
The RS-232 Driven DMX Engine

Simple Yet Powerful Lighting Control

Firmware 5.04, October 2016



Table of Contents

Simple Yet Powerful Lighting Control	1
Table of Contents	2
Firmware Change Summary	3
System Overview	4
System Overview	5
System Specifications	6
Updating Firmware - Windows PC	7
Set Startup Scene & Timeout Delay	8
Startup	10
Building a New Scene	12
Adding to a New Scene	14
Storing a Scene in Permanent Memory	16
Recalling a Stored Scene	19
Jog a Channel	21
Streaming Mode	22
Clear All Channels, No Delay, No Fade Time	24
Reset	24
Set System Baud Rate	24
Query Current DMX Values	25
Halt	26
Simultaneous Fade Speeds in One Command	27
Automatic Color Wheel Cycle Effect	28
Group Fade	30
Debug Verbosity	31
Selectable DMX Output Speed	32

Firmware Change Summary

5.04

- Add power-on start scene and delay
- Add DMX standard and 'slow' output speed, switchable
- Add automatic RGB color wheel driver
- Add group fade command
- Update: dimming engine calculates levels at 100 frames second
- Update: fix error when processing 'mask' scene recall command where not all values were correct

4.85

- Add simultaneous, multi-speed fading
- Add Halt command and feedback
- Add 'commend ignore'
- Add verbose debugging

4.07

- Baud rate checking, system forces to 9600 after reboot if a nonstandard baud rate is specified.
- Add command for software reboot
- Add command to pause mid-fade
- Add command for jogging individual DMX channels up and down
- Add command for 'scene masking' when pre-stored scenes are recalled
- Add command for verbose debug / serial feedback

4.05

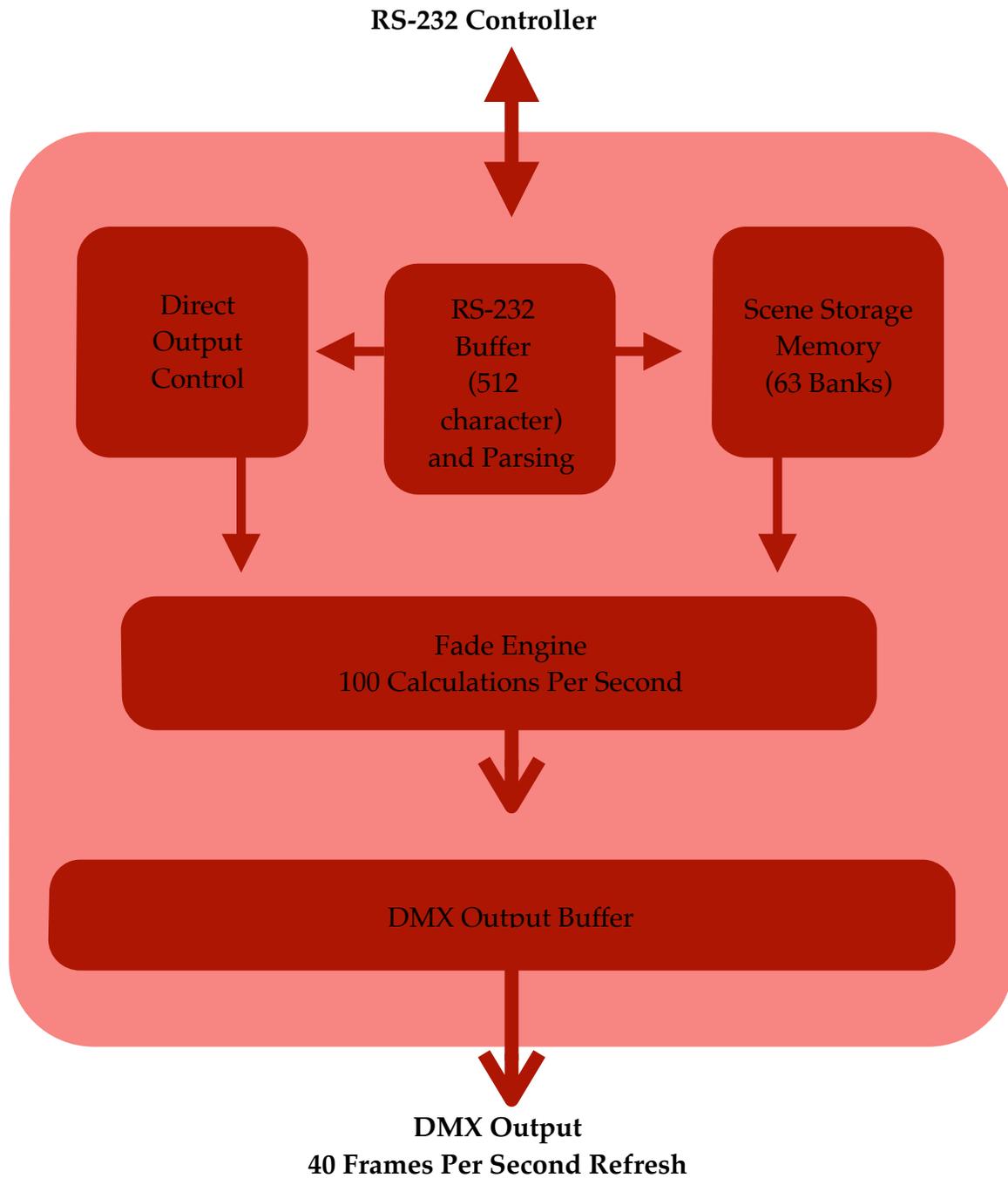
- Add scene storage
- Add scene recall
- Add baud rate selection
- Add channel query
- Add 'layer' A command for building a scene channel by channel using multiple commands

