LogicLoader™ Command Description Manual

A LogicLoader Command Reference
(LogicLoader Version 2.3)

Logic Product Development
Published: May 2005
Last Revised: July 2007

This file contains source code, ideas, techniques, and information (the Information) which are Proprietary and Confidential Information of Logic Product Development, Inc. This information may not be used by or disclosed to any third party except under written license, and shall be subject to the limitations prescribed under license.

No warranties of any nature are extended by this document. Any product and related material disclosed herein are only furnished pursuant and subject to the terms and conditions of a duly executed license or agreement to purchase or lease equipments. The only warranties made by Logic Product Development, if any, with respect to the products described in this document are set forth in such license or agreement. Logic Product Development cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, indirect, special or consequential damages.

Logic Product Development may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering the subject matter in this document. Except as expressly provided in any written agreement from Logic Product Development, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

© Copyright 2005-2007, Logic Product Development, Inc. All Rights Reserved.

REVISION HISTORY

REV	EDITOR	REVISION DESCRIPTION	LoLo VER.	APPROVAL	DATE
A	James Wicks	Release	2.0	ME	06/03/05
B	Bruce Rovner	Added ‘setvar’ command; Updated ‘source’ command; Updated ‘echo’ command; General editing	2.0.5	HAR	01/06/06
C	Jed Anderson, Eric Nelson	Corrected typo in ‘bitmap’ command purpose to read “4, 8, or 24 bpp”; Updated ‘add-yaffs’ command; Updated ‘erase’ command	2.0.5	MT	05/08/06

D	Jed Anderson, Peter Barada	Updated for 2.3.0 release;
		Added commands: if, set, unset, while;
		Added more Return values for: bitmap, burn, draw-test, slide-show, video-clear;
		Updated commands: config, echo, exec, load
			2.3.0	MT	01/16/07

E	Jed Anderson	- Section 2.1: Corrected example for ‘add-yaffs’ command by adding a ‘B’ before the NAND flash block location and block length - Section 2.5: Corrected order of device and start block in ‘burn’ command usage	2.3.0	RGL	07/25/07

Table of Contents

1 Introduction to LogicLoader^{TM} Commands

This document briefly explains each LogicLoader command. Each command is described within these categories:

- Purpose
- Usage
- Examples
- Return
- Notes

You may view all of the available commands for your System on Module (SOM) by entering ‘help all’ at the losh prompt. Typing the entry ‘help’ at the losh prompt will print a listing of the available sub-menus. The sub-menu listings are intended as a prompt to the user when needed.

1.1 LogicLoader User’s Manual

Before reading this document, please reference the *LogicLoader User’s Manual* available on Logic’s website at https://www.logicpd.com/auth/. The *LogicLoader User’s Manual* includes a product brief and provides an overview of LogicLoader usage. It also describes the YAFFS file system, configuration block, video interface, and boot time scripts.

1.2 Glossary of Abbreviations, Acronyms, and Symbols

?	Built-in shell variable holding return value of last command
@	Built-in shell variable holding auxiliary value
< >	Required argument
[ ]	Optional argument
[[ ]]	Optional argument within another optional argument
ARGS	Arguments
ARGV	Argument vector, all arguments passed to the command
CF	CompactFlash®
BWH	hexadecimal byte, half-word, or word ▪ 'b' = 8-bit value ▪ 'h' = 16-bit value ▪ 'w' = 32-bit value
DST/DEST	Destination
ERRNO	Error number
FOO	Example name meaning ‘file name,’ or ‘variable name’; its counterpart is ‘bar’
GPIO	General Purpose Input Output
LEN	Length
LoLo	LogicLoader
LwIP	Lightweight implementation of the TCP/IP protocol stack
MEM	Memory
NFS	Network File System
ODUX	Octal/decimal hexadecimal
RAM	Random Access Memory
RTC	Real-time clock
SRAM	Static Random Access Memory
SRC	Source
STDIN	Standard input stream
STDOUT	Serial port, typical interface to the machine
TFTP	Trivial File Transfer Protocol
TLB	Translation Look-aside Buffer
YAFFS	Yet Another Flash File System

1.3 Command Usage Overview

The losh command set may vary according to the different features and requirements of the different engines. In LogicLoader, all commands in all categories that have been implemented for your engine are always available from the losh prompt.

1.3.1 Losh Command Annotation

Losh commands are listed in Section 2, below, with required or optional arguments. Arguments required by a command are noted inside angle brackets '< >'. Arguments optional to the command are designated by square brackets '[ ]'. For example, the 'load' command requires an argument that specifies the type of file to load, but optionally, a user may also specify an input stream or filename.

