

# **SPROG DCC Decoder Programmer**

Operating Manual  
Firmware Version 3.4  
April 2004

© 2004 Andrew Crosland

web: <http://www.sheerstock.fsnet.co.uk/dcc/sprog>

e-mail: dcc@sheerstock.fsnet.co.uk

## **Disclaimer**

You build, test and use SPROG entirely at your own risk. No responsibility is/can be accepted for any loss or damage whatsoever arising from any attempt to build, test or use any item described in this document.

## Table of Contents

1	Introduction.....	2
2	Firmware Overview.....	2
2.1	Basic Operation With Terminal Emulator.....	2
2.1.1	Command Set Summary .....	2
2.1.1.1	General Commands .....	3
2.1.1.2	Programmer Commands .....	5
2.1.1.3	Rolling Road Tester Commands .....	5
2.1.1.4	Bootloader Command.....	6
2.2	Operation With DecoderPro Software .....	6
2.3	Upgrading SPROG Firmware .....	6
3	Useful Links.....	7

## 1 Introduction

This document describes the version 3.4 firmware for the SPROG DCC decoder programmer.

Documentation for DecoderPro is available from the DecoderPro web pages <http://jmri.sourceforge.net/>

The latest version of this, and other, documents can be downloaded from the SPROG web pages <http://www.sheerstock.fsnet.co.uk/dcc/sprog>.

## 2 Firmware Overview

SPROG uses a PIC microcontroller to translate commands sent over the USB or RS-232 interface into DCC packets to be driven by a power MOSFET output stage configured as a H bridge. The PIC forms an integral part of the booster, generating the correct switching waveforms for the MOSFETs and sensing the output current through its internal A/D (analogue-to-digital) converter.

A regular interrupt sets the DCC timing at 58us for a '1' half-bit and 116us for a '0' half-bit. During each period of interrupt processing the next DCC bit is generated. The A/D converter is read regularly and set to do a new conversion and the UART is polled to check for new data received from the USB or RS-232.

Output current is sensed by a small value resistor in the ground of the H-bridge. The voltage developed across this resistor is filtered and sampled by the PICs A/D converter to detect programming acknowledge pulses and short circuit conditions. A number of readings are taken and averaged to avoid false overload detection due to noise.

SPROG may be operated with a terminal emulator program such as windows HyperTerminal, see 2.2.

The preferred method of operation is to use DecoderPro, see 2.2.

### 2.1 Basic Operation With Terminal Emulator

#### 2.1.1 Command Set Summary

All commands must be entered on a single line terminated by carriage return. Maximum input line length is 20 characters, including carriage return. Format of parameters is dependant upon the command. The maximum number of parameters on any line is 6.

*Commands in italics are either unimplemented or implemented but not finalized. The format of input parameters or output may change.*

#### General Commands

**M [n]** – Display [Set] operating mode  
**R** – Read mode from EEPROM  
*S* – *Display Status*  
**W** – Write mode to EEPROM  
**Z [n]** – ZTC Compatibility Mode  
**?** - Display Help  
**ESC** - immediately shutdown power to track

#### Programmer Commands

**C CV [Val]** - Read [program] CV using direct bit mode  
**V CV [Val]** - Read [program] CV using paged mode

#### Rolling Road Tester Commands

**A [n]** – Display [Set] Address  
*F n* - *Toggle Function output (not yet implemented)*  
**O byte [byte] [byte] [byte]** - Output bytes as DCC packet.  
 +- Track power on  
 -- Track power off  
 <[step |<] - Reverse speed step[s]  
 >[step |>] - Forward speed step[s]

#### Bootloader Command

**B a b c** – Start Bootloader

#### Input Format

Input values are always parsed as decimal, unless overridden with 'b' or 'h' prefix for binary or hexadecimal, respectively. E.g. h15 is equivalent to 21 decimal.

#### Acknowledgement Messages

CV values are given in hexadecimal.

