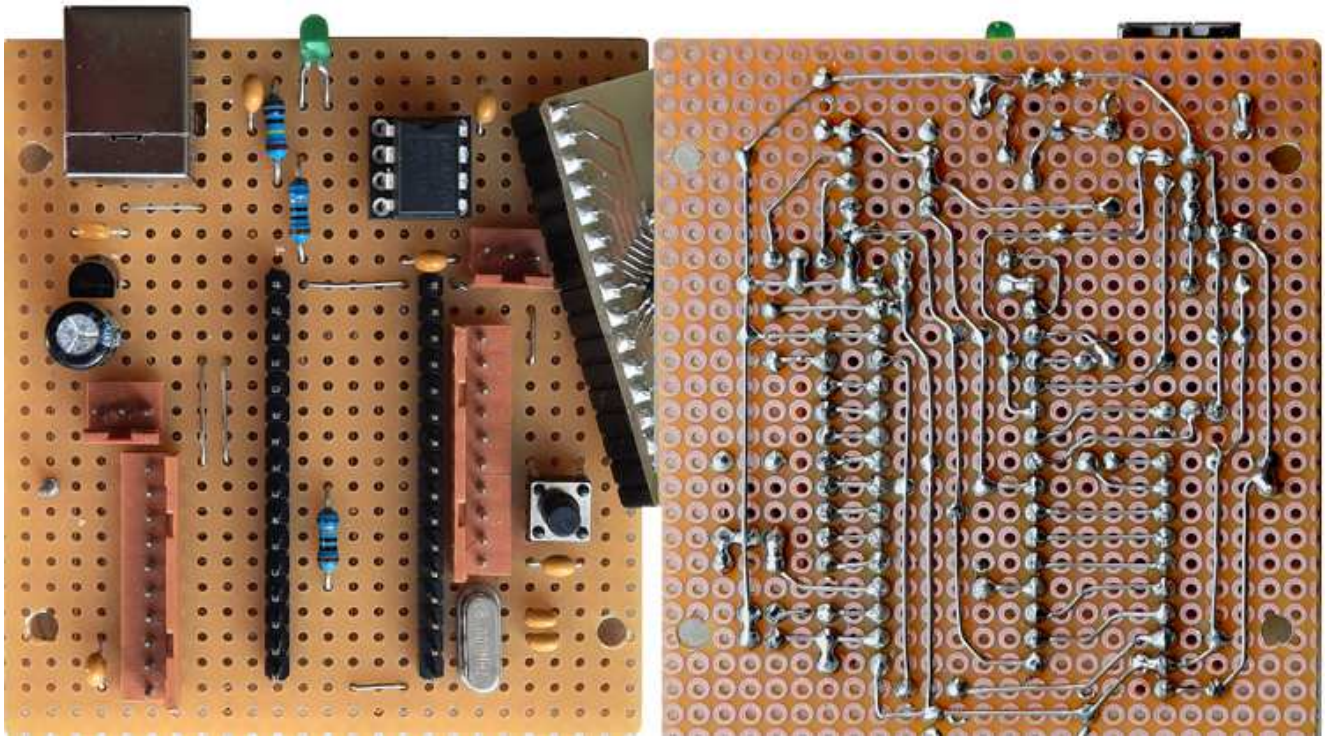
Schematic diagram and:



STM32G431 on 32pin breakout on VeroBoard



view from component side



The I/O ports connected with CN4 and CN5 are exclusively supported by CForth firmware, not by the ASCII and MIDI command sets described below in this manual
The PCB layout and assembly drawing is shown from the **component side**!

Special parts:

IC1: 3.3 V regulator TO92 case:preferably LP2950ACZ3.3, source Reichelt and others - or LE33
IC2: STM32G431KB or STM32G441KB , source: Reichelt, tme.eu, Mouser
IC3: MAX487 CPA, source Reichelt, Mouser
Connectors CN2, CN3: source Reichelt PS 25/3G BR, Conrad 741221
Connectors CN4, CN5: source Reichelt PS25/10G BR, Conrad 741264
LED: 3mm, low current (specified for 2mA. For blue or white LED: R1=10kOhm), source Reichelt and others

Programming:

Because no RS-232 port is available on these boards, programming is possible only with an SWD programmer.