The command's syntax is documented as: load <type> [source]. In most cases, optional arguments are filled with default values if not specified by the user. For example, if a user does not specify the 'source' argument to the load command, the load is assumed to come from the standard input stream (stdin).

1.3.2 Losh Command Return Values

All losh commands return a value. The returned value from the last command executed is stored in the built-in shell variable '?'. A value of zero indicates that the command executed successfully. A non-zero indicates that an error prevented the command from succeeding.

Some commands return a second value. This value is stored in the built-in shell variable '@'.

For complete information concerning shell environment variables, see the *LogicLoader User's Manual*.

2 Losh Commands

2.1 add-yaffs

Purpose:

This command adds a YAFFS partition into LoLo's partition table. You must execute this command **before** attempting to mount the YAFFS partition itself. When creating a partition in a NOR flash device, the start location is a memory address and the length is in bytes. When creating a partition in a NAND flash device, the start location is the start block and the length indicates the number of blocks.

Usage:

add-yaffs <name> <nor|nand> <start> <length> where:

- <name> is any string you want to refer to the partition
- <nor|nand> indicate the flash device type
- <start> the start location of the partition
- <length> the length of the partition

Examples:

- losh> add-yaffs boot nor 0xC0000 0x800000
- losh> add-yaffs store nand B9 B500

Return:

- 0 - success
- 2 - unknown memory type
- 10 - wrong number of arguments
- 60 - out of memory
- 70 - partition table appears full
- 71 - failed to initialize

Note:

This command must be called after a reset in order to make LoLo aware of a YAFFS partition. To create a new YAFFS partition, use the erase command to erase the range of addresses containing the YAFFS file system in flash memory before calling add-yaffs.

2.2 bench-mark

Purpose:

This command measures the amount of time it takes for a processor to complete a loop of [increments] [reps] times. The default is to perform 10 repetitions of a 1000000 increment loop.

Usage:

bench-mark [increments [reps]]

Examples:

- losh> bench-mark
- losh> bench-mark 10 500

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2

2.3 bitmap

Purpose:

This command displays a bitmap on the screen. Only Windows expanded Device Independent Bitmaps are supported. The bitmap should be a standard 4, 8, or 24 bpp with no compression. The [address] parameter may be used to specify an alternate frame buffer address in ram for drawing the bitmap. The video-fb command may then be used to load the alternate frame buffer address into the video controller. The [address] parameter must be placed immediately after the <file_name>, before any other optional parameters. The [tl_x, tl_y] and [br_x, br_y] parameters optionally set the top-left x, top-left y, bottom-right x, and bottom-right y coordinates to specify the bounds of the bitmap on the screen. These comma separated parameters refer to an origin at the top-left of the screen. Specifying a bitmap file 0,0 640,480 will display a bitmap which covers an entire 640x480 screen. The default is to show as much of the bitmap as the screen will allow.

Usage:

bitmap <file_name> [address] [tl_x,tl_y [br_x,br_y]]

Examples:

- losh> bitmap /cf_card/TEST_FILE.BMP
- losh> bitmap /cf_card/TEST_FILE.BMP 0xa0400000
- losh> bitmap /cf_card/TEST_FILE.BMP 0,0 640,480

Return:

- 0 - success
- 2 - bad argument 2
- 3 - bad argument 3
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 23 - not a file
- 24 - failed to read
- 31 - no open display
- 32 - failed to get window handle
- 33 - not a Windows Device Independent Bitmap
- 34 - invalid window
- 35 - bad frame buffer
- 60 - out of memory

2.4 bootme

Purpose:

This command initiates the transfer of a Windows CE image between the device and a host running Platform Builder.

Usage:

bootme

Examples:

- losh> bootme

Return:

- 0 - success
- 51 - no active network interfaces
- 52 - unable to create socket
- 53 - unable to bind socket
- 54 - ethernet option not installed
- 60 - out of memory

2.5 burn

Purpose:

This command burns a loaded image to flash. If the device is a block device, such as NAND flash, then the [start block] parameter is the location where the data will be burned. For NAND devices, the data will be burned starting at [start block], and skipping any bad blocks. No spare areas will be programmed. This is a common method used by bulk programmers. NAND flash devices will typically have some bad blocks marked by the manufacturer. Bad blocks can also develop over time. For these reasons, we recommend using the YAFFS file system for data storage. The purpose of using the burn command for NAND is to simulate data that may have been programmed by a bulk programmer. The burn command as used with NAND devices is only intended to be used for debugging bulk programming algorithms. NOR flash devices do not require bad block management & can be burned and read back without the strict need of the YAFFS file system.

