Mellanox Firmware Tools (MFT) User's Manual

Rev 1.50

www.mellanox.com

NOTE:
THIS HARDWARE, SOFTWARE OR TEST SUITE PRODUCT (“PRODUCT(S)”) AND ITS RELATED DOCUMENTATION ARE PROVIDED BY MELLANOX TECHNOLOGIES “AS-IS” WITH ALL FAULTS OF ANY KIND AND SOLELY FOR THE PURPOSE OF AIDING THE CUSTOMER IN TESTING APPLICATIONS THAT USE THE PRODUCTS IN DESIGNATED SOLUTIONS. THE CUSTOMER'S MANUFACTURING TEST ENVIRONMENT HAS NOT MET THE STANDARDS SET BY MELLANOX TECHNOLOGIES TO FULLY QUALIFY THE PRODUCTO(S) AND/OR THE SYSTEM USING IT. THEREFORE, MELLANOX TECHNOLOGIES CANNOT AND DOES NOT GUARANTEE OR WARRANT THAT THE PRODUCTS WILL OPERATE WITH THE HIGHEST QUALITY. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL MELLANOX BE LIABLE TO CUSTOMER OR ANY THIRD PARTIES FOR ANY DIRECT, INDIRECT, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OF ANY KIND (INCLUDING, BUT NOT LIMITED TO, PAYMENT FOR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY FROM THE USE OF THE PRODUCT(S) AND RELATED DOCUMENTATION EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mellanox Technologies
350 Oakmead Parkway
Sunnyvale, CA 94085
U.S.A.
www.mellanox.com
Tel: (408) 970-3400
Fax: (408) 970-3403

Mellanox Technologies, Ltd.
Beit Mellanox
PO Box 586 Yokneam 20692
Israel
www.mellanox.com
Tel: +972 (0)4 909 7200 ; +972 (0)74 723 7200
Fax: +972 (0)4 959 3245

Mellanox Firmware Tools (MFT) User’s Manual

Table of Contents

Revision History

Printed on December 29, 2011.

Table 1 - Revision History Table

Date	Revision	Description
December 2011	1.50	Added the following note “This step in not required in Windows.” to the following sections: • Section 1.3, “MFT Access to Hardware Devices,” on page 8 • Section C.1.2, “Operation,” on page 43 • Section C.2.1, “Operation,” on page 45 • Section C.3.1, “Operation,” on page 46 • Section C.4.1, “Operation,” on page 47 • Section E.3, “Software Installation,” on page 53
July 2011	1.40	• Updated Section 1.1, “Supported Operating Systems,” on page 6 • Updated Section 5, “Supported Mellanox Devices,” on page 8 • Updated “Mlxburn” format • Added Section 2.4.2, “SwitchX™ Switch Examples,” on page 16 • Added Section 3.2.3.4, “Disabling/enabling Access to the Hardware,” on page 35 • Updated Section 3, “flint – Firmware Burning Tools,” on page 20 • Updated Section 3.2.2.1, “Burning a FW Image,” on page 24 • Updated Section 3, “flint – Firmware Burning Tools,” on page 20 • Added the “-striped_image” flag to Section 3.2.3.1, “Setting GUIDs and MACs,” on page 27
December 2010	1.30	• Updated table in Section 2, “MFT Software Dependencies on Linux” • Removed [-sw_sys] entries from the document • Added device 25438 - for MT26438 ConnectX-2 VPI w/ Virtualization+ • Added Section 3.2.3.2, “Preparing a Binary Firmware Image for Pre-assembly Burning,” on page 32 • Added section “On 4th Generation Devices” on page 29. • Updated sections “Install MFT On Linux OS” on page 10. and “Uninstall MFT (Linux)” on page 10. • Removed section isw Utility • Added “Preparing a Binary Firmware Image for Pre-assembly Burning” on page 32 • Added the “-striped_image” flag in sections “mlxburn Synopsis” on page 13 and “Switch Descriptions” on page 20
October 2009	1.20	• Added support for Mellanox MT25408 ConnectX-2, MT25408 ConnectX-2 EN, MT25458 ConnectX-2 EN*t*, MT64102 BridgeX, and MT1016 PhyX devices • Removed ibspark text (no longer supported) • Added the option ‘-fw_dir’ to mlxburn • Added Section 2.4.4, “BridgeX Gateway Examples” • Added support for Expansion ROM images - see Section 3.2.2.4, “Managing an Expansion ROM Image” • Added Section C.3, “mlxi2c Utility” • Added Section C.5, “mget_temp Utility”
December 2008	1.10	• Added support for In-Band device access for the Windows operating system. See Section 1.2 and Section 1.3. • Added Appendix E, “MTUSB-1 USB to I2C Adapter”
November 4, 2008	1.01	• Added VMware ESX Server 3.5 support. See Section 1.2.3, Section 1.3, Section 1.4.5, Section 1.4.6, and Section 2.1 • Added the -ul flag to *mlxburn* -see Section 2.3