Instead of buying a dedicated ST-LINK module it is recommended to get a STM Nucleo-64 module (best choice: STM32L476 or STM32F446), which is less expensive, can be configured as programmer and additionally be used for other experiments with STM32 microcontrollers. How to program external parts with the Nucleo, read its manual. When the programmer is cut off and the programming software sends a problem like "no target voltage", connect the 3.3V output (pin next to text U1) of the regulator (5 pins) on the ST-Link part with R23 (4.7 kOhm, pad directed towards the SWD connector) to pretend a supply voltage of the programmed device.

Build a short programming adaptor (see picture above, wire length max 30 cm)**:**

--- connect the 2nd pin of the Nucleo SWD connector (counted from the side towards the Mini USB connector) with pin PA14 of our board.
--- connect the 3rd pin with Ground of our board.
--- connect the 4th pin with pin PA13 of our board.
These pins are broken out at connector CN2. Connection of Reset is not necessary.
--- **start the ST-LINK software**. Connect the Reset pin with Ground. Click item "Connect" of the "Target" menue, after 1 second release Reset. Then a screen with the connection report and a listing should appear. (If the microcontroller has been used with other projects, first perform "Chip Erase".) When you are connected, select "Program&Verify" from the "Target"

menue and upload the hex code. After programming, click item "Disconnect" from the "Target" menue (or it gets disconnected autmatically) and remove the ST-Link adaptor.

By default the USB interface works with the STM Vid/Pid, **but this is exclusively allowed for test and evaluation! For any public use, your individual Vid/Pid must be activated !** (how to do this see ASCII command 'U' page 6)

MIDI and ASCII text modes of operation and commands are described starting from next page.

Microcontroller I/O ports, which are connected to CN4 and CN5, are not supported by these command sets.

More flexible operation combining DMX with use of microcontroller ports (CN4 and CN5) is provided with the **CForth interpreter** firmware (www.midi-and-more.de/more/cforth.htm). Simple time sequenced and/or interactive lighting shows can be programmed this way, which specifically may be used for media installations and objects.

Modes of Operation:

ASCII command set

is active when **PA14 (SWD-Clk pin) is NOT connected with Ground** (no jumper or switch). The actual firmware revision 2.5 (September 2021) provides some new features like a simple Chaser and simplified setting of RGB lamps.

Short reference of all ASCII commands

Detailled description of all ASCII commands:

Every control command and every state message is assigned with a single characteristic letter. If a command expects a parameter, it is listed after the command letter in acute angular brackets <..:>. **Number values are always in decimal format and are sent via USB as ASCII text.**
This compact format is suitable to enter commands manually as well as for automatic generation and parsing in an application software.

Address the DMX channel to be manipulated with following commands:

S <DMX channel number>

The **parameter addresses a DMX channel**, on which subsequently described commands have an effect.

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

Parameter: DMX address (range 1 to 512) is the number of the DMX channel (slot) to be manipulated with subsequent commands. No action is started immediately.

Transmit buffer manipulation:

V <level>

Write parameter into the transmit buffer of the actually adressed DMX channel.

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

Example: V34 sets the DMX level to 34 at the DMX channel which is actually addressed

---

, (comma) **<level>**

First this command increases the actually addressed DMX channek automatically, then it writes the parameter into the transmit buffer for the new DMX channel'.

Parameter: level (range 0 to 255) is the value or intensity which will be immediately transmitted at the incremented and now actual DMX channel

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

+ (no parameter)

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

Comment: The level cannot be made greater than decimal 255. If it is already equal to 255, the + command is ignored. If a fade process is active at this DMX channel, only the final value is increased.