2.00

- Original release

System Overview

The RS-232 DMX Engine is a self-contained DMX controller. It accepts simple, human-readable commands via RS-232 and uses these commands to generate, recall and crossfade between DMX scenes.



System Overview

The DMX Engine accepts RS-232 commands at any standard baud rate. Incoming serial data is buffered in the system's memory until a carriage return is received, which signifies the end of a command.

Once the command has been received, it is parsed. Commands containing syntax errors are ignored, and an error message is transmitted back to the RS-232 host. Valid commands are processed immediately.

Commands can contain information directing the system to

- Set DMX channels to certain levels.
- Store scenes (a snapshot of the DMX output buffer) in memory.
- Recall scenes from memory using a very simple command syntax.
- Fade between different scenes. The system automatically calculates all intermediate DMX channel values, and requires only a crossfade time from the end user.
- Report the state of any DMX channel back to the user via RS-232.

More significantly, commands may be processed simultaneously, with the only limit being (a) the RS-232 port speed and (b) the 512-byte receive buffer. This means that, for example, a group of channels can fade between levels at one rate while a second group of channels can fade to different levels at a completely different rate.

Users of high-end theatrical lighting consoles may recognize this feature as 'simultaneous cue stacks,' and it adds a great deal of flexibility to the system's operation.

System Specifications

Power Supply

9-15v DC, 200 mA, center positive. Barrel size is 2.1 x 5.5 mm. System ships with a switching power supply, 9V DC output, 300 mA, 80-240V AC input. It will work worldwide, though an adapter for the two-pin American-style plug may be required. Check a travel shop, or ask at a hotel's front desk if they have any in Lost & Found.

DMX Output

Standard XLR pinout per USITT. Neutrik XLR jacks.

- Pin 1 ground
- Pin 2 Data -
- Pin 3 Data +
- Pins 4 & 5, of XLR5 jack is installed are not connected
- Each DMX output is driven by its own transmitter.

Serial Command Protocol

Baud Rate: 9600, 19,200, 38,400, 57,600 or 115,200, 8N1, user selectable. Default is 9600.

- Pin 2 is system transmit
- Pin 3 is system receive
- Pin 5 is ground

Pin 4 (DTR) is used to update the system's firmware. This allows new features to be added in the field, without requiring the hardware to be shipped back to our shop. If asserted (set high) by the PC or other controlling equipment, the system will wait in 'reset' mode for new firmware and the system will neither accept serial commands nor output DMX. In normal use, either physically disconnect this wire or ensure that DTR is cleared.

Depending on which external devices is sending the RS-232 data, a crossover cable (which swaps pins 2 & 3) *may* be required. When connected directly to a PC, a 'straight' cable works properly.

In the following pages, the text [cr] is used to represent a single data byte, the carriage return. This value is decimal 13 or hex 0x0D.

Many screenshots show data sent / received through a terminal program, as well as a screenshot from a DMX monitoring program. Data sent from the PC to the system is displayed in green text, and data generated by the system is yellow. Thus, system output can be easily compared with RS-232 input.

