DecaBox - Field Programmable DMX to RS-232

Getting started & field troubleshooting.

- Getting Started
- Sample Patch File
- Purchasing

Getting Started

This guide should serve as a road map for initial installation of the DecaBox DMX to RS-232 Field Programmable system. Unfortunately, since every installation is unique, it's impossible to provide a one-size-fits-all solution. However, when we help customers via telephone or remote desktop, here are the general steps we follow.

For purposes of this document, we'll walk step by step through controlling a Digital Projection M-Vision Laser 18k Video Projector via several channels of DMX data.

The general routine looks like this::

1. Obtain the RS-232 command set for the device in question and test these commands using a PC and serial interface.
2. Add the commands to the DecaBox patch file.
3. Confirm that the DecaBox output strings are correct
4. Connect the DecaBox to the projector and verify proper operation.

Each step will be addressed below.

Step 1: Obtain a Command Set and Test Using a PC

A detailed command set for this projector can be found at this link. For demonstration purposes, we will implement 'Power On' and 'Power Off' commands.

On page 33 of the document we learn the required baud rate and serial settings:

The detailed generic command structure is on page 34:

Serial Port setup

- Baud rate   9,600 bp
- Data length   8 bits
- Stop bits   one
- Parity   none

Protocol commands

Commands are used to simulate menu operations and determine the settings of the projector, and use the following format:

- All commands consist of ASCII text strings starting with an asterisk✳ and ending with an ASCII Carriage Return character↵ (code 13):
  `*command operator <value>`↵
- The <command> string determines which setting the command will affect.

Thus, all commands begin with the * character and end with a carriage return. There are spaces between characters for easy readability:

Examples

|	
*orientation = 3↵	sets the orientation to Rear Ceiling (for a ceiling mounted projector positioned behind the screen)
*aspect.ratio ?↵	asks what the current aspect ratio is
*zoom.in↵	commands the projector to zoom in
*orientation=3↵	is an invalid instruction because of the missing spaces before the operator and the value

On page 58 we see the power command:

Command	Operators allowed					Values accepted / Format of response – per model		
	Set =	Get ?	Inc +	Dec -	Exe	HL Laser II 3D	HL Laser 4K	M-Vision Laser 18K
Miscellaneous								
power	●	●	✕	✕	✕	0 = Off 1 = On		n/a
...	●	●	✕	✕		0 = Open 1 = Close		
...						1 = Warm Up 2 = Imaging 3 = Cooling 4 = Error		
errcode	✕	●	✕	✕	✕	string		
sys.info	✕	●	✕	✕	✕	n/a		string

Continues on next page...

Thus, the actual data we need to send to the projector looks like this. In this example, [cr] stands for the carriage return character, which is decimal 13 or hex 0x0D:

```
*power = 0[cr]    ← Off
*power = 1[cr]    ← On
```