- (minus, no parameter)

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

Comment: The level cannot be made less than 0. If it is already zero, the - command is ignored.
If a fade process is active at this DMX channel, only the final value is decreased.

---

^ <increment>

Add <increment> to the transmit buffer addressed by SLOT (and start a fade process)

Comment: The level cannot be made greater than decimal 255 If it is already equal to 255, the command is ignored. If FadeTime != 0, only the final value is increased and a new Fade Process is started.

_ <decrement>

subtract <decrement> from the transmit buffer addressed by SLOT (and start a fade process)

Comment: The level cannot be made less than 0. If it is already zero, the command is ignored.
If FadeTime != 0, only the final value is decreased and a new Fade Process is started.

= <block length>

This command writes the final level of the addressed DMX channel into the number of <block length> DMX channels starting from (addressed channel+1). A new fade to this final level is started from the actual level of each of these channels. 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 <block length>, channel 512 is never exceeded

---

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.

Every new setting of RGB-hue, color saturation and luminance is applied immediately to the 3 DMX channels addressed by SLOT, furthermore each is stored in a global register (not individuall per DMX channel). During every new setting of hue, saturation and luminance, the stored global values ot the other color components are applied, too.

If set, the fade time gets also also applied in combination with the H command. But the fade transition from the previous color tone to the new one is performed **along a straight line through the color space**, not along the spectral color circle. So, if is faded between very different colors, disagreable desaturated color tones may appear. To get a perfect color transition, up tp 6 subsequent fade steps between neighboured colors have to be performed.The technical handling can be simplified by use of the chaser effect.

W <saturation>

Sets the color saturation (adds White) for 3 subsequent DMX channels (RGB lamp).

Comment: The parameter of <saturation> may take values between 0 and100. The maximum value 100 (default) 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.

B <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 100. The value 100 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.

T < tenths of seconds>

Enter parameter into FADETIME. No action is started directly.

Parameter: FADETIME is always entered in tenths of seconds. Maximum fadetime is 12.7 seconds. This is valid for all subsequent level changes until a new T command is given. The FadeTime is stored as global Parameter (= 0 at virgin Flash) and is reloaded at system start.

Example: T113 sets the Fade Time to 11.3 seconds

X (no parameter)

Stop (freeze) all fading processes immediately at their present levels

---

A <cycle length>

Set the length of the chaser cycle (n=2 to 220) **and start the chaser**

<cycle length> = 0 switches the chaser OFF

Comment: The chaser feature works as follows: a sequence of presets (=lighting scenes) is loaded in a cyclic manner to DMX channels 1 to 128.

Before the chaser can be started, the **step duration** (command P) as well as the **start scene** (command G) has to be adjusted. Default at power on: 2 (i.e. switching between 2 lighting scenes)
DMX channels 129 to 512 are not modified by the chaser and may be used for chaser indepent steady lighting.
The actual setting of the fade time is applied by the chaser.

Example: if the chaser cycle is set to 4 and the the chaser start is set to 64, then presets no. 64,65,66,67 are loaded partially, then preset no. 64 again and so on

---

P <chaser step duration>

Set duration of chaser step in 1/10 s units (step duration = 0 to 25.5 seconds)

Comment: Default at power on: 20 (= 2 seconds). After the duration of a chaser step is over, the chaser automatically loads the next lighting scene in the cycle. After <cycle length> presets were loaded in sequence, the procedure is repeated form <chaser start> .
Step duration 0 only virtually stops the chaser. It can be forwarded by one step with command "N".

G <chaser start>

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

Accepted start values: 0 to 219. Default at power on:50. If the chaser would access a preset beyond 220, the sequence continues with loading scene <chaser start>

N (no parameter)

Forward the chaser immediately (asynchronously) by one step

D (no parameter)

Sets all DMX channels immediately to level 0. Panic function, no soft fade.