Usage:

burn [device]
burn [device] [start block]

Examples:

- losh> burn
- losh> burn /dev/flash0

Return:

- 0 - success
- 10 - wrong number of arguments
- 22 - flash device could not be opened
- 64 - address space invalid
- 80 - loaded image does not fit in: flash device
- 81 - burn failed
- 90 - image failed to load
- 250 - skipping burn

2.6 cache-flush

Purpose:

This command flushes the processor's cache.

Usage:

cache-flush

Examples:

- losh> cache-flush

Return:

- 0 - success

Note:

This command always returns 0.

2.7 cache-off

Purpose:

This command stops the processor's instruction and data cache. On certain processors the data cache must be enabled to access internal SRAM. In that case it will turn off the instruction cache but leave the data cache enabled.

Usage:

cache-off

Examples:

- losh> cache-off

Return:

- 0 - success

Note:

This command always returns 0.

2.8 cache-on

Purpose:

This command starts the processor's cache.

Usage:

cache-on

Examples:

- losh> cache-on

Return:

- 0 - success

Note:

This command always returns 0.

2.9 cat

Purpose:

This command will print the contents of the specified file to stdout.

Usage:

cat <file>

Examples:

- losh> cat /cf/foo.txt

Return:

- 0 - success
- 2 - bad argument 2
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 23 - not a file
- 24 - failed to read

Note:

This command will work on any type of file. Use the 'hd' command to view the contents of binary files.

2.10 cd

Purpose:

This command changes the current working directory.

Usage:

cd <directory>

Examples:

- losh> cd /dev

Return:

- 0 - success
- 10 - wrong number of args
- 20 - directory doesn't exist

2.11 cfstat

Purpose:

This command prints the state of CompactFlash slots.

Usage:

cfstat

Examples:

- losh> cfstat

Return:

- 0 - success
- non-zero indicates failure

Note:

This command is not supported on all architectures.

2.12 config

Purpose:

This command saves and/or displays configuration information in the config device. The config device starts at offset 0x40000 in the boot flash device. The config device may be used to store a boot script, the debug port baud rate settings, user defined video settings, and the debug Ethernet settings. Please consult the *LogicLoader User's Manual* for a full explanation of the config block device.

Usage:

```
config <b|s|v|e|B|S|V|E|C|CREATE> [data] baud script video ethernet config
    Use the lower case to display the contents, use the upper case to save new contents.
    bB – baud       -       see the LogicLoader addendum for supported baud rates
    sS – script
    vV – video
    eE – Ethernet
    C -  load a config block
    CREATE – create a new config block
```

Examples:

- losh> config b :displays baud rate of debug serial port
- losh> config B 57600 :saves debug serial port baud
- losh> config E 0 :saves the current ethernet settings
- losh> config V name x y :saves current video settings
- losh> config CREATE :save a default configuration
- losh> config C 32768 /cf/myconfig :load a new config block form the CF card
- losh> config S 468 :load a new script from stdin
- losh> config <C|S> <length> [source] to load

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 4 - bad argument 4
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 24 - failed to read
- 25 - failed to write
- 26 - unable to create
- 27 - unable to remove
- 28 - unable to seek
- 40 - config device is not valid
- 41 - index or name is not valid
- 60 - out of memory

2.13 cp

Purpose:

This command is used to copy a file.

Usage:

cp <src> <dest>

Examples:

- losh> cp /cf/foo /yaffs/bar

Return:

- 0 - success
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 23 - not a file
- 24 - failed to read
- 25 - failed to write
- 26 - unable to create
- 60 - out of memory

Note:

The destination file must be on a writable file system.
This command does **not** work on directories.
This command sets $@ to the number of bytes copied.

2.14 date

Purpose:

This command displays the number of seconds since boot.

Usage:

date

Examples:

- losh> date

Return

- 0 - success

2.15 draw-test

Purpose:

This command draws framed red, green, blue and stipple test patterns.

Usage:

draw-test

Examples:

- losh> draw-test

Return:

- 0 - success
- 31 - no open display
- 32 - failed to get window handle
- 34 - invalid window
- 35 - bad frame buffer

2.16 echo

Purpose:

This command echoes a string to standard output or to a file.