Table 1 - Revision History Table

Date	Revision	Description
August 2008	1.0	• Added support for Mellanox InfiniScale IV switch device • Expanded In-Band support with the command 'mst ib add' • Modified the Windows MFT installation (now it is a standalone installation) • Added the -qq flag to flint
April 2008	0.65	• Added MFT installation instruction in Section 1.4, "MFT Installation," on page 10 • mlxburn tool: added Expansion ROM auto-detection description • flint tool: added the '-blank_guids' flag and the 'sg' command • Added Section 3.2.3.1 describing how to set GUIDs/MACs on a Flash device with blank GUIDs/MACs
August 2007	0.60	• Updated tool usage examples to use ConnectX devices • Added Appendix C, "Debug Utilities"
June 2007	0.50	• Added the '-mac' flag to the *flint* and *mlxburn* tools to support the ConnectX EN 10GigE adapter • Added Section 4.3, "ibspark," on page 29 that describes *ibspark*, the In-Band firmware burning tool for InfiniScale III switches • Modified Chapter 2, "mlxburn - FW Image Generator & Burner" on page 18 to describe *mlxburn* support for burning switch systems with multiple InfiniScale / InfiniScale III switch devices • Added Appendix B, "Flow Examples - mlxburn" to describe burning switches In-Band and via a direct I2C connection
January 2007	0.40	• MFT for Windows is now part of the WinIB software package; therefore, to install MFT on a Windows machine, you need to install WinIB and enable MFT. See Section 1.4.3, "Install MFT On Windows OS," on page 10. • Added ***flint*** flag: -use_image_ps • Removed ***flint*** flags: -crc, -bsn
January 2006	0.30	(MFT version 1.0.1) • Added querying options for VPD for mlxburn • Added examples to demonstrate support of MT43132 InfiniScale device by mlxburn and spark • Reorganized the "flint – Firmware Burner," on page 26 chapter • Added the Appendix "PSID Assignment"
October 2005	0.20	Added Windows distribution to MFT (MFT version 0.5.1)
		Added the following sections: • Section 1.1, "Supported Operating Systems," on page 6 • Section 1.3, "MFT Access to Hardware Devices," on page 8 • Section 1.2, "Software Prerequisites," on page 7 • Section 1.4, "MFT Installation," on page 10
August 2005	0.10	First release (Linux distribution only) (MFT version 0.5.0)

1 Introduction

The Mellanox Firmware Tools (MFT) package is a set of firmware management tools for a single InfiniBand node. MFT can be used for:

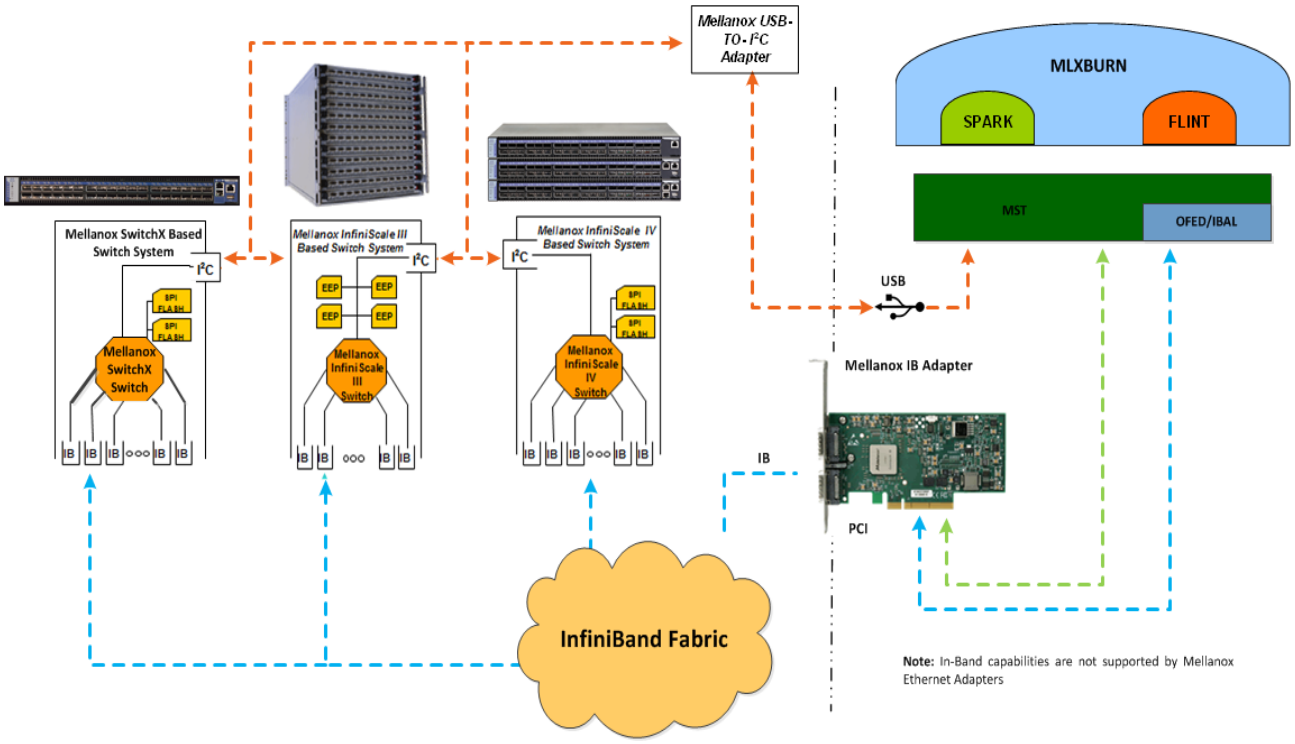
- Generating a standard or customized Mellanox firmware image
- Querying for firmware information
- Burning a firmware image to a single Mellanox device

The following is a list of the available tools in the package, together with a brief description of what each tool performs. The tools apply to single Switch Systems or adapter cards, but not to clusters.

Table 1 - Mellanox Firmware Tools (MFT) Available Tools

mlxburn	This tool provides the following functions: • Generation of a standard or customized Mellanox firmware image for burning—in .bin (binary) or .img format • Burning an image to the Flash/EEPROM attached to a Mellanox HCA or switch device • Querying the firmware version loaded on an HCA board • Displaying the VPD (Vital Product Data) of an HCA board
flint	This tool burns a firmware binary image or an expansion ROM image to the Flash device of a Mellanox network adapter/bridge/switch device. It includes query functions to the burnt firmware image and to the binary image file.
spark	This tool burns a firmware *binary* image to the EEPROM(s) attached to an InfiniScaleIII® switch device. It includes query functions to the burnt firmware image and to the binary image file. The tool accesses the EEPROM and/or switch device via an I2C-compatible interface or via vendor-specific MADs over the InfiniBand fabric (In-Band tool).
Debug utilities	A set of debug utilities (e.g., itrace, mstdump, mlxi2c, and i2c)

Figure 1: Mellanox Firmware Tools – A Scheme of Operation



1.1 Supported Operating Systems

MFT distributions are available for the following operating systems: *Linux*, *Windows*, and *VMware® ESX Server 3.5*. Please refer to the release notes of your version for supported platforms and kernels.

Unless explicitly specified, the usage of the tools is identical for all operating systems.

1.2 Software Prerequisites

1.2.1 On Linux

Table 2 - MFT Software Dependencies on Linux

Software Package	Required Version
Kernel sources	Machine's kernel version
OFED / MLNX_OFED[1, 2]	1.1 or later
Perl	5.6 or later
zlib	1.1.4 or later

1. OFED can be downloaded from http://www.openfabrics.org. Note that installing OFED is *not* required if you wish to install MFT without In-Band capabilities.
2. For the 'mst ib add' command to run, one of the OFED packages "ibutils" or "ibutils2" or "infiniband-diags" should be installed and available in the PATH. (For details on OFED installation, see the link in Section 1.5, "Reference Documents and Downloads," on page 11.

1.2.2 On Windows

Table 3 - MFT Software Dependencies on Windows

Software Package	Required Version
I2CBridge[1] (Dimax's Driver for USB to I2C Adapter)	0.1.4 or later
WinOF [2] (optional)	2.0.0 or later NOTE: The tools package must also be installed as part of the WinOF installation.