The key **must be pressed a second time** after 300ms to avoid simple mis-operation

---

R <number of bytes>

Poll <number of bytes> of the DMX transmit buffer starting from the actually addressed DMX channel and send them via USB.

Parameter: number of polled bytes (1 to max. 512) . When the level of DMX channel no.512 is sent, the output stops.

Syntax of the resulting state message:

R<number of bytes> from S <1st channel no.>: V <DMX level> [**,<**DMX level>…] <CR >

Q (no parameter)

Display level of the actually selected DMX channel, Fade Time and DMX cycle

Additionally the level of the next 2 DMX channels is displayed,
to get quick information about the state of a typical RGB lamp.

Finally the Chaser setup is displayed via Terminal.

L <number of DMX channels>

Limits the DMX cycle to a specific number of DMX channels.

Minimum is 24, maximum is 512

The DMX cycle length is stored in the "Globals" Flash page and reloaded at system start. (default at virgin Flash = 512)

~ <preset number>

Saves the current content of the transmit buffer permanently in the microcontroller flash memory.

Preset # 1 to 220 is supported

Preset #0 (all DMX channels = 0) is not stored in Flash, but simulated by firmware.

@ <preset number>

Loads the permanently stored lighting scene into the DMX transmit buffer.

Preset # 0 to 220 is supported

Preset #0 set all DMX channels to zero ("dark scene"). **Gets Loaded at System Star**t

A soft fade from the previous lighting scene is performed.

C <preset number>

Sets the MIDI channel which is used by the MIDI command set

Range 1 – 16.

The MIDI channel is stored in the "Globals" Flash page and reloaded at system start

---

U <USB Vid/Pid>

Sets the Vid/Pid identifiers for USB connection after next system start

First letter V is displayed. Enter your Vid as a **4 digit hex number**
Next letter P is displayed. Enter your Pid as a **4 digit hex number**
Leading zeroes must be entered.
Finally the input must be confirmed with upper case Y, else the action is cancelled.

The USB Vid/Pid is stored in the "Globals" Flash page and reloaded at system start.

Comment: This action is critical because a corresponding .inf file must be installed on the PC. Else USB connection will not work any more !
Rescue if no .inf file or wrong number entered: Connect **Pin A1** of the microcontoller with **Ground** while system start (e.g. grounded needle). This will erase the corresponding Flash page and reset USB Vid/Pid, MIDI channel, Fadetime and DMX cycle length to default.

? (no parameter)

Display actual values of Global Parameters:

Fadetime, DMX cycle length, MIDI Channel, USB Vid/Pid

MIDI Channel Message protocol of the DMX512 transmitter

is active when **PA14 (SWD-Clk pin) is connected with Ground** (jumper or switch).

By default MIDI channel no. 1 is the base channel for command input. This may be changed with ASCII command 'C' and stored in Flash as Global Param. Details see above.

Quick start and basic commands:

The most frequently used application is **lighting control with NOTE ON messages from a sequencer**.

To address any of the DMX channels 1 to 127, control data have to be sent on the selected MIDI base channel. How to access DMX channels 128 to 256 see below.

The 1st data byte of the MIDI command defines the DMX channel to be addressed.
The 2nd data byte of the command describes the DMX level (light intensity) to be set.
the 2nd MIDI data byte is multiplied by 2 inside the USB/DMX Interface.
Exception: 2nd data byte 127 sets the DMX level to 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.**.

To write data into DMX channels 128 to 256 with these commands, the MIDI commands are sent on the **next higher MIDI channels** as described in the table:

coded MIDI channel	**1st data byte**	**addresses DMX channel #-**	**calculation of 1st data byte**
as selectred base channel	1 to 127	1 to 127	= DMX channel
as base channel + 1	0 to 127	128 to 255	= DMX channel minus 128
as base channel + 2	0 to 127	256 to 383	= DMX channel minus 256
as base channel + 3	0 to 127	384 to 511	= DMX channel minus 384
as base channel	**0 special case !**	**512**	**= 0**

This means: on a sequencer program you have to **reserve a block of 4 MIDI channels** for full control of the USB / DMX Interface 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.