Usage:

echo <string> [filename [offset]]

Examples:

- losh> echo "Hello world"
- losh> echo "Save this string" /dev/serial_eeprom
- losh> echo "Append this string" /dev/serial_eeprom

Return:

- 0 - success
- 3 - bad argument 3
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 25 - failed to write
- 28 - unable to seek

Note:

This command sets $@ to the number of characters written when no failure occurs.

2.17 erase

Purpose:

This command erases non-volatile <device>. When using a memory mapped device (such as NOR flash) the <start address> and <length> parameters indicate the memory address and the length in bytes. When using a block device (such as NAND flash) the <start block> and <number of blocks> indicate the first block number and the number of blocks to erase. Note; some devices, such as NAND flash, are marked with bad blocks by the device manufacturer, these blocks will not be erased and the erase command will indicate which blocks have been marked bad.

Usage:

erase <start_address> <length>
erase <offset> <length> <device>
erase <start block> <number of blocks> <device>

Examples:

- losh> erase 0x400c0000 1024
- losh> erase 0 128 /dev/serial_eeprom
- losh> erase 0 512 /dev/nand0

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 83 - erase failed
- 250 - skipping erase

2.18 exec

Purpose:

This command allows the processor to jump to an OS image loaded in memory or to a given address. Before the jump, interrupts, memory caching, and mapping are disabled. -t is used to modify the calling sequence with the architecture ID in the 2nd arg, and create an ARM Linux ATAGS structure that is passed in as the third argument to the kernel. If -t is not specified, then the third argument is the command line string.

Usage:

exec [-t] [-i arch_id] [-a atag_addr] [addr -] [kernel command line]

Examples:

- losh> exec
- losh> exec -t -i 389 0x400c8000 –
- losh> exec -t -a 0xc0001000 0x400c8000 - "root=nfs"
- losh> exec 0x400c8000 -
- losh> exec 0x400c8000 - "root=nfs"
- losh> exec "root=/dev/mtd/2 rootfstype=yaffs"
- losh> exec -t "root=/dev/mtd/2 rootfstype=yaffs"

Return:

- 0 (or return value of the executed code) - success
- 1 - bad argument 1
- 2 - bad argument 2
- 10 - wrong number of arguments
- 60 - no memory for command line
- 108 - no loaded image found

Note:

Returns the return value of the executed code and sets $@ to the jump address when no losh errors occur. The memory for the command line string is taken out of LoLo's internal heap. If there is no command line, then the pointer is to a null-terminated string.

2.19 hd

Purpose:

This command prints the contents of a file to stdout in hex format. This command is useful for looking at the contents of binary files.

Usage:

hd <filename> [len [offset]]

Examples:

- losh> hd /dev/serial_eeprom

Return:

- 0 - success
- 2 - bad argument 2 (length)
- 3 - bad argument 3 (offset)
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 24 - failed to read

Note:

This command functions similarly for both binary and text files. When looking at text, use the 'cat' command.

2.20 help

Purpose:

This command provides information about the usage of a specific command or group of commands.

Usage:

```
help <test|file|dir|video|net|thread|all|cmd_name>
```

Examples:

- losh> help dir
- losh> help ifmac

Return:

- 0 - success
- 1 - bad argument 1

2.21 if

Purpose:

To alter control flow of a script.

Usage:

The first form of the ‘if’ statement has only a ‘then’ clause that is executed if the conditional expression is true:

```
if <conditional_expr>
        <true_statements>
endif
```

The second form of the ‘if’ statement contains both a ‘then’ clause that is executed if the conditional expression is true, and a ‘else’ clause that is executed if the conditional expression is false.

```
if <conditional_expr>
    <true_statements>
else
    <false_statements>
endif
```

Examples:

-
  ```
  if ($a != 0)
     echo "A not zero"
     b = -1;
  else
     echo "A is zero"
  endif
  ```

Return:

- The return value is the return of the last statement executed in the body.

Note:

An ‘if’ statement can be nested in the body of either the ‘then’ or ‘else’ clause of an ‘if’ statement.

2.22 ifconfig

Purpose:

This command configures a network interface

Usage:

```
ifconfig [interface]
ifconfig <interface> <ip> <netmask> <gw>
ifconfig <interface> [up|down|dhcp|def|/dev/config]
```

Examples:

- losh> ifconfig (display current status of interfaces)
- losh> ifconfig sm0 dhcp (bring up an interface using DHCP)
- losh> ifconfig sm0 1.1.1.1 255.255.255.0 1.1.1.0 (manual configuration)
- losh> ifconfig sm0 down (bring an interface down)

- losh> ifconfig sm0 up (bring an interface up)

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 4 - bad argument 4
- 10 - wrong number of arguments
- 11 - badly formatted ip address
- 22 - file or device could not be opened
- 24 - failed to read /dev/config
- 44 - unable to set ethernet mode (/dev/config)
- 50 - unable initialize network device
- 54 - ethernet option not installed
- 55 - unable to find network device
- 60 - out of memory

2.23 ifmac

Purpose:

This command programs and/or displays the MAC address for a network interface. When the user only provides bytes four, five, and six, the first three bytes will default to Logic's 0x00:0x08:0xEE.