Connect the projector to a PC running a terminal program (we like RealTerm) and send the data to the projector, confirming that the command syntax is correct. The projector should respond to commands.
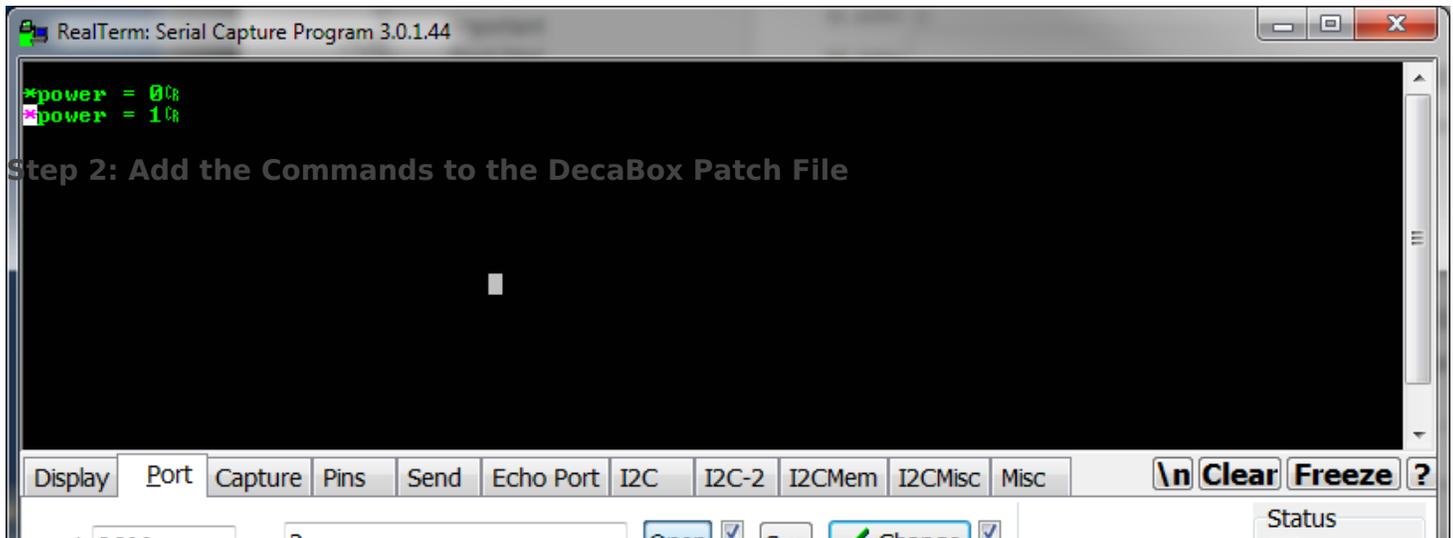
Step 2: Add the Commands to the DecaBox Patch File

The DecaBox uses a file called 'patch3.txt' to bind serial strings to specific DMX channel:value combinations. A sample file is included at the end of this document. The file is also shipped on the SD card of each DecaBox. Remove the four screws holding the chassis together to access the SD socked.

This file **must be edited** in NotePad or TextEdit. Don't use Word or other complex programs - they introduce garbage characters which will cause the system to fail.

First, we set the system baud rate for this projector:

```
; Set the global baud rate.  Default here is 8N1, no handshaking; Regular options are 9600 19200
38400 57600 115200
; no commas are needed
;
baud 9600
;
; This document is processed at startup.
;
```

Then, let's assign the power command two ways:

1. Assign power on and power off to a pair of DMX channels at 95%+
2. Assign power on and power off to a single DMX channel in two sections of the [0 255] range

We'll add these commands to the end of the file. Note the use of $0D to designate the carriage return:

```
;Assign DMX channels 10 and 11 at 95%+ to trigger on and off
10 250 255|*power = 0$0D
11 250 255|*power = 1$0D
; Assign DMX channel 12 to trigger on and off throughout the range:
12 10 127|*power = 0$0D
12 128 255|*power = 1$0D
```

When the dust settles, 'bumping* channel 10 or 11 will trigger a single transmission of either command. Running channel 12 between 0 and 100% will generate many, many copies of the command. When programming cues in a lighting console, an example cue might look like this:

CH12 @ 40% Time 0

CH12 @ 100% Time 0

Step 3: Connect the DecaBox to a PC and Test Using a Terminal Program

Now that the commands have been created, save the patch3.txt file back to the SD card, insert it in the DecaBox, and re-add power. After several seconds, the system will be stable and ready to process DMX data.

When connecting the DecaBox to a PC, a null modem / crossover cable *is required*. The reason for this is that the DecaBox pretends to be a PC as far as the serial port wiring is concerned. When connecting two PCs together, a cable which swaps pins 2 and 3 from end-to-end will allow them to talk with each other. As printed on its panel, the DecaBox transmits data on pin 3. In this firmware build, it does not receive serial data, but that data would be accepted on pin 2. Ground is pin 5.

Here we connect a DMX source to the DecaBox and then bump channels 10 and 11 to 100%:

On the DecaBox, the left LED will illuminate and pulse when valid DMX data is present. When an RS-232 command is triggered, the right LED will flicker briefly as the string is transmitted.

Comparing the received commands to those generated earlier indicates that they are correct.

Step 4: Connect the DecaBox to the RS-232 Equipment and Test

This can be the trickiest step. Depending on the receiving equipment's serial pinout, a crossover cable may or may not be required to make the final connection. Try both to be sure.