This limitation can be worked around with somewhat **more complex CONTROL CHANGE commands, which allow to adjust any DMX channel 1 to 512 with full 8 bit accuracy using only the MIDI base channel**. See description of the appropriate commands below.

The response of the USB/DMX Interface to NOTE ON messages with **velocity=0** or to any **NOTE OFF** messages is prevented with PROGRAM CHANGE command 120 (default state). This way you can "play in" DMX level timing with NOTE ON on a keyboard or sequencer without care about note ends. Setting DMX level to 0 with NOTE OFF or zero velocity is allowed with PROGRAM CHANGE command 121. This allows playing "realtime lighting piano".

Survey of all MIDI Channel Commands (MIDI Implementation Chart)

Abbreviations:

DB means "data byte", DB1 means "1st data byte" (note value, controller number), DB2 means "2nd data byte" (velocity, controller value)

When using **PROGRAM CHANGE** commands you should take into account, that most MIDI devices and software physically send the data byte value "0" when "program no.1" is selected and so on ! **In the table "DB" denotes the physically transferred data byte.**
Check your equipment if the parameter has to be entered by 1 higher

MIDI message configured MIDI channel plus up to 3 additional MIDI channels	**special data values**	**function /effect**	**p.**
NOTE OFF	**DB1** = DMX channel ++ **DB2** = ignored	DMX level --> 0.	12
NOTE ON MIDI channel = base ch. + up to 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
CONTROL CHANGE MIDI channel = base ch.	**DB1**= 1 **DB2** = 0..127	set Fade Time 0 …12.7 sec (DB2 / 10) in steps of 1/10 sec	13
	DB1 = 16 (hex 10) **DB2 =** step in 1/10 sec	set step duration of the chaser. 0 = chaser OFF. Controlled by an internal timer.	15
	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	15
	DB1 = 19 (hex 13) **DB2 =** start preset	set start scene (preset no.) 0-127 of the chaser cycle. See detailled description	15
	DB1 = 20 (hex 14) **DB2 =** start preset-128	set start scene (preset no.) 128-219 of the chaser cycle. See detailled description	15
	DB1 =36 (hex24) **DB2** = 1 to 9	poll (entry of DB2) levels of DMX OUT starting from the actually addressed DMX channel Response is a series of Control Change messages	16
	DB1 =37 (hex25) **DB2** = 1 to 127	poll (entry of DB2) levels of DMX OUT Response is a SysEx message like ASCII Cmd "R"	16
	DB1 = 40 (hex 28)	fill a block of DB2 DMX channels starting from ch. SLOT+1 with the final level of DMX ch. "SLOT"	14
	DB1= 48 (hex30) **DB2** = 0..127	limit number of transmitted DMX channels to **(DB2+1) * 4** (DB2 min= 5 →24DMX channels)	16
	DB1 = 64 (hex 40) **DB2** = 0 to 127	set hue (color tone) of an RGB lamp (3 consecutive DMX channels) see detailled description !	14
	DB1 = 65 (hex 41) **DB2** = 0 to 100 !!	set color saturation of an RGB lamp i.e.add grey- or white component (3 consecutive DMX channels)	14
	DB1 = 66 (hex42) **DB2** = 0 to 100 !!	set brightness/luminance of an RGB lamp (3 consecutive DMX channels)	15
	DB1= 80–83 (hex50-53) **DB2** = 0..127	address DMX channel using only one single MIDI channel. "	12
	DB1 = 84 (hex 54) **DB2** = 0..127	set DMX level at addressed DMX ch. to **DB2** *2 adjust DMX level with 8 bit resolution to **even** value	13
	DB1 = 85 (hex 55) **DB2** = 0..127	set DMX level at addressed DMX ch to **DB2** *2 **+ 1** adjust DMX level with 8 bit resolution to **odd** value	13
	DB1 = 86 (hex 56) **DB2** = 0..127	First **increase addressed DMX channel** there set DMX level to **DB2** *2**. (**8 bit resolution **even** val.)	13
	DB1 = 87 (hex 57) **DB2** = 0..127	First **increase addressed DMX channel** there set DMX level to **DB2** *2 **+ 1** adjust DMX level with 8 bit resolution to **odd** value	13
	DB1 = 96 (hex 60) **DB2**= preset number	load preset (Lighting Scene) 0-127 from Flash memory	17
	DB1 = 97 (hex 61) **DB2**= preset num - 128	**load preset** (Lighting Scene) **128-220** (data byte = preset# - 128)	17
	DB1 = 112 (hex70) **DB2** = preset number	save preset (Lighting Scene) 1-127 nonvolatile in flash memory**.** Preset#0 (all channels=0) is generated by firmware and not stored in Flash	16
	DB1 = 113 (hex71) **DB2** = preset num -128	**save preset** (Lighting Scene) **128-220** (data byte = preset# - 128)	16
	DB1 = 119 (hex 77)	**DB2** has the same meaning as the data byte of the corresponding PROGRAM CHANGE command.	