Usage:

ifmac <interface> [byte 4:byte 5:byte 6]
ifmac <interface> [byte 1:byte 2:byte 3:byte 4:byte 5:byte 6]

Examples:

- losh> ifmac sm0
- losh> ifmac sm0 0x01:0x1a:0xee

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 24 - failed to read
- 24 - failed to write
- 44 - unable to set Ethernet mode (/dev/config)
- 50 - unable initialize network device
- 54 - Ethernet option not installed
- 55 - unable to find network device
- 60 - out of memory

Note:

This command sets $@ to the error code resulting from saving to the config block.

2.24 info

Purpose:

This command prints information from a chosen category.

Usage:

info <version|arch|mem|net|cpu|intr|var|yaffs>

Examples:

- losh> info version

Return:

- 0 - success
- 1 - bad argument 1

2.25 jump

Purpose:

This command allows the processor to jump to an image loaded in memory or to a given address.

Usage:

jump [addr]

Examples:

- losh> jump
- losh> jump 0x40040000

Return:

- 0 (or return value of the jumped to code) - success
- 1 - bad argument 1
- 10 - wrong number of arguments
- 108 - no loaded image found

Note:

Returns the value returned from the executed code and sets $@ to the jump address when no losh errors occur.

2.26 kill

Purpose:

This command stops the specified thread(s) from running. Use the 'ps' command to view the list of threads.

Usage:

kill <thread_id> [thread_id] ...

Examples:

- losh> kill 2 (stops thread number 2)

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 4 - bad argument 4
- 5 - bad argument 5
- 6 - bad argument 6
- 7 - bad argument 7
- 8 - bad argument 8
- 9 - bad argument 9
- 10 - wrong number of arguments

Note:

This command returns the last argument number it could not kill and 0 if it was able to kill the entire argument list.

2.27 load

Purpose:

This command opens the requested source file to load and then calls out to various binary format parse functions to perform the load.

Usage:

```
load <type> [source]
load <type> /tftp/<server:filename>
load raw <dest_addr> <length> [source]
load <type> -dhcp
```

Examples:

- losh> load elf

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 10 - wrong number of arguments
- 100 - bad checksum
- 101 - file has bad magic
- 102 - file has bad md5sum
- 103 - file for wrong platform
- 104 - file has wrong version
- 105 - file has too many sections
- 106 - dhcp inactive
- 107 - general load failure
- 108 - no loaded image found
- 109 - unknown type

2.28 ls

Purpose:

This command will list the contents of the current directory, the directory specified on the command line, or the file specified on the command line.

Usage:

```
ls [dir|file]
```

Examples:

- losh> ls
- losh> ls /dev

Return:

- 0 - success
- 20 - the file or directory doesn't exist

2.29 md5sum

Purpose:

This command calculates an MD5 hash on a file or memory chunk.

Usage:

```
md5sum <filename | address> [read-size]
```

Examples:

- losh> md5sum foo.txt

- losh> md5sum /dev/serial_eeprom
- losh> md5sum 0x0 512

Return:

- 0 - success
- 2 - bad argument 2
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 23 - not a file
- 60 - out of memory

2.30 mem-cmp

Purpose:

This command compares two memory areas over the <length> of bytes specified.

Usage:

mem-cmp <addr1> <addr2> <length>

Examples:

- losh> mem-cmp 0xc0000000 0x40000000 1000

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 10 - wrong number of arguments
- 91 - test failed

Note:

This command sets $@ to the offset where the match failed.

2.31 mem-copy

Purpose:

This command copies memory from <src> address to <dst> address for <count>, in sizes of [bhw] (byte, half-word, word). The default size is a word. A source or destination address that is unaligned with the write width will return an error number indicating an unaligned address.

Usage:

mem-copy <src> <dst> <count> [/bhw]

Examples:

- losh> mem-copy mem-copy 0xc0000 0xd0000 0x100 /h
- losh> mem-copy mem-copy 0xc0000 0xd0000 0x100

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 4 - bad argument 4
- 10 - wrong number of arguments
- 11 - bad formatting
- 61 - unaligned memory access

2.32 mem-fill

Purpose:

This command fills a memory area with a certain value.

