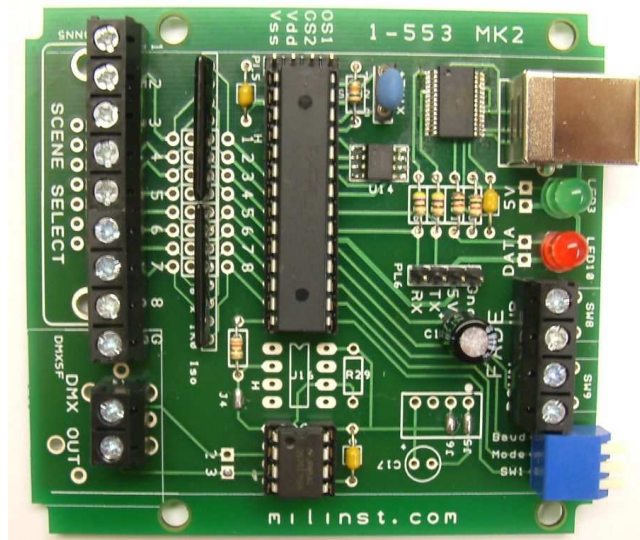


512 Channel Serial to DMX Transmitter PRO



The transmitter is a full speed serial to DMX512 transmitter. The on-board processor generates the full specification DMX-512 data transmission with timing independent of any connected PC source. The transmitter may also be used as a stand alone device with either a switched scene or script running facility.

- 512, 256 or 128 channels per scene capability
- On-board scene storage of 13 scenes at 512 channels/scene, 55 scenes at 128 channels/scene
- **New** Run Script mode with script switching function
- **New** internal Scene Merge, X-Fade and Scene fade/dimming functions
- In Command Mode, control either by RS232 commands or manual switch selection
- RS232 baud rates of 9.6 or 19.2kB
- USB serial interface and optional TTL interface for embedded applications
- PCB layout for 5-pin Female XLR or 2-pin terminal block DMX output
- Sample PC control software available.

Connections

DMX Out	3-way terminal block (Data +, Data -, Screen)
Power In	USB Type B socket
Serial Data	USB type socket (TTL level input interface available)
Manual Scene select	9-way terminal block (Scenes 1-8 and ground). Active low.
Mode select	2-way switch: On: Run-script mode, Off: command mode
Baud Rate select	2-way switch: ON: 9.6kb, Off: 19.2kb.
SW1	Not used
Fade	Twin 2-way terminal blocks: Fade up and down inputs (active low)

Operation

The transmitter may be operated in either a Command Mode or Run (script) Mode by setting the Mode switch to the appropriate position. In Command Mode, DMX scene switching is based on manual switching or serial commands. In Run-Script Mode the scene sequence is executed along with the scene duration and x-fade times according to a user generated stored script.

Data must be loaded into the transmitter using the Command mode before the Run-Script mode may be used.

Command Mode

Loading Data

Download and install the DMXGrid software package from the Downloads section of our web site.

Set the board Mode switch to Command (Cmd) mode, select the required baud rate and connect a suitable DMX output cable. Attach a suitable USB connection to the transmitter to power up the device. The transmitter will start generating DMX data.

Run the DMXGrid software, select the transmitter serial port (you may need to check this with Windows Device Manager) and set the baud rate to match that on the 1-553 transmitter.

The software will begin by importing all stored data from the transmitter. The output will then default to Scene 1 (high-lighted) - this is the DMX data that will be initially send by the transmitter.

Before attempting to enter/change scene data, check the default channels/scene value (ie 512, 256 or 128) is the value you require. If not then select **New** from the menu- this procedure will prompt for the new value of channels per scene and then automatically clear all data from the eeprom.

Enter data changes to Scene 1 as required- any changes will immediately appear on the DMX output. Once all data changes have been made to a particular scene, use the **Save Scene** button to save the data to the on-board eeprom. [**Note: this is a change from MK1 firmware!**]

Use the **Goto Scene** button to change to other scenes to input / save data.

The transmitter will load and transmit the data from the next selected scene.

The data stored in the on-board eeprom can be checked by pressing the **Read Data** button.