PROGRAM CHANGE	DB = 1	stop (freeze) all fader processes immediately	14
MIDI channel = base ch.	DB = 8	decrease DMX level (minus 1) at addressd channel	14
	DB = 9	increase DMX level (plus 1) at addressed channel	14
	DB = 16 (hex 10)	forward Chaser by 1 step immediately	15
	DB = 119 (hex 77)	send the actual setup as SysEx message (similar content as ASCII command "Q").	16
	DB = 120 (hex 78)	Note Off and Note velocity 0 is ignored (default)	17
	DB = 121 (hex 79)	Note velocity 0 and Note Off is set as DMX level	17
	DB = 127 (hex7F)	Panic: set DMX level of all channels to zero	14
PITCH CHANGE MIDI channel = base ch.	**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

Simple setting of DMX channel and DMX transmit level

NOTE ON (NOTE OFF)

The first MIDI data byte (note value/pitch) **sets the DMX channel.**
The 2nd data byte (note velocity) **describes the light intensity to be set.**

The DMX level is **twice of** the second MIDI databyte.
Or vice versa: the **the 2nd MIDI data byte is HALF OF the intended DMX level (0 to 255).**

Following exceptions do apply:

Velocity 127 sets the DMX level to 255
after PROGRAM CHANGE 121 is sent, Note OFF and NOTE ON with velocity = 0 are ignored
BUT: NOTE ON with velocity = 1 sets DMX level to 0

This simple method works only for DMX channels 1 to 127.
To address DMX channels 128 to 512, the USB / DMX Interface expects control commands on a higher MIDI channel corresponding with following table:

coded MIDI channel	**1st data byte**	**addresses DMX channel-**	**calculation of 1st data byte**
as selected base channel	1 to 127	1 to 127	= DMX channel
as base channel + 1	0 to 127	128 to 255	= DMX channel minus 128
as base channel + 2	0 to 127	256 to 383	= DMX channel minus 256
as base channel + 3	0 to 127	384 to 511	= DMX channel minus 384
as base channel	**0 special case !**	**512**	**= 0**

Address or select the active DMX channel with:

CONTROL CHANGE

This alternative command version is used, **when all MIDI messages shall be sent on the MIDI channel = selected base channel**.

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

1st data byte	**addresses DMX channel no.**	**2nd data byte**	**calculation of 2nd 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 (hex53)	256 to 2383	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**

This command does not directly trigger any action. But the updated active DMX address register will be executed together with subsequent commands.