Usage:

mem-fill mem-fill <addr> <count> <value> [/bhw]

Examples:

- losh> mem-fill 0x200c0000 100 0xac /b
- losh> mem-fill 0xc0000 0x1000 0xabcd /h

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 4 - bad argument 4
- 10 - wrong number of arguments
- 11 - bad formatting
- 61 - unaligned memory access

2.33 mount

Purpose:

This command mounts a filesytem of type <fstype> onto LoLo's root filesystem at point <point>. If the mount command is successful, you may use other shell commands to access the new filesystem.

Usage:

mount <fstype> [drive addr] <point>

Examples:

- losh> mount fatfs /cf
- losh> mount yaffs /rom

Return:

- 0 - success
- 2 - bad argument 2
- 10 - wrong number of arguments
- 21 - unable to mount
- 22 - file or device could not be opened

2.34 ping

Purpose:

This command pings a remote host via the ICMP network protocol.

Usage:

ping <ip-address> [reps]

Examples:

- losh> ping 192.168.1.1 (pings the host once)
- losh> ping 192.168.1.1 10 (pings the host ten times)

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 10 - wrong number of arguments

- 51 - no active network interfaces
- 54 - ethernet option not installed

Note:

The parameter <ip-address> **must** be quoted.

2.35 ps

Purpose:

This command displays a list of the currently executing threads.

Usage:

ps

Examples:

- losh> ps

Return:

- 0 - success

2.36 pwd

Purpose:

This command prints the current working directory to stdout.

Usage:

pwd

Examples:

- losh> pwd

Return:

- 0 - success
- 20 - the file or directory doesn't exist

2.37 remap

Purpose:

This command remaps 1M chunks of memory from physical to virtual for length. The optional 'c' parameter can be used to turn on caching.

Usage:

remap <phys> <virt> <length> [c]

Examples:

- losh> remap 0x72000000 0x72000000 0x200000
- losh> remap 0x64000000 0x64000000 0x400000 c

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 10 - wrong number of arguments

Note:

This command is not supported on all architectures.

2.38 rm

Purpose:

This command removes a file or files.

Usage:

rm <file> [file ...]

Examples:

- losh> rm /yaffs/foo

Return:

- 0 - success
- 10 - wrong number of arguments
- 20 - the file or directory doesn't exist
- 27 - unable to remove

Note:

The file must exist on a writeable file system.
This command sets $@ to the number of files removed.

2.39 set

Purpose:

The 'set' command can be used to modify several internal variables affecting script execution. These function similarly to the Unix shell scripting analog, where a '-' causes the following flags to be set, and a '+' causes them to be unset. It is highly recommended during development, that one set the '-w' flag to receive warnings about common scripting errors.

The flags available are:

e	Exit script execution immediately when commands fail
n	Read commands, but don't execute; ignored by interactive shells.
q	Don't print LoLo error messages
u	Exit on expansion of unset variables
v	Echo input lines as they are read
w	Print warnings for possible errors
x	Echo all user commands before executing them

Usage:

set set -[enuvx]

Examples:

- losh> set

2.40 setvar

Purpose:

This command manipulates environment variables; it can create new variables and it can set the value of a new or existing variable. This command can also increment or decrement a numeric variable through use of the [-i] (increment) and [-d] (decrement) options. When setvar is used to create a new variable, the [value] field will initialize the variable to the value specified, [-i] will initialize the variable to ‘1’, and [-d] will initialize the variable to ‘0’. The value specified can be any expression that is valid to use with the ‘=’ assignment statement.

Usage:

setvar <varname> [-i | -d | value]

Examples:

- losh> setvar my_var 3

- losh> setvar my_var -i
- losh> setvar my_var -d
- losh> setvar my_file /cf/nk.bin

Return:

- 0 - success
- 2 - bad argument 2
- 10 - wrong number of arguments
- 26 - unable to create the variable

2.41 sleep

Purpose:

This command sleeps for the given number of milliseconds.