1. Visit http://www.diolan.com to download this driver. This driver is required for the first use of the MTUSB-1 device. It is not required for MFT software installation.
2. WinOF is required only for In-Band access. The package can be downloaded from www.mellanox.com > Products > InfiniBand SW/Drivers.

1.2.3 On VMware ESX Server 3.5

Table 4 - MFT Software Dependencies on VMware ESX Server 3.5

Software Package	Required Version
OFED's *mstflint* RPM [1]	OFED 1.1 or later

1. OFED's *mstflint* RPM can be downloaded from http://www.mellanox.com/products/management_tools.php.

1.3 MFT Access to Hardware Devices

Table 5 lists the Mellanox devices supported by MFT, the supporting tools, and the access methods to these devices.

Table 5 - Supported Mellanox Devices

Device Type	Product Name	Supporting Tools	HW Access Method		
			PCI	I2C	In-Band
HCA (InfiniBand)	MT23108 InfiniHost	mlxburn, flint	V	V	
	MT25208 InfiniHost III Ex		V	V	
	MT25204 InfiniHost III Lx		V	V	
VPI Network Adapter	MT25408 ConnectX	mlxburn, flint	V	V	V
	MT25408 ConnectX-2		V	V	V
	MT25408 ConnectX-3		V	V	V
Ethernet Adapter (NIC)	MT25408 ConnectX EN	mlxburn, flint	V	V	
	MT25408 ConnectX-2 EN				
Switch	MT58100A0 SwitchX	mlxburn, flint	V [1]	V	V
	MT48436 InfiniScale IV		V[1]	V	V
	MT47396 InfiniScale III	mlxburn, spark		V	V
	MT43132 InfiniScale			V	
Bridge	MT64102 BridgeX	mlxburn, flint	V[1]	V	V

1. For managed switch products only.

MFT tools access Mellanox devices via the PCI-X / PCI Express interface, via a USB to I2C adapter (Mellanox P/N: MTUSB-1), or via vendor-specific MADs over the InfiniBand fabric (In-Band).

In-Band device access requires the local IB port to be in the ACTIVE state and connected to an IB fabric.

All MFT tools address the target hardware device using an *mst device name*. This name is assigned by running the command ‘mst start’(In Windows, it is not required to run the "**`mst start`**" command) for PCI and I2C access. In-Band devices can be assigned by running the 'mst ib add' command.

To list the available mst device names on the local machine, run ‘mst status’ (on Linux and Windows only).

Notes for Windows:

- The mst service, which provides PCI access to the target adapter, is run automatically upon boot.
- To access an adapter or switch device via the USB bus: A USB to I2C Adapter should be used to connect the host USB port and the I2C port of the target device. Upon the first usage of this interface, you will be requested to install the USB to I2C Adapter driver (I2CBridge[1]).

Note for VMware ESX Server 3.5:

- O*nly* PCI user level access to Mellanox devices is supported on VMware ESX Server 3.5 machines. (See below.)

The format of an mst device name is as follows:

- ***Via PCI:*** mt<dev-id>_pci<_crX|confX>

  where:

  X is the index of the adapter on the machine.

  _crX devices access the adapter directly (recommended if possible)

  confX devices use configuration cycles to access the adapter

  For example:

```
mt25418_pci_cr0
```

- **Via USB to I2C adapter:** For example, mtusb-1.
- **Via In-Band:** <string>lid-<lid_number>.

  For example:

```
"CA_MT25418_sw005_HCA-1_lid-0x000c" or simply "lid-12"
```

  The “mst ib add” command adds devices in the format:

  - for adapters and bridges:

```
CA_<device id >_<ib node description>_lid-<lid number>
```

  - for switches:

```
SW_<device id >_lid-<lid number>
```

  See Step 3 in Appendix B,“Flow Examples - mlxburn,” on page 38 for instructions on how to obtain the device LID

1. Visit http://www.xdimax.com to download this driver.

See Appendix D,"In-Band Access to Multiple IB Subnets," on page 52 for details on how to access devices in multiple IB subnets.

- **Via PCI user level:** <bus:dev.fn>

The mst device name obtained in this method can only be used with the mlxburn tool with the '-ul' flag.