Set DMX level with 8 bit resolution at the addressed DMX channel with:

Method 1:

PITCH WHEEL CHANGE

MIDI channel as selected base channel

1st data byte: if equal or greater than 64, "1" is added to the DMX level
if less than 64 (hex40), nothing is added to the DMX level

2nd data byte: desired DMX level divided by 2
inside the USB / DMX Interface, this value is multiplied by 2 and incremented (or not) according to value of 1st data byte before it is written into the DMX transmit buffer

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

Method 2:

CONTROL CHANGE

MIDI channel as selected base channel

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**

Set Fade Time with:

CONTROL CHANGE

MIDI channel as selected base channel
1st data byte = **1**, then
2nd data byte = 0 – 127: fade time in 1/10second units
(setting range 0 –12.7 seconds).

The actual value of Fade Time is copied into the respective resource when the fade process is started. Immediately after then the Fade Time can be modified without retroactivity on running fade processes.

Simple modifications of the DMX level with:

PROGRAM CHANGE

MIDI channel as selected base channel

data byte = 1 stops (freezes) all active fader processes immmediately.

data byte = 8 decrease (subtract 1 from) the level of the addressed DMX channel

data byte = 9 increase (add 1 to) the level of the addressed DMX channel

These comands are useful to perform extremely slow fade transitions and to make fine level adjustmets

data byte = 127 (hex 7F) sets all DMX channels immediately to level 0 ("Panic switch")

Fill block of DMX channels starting from (addressed channel+1) with the final level of the actually addressed DMX channel with:

CONTROL CHANGE

MIDI channel as selected base channel

1st data byte = 40 (hex28)

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

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

CONTROL CHANGE

MIDI channel as selected base channel

1st data byte = 64 (hex 40): set hue at the addressed DMX channel + next 2

2nd databyte = hue (color tone) 0 - 127

This will approximately result in following colors. Intermediate hue values will result in intermediate colors:
2nd databyte= 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.

Comment: This command influences the actually addressed DMX channel (actual entry to the SLOT register set before by a NOTE ON or CONTROL CHANGE message) and the two next higher DMX channels. It is provided that the RGB setting of the respective lamp is done on these 3 subsequent DMX channels. All features else of a complex lamp ("fixture") may be used independently. Later each DMX channel can be accessed individually with any command.

If set, the fade time gets also also applied in combination with these commands. But the fade transition from the previous color tone to the new one is performed along a straight line through the color space, not along the color circle. So, if is faded between very different colors, disagreable intermediate color tones may appear. To get a perfect color transition, up tp 6 subsequent fade steps between neighboured colors have to be performed.The technical handling can be simplified by use of the chaser effect.

Set color saturation of a RGB lamp with :

CONTROL CHANGE

MIDI channel as selected base channel

1st data byte = 65 (hex 41): set saturation at the addressed DMX channel + next 2

2nd databyte = color saturation 0 – 100 ("percent")

Comment: "Saturation" describes the amount of white or grey in a color tone (pastel shade). Saturation=100 gets a pure color, saturation=0 gets white or grey without specific color tone.

Set brightness/luminance of a RGB lamp with :

CONTROL CHANGE

MIDI channel as selected base channel
1st data byte = 66 (hex 42): set luminance at the addressed DMX channel + next 2
2nd databyte = brightness/luminance 0 – 100 ("percent")

Set chaser step duration with :

CONTROL CHANGE

MIDI channel as selected base channel
1st data byte = 16 (hex 10)
2nd data byte = (1 - 127) chaser step duration in 1/10 second units
by **default step duration of 2 seconds is active**
Step duration 0 only virtually stops the chaser. It can be forwarded by one step with PROGRAM CHANGE 16 at any time.

Comment: After the duration of any chaser step is over, the chaser automatically loads the next preset in the cycle.