Updating Firmware - Windows PC

From time to time new firmware may be released. Follow these steps to update the DMX Engine:

You should have been provided these three files, or perhaps a .zip file containing them:

- Click Me to Update, a wWindows batch file
- Updater, a Windows .exe
- XXX.eeprom, the firmware file

Save these three files to your desktop or in a folder where they can be easily accessed.

Connect DMX engine to the PC using a straight serial cable. Pin 4 (DTR) must be present for the firmware to load. Add power to the DMX engine using the included wall supply.

Double click on the 'Click Me to Update' file. Update takes about ten seconds. You may need to authorize Windows to allow this file to run. In rare instances, right-click on the batch file and choose 'Run as Administrator.'

System will restart, and current firmware version is displayed through the serial port at startup. When the engine is running, the DMX LED will pulse rapidly.

Set Startup Scene & Timeout Delay

It may be useful for the DMX engine to auto-load a stored scene immediately after a power cycle. Often a control system (Crestron, Control4, etc) may take tens of seconds, or even several minutes, to restore its state after an unplanned outage. For this reason, the DMX Engine can load a pre-saved scene immediately after power is applied.

UX,T[cr] ← set startup scene and delay time, if needed.
U?[cr] ← query the system for current startup scene

Where

- U is a capital U
- X is the pre-saved scene number, [0 60]. Leading zeros not required.
- If X = 0, no scene will load and system will run normally. The default startup behavior is to set all channels to zero. If X is set to zero, T should be zero as well.
- X is followed by a comma [,]
- T is the approximate time, in seconds, the system will wait after startup to load the scene. DMX output will be live and the system will respond to commands during this period. If any regular serial command is received during this time, the startup scene will not be recalled. Value here is [0 255], leading zeroes not required.
- [cr] is a carriage return, hex \$0D or decimal 13.
- U?[cr] queries the startup scene status, and an ASCII string is returned to the user.

Notes:

- Prior to defining a startup scene, the scene should be built and stored using the regular 'F' or 'A' commands, then stored with 'S'. See here for a PDF file containing the full protocol description.
- An uninitialized scene will probably / possibly default to 'all channels on.' This is because a raw-from-factory memory chip is loaded with '255' or '0xFF' as values. So be careful what you choose to load without first pre-programming.
- The timing value 'T' is approximate and may vary by +/- 10% or so.

Example:

Load scene #4 immediately after a power cycle:

U4,0[cr]

Load scene #1 five seconds after a power cycle

U1,5[cr]

Change to zero startup scene. All DMX channels are zero until the connected equipment (Crestron, AMX, Control4, etc) sends commands:

U0,0[cr]

Also at startup, the system transmits a 'splash screen' through the RS232 port. This confirms proper system operation, and also displays the current firmware revision.

As shipped from the factory, the system defaults to 9600 baud, 8 bits, no parity, 1 stop bit. It's designed to connect directly to a PC or other source hardware. This means that a standard 'straight' serial cable will work properly for initial testing. Depending on the system pinouts of other control hardware being used (ie Crestron, Elan, etc) a crossover cable may also be required.

The following pages will discuss the process of building, storing and recalling a DMX scenes.

Building a New Scene

`FXXX@YYY:TTT[cr]`
`FXXX@YYY,AAA@BBB,CCC@DDD:TTT[cr]`

- F is a capital F
- @ is the ASCII 'at' character, hex 0x40
- XXX, AAA, CCC are three digit DMX channel numbers with range [001 512]. As only channels 1-512 exist in a typical DMX universe, channel 000 can be used to select all channels.
- YYY, BBB, DDD are three digit channel values with range [000 255] (: is the ASCII colon character, hex 0x3A)
- TTT is a three digit time value, in tenths of a second, range [000 999]
- [cr] is the carriage return character, decimal 13 or hex 0x0D.

For 'F' commands, the channel:value information contained in each string *replaces* everything in the active DMX output buffer. This buffer is refreshed ~40 times per second, regardless of incoming serial data. If a channel is not specifically mentioned in an F command, it is set to zero when the command is executed.

For example, following the completion of these (one with a zero fade time, one with 2.4 seconds and one with 1.0 second crossfades):

`F001@100:000[cr]`
`F010@128,011@127,012@126:024[cr]`
`F007@010:010[cr]`

... the DMX output buffer would have the following values:

CH1 - CH6: 0
CH7: 010
CH8-CH512: 0

NOTE: ANY CHANNEL NOT MENTIONED in an 'F' command is set to zero. Consider using the 'A' command when building up a scene channel-by-channel.