For example, if you run *lspci -d 15b3*: Mellanox devices and PCI Device IDs will be displayed.

```
> /sbin/lspci -d 15b3:

02:00.0 Ethernet controller: Mellanox Technologies Unknown device 6368 (rev a0)
```

1.4 MFT Installation

1.4.1 Install MFT On Linux OS

From MFT version 2.6.2, MFT installation method on Linux OS is RPM based. MFT applications are installed using a pre-compiled binary RPM, and Kernel modules are distributed as a source RPM and compiled by the installation script.

To install MFT, perform the following:

1. Download the Linux MFT package from the Mellanox Management Tools webpage:http://www.mellanox.com/products/management_tools.php
2. Untar the downloaded package
3. Run 'install.sh'
4. Start the mst driver by running: mst start

It is possible to customize some installation parameters (such as the target installation path). Run 'install.sh --help' for details.

1.4.2 Uninstall MFT (Linux)

To uninstall MFT (Linux) run:

```
> rpm -e mft kernel-mft
```

1.4.3 Install MFT On Windows OS

1. Download the Windows MFT MSI from http://www.mellanox.com/products/management_tools.php.
2. Double-click the MSI and follow the instructions for installation.

1.4.4 Uninstall MFT (Windows)

To uninstall WinMFT, perform one of the following:

1. Activate the ‘Add or Remove Programs’ utility of Windows and click the ‘Remove’ button of the MFT tool entry.
2. Click start > Programs > Mellanox > WinMFT > uninstall.

1.4.5 Install MFT On VMware ESX Server 3.5

1. Download the *mstflint* for VMware ESX Server 3.5 RPM from the Mellanox Management Tools webpage:
   http://www.mellanox.com/products/management_tools.php
2. Download the MFT for VMware ESX Server 3.5 RPM from the Mellanox Management Tools webpage.
3. To install the RPMs, run:

```
> rpm -i mstflint-X.X.rpm

> rpm -i mft-X.X.Xvmware_<kernel version>.rpm
```

1.4.6 Uninstall MFT (VMware ESX Server 3.5)

1. To uninstall MFT, run:

```
> rpm -e mft

> rpm -e mstflint
```

1.5 Reference Documents and Downloads

- To download firmware images and their release notes, see http://www.mellanox.com under ‘Firmware’ downloads.
- Mellanox OFED (for Linux) is a software stack that can be downloaded from http://www.mellanox.com > Products > InfiniBand SW/Drivers.
- Mellanox WinOF (for Windows) is a software stack that can be downloaded from http://www.mellanox.com > Products > InfiniBand SW/Drivers.
- *ibdiag* tools – run ‘man ibdiagnet’ for details on a machine with OFED installed.

2 mlxburn - FW Image Generator & Burner

2.1 Overview

mlxburn is a tool for firmware (FW) image generation and/or for burning a firmware image to the Flash/EEPROM attached to a Mellanox device. Both functions or a single function of mlxburn can be activated by means of command line options (see Section 2.3, “mlxburn Synopsis”). It can also query for firmware attributes (e.g., firmware version, GUIDs, etc.) and VPD info of adapter cards and switch systems.

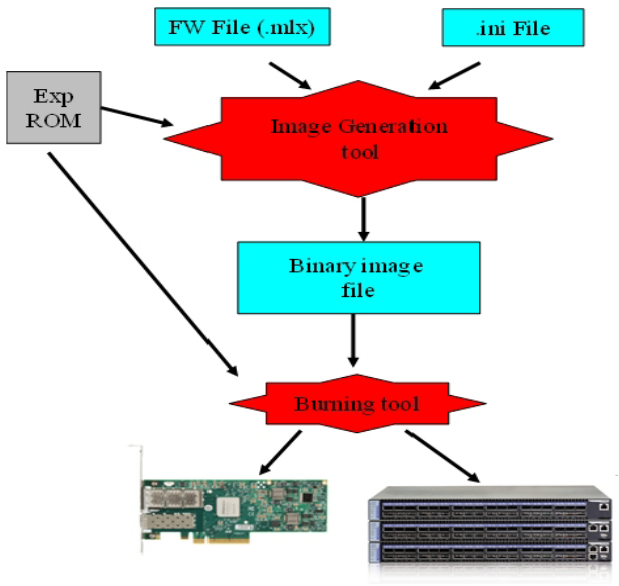
mlxburn allows for cusomization of standard Mellanox firmware for OEM specific needs (e.g., a specific adapter board type). See Section 2.2.1, “Firmware Customization,” on page 13.