Set chaser cycle length and start it with :

CONTROL CHANGE

MIDI channel as selected base channel
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 chasecan be started, the **step duration** and **start scene** has to be adjusted
As soon as <cycle length> scenes are loaded in sequence, the procedure repeats from <chaser start>.
If the chaser would access a preset beyond 220, the sequence continues with loading <chaser start>

Set chaser start preset (Lighting Ccene) with :

CONTROL CHANGE

MIDI channel as selected base channel
1st data byte = 19 (hex13) ,
2nd data byte: = chaser cylcle start preset (0 - 127)
for higher preset numbers:
1st data byte = 20 (hex14)
2nd data byte = 0 - 91 (preset number -128) saves scene as preset# 128 . 219

The chaser loads only the first 128 DMX channels of the preset. The remaining upper DMX channels can be used for chaser-independent steady light.

Forward chaser immediately (asynchronously) by 1 step with :

PROGRAM CHANGE

MIDI channel as selected base channel
data byte = 16 (hex 10) forward chaser immediately by 1 step

Poll DMX transmit buffer at/from the actually addressed DMX channel:

CONTROL CHANGE

Method 1:

MIDI channel as selected base channel
1st data byte = 36 (hex 24)
2nd data byte 1 to 10
The response comes as a sequence of CONTROL CHANGE messages:
first CC: actually addressed DMX channel, format as described for CC 80-83
second CC: 1st data byte = 1, 2nd data byte = actual Fade Time
following CCs:DMX level at this and subsequent channels, format as CC 84,85

Method 2:

MIDI channel as selected base channel
1st data byte = 37 (hex 25)
2nd data byte 1 to 127
The response comes as a SysEx message with same content as ASCII command "R"
The SysEx Manufacturer ID is simply 1 byte hex 7D (non commercial ID)

Poll actual DMX setup:

PROGRAM CHANGE

MIDI channel as selected base channel
data byte = 119 (hex 77)
The response comes as a SysEx message with same content as ASCII command "Q"
The SysEx Manufacturer ID is simply 1 byte hex 7D (non commercial ID)

Limit the number of transmitted DMX channels ("DMX cycle") with:

CONTROL CHANGE

MIDI channel as selected base channel
1st data byte = 48 (hex30)
2nd data byte = 5 … 127 (hex 7F)
The number of transmitted DMX channels (DMX cycle length) is calculated as follows:
cycle = (2nd data byte+1) * 4. The minimum 2nd data byte is =5 -> 24 DMX channels

---

Save, store preset (= actual Lighting Scene) with:

CONTROL CHANGE

MIDI channel as selected base channel
1st data byte = 112 (hex70)
2nd data byte = 1 … 127 (preset/scene number)
The actual lighting scene is stored nonvolatile in the microcontroller flash memory.
Preset#0 (all DMX channels = 0) is generated by firmware and not stored in Flash.
for higher preset numbers:
1st data byte = 113 (hex71)
2nd data byte = 0 … 92 (preset number - 128) saves scene as preset# 128 … 220

Load preset (=Lighting Scene) into DMX transmit buffer with:
CONTROL CHANGE

MIDI channel as selected base channel
1st data byte = 96 (hex60)
2nd data byte = 0 … 127 (preset/scene number)
for higher preset numbers:
1st data byte = 97 (hex61)
2nd data byte = 0 … 92 (preset number - 128) loads scene as preset# 128 … 220

Change mode of operation with :

PROGRAM CHANGE

MIDI channel as selected base channel

data byte = 1

stops (freezes) all active fader processes immmediately.

data byte = 120 (hex 78**)**

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

data byte = 121 (hex 79)

NOTE ON messages with 2nd data byte (velocity) = 0
are accepted and set the DMX level to 0
This mode is active, until it is revised with PROGRAM CHANGE 120 or device new start

General comment on PROGRAM CHANGE messages:

When formatting a PROGRAM CHANGE message, many MIDI sequencers and other MIDI control equipment demand that the data value (program number) has to be entered one higher than the physically transmitted data byte.(Example: user entered program #1 is transmitted as hex C0 00), **The USB / DMX Interface evaluates PROGRAM CHANGE messages with the physical parameter as described in the manual. Check your equipment if the parameter has to be entered by 1 higher**

---

contact: wschemmert@t-online.de