Data from individual or all scenes may be deleted using the **Reset** button from the menu.

If you intend to use the Run Script Mode- note the following:

In Run-Script mode, the last 4 channels of each scene are used to control the run-script actions:

The last scene byte is reserved- any value here is disregarded.

The last byte -1 is the next scene to be shown in the sequence

The last byte -2 is the x-fade time (in units of 65msecs) to the next scene

The last byte -3 is the duration of the current scene (units of 2secs) until switching occurs

Channel #	Scene 1	Scene 2	Scene 3	Scene 4	Scene 5	Scene 6	Scene 7	Scene 8
253	2	10	40	5	200	156	7	45
254	255	50	50	0	0	150	100	35
255	3	4	5	6	7	8	2	1
256	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

The above example shows a typical script running data that could be entered for an application using scenes 1 through 8 with 256 channels per scene.

The running order would be 1,3,5,7,2,4,6,8,1 etc. [Scripts must always start with scene 1]

Scene 1 would run for $2 \times 2 = 4$ secs before switching to scene 3 in a slow x-fade of $255 \times 65 \text{msecs} = 16.5$ seconds.

Scene 4 would run for $2 \times 5 = 10$ seconds before switching immediately ($0 \times 65 \text{msecs}$) to scene 6.

In Script running mode these last 4 operating bytes are not transmitted in the DMX stream- ie in the above example, only DMX channels 1 to 252 would be transmitted.

Scene Switching

Once data has been entered into the transmitter, scenes may be selected either by using the 8-way terminal block (for scenes 1 through 8) or via serial commands. Pins 1-8 are weakly pulled to +5V and are active low.

To switch scenes manually, temporarily ground the relevant scene number terminal- the DMX transmission will change from the current scene to the newly selected scene. The rate of change between scenes is controlled by the scene number pins: for a gradual change between scenes, just ground and release the pins. For a faster change, ground and hold the pin until the change is complete.

Scene Dimming

The current scene may be dimmed using the Dim UP/Down connections. Short and hold relevant the pin pair until the required level is achieved.

Shorting the Up pins will not increase the DMX data values above the original stored scene values.

Serial Commands

The following serial commands are supported in Command Mode. Data should be sent in the 8-N-1 format at the selected baud rate.

[If sent via the optional TTL input header, data should be NonInverted RS232 polarity (ie as if from a line driver) at TTL levels.]

Load data command (\$01)

Sets the value of the specified channel on the currently active scene.

Requires 4 bytes, Load command (\$01),
 High byte address (\$HH)
 Low Byte address (\$LL)
 Data Byte (\$XX)

Eg string \$01,\$00,\$02,\$0F will write the value of 15 to channel number 2 on the currently active page. Address values greater than the current page size will be wrapped round.

The transmitter will return the byte \$55 plus the last byte received (in the above case, the return string would be \$55,\$0F).

To load data to sequential data locations continue to send the data bytes within 50 msecs of one another. If no data is received within 50 msecs of the last data byte, the serial link will time out and respond with the \$55,\$last_data_ byte string.

Eg to send \$32,\$45,\$31,\$56 to successive locations starting at channel \$00,\$76:

Send: \$01,\$00,\$76,\$32,\$45,\$31,\$56

with a maximum of 50msecs between successive data bytes.

After 50msecs, the transmitter will return \$55,\$56.

Note: If you wish to save any data changes to the eeprom, finish with the 'Save Current Scene Data' (\$07) command – if not, any data changes will be lost when changing to another scene.

Set Number of DMX Channels per Scene (\$02)

Sets the number of transmitted DMX channels per scene.

Requires 2 bytes:

Command Byte \$02
Data Byte \$XX

Where \$XX can have one of 3 values:

Data Byte	Number of Channels	Maximum stored Scenes
\$00	512	13
\$01	256	27
\$02	128	55

Other data byte values are ignored and the comms link will timeout.

Once changed, the number of channels per scene is stored in non-volatile memory and is read at power up.

Transmitter returns with \$55 after command executed or \$00,\$XX if the command has timed out between successive command bytes.

Set the Current Scene Number (\$03)

Sets the currently active scene

Requires 2 bytes

Command byte \$03
Data Byte \$XX

Where \$XX is the required scene number and is limited to the maximum allowable value based on the Set Number of Channels.