On VMware ESX Server 3.5, you must use mlxburn with the ‘-ul’ flag.

2.2 Firmware Generation and Burning with mlxburn

The **mlxburn** firmware update flow is composed of two separate stages: image generation and image burning. In the image generation stage, a given Mellanox firmware release (in .mlx format for adapters, bridges and 4th generation switches, and in .BIN file format for InfiniScale III switches) is processed together with a board-specific configuration (.ini) file to generate a ‘burnable’ firmware image. This image is burnt to the Flash/EERPROM attached to a Mellanox device in the second stage. The burning process retains device specific data such as GUIDs, UIDs, MACs, VSD, and BSN. Also, the burn process is failsafe by default.

Figure 2: FW Generation and Burning



mlxburn runs both stages by default, but it may perform only one by means of command options. If the '-wrimage' is specified (see Section 2.3, "mlxburn Synopsis"), only image generation is performed. Specifying the '-image' option skips the image generation stage and loads the provided image (generated in a previous run of **mlxburn** using the '-wrimage' option).

When generating an image file for a Mellanox InfiniScale III switch device, the produced image file name must end with a .img extension

2.2.1 Firmware Customization

A Mellanox firmware image can be customized (usually) to fit a specific board type. The customization is done by using a FW parameter-set file in the image generation stage. This file has a .ini format. Each parameter-set file has a unique parameter-set ID (PSID), which is kept in the device Flash/EEPROM and allows retaining device configuration during future FW updates.

During a device FW update, **mlxburn** reads the PSID from the device and uses the corresponding .ini file when generating the FW image. **mlxburn** searches for the files in the same directory of the FW release. When **mlxburn** is used to generate an image file, or when no corresponding parameter-set file is found, the user should explicitly specify which parameter-set file to use.

To produce an image file the user needs to provide the option '-wrimage <target file>'. To actually burn the image to the Flash/EEPROM attached to a Mellanox adapter or switch device, the user needs to specify the option '-dev <mst device>' (see the synopsis section below).

If run in burning mode, **mlxburn** auto-detects the firmware parameter-set with which the device was previously burnt. It locates and uses this parameter-set file to generate the appropriate image for the device (by merging the FW release with the specific parameter-set required).

To inhibit image generation, the '-image <pre-generated-image-file>' should be used. It instructs **mlxburn** to use the given file for burning the device.

2.3 mlxburn Synopsis

mlxburn

```
[-h][-v] <-dev mst-device|-wrimage fw-image> <-fw mellanox-fw-file|-image fw-image|-img_dir
img_direcory|-fw_dir fw_dir> [-conf fw-conf-file][-nofs][-nofs_img][-format BINARY|IMAGE][-
dev_type device-type] [-exp_rom <exp_rom_file>][-exp_rom_dir <exp_rom_dir>][-force][-conf_dir
<conf_dir>][-fwver] [-vpd][-vpd_rw][-vpd_prog_rw <rw-keywords-file>][-vpd_set_keyword <keyword-
assignment>] [-set_pxe_en <(port1|port2)=(enable|disable)>] [-query][-ul]
```

where:

-h	- Display a short help text
-v	- Print version info and exit
-V <INFORM|WARNING|DEBUG>	- Set the verbosity level. Default is WARNING.
-dev <mst-dev>	- Burn the image using the given MST device