Adding to a New Scene

AXXX@YYY:TTT[cr]
AXXX@YYY,AAA@BBB,CCC@DDD:TTT[cr]

- A is a capital A
- XXX, AAA, CCC are three digit DMX channel numbers with range [001 512]
- YYY, BBB, DDD are three digit channel values with range [000 255]
- TTT is a three digit time value, in tenths of a second, range [000 999]

For 'A' commands, the channel:value information contained in each string *adds to* data in the DMX output buffer. Higher values take precedence. Thus, lighting data may be added to existing scenes a few channels at a time. In a photo editing application, this process would be similar to adding layers to an image.

If a channel is not specifically mentioned in an A command, its value remains the same as it as before the command was processed.

F001@100:000[cr]
A002@255:000[cr]
A010@128,011@127,012@126:024[cr]
A007@010:010[cr]

... the DMX output buffer would have the following values:

CH1 : 100
CH2 : 255
CH10 : 128
CH11 : 127 CH12 : 126 CH7 : 10

For example, following the completion of these commands:

```
F001@100:000[cr]  
A002@255:000[cr]  
A010@128,011@127,012@126:024[cr]  
A007@010:010[cr]  
M22[cr]
```

... the DMX output buffer would have the following values: CH1 : 100

CH2 : 255

CH10 : 128

CH11 : 127 CH12 : 126 CH7 : 10

...

and memory bank #22 may be recalled at any time in the future.

Recalling a Stored Scene

SXXX:TTT[cr]

- S is a capital S
- XXX is the desired scene number, valid range is [1 063]
- TTT is the scene crossfade time, valid range is [000 999] tenths of a second

Up to 63 DMX scenes may be saved in memory. This memory is permanent and will survive a power cycle. Scenes may be recalled from memory and faded in over a specific time. Scene recall commands *overwrite* the DMX output buffer with stored data. Using this basic command, all channels are changed from their 'live' value to that which was stored in memory.

Example: recall stored scene #1 instantly:

S001:000[cr]

Example: crossfade from the current DMX output buffer with the contents of scene 32 over 5.7 seconds:

S032:057[cr]

However, in some instances it is desirable to recall a stored scene, but only apply it to part of a DMX universe. For example, consider a three-room home automation installation. Each room uses 15 channels of DMX for RGB accent lighting. It might be useful to recall a scene but only apply that data to a specific range of DMX channels. One room can change color while leaving the other rooms unaffected.

For that purpose, the S command can be overloaded with mask values:

SXXX:TTT,L,H[cr]

In this case, L and H represent the lower and upper bounds of the DMX channel range to be copied from system memory to the constantly repeating DMX output buffer. Other valid commands could include:

Recall scene six, three second crossfade.

S6:030[cr]

As above, but only copy channels between 10 and 30 from memory to the DMX output buffer:

S6:030,10,30[cr]

As above, but only copy channels 100-150 from memory to output. All other channel information carries through. Zero fade time.

S6:000,100,150[cr]

Jog a Channel

Individual DMX channels may be jogged up and down. In this mode, it is not required to know the channel's exact value. Rather, it can be added to or subtracted from in arbitrary amounts, up to the limits of 0 or 255. This command may be useful when implementing bump buttons on a human interface control panel.

```
JN:X[cr]  
JN:-X[cr]  
JN:+X[cr]
```

- J is a capital J
- N is the DMX channel number, range is [1 512]
- X is a decimal number, either positive or negative
- [cr] is a carriage return.

Streaming Mode

In this mode, it's possible to transmit data directly to the DMX output buffer, with a bare minimum of protocol overhead.

The first step is to tell the system how many channels will be transmitted. This is called the 'streaming mask.' Since it may not be necessary or even desirable to control an entire universe at one time, a selection of channels may be chosen. The command is

```
X:LLL:SSS[cr]
```

- X is a capital X
- LLL is a three digit lower-bound of the DMX range to control. Leading zeroes should be included.
- SSS is the size of the mask, in DMX channels.

For example, to control channels 1-10

```
X:001:10[cr]
```

... and to control channels 1-300

```
X:001:300[cr]
```

To control channels 30-45

```
X:030:15[cr]
```

A query command can also be sent:

```
X?[cr]
```

The system will return the current channel mask. Default is [1 512] at powerup, and masks are not saved through a power cycle.

Once the range has been set, data can be sent directly to the serial port. Some care is required on the part of the user, however. A set of streaming data begins with the character 'x' and then is followed by binary bytes, the number of which corresponds to the channels being controlled.