Transmitter returns with \$55 after command executed or \$00,\$XX if the command has timed out between successive command bytes.

On power-up the default start scene is scene 1

Read Current Scene Data (\$04)

Transmits the data bytes from the currently active scene to the PC.

Single byte command.

Data begins at Channel 1 and continues until the last channel number data is sent.

Returns with \$55 after command executed

The DMX transmission is suspended during this command so it is advisable to insert a short time delay between successive 'Reads' to allow the DMX transmission to re-start and maintain the equipment on the network. (Equipment should be capable of withstanding an extended break of up to 1second according the DMX512 specification)

To read the whole stored data, set the active scene to 1, read the data, increment the scene number and re-read the data. Repeat until all scenes have been read.

Read current scene number (\$05)

Returns the currently set scene number as a single byte followed by \$55.

Read the number of channels (\$06)

Read the current number of channels being transmitted as a 2-byte string (low byte first) followed by \$55 eg for 512 channels: \$00,\$02,\$55

Save Current Scene Data (\$07)

Saves the current DMX data to the currently selected scene.

Returns with \$55 after command executed.

Merge the current scene with the next scene (\$08)

Requires three bytes: \$08, next_scene, rate

If next_scene is a valid scene number then the operation merges the current scene data with the next scene values according to:

$$\text{current_scene_value} * \text{rate} / 255 + (255 - \text{rate}) * \text{next_scene_value} / 255$$

where rate is between 0 and 255. If rate =0 then the data is solely that of the next scene, if rate = 255 then the data is solely the current scene.

If next_scene=0 then the operation is a dimming function only of the current scene data according to:

$$\text{current_scene_value} * \text{rate} / 255$$

The operation takes 15msecs for 128 channels, 25 for 256 and 50msecs for 512 channels.

Returns with \$55 after command executed.

Note: This command does not alter either the current_scene or next_scene stored data values.

X-Fade from current to next scene (\$09)

Requires 3 bytes:

Command	\$09
Next_scene	\$YY
X-Fade rate	\$ZZ

Next_scene is the value of the next scene to be switched to.

X-Fade rate is the time for the x-fade in units of 65msecs. A value of \$00 would give an instant change to the new scene. A value of \$FF would change to the new scene in 16.5 seconds.

Returns with \$55 after command executed.

Clear the data from the current Scene (\$0A)

Requires one byte: \$0A

All channels from the currently selected scene are re-set to zero.

Returns the ACK byte (\$55) when the command has been finished

Clear all the data from all scenes (\$0B)

Requires one byte: \$0B

All channels from all scenes are re-set to zero.

Returns the ACK byte (\$55) when the command has been finished

Identify Version (\$0C)

Requires one Byte: \$0C

Returns the ACK byte (\$55) with the MK2 firmware, no response from the MK1 firmware

It is strongly recommended that the board be powered down before changing from Command to Run-Script Modes.

Set the Mode switch to the Run (Script) position.

The transmitter will start at Scene 1 and then continue to execute the sequence as defined in the last 4 channels of each scene (see above for details of these bytes). The 4 sequence control bytes will not be included in the DMX transmission.

The manual scene switching pins are active during Run-Script Mode and are read continuously except during the x-fade period. If a manual switch is detected, the current scene duration will be prematurely shortened and a x-fade made to the newly selected scene. Eg

Assuming the following scene sequences are stored:

1,10,11,12,13,14,15,16,1
3,21,22,23,24,25,26,3

Then the start-up running sequence would be

1,10,11,12,13,14,15,16,1,10,11,.....

If switch 3 is pressed whilst scene 12 is active, the sequence would change as follows

1,10,11,12,.....12,3,21,22,23,24,25,26,3,21,22,...

-and if switch 1 is pressed during scene 24:

21,22,23,24,1,10,11,12,13....

The sequence returns to the original play-back sequence.

The 8 inputs therefore allow up to 8 separate sequences to be switched in depending on external triggers (eg PIR, Time-clock).

Mechanical

PCB size: 81x75mm Aprox: 3-1/4" L x 3" w x 1/2" H