Message	Meaning
!O	Overload
!E	Error
No-ack	No acknowledge pulse received during programming
OK	Programming operation completed

**Table 4 Acknowledgement Messages**

### 2.1.1.1 General Commands

**M** - Display operating mode

**Mn** - Set operating mode to n

The mode word, n, is interpreted as a 16 bit binary value with each bit corresponding to a particular feature, as shown in Table 5.

Bit	Name	Feature
0	spare	Do not use, always set to 0 for future compatibility
1	ECHO_ON	SPROG echoes all received characters if this bit is set
2	spare	Do not use, always set to 0 for future compatibility
3	CALC_ERROR	Set to calculate error byte for O command. If clear then error byte must be supplied on the command line
4	RR_MODE	Set for rolling road/test mode
5	ZTC_MODE	SPROG uses modified DCC timing for older ZTC decoders
6,7	spare	Do not use, always set to 0 for future compatibility
8	DIR	Direction for rolling road/test mode and booster mode. Set for reverse
9	SP14	Select 14 speed step mode for rolling road/test mode and booster mode.
10	SP28	Select 28 speed step mode for rolling road/test mode and booster mode.
11	SP128	Select 128 speed step mode for rolling road/test mode and booster mode.
12	LONG	Use long addresses in rolling road/test mode and booster mode
13-15	spare	Do not use, always set to 0 for future compatibility

**Table 5 Mode Word Bits**

*Examples required*

See 'Reset' for a description of the reset state of the mode word.

**R – Read Mode from EEPROM**

Read a previously saved operating mode from EEPROM, see M command.

**S – Display Status**

Output SPROG status, including A/D (Analogue-to-digital converter) readings and throttle selector inputs.

A/D readings are output as 4-digit hex values in the order:

AN0: Vin sense input

AN1

\*\*\*

**W – Write Mode to EEPROM**

Write current operating mode to EEPROM, see M command.

**Z [n] – ZTC Compatibility Mode**

Some older ZTC decoders (e.g. ZTC401) require modified DCC timing.

Z 0 – return to normal DCC timing

Z 1 – Enable ZTC compatibility mode.

#### **? - Display Help**

Displays the SPROG firmware version.

SPROG Ver 3.4

>

#### **ESC - Shutdown**

DCC output is turned off immediately.

### **2.1.1.2 Programmer Commands**

**C CV** - Read a CV using direct bit mode

**C CV Val** - Program a CV using direct bit mode

**V CV** - Read a CV using paged mode

**V CV Val** - Program a CV using paged mode

If no VAL value is given, then this command reads the specified CV and displays it in the form CV <hexadecimal>, otherwise writes the value Val to the specified CV.

### **2.1.1.3 Rolling Road Tester Commands**

#### **A – Display Address**

#### **A n – Set Address**

Display or set the decoder address (decimal) to be used in speed/direction packets. If a new address is set then current speed step will be reset to zero. This command does not perform any programming of decoder CVs.

#### **I – Display output current limit**

#### **I n – Set output current limit**

*Display or set the booster stage output current limit for rolling road/test mode and booster mode.*

*Not implemented yet.*

#### **F n - Toggle Function output**

*Send DCC packet to toggle the state of function output Fn.*

*Not implemented yet.*

#### **O byte [byte] [byte] [byte] - Output bytes as DCC packet.**

Any arbitrary DCC packet may be generated using this command. SPROG will add the correct pre-amble bits, start bits, and error byte. Note that all address and data bytes and, optionally, the error byte must be given on the command line, this command does not use the address set by the 'A' command. If the mode word CALC\_ERROR bit is set then SPROG will calculate the correct error byte which must not be given on the command line. If CALC\_ERROR is not set then the error byte must be given on the command line, allowing erroneous packets to be generated for decoder testing.

#### **+ - Track power on**

Turn on track power and check for overload condition after 100ms. Twenty-four reset packets will be transmitted. At all other times when there is no DCC data being transmitted, DCC pre-amble will be transmitted.

#### **-- Track power off**

Turn off track power.