When 'x' is received, the system begins counting subsequently-received bytes and storing them in a temporary buffer. When the appropriate number of bytes have been received, the temporary buffer is copied to the DMX output buffer. DMX channels which fall outside the streaming mask are unaffected.

Clear All Channels, No Delay, No Fade Time

C[cr]

Reset

I[cr]

Initiates a software reboot, which has the same effect as a physical power cycle.

Set System Baud Rate

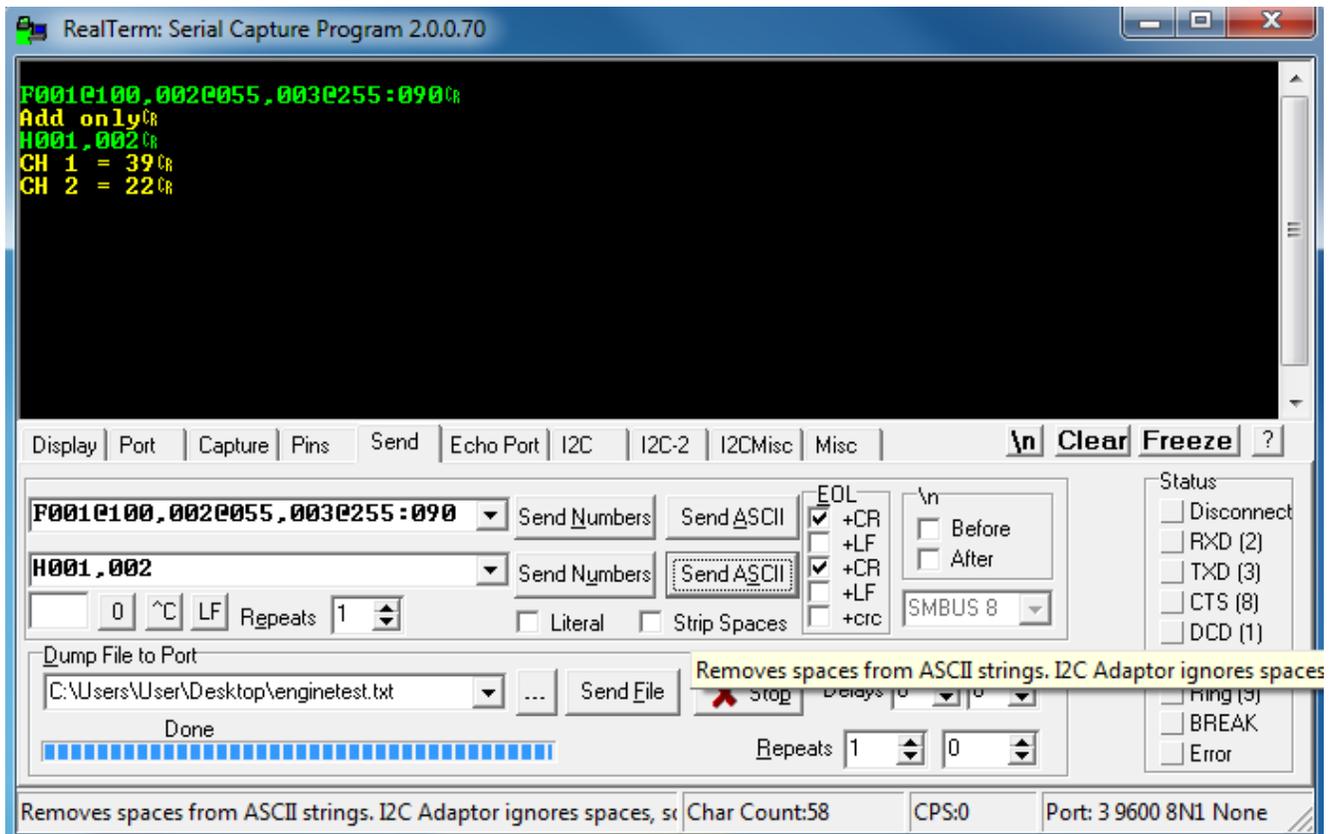
B9600[cr]
B19200[cr]
B38400[cr]
B57600[cr]
B115200[cr]

Change takes place after a power cycle or software reset.

Halt

Hx[cr]
Hx,y,z...[cr]

The halt command pauses specific channels mid-fade. It also returns the value of the paused channels in an easy-to-parse format. This command use useful when building human interface panels. Channels can be set to fade slowly to a certain point, and the operator can pause or stop the fade when the desired levels are reached.



