Propeller Loader

by David Betz and Steve Denson

May 23, 2012

Table of Contents

2 Introduction 3

3 Common Use Cases 3

3.1 Loading COG or LMM Program 3

3.2 Loading an XMM Program 3

3.2.1 Cache Drivers 4

3.3 Using the SD Loader 4

3.4 Using the SD Cache Driver 5

3.5 Writing a File to the SD Card 5

3.6 Creating a PEX file 5

3.7 Creating a Spin Binary File 5

4 Options 5

4.1 -b <type> Select target board 5

4.2 -D <var>=<value> Define a board configuration variable 5

4.3 -e Write the program into EEPROM 6

4.4 -f Write a file to the SD card 6

4.5 -I <path> Add a directory to the include path 6

4.6 -l Write a program to the SD card and use the SD loader 6

4.7 –P List available serial ports 7

4.8 -p <port> Select serial port 7

4.9 -q Quit on the exit sequence 8

4.10 -r Run the program after loading 8

4.11 –S or –S<n> Slow down the loader by adding a delay 8

4.12 -s Write a spin .binary file for use with the Propeller Tool 8

4.13 -t or –t<baud> Enter terminal mode after running the program 8

4.14 -v Verbose output 9

4.15 -x Write a .pex binary file for use with the SD loader or SD cache 9

4.16 -z Write a program to the sd card and use the SD cache 9

4.17 -? Display a usage message and exit 9

5 Configuration Files 9

5.1 Board Types and Subtypes 10

6 Expressions 10

7 Variable Patching 11

# Introduction

The Propeller Loader, propeller-load, is a command line program you use to load programs generated by the PropGCC toolchain into a Propeller board over a serial connection from a PC running Windows, Mac OS x, or Linux. It can also load Spin binary programs generated by programs like the Propeller Tool or BST.

# Common Use Cases

## Loading COG or LMM Program

Loading a COG (-mcog) or LMM (-mlmm) mode program is done in a single stage using the Propeller chip’s boot loader. The only board configuration parameters that are used for this type of load are the *baudrate, clkfreq,* and *clkmode* settings. The –b option can be omitted if the *default* board configuration is adequate. In other words, if the board uses an 80mhz clock with clock mode XTAL1+PLL16X and a baud rate of 115200 the –b option can be omitted. Also, if the program being loaded makes use of variable patching as described later in this document the board type should be specified in the load command. In general, it’s best to always specify the board type.

Examples

propeller-load –b c3 myprog.elf –r –t

This command loads the program *myprog.elf,* starts it running, and then enters the terminal emulator.

propeller-load –b c3 myprog.elf –e –r -t

This command loads the program myprog.elf, writes it to the EEPROM, starts it running, and then enters the terminal emulator.

## Loading an XMM Program

Loading an XMM (-mxmmc, -mxmm-single, or –mxmm-split) program is done in two stages. The first stage uses the Propeller chip’s boot loader to load a helper program that contains a driver that knows how to write into the target board’s external memory. In the second stage, the loader talks to this helper program to load the XMM program into external memory. A board type is always required for XMM loads since the board configuration file contains the name of the driver to use to access external memory. This is called the cache driver and it is also used once the XMM program is running to allow the XMM kernel to access external memory.

Examples

propeller-load –b c3 myprog.elf –r –t

This command loads the program *myprog.elf,* into external memory starts it running, and then enters the terminal emulator.

propeller-load –b c3 myprog.elf –e –r -t

This command loads the program myprog.elf into external memory, writes a flash loader to the EEPROM, starts the program running, and then enters the terminal emulator. The –e option only makes sense when external flash memory is available on the target board. This is because the loader that is written to EEPROM assumes it will find the XMM program in external memory and that requires that at least the portion of the external memory that contains code be non-volatile. The –e option can be used on a board that has both external flash and SRAM if either the –mxmmc or –mxmm-split memory models are used since in those models the code is written to the flash.

### Cache Drivers

Cache drivers are used by XMM programs to write the program to external memory during loading and also to access external memory at runtime. There is usually at least one dedicated cache driver for each type of target board although some generic cache drivers exist that will work with any board with specific memory parts. These generic cache drivers include an EEPROM cache driver that works with boards that have EEPROMs of 64k or larger, an SD cache driver that works with any board with an SD card slot, and drivers that work with boards that have specific types of SPI flash or SRAM chips.

## Using the SD Loader

The SD loader provides a way to load XMM programs from an SD card. This is mostly useful for boards that have external RAM but not flash.

Examples

propeller-load –b c3 –l myprog.elf –r –t

This command writes the program *myprog.elf* to the SD card as *autorun.pex,* loads a helper program that loads the program from the SD card into external memory, starts it running, and then enters the terminal emulator.

propeller-load –b c3 –l myprog.elf –e –r -t

This command loads the program myprog.elf, writes it to the EEPROM, starts it running, and then enters the terminal emulator.

## Using the SD Cache Driver

## Writing a File to the SD Card

## Creating a PEX file

## Creating a Spin Binary File

# Options

## -b <type> Select target board

Use this option to select the target board type. This determines which board configuration file is used. If this option is not specified, the value of the environment variable *PROPELLER\_LOAD\_BOARD* is used. If that environment variable is not defined, the “default” board type is used. The configuration file for the selected board type is located by first looking in

* the directory containing the file being loaded
* the directory given in the environment variable *PROPELLER\_LOAD\_PATH*
* the directory containing the propeller-load program
* the directory /opt/parallax/propeller-load

Example

-b c3

This will select the *c3.cfg* board configuration file.