Usage:

sleep <ms>

Examples:

- losh> sleep 10

Return:

- 0 - success
- 1 - bad argument

2.42 slide-show

Purpose:

This command displays a bitmap slide show on the device's screen using a configuration file with each entry having the following format:

<FILE.BMP>[:tl_x,tl_y:br_x,br_y:Tseconds]

example entries:

```
FILE.BMP
FILE.BMP:T5
FILE.BMP:5,5:90,90:T3
```

The T parameter must be last, it defaults to 3

Usage:

slide-show <configuration script file name> [repeat]

Examples:

- losh> slide-show CONFIG.TXT

Return:

- 0 - success
- 2 - bad argument 2
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 23 - not a file
- 24 - failed to read
- 31 - no open display
- 32 - failed to get window handle
- 33 - not a Windows Device Independent Bitmap
- 34 - invalid window
- 35 - bad frame buffer
- 60 - out of memory
- 251 - bitmap error

2.43 source

Purpose:

This command executes a series of losh-shell commands listed in <file>. Follow the source command with an [s] to run the script in silent mode. Follow the source command with a variable name [-varname] to run the script the number of times indicated by the variable's value. The source command will decrement the value by one each time it runs the file, and will continue to call the script until the value is equal to zero. The variable's value may also be updated by commands within the file itself in order to control the number of times the file is sourced. When using the [-varname] option, the variable must have been initialized prior to using the variable with the source command.

Usage:

source <file> [s] [-varname]

Examples:

- losh> source /cf_card/STARTUP
- losh> source /dev/serial_eeprom s (s for silent)
- losh> source /cf_card/STARTUP -my_counter
- losh> source /dev/serial_eeprom s -my_counter

Return:

- 0 - success
- 2 - bad argument 2
- 3 - bad argument 3
- 22 - file or device could not be opened
- 45 - setting is not valid
- 60 - out of memory

2.44 test-dev

Purpose:

This command opens and calls the test function exported by a specified device. Some devices do not export a test.

Usage:

test-dev <device> [test arg list]

Examples:

- losh> test-dev /dev/serial_eeprom
- losh> test-dev /dev/flash0 ? (prints help about the flash test)

Return:

- 0 - success
- 2 - bad argument 2
- 3 - bad argument 3
- 4 - bad argument 4
- 5 - bad argument 5
- 6 - bad argument 6
- 7 - bad argument 7
- 8 - bad argument 8
- 9 - bad argument 9
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 90 - unable to start test
- 91 - test failed
- 92 - test timed out

Note:

Some tests are destructive.

2.45 test-mem

Purpose:

This command runs a memory test on all memory outside of LoLo. A test-mem with the 't' option will require a subsequent call with the 'r' option to obtain the results.

Usage:

test-mem [v|t|r]

Examples:

- losh> test-mem v (v for verbose)
- losh> test-mem t (t for threaded)
- losh> test-mem r (r for results)

Return:

- 0 - success
- 1 - bad argument 1
- 10 - wrong number of arguments
- 91 - test failed
- 92 - test timed out

2.46 test-reg

Purpose:

This command writes & reads a pattern into a register 1M times and looks for errors.

Usage:

test-reg <reg addr> <reg size in bytes> <bitmask_of_valid r/w bits>

Examples:

- losh> test-reg 0xc0000000 1 0xff

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 10 - wrong number of arguments
- 91 - test failed

2.47 test-rtc

Purpose:

This command starts or stops the rtc test and reports results.

Usage:

test-rtc

Examples:

- losh> test-rtc

Return:

- 0 - success
- 90 - unable to start test
- 91 - test failed

Note:

This command is not supported on all architectures.

2.48 tlb-flush

Purpose:

This command flushes the processor's tlb.

Usage:

tlb-flush

Examples:

- losh> tlb-flush

Return:

- 0 - success

Note:

This function is unique to each architecture and always returns 0.

2.49 tsleep

Purpose:

This command causes its calling thread to sleep for <ms> number of milliseconds.

Usage:

tsleep <ms>

Examples:

- losh> tsleep 1

Return:

- 0 - success
- 1 - bad argument 1

2.50 unset

Purpose:

Delete the losh variable <my_var> to reclaim its memory.

Usage:

unset <my_var>

Examples:

- losh> unset my_var
- losh> unset temporay_network_drive_name

2.51 update

Purpose:

This command loads and installs an update image.

Usage:

update [filename]

Examples:

- losh> update
- losh> update /cf_card/1001234_lolo.upd

Return:

- 0 - success
- 10 - wrong number of arguments
- 22 - file or device could not be opened
- 24 - failed to read
- 80 - image does not fit in flash
- 81 - burn failed
- 82 - erase failed
- 83 - verify failed
- 101 - file has bad magic
- 102 - file has bad md5sum
- 103 - file for wrong platform
- 104 - file has wrong version
- 105 - file has too many sections
- 107 - general load failure
- 108 - no loaded image found
- 109 - unknown type
- 110 - unable to read master header
- 111 - unable to read section header
- 112 - unknown record type
- 113 - wrong arch type
- 114 - file has too many headers

2.52 video-clear

Purpose:

This command clears a device's screen to white or an optional color.