Simultaneous Fade Speeds in One Command

`ZXXX@YYY:T111,ZAAA@BBB,T22,ZCCC@DDD:T33[cr]`

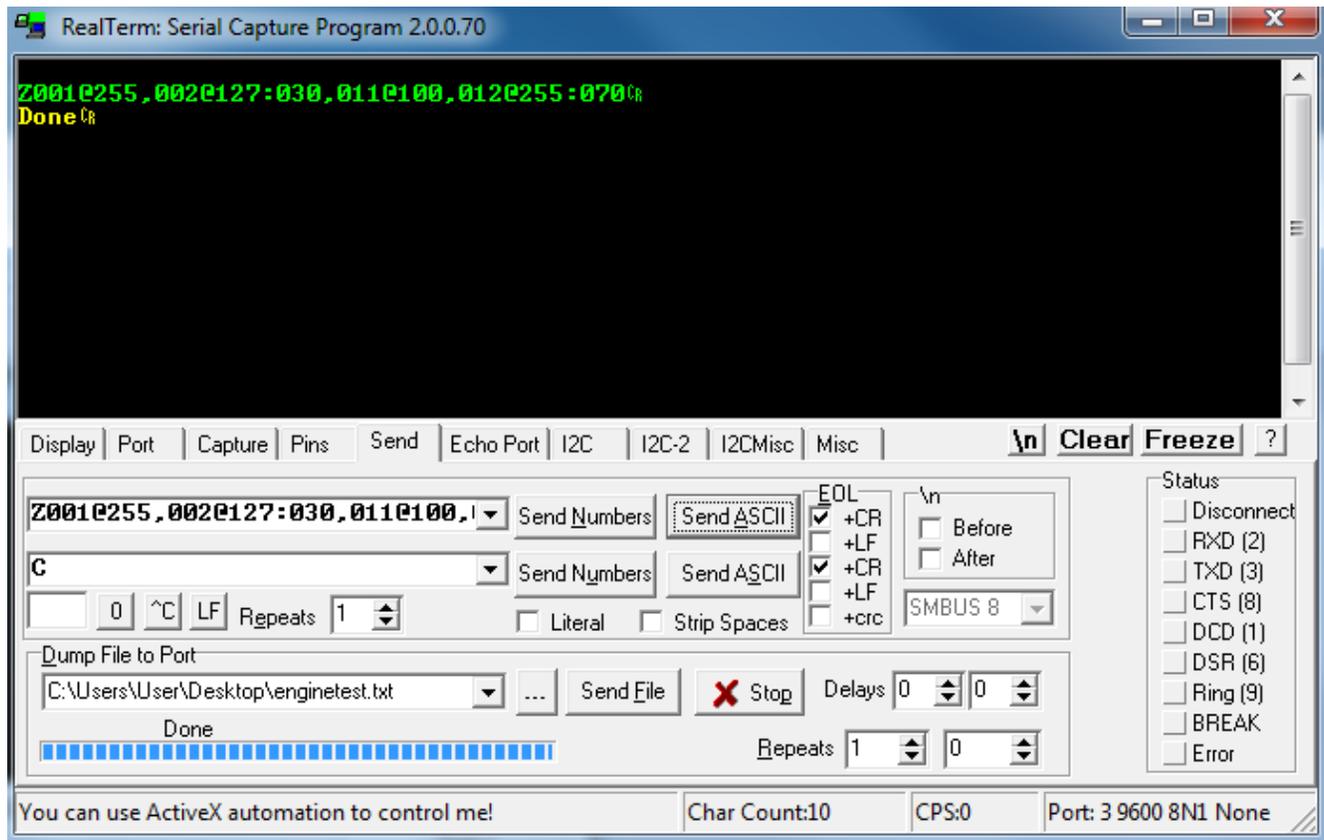
Although the system can execute commands simultaneously and in parallel, some designers requested the ability to send fades at multiple speeds in a single command. Z command acts similarly to the A command, in that the DMX output buffer is not completely overwritten.

Channel numbers, channel levels and fade speeds *must* be 3-digit decimal values. Leading zeros may be required.

Note that the RS-232 receive buffer is 512 characters long.

This command sets channels to levels at three different and simultaneous speeds:

`Z001@255,002@127,003@064:010,004@127,005@255:030,
011@100,012@127,013@200,019@255:070[cr]`



Automatic Color Wheel Cycle Effect

This set of commands allows the DMX Engine to automatically cycle through an RGB color wheel, at a user-selectable speed. The wheel can be stopped at any time, or run indefinitely, and the DMX channels queried or an entire scene saved. This makes it easy for users to choose colors, or perhaps would allow a designer to run a rolling color cycle as a background effect, etc.