#### **<<[<] - Reduce/Reverse speed step[s]**

#### **>>[>] - Increase/Forward speed step[s]**

Adjust speed step relative to current speed. If decoder is running in reverse then Reduce/Reverse will increase the reverse speed and Increase/Forward will decrease the reverse speed. If decoder is running forward then Reduce/Reverse will decrease the forward speed and Increase/Forward will increase the

forward speed. Increment or decrement is determined by the number of '<' or '>' characters in the command. Speed will not increment/decrement past maximum forward or reverse speed step nor through zero. The current speed step will be reported after performing this command:

< – **display Reverse speed step**

< **step** – **Set Reverse speed step**

> **display Forward speed step**

> **step** – **Set Forward speed step**

Set forward or reverse speed step directly.

### **2.1.1.4 Bootloader Command**

#### **B a b c – Start Bootloader**

Exactly three arguments, a, b, c, must be given with the B command but their values are not checked. This helps prevent inadvertent issuing of the b command. The B command enters the bootloader ready to receive updated SPROG firmware. See 2.3 for full details of how and when to use the bootloader.

## **2.2 Operation With DecoderPro Software**

DecoderPro is a program written in the Java programming language, allowing it to be used on most popular operating systems. It supports a wide range of DCC hardware and Version 1.1.6 and above includes support for SPROG.

To use SPROG with DecoderPro you must first ensure that character echoing is disabled by clearing bit 1 of the mode word (see description of M command) using a terminal emulator to connect to SPROG. Then issue a W command to save the mode word in EEPROM. The default state of the programmed PICs supplied in SPROG kits is correct for operation with DecoderPro.

You must have the Java runtime environment installed on your computer to use DecoderPro and it must be installed before DecoderPro is installed.

- Connect SPROG and power up
- Start DecoderPro
- The first time you run DecoderPro the preferences window should open. If not it can be found under the DecoderPro Edit menu
- Select SPRG as the layout connection and specify the COM port to which it is connected
- Save your preferences, exit and restart DecoderPro
- Click on “Use programming Track”
- Select the decoder type to be programmed and click “open programmer”
- Select the programming mode either paged or bit direct depending on the decoder being programmed

You are now ready to edit CVs, speed tables, function mapping, etc., for more information see the DecoderPro web pages.

## **2.3 Upgrading SPROG Firmware**

Occasionally it will be necessary to release new versions of the SPROG firmware, the program that is programmed into the PIC micro-controller, in order to fix bugs or add new features. SPROG includes a feature known as a bootloader which makes this easy to do without requiring any special programming hardware or removing the PIC. New firmware will be supplied as a text file (a .hex file), available from the SPROG homepage.

Normally the bootloader would be started using the B command. If a SPROG firmware download fails or is corrupted somehow then SW100 may be used in conjunction with the reset switch to start the bootloader. Hold SW100 closed whilst resetting SPROG to start the bootloader.

SPROG needs to be connected to a terminal emulator program that can transmit the text file, e.g. Windows HyperTerm. Enter 'B 1 1 1<return>' to start the bootloader, you should see the bootloader prompt:  
L>

Use the terminal emulator to transmit the .hex file. As each line of the file is received, SPROG will decode the data and re-program it's internal program memory. It is not possible to re-program the bootloader portion of the memory in this way. When the download of the new firmware is complete, reset the SPROG to resume normal operation. The previous setting of the mode word will be retained after a firmware upgrade.

If the bootloader operation is interrupted or corrupted in some way such that SPROG is left with a corrupt program, it is still possible to start the bootloader by pressing and holding the boot switch, SW100, whilst resetting SPROG.

### 3 Useful Links

SPROG homepage <http://www.sheerstock.fsnet.co.uk/dcc/sprog.htm>

FTDI <http://www.ftdichip.com>

Microchip <http://www.microchip.com>

Model Electronics Railway Group (MERG), <http://www.merg.org.uk>.

Java Model railroad Interface (for DecoderPro) <http://jmri.sourceforge.net/>

Sun Microsystems (for Java) <http://java.sun.com>