Usage:

video-clear [r|g|b|y|l]

Examples:

- losh> video-clear (clear to white)
- losh> video-clear r (clear to red)
- losh> video-clear l (clear to black)

Return:

- 0 - success
- 31 - no open display
- 32 - failed to get window handle
- 34 - invalid window
- 35 - bad frame buffer

2.53 video-close

Purpose:

This command closes the default video device. It disables power to the display, disables the video controller, and attempts to restore any GPIO connections to their default values.

Usage:

video-close

Examples:

- losh> video-close

Return:

- 0 indicates success

2.54 video-fb

Purpose:

This command sets the address for the current frame buffer. This address will be used for drawing and displaying by the other video commands. A display must have been initialized by using either the 'video-open' or 'video-init' commands before using this command. When no address is given it just prints the current frame buffer address.

Usage:

video-fb [address]

Examples:

- losh> video-fb 0xa0400000

Return:

- 0 - success
- 1 - bad argument 1
- 31 - no open display

Note:

This command sets $@ to the frame buffer address when a valid display exists.

2.55 video-init

Purpose:

This command initializes the default video device. It configures all gpio for the video controller but does not power up the display or enable the controller. To display an image it must be followed by a 'video-on' command.

Usage:

video-init <display> <bpp>

Examples:

- losh> video-init 5 16
- losh> video-init myscr 8

Return:

- 0 - success
- 2 - bad argument 2
- 10 - wrong number of arguments
- 30 - found no suitable display driver

2.56 video-off

Purpose:

This command powers down and disables the video controller. The controller remains configured for the display and may be turned back on again with the 'video-on' command.

Usage:

video-off

Examples:

- losh> video-off

Return:

- 0 - success

2.57 video-on

Purpose:

This command powers up the display and enables the video controller. The 'video-init' or 'video-open' commands must have been issued prior to using this command.

Usage:

video-on

Examples:

- losh> video-on

Return:

- 0 - success
- 31 - no open display

2.58 video-open

Purpose:

This command opens the default video device. It configures all GPIO and enables the video controller. It is equivalent to 'video-init' followed by a 'video-on' command.

Usage:

video-open <display> <bpp>

Examples:

- losh> video-open 5 16
- losh> video-open myscr 8

Return:

- 0 - success
- 2 - bad argument 2
- 10 - wrong number of arguments
- 30 - found no suitable display driver

2.59 w

Purpose:

This command writes a hex byte, half-word, or word to a memory location. The default is word.

Usage:

w [/[bhw]] <addr> <data>

Examples:

- losh> w /w 0x60000000 0x12345678

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 10 - wrong number of arguments
- 61 - unaligned memory access

Note:

This command sets $@ to the location written.

2.60 while

Purpose:

To repeatedly execute a set of statements until the condition is false. The while statement first evaluates the <conditional expression>, and if true, execute the commands in <body>. Then control passes back to the top where the 'while' statement reevaluates the <conditional expression> and breaks out of the loop if it is false.

Usage:

```
while <conditional expression>
      <body>
done
```

Note that

Examples:

The following computes the GCD (Greatest Common Divisor) of the variables A and B leaving the result in A, and using 'tmp' as a temporary varialbe:

```
while ($B != 0)
tmp = $B
B = $A % $B
A = $tmp
done
```

The following example continually examines a value in memory (say an I/O register) once per second and breaks out of the loop if the low bit is zero. It assumes the register address is in the variable REG_ADDR, and the register is 16 bits in size:

```
while !(*(short)$REG_ADDR & 1)
        sleep 1000
done
```

Return:

- The return value is the return of the last statement executed in <body_statements>.

Note:

A ‘while’ statement can be nested in the body of a ‘while’ statement.

2.61 x

Purpose:

This command examines memory with [width] using [format] at an address for a [length].

Usage:

```
x [/[bhw][odux]] <addr> [length]
```

Examples:

- losh> x /h 0x40000000 64
- losh> x /b 0x40000000 128

Return:

- 0 - success
- 1 - bad argument 1
- 2 - bad argument 2
- 3 - bad argument 3
- 10 - wrong number of arguments

Note:

This command sets $@ to the value located at the first address read. . Even though $@ is a 32 bit value, the address is read using a cast to the width specified and the unused bits are set to zero.