The color wheel has 512 steps total. Since the DMX Engine is generating all of the color intensities internally, the system controller (Control4, Crestron, etc) is free to perform other tasks.

Up to 20 x 3-channel RGB LED fixtures can be controlled simultaneously. This firmware release doesn't specifically support LED fixtures which contain a 'master intensity' control, etc. However, those DMX channels can certainly be set on a channel-by-channel basis if needed, and then the color cycle can run on top of them. DMX channels not specifically mentioned in the RGB arrays (see below) are unaffected.

Commands:

WRa,b,c,d[cr]	← Define the red array of channels, separated by commas.
WGa,b,c,d[cr]	← Define the green array of channels, separated by commas.
WBa,b,c,d[cr]	← Define the blue array of channels, separated by commas.
WTa[cr]	← Set the color wheel cycle time, in seconds.
WX[cr]	← Execute (start) the color wheel
WS[cr]	← Stop the color wheel
WC[cr]	← Clear color arrays, to allow reloading or change
W?[cr]	← Query the color system for DMX channels assigned to each color

These commands do not survive a power cycle, so they must be generated for each use. First, set an array of DMX channels corresponding to the red LEDs in each fixture. For example, assume six fixtures which are sequentially addressed. Their start addresses would be 1, 4, 7... Each fixture uses three channels, one each for red, green and blue. The arrays only need to be defined once per power cycle.

Example:

First set the red channels:

```
WR1,4,7,10,13,16,19[cr]
```

Then set the green channels:

```
WG2,5,8,11,14,17,20[cr]
```

Then set blue:

```
WB3,6,9,12,15,18,21[cr]
```

Set the cycle time (for example, 12 seconds) with the command

```
WT12[cr]
```

Once this is done, the color wheel can be started and stopped with

```
WX[cr]
```

```
WS[cr]
```

If the color wheel needs to be applied to one group of channels, and then a second group, the array can be set, stored and then re-set:

```
WR1,4,7,10,13,16,19[cr]
```

```
WG2,5,8,11,14,17,20[cr]
```

```
WB3,6,9,12,15,18,21[cr]
```

```
WT8[cr]
```

```
WX[cr]
```

... wait for the correct color to appear

```
WS[cr]
```

... store or otherwise manipulate the DMX scene

```
WC[cr]
```

```
WR22,25,28,31[cr]
```

```
WG23,26,29,32[cr]
```

```
WB24,27,30,33[cr]
```

```
WX[cr]
```

... wait for the new color to appear

```
WS[cr]
```

... store or otherwise manipulate the DMX scene

Group Fade

Occasionally it is useful to set a large group of DMX channels to the same level. While this can be accomplished using the standard 'F' or 'A' commands, the new 'G' command allows the channels to be adjusted in a less verbose way.

GA-B@C:T[cr]

This command sets channels in range [A B] to value C and crossfade time T. A and B are valid between [1 512]. Leading zero is not required. C is the channel value, range [0 255]. T is the time, in tenths of a second, for the crossfade to take place. Range for T is [0 999] or 99.9 seconds maximum.

Example: Set DMX channels 37 - 126 to 50% over 7.6 seconds:

G37-126@127:76[cr]

GA-B/S@C:T[cr]

As above, but the slash and S allows for a group of channels in a particular pattern to be set. For example, if a group of RGB fixtures had sequential start addresses (1, 4, 7, 10...) the red channel in each fixture could be controlled easily, skipping through the array.

Example:

5 RGB fixtures are sequentially connected, and the first start address is 100. Turn on all the green and blue elements over 2.5 seconds:

G101-114/3@255:25[cr]
G102-114/3@100:25[cr]

When this command finishes, channels will have the following levels:

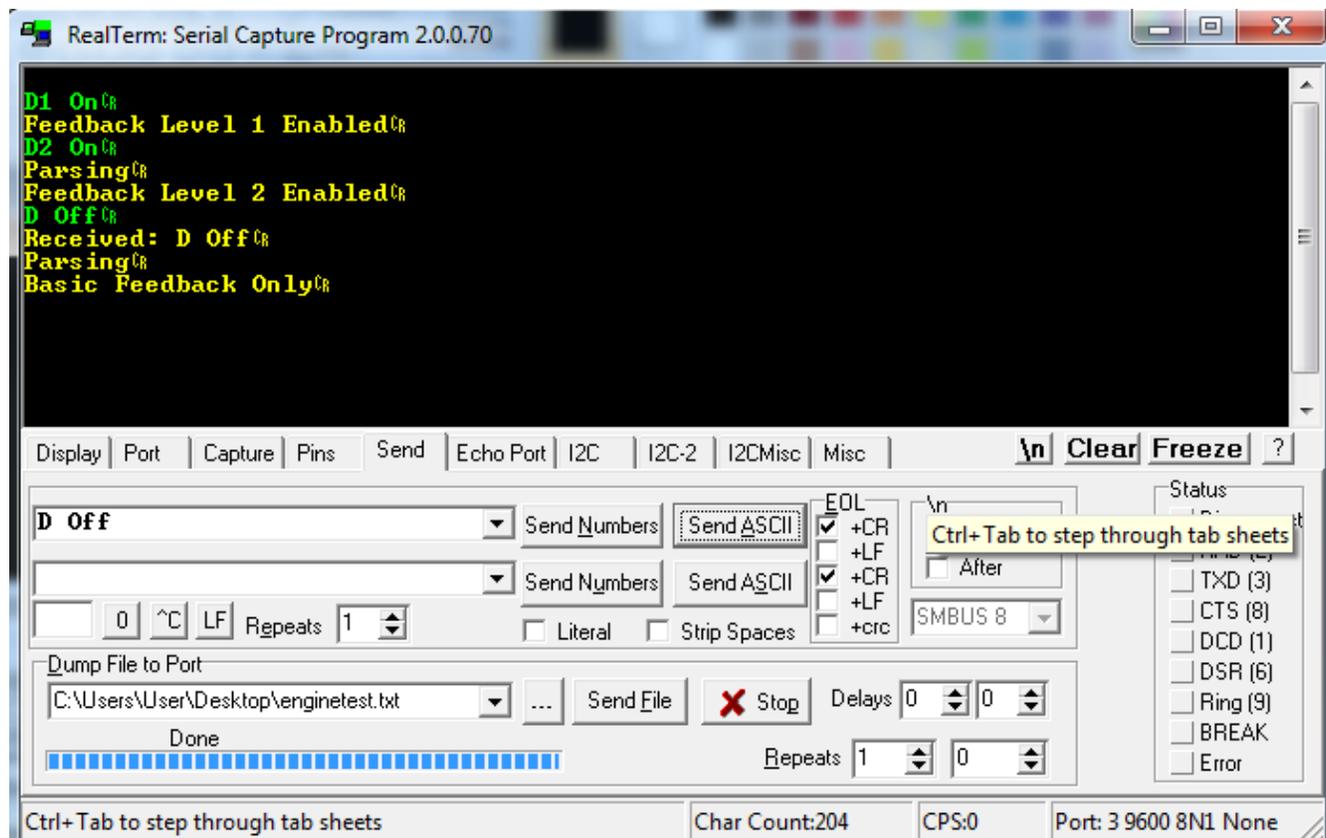
Fixture #	Channel	Value (Red)	Channel	Value (Green)	Channel	Value (Blue)
1	100	0	101	255	102	100
2	103	0	104	255	105	100
3	106	0	107	255	108	100
4	109	0	110	255	111	100
5	112	0	113	255	114	100

Debug Verbosity

The system generates feedback through the RS-232 port. This feedback can be used to troubleshoot a program if needed. There are multiple levels of feedback available.

```
D1 On[cr]
D2 On[cr]
D Off[cr]
```

The debug value defaults to D1 (simple feedback) after a power cycle. D2 is more verbose, and can be toggled on or off as needed.



Selectable DMX Output Speed

By default, the DMX engine transmits full-speed, full-frame DMX, and speeds close to the maximum available by the USITT spec.

Unfortunately, we've learned that some DMX LED fixtures, particularly Amazon / eBay / Alibaba 'bargain bin' specials, have a hard time keeping up with this datastream.

To generate DMX with relaxed packet and inter-byte timing, ~ 15 frames per second, send the command:

`Slow[cr]`

For full frame DMX at ~40 frames per second, send the command:

`Fast[cr]`

The output speed command survives a power cycle. Immediately after a command is received, the system will reboot. The reboot process takes 2-3 seconds to complete. The DMX LED on the chassis will flash in sync with DMX packets, providing additional user feedback of the current setting.

For a more in-depth discussion of this issue, please visit

www.response-box.com/gear/decabox-dmx-slowdowner/