## -D <var>=<value> Define a board configuration variable

Use this option to define or redefine a configuration variable. The loader will use values set using the –D option instead of the corresponding values from the selected board configuration file. The –D option can also be used to define new variables that are not in the board configuration file. This could be useful if the program being loaded makes use of the values of these additional variables through the loader’s variable patching facility.

Example

-D baudrate=9600

This will use 9600 as the baud rate overriding the value for *baudrate* in the selected configuration file.

## -e Write the program into EEPROM

Use this option to write the program being loaded to EEPROM. This option works differently depending on the type of program being loaded.

If a COG or LMM program is being loaded, the entire program is written to the EEPROM.

If an XMM program is being loaded, only a loader is written to EEPROM. The program itself is written to external memory. This requires that the external memory be non-volatile. In other words, it must be flash memory.

## -f Write a file to the SD card

Use this option to write a file to an SD card inserted in the target board. If this option is given then no program is loaded. The only action is to write a file to the SD card.

Example

-f myprog.pex

## -I <path> Add a directory to the include path

Use this option to add a directory to the include path. This is the path that the loader uses to locate board configuration files and drivers. The directories specified using the –I option will be searched before the standard directories.

Example

-I foo/bar

## -l Write a program to the SD card and use the SD loader

Use this option to load code that can load the program *autorun.pex* from the root directory on the SD card into external memory. If a filename is given in the command, that file will be written to the SD card as *autorun.pex.* It should be an XMM program.

Examples

-l

Load code to load the program *autorun.pex* which should already be on the SD card

-l myprog.elf

Write the program myprog.elf to the SD card as *autorun.pex* and load code to load it from the SD card.

## –P List available serial ports

Use this option to list all serial ports. Not all of the ports listed will necessarily be connected to boards containing Propeller chips.

## -p <port> Select serial port

Use this option to select the serial port that is connected to the Propeller board you wish to load. If this option is not specified and the environment variable *PROPELLER\_LOAD\_PORT* is defined, it’s value is used. If it is not defined, the loader will search for ports attached to a Propeller board. The first one found is used.

Examples

-p COM12

-p /dev/ttyUSB12

-p /dev/cu.usbserial-12

If *<port>* begins with a digit under Windows or something other than a ‘/’ under Linux or Mac OS X, it is interpreted as a shorthand for the full port name. In that case, the system-specific prefix is added to *<port>* to form the full name.

Port prefixes

* Windows: “COM”
* Linux: “/dev/ttyUSB”
* Mac OS X: “/dev/cu.usbserial-“

Examples

-p12

-p 12

Under Windows these would be interpreted as COM12.

Under Linux they would be interpreted as /dev/ttyUSB12.

Under Mac OS X they would be interpreted as /dev/cu.usbserial-12.

## -q Quit on the exit sequence

Use this option to cause propeller-load to exit terminal mode when it receives the byte sequence (0xff, 0x00, status) from the target board. This is primarily intended for use in automated test scripts.

## -r Run the program after loading

Use this option to start the program running after loading has completed.

## –S or –S<n> Slow down the loader by adding a delay

Use this option to introduce a time delay during the initial phase of loading that uses the Propeller boot protocol. If *<n>* is not given, a delay of 5ms is used. The *<n>* must be immediately adjacent to the –S with no intervening space.

Examples

-S

This will cause a delay of 5ms to be used.

-S12

This will cause a delay of 12ms to be used.

## -s Write a spin .binary file for use with the Propeller Tool

Use this option to write a Spin .binary file for use with the Propeller Tool or any other loader that can handle the Spin binary format.

Example

-s myprog.elf

This will write myprog.binary.

## -t or –t<baud> Enter terminal mode after running the program

Use this option to cause the loader to enter a simple terminal emulator after loading is complete. If *<baud>* is given the baud rate is changed to this value before entering terminal mode. The *<baud>* must be immediately adjacent to the –t with no intervening space.

## -v Verbose output

Use this option to produce more verbose progress information.

## -x Write a .pex binary file for use with the SD loader or SD cache

Use this option to write a Propeller executable file (.pex). This file can then be transferred to an SD card to be run using either the –z or –l option.

Example

-x myprog.elf

This will read the program myprog.elf and write myprog.pex.

## -z Write a program to the sd card and use the SD cache

Use this option to load code that can run the program *autorun.pex* from the root directory on the SD card. If a filename is given in the command, that file will be written to the SD card as *autorun.pex.* It should be an XMM program.

Examples

-z

Load code to run the program *autorun.pex* which should already be on the SD card

-z myprog.elf

Write the program myprog.elf to the SD card as *autorun.pex* and load code to run it from the SD card.

## -? Display a usage message and exit

Use this option or just invoke propeller-load with no parameters to display a short usage message.

# Configuration Files

## Board Types and Subtypes

Target board type can be either a single identifier like 'propboe' in which case the subtype defaults to 'default' or it can be of the form <type>:<subtype> like 'c3:ram'.

Variables that can be set with -D are:

* clkfreq
* clkmode
* baudrate
* reset
* rxpin
* txpin
* tvpin
* cache-driver
* cache-size
* cache-param1
* cache-param2
* sd-driver
* sdspi-do
* sdspi-clk
* sdspi-di
* sdspi-cs
* sdspi-clr
* sdspi-inc
* sdspi-start
* sdspi-width
* spdspi-addr
* sdspi-config1
* sdspi-config2
* eeprom-first

# Expressions

Value expressions for -D can include:

* rcfast
* rcslow
* xinput
* xtal1
* xtal2
* xtal3
* pll1x
* pll2x
* pll4x
* pll8x
* pll16x
* k
* m
* mhz
* true
* false

an integer or two operands with a binary operator + - \* / % & | or unary + or -

all operators have the same precedence

# Variable Patching

int \_cfg\_baudrate = -1;