Sample Patch File

This sample patch3.txt file can be copied and pasted as needed.

```
; DecaBox Protocol Converter
; Engineering Solutions Inc
; www.response-box.com
;
; Firmware for converting DMX512 input
; into RS232 output in a user configurable way
;
; This file is called patch3.txt and should be edited; in either TextEdit (OS X) or Notepad
(windows).  Other programs,; such as Word, may introduce unwanted garbage characters in the file.
;
; Note that lines which begin with a semicolon are comments, and; are ignored by the firmware. A
line can contain a comment or a command,
; but not both.
;
;Also, lines which contain commands MUST have the command begin in column 1.;No leading spaces
allowed.
;
;This is fine
; But this will throw an error. Make sense?
;
; Set the global baud rate.  Default here is 8N1, no handshaking; Regular options are 9600 19200
38400 57600 115200
; no commas are needed
;
baud 115200
;
; This document is processed at startup.
;
```

```
;
; Up to 512 characters may be assigned to a single DMX channel:value combination
;
; Commands are divided up into several columns as follows:; and columns are separated by the
space character ' ' in the table below.
;
; All data after the '|' character is considered part of the string
; to be transmitted
;
; Channel_Number Lower_Limit Upper_Limit |Data to Transmit
;
; Channel number is a 1-3 digit DMX channel value, range is [1 512]; The serial commands are
tied to this channel.
;
; Lower and upper limit refer to the range of DMX channel values to which; an RS232 string will
respond.  Often users like to have a range (say, 5% - 10%); or only trigger above a specific
value.  Value valid range is [1 255]
; Each command needs a valid lower and upper range
;
; Finally, add the desired serial string:; Characters which aren't human readable and editable
in a
; text editor (such as carriage return, line feed, STX,; ETX and so forth, are defined by the
escape sequence $AB, where AB is a two digit; hex representation of a single byte to be
transmitted. Valid range of AB is [00 FF]; Leading zeroes may be required, depending on desired
value.
;
;
; The protocol is not case sensitive.  $AB == $ab == $Ab == $aB
;
; Strings made entirely of hex characters can also be built using the abovementioned; escape
sequences.  Here's how to send the five bytes $02 $04 $10 $22 $38:
;
; $02$04$10$22$38
;
; If the $ character itself needs to be transmitted as part of a serial; string, it must be
encoded as the hex character $24.
;
```

```
; $02$04$10$24$38$2F
;
;The actual DMX to RS232 Patch Chart begins here:
;
;
; Send 'Hello World' and a carriage return one time,
; when channel 3 is between 225 and 255:
;3 10 128|Hello World$0D
;3 129 255|Second String Here!$0D
;
; Send 'Projector Off" when channel 4 is at any value greater than 5:
;4 5 127|ProjectorOff$0D
;4 128 255|Projector On$0D
;
; Send six hex bytes when channel 5 is between 100 and 200
;
;5 100 200|$00$01$02$03$23$24$25$26$27
;
; etc etc etc
; Sample files for Barco:
;
;; power on off
;15 96 127|{"jsonrpc":"2.0","method":"system.poweron","params":{}};15 128
159|{"jsonrpc":"2.0","method":"system.poweroff","params":{}}
;
;Zoom in/out
6 1 255|{"jsonrpc":"2.0","method":"optics.zoom.stepforward","params":{"steps":200}}1 71
80|{"jsonrpc":"2.0","method":"optics.zoom.stepreverse","params":{"steps":200}}
;
;Laser projector - Test Commands
;Assign DMX channels 10 and 11 at 95%+ to trigger on and off
10 250 255|*power = 0$0D
11 250 255|*power = 1$0D
; Assign DMX channel 12 to trigger on and off throughout the range:
12 10 127|*power = 0$0D
12 128 255|*power = 1$0D
```

Purchasing

If you've landed here via a web search, don't already own one of these systems, but would like to add one to your collection, our online store is here:

www.response-box.com/gear/shop

And our main site is here, which includes links to distributors, etc:

www.response-box.com/gear