-wrimage <fw-image-file>	- Write the generated binary image to the provided file name
-fw <mellanox-fw-file>	- Specify the Mellanox firmware file to use (file extension is .mlx)
-format <BINARY|IMAGE>	- Specify which image format to use. Can be used only with the -wrimage flag. Default is BINARY.
-image <fw-image-file>	- Use the given firmware image to burn (file extension is .bin or .img)
-conf <parameter-set-file>	- Firmware configuration file (.ini). Needed for image generation (not using the -dev flag) or if auto-detection of configuration fails.
-conf_dir <dir>	- Instruct the burn operation to look for auto-detected configuration files under the specified directory rather than under the firmware file directory.
-fw_dir <dir>	- Instruct the burn operation to look for auto-detected firmware files under the specified directory.
-nofs	- When specified, the burn process will not be failsafe. A non-failsafe burn is required (on the rare occasion) when a new firmware version has modifications in the Invariant Sector
-nofs_img	- When specified, the generated image will not be failsafe. If burning is also specified, it will not be failsafe either.
-exp_rom <exp-rom-file>	- Integrate the given expansion rom file to the FW image. The given file may be in .img or .bin/.rom (raw binary) format. If the exp-rom-file is set to "AUTO", expansion rom file is auto detected from the files rom in the exp_rom_dir (see below). **Note:** Exp rom auto detection is done for devices that are already burned with an exp-rom image. If "-exp_rom AUTO" is specified for a device with no exp-rom, it would be burnt with no exp rom. To add exp-rom to a device, manually supply the exp rom file to use.
-exp_rom_dir <exp_rom_dir>	- The directory in which to look for expansion rom file when "-exp_rom AUTO" is specified. By default, exp-rom files are searched in <fw file directory>/exp_rom/*
-fwver	- When a device is given: Display current loaded firmware version. When a FW file is given (-fw flag): Display the file FW version.
-force	- Run mlxburn in non-interactive mode
-vpd [1]	- (**on Linux only**): Display the Read Only section of the PCI VPD (Vital Product Data) of the given device
-vpd_rw [1]	- (**on Linux only**): Display the Read Only and Read/Write sections of the PCI VPD of the given device
-vpd_prog_rw [1] <rw-fields-file>	- (**on Linux only**): Program the VPD-W tag (the writable section of the VPD) with the data given in the rw-keywords-file. File lines format: "KEYWORD = VALUE". In order to set binary data to a keyword, add ":BIN" to the keyword name. in this case, the data is a hexadecimal string of even length. Example file: V1 = MY-ASCII-KEYWORD

1. The VPD query may not be enabled on certain board types. Also, VPD operations are available only for devices with a PCI interface.

V2:BIN = 1234abcd

White spaces before and after VALUE are trimmed.

-vpd_set_keyword [1] <keyword-assignment> - (**on Linux only**): Add or change a keyword value in the VPD-W tag (the writable section of the VPD) with the data given in the keyword-assignment string. The string format is identical to a line in the rw-keywords-file described above. Other keywords in the VPD-W tag are not affected by this operation

-query - Query adapter or switch devices for firmware information

-ul - (**on Linux only**): Use this flag to access hardware using the PCI user level method. When using this flag, the argument of the -dev flag should be the PCI number of the target device in the format: <bus:dev.fn>

-striped_image - For image generation only. The "striped image" is used as its file layout is similar to the image layout on the flash. The striped image can be used for pre burning using an external burner or flint "bb" command.

-dev_type <Mellanox-Device-ID> - **mlxburn** must know the Mellanox device ID in order to work properly. This option should be used if auto-detection of the device type (taken from the firmware file) fails. The following is the list of supported device IDs:

23108 - For MT23108 InfiniHost based HCA cards (Cougar family)

25208 - For MT25208 InfiniHost III Ex in InfiniHost-mode (with local attached memory) HCA cards (Lion Cub family)

25218 - For MT25208 InfiniHost III Ex in MemFree-mode HCA cards (Lion Mini family)

25204 - For MT25204 InfiniHost III Lx HCA cards (Tiger/Cheetah families)

25408 - For MT25408 ConnectX Dual 10Gb/s InfiniBand (VPI) Port Adapter Cards

25418 - For MT25408 ConnectX Dual 20Gb/s InfiniBand (VPI) Port Adapter Cards

26418 - For MT25408 ConnectX Dual 20Gb/s InfiniBand (VPI) Port Adapter Cards with PCI Express Gen-2

26428 - For MT25408 ConnectX IB Dual 40Gb/s InfiniBand (VPI) Port Adapter Cards with PCI Express Gen-2

25448 - For MT25408 ConnectX EN (NIC) cards

26448, 26468- For MT25408 ConnectX EN 10GigE cards with PCI Express Gen-2

26478- For MT25408 ConnectX-2 EN 40GigE cards with PCI Express Gen-2

25458 - For MT25458 ConnectX EN 10GBASE-T cards

26458 - For MT25458 ConnectX EN 10GBASE-T cards with PCI Express Gen-2

25438- For MT26438 ConnectX-2 VPI w/ Virtualization+

43132 - For MT43132 InfiniScale based switch systems

47396 - For MT47396 InfiniScale III based switch systems

48436 - For MT48436 InfiniScale IV based switch systems (IB @ SDR)

48437 - For MT48436 InfiniScale IV based switch systems (IB @ DDR)

48438 - For MT48436 InfiniScale IV based switch systems (IB @ QDR)

64102, 64122 - For MT64102 BridgeX based gateway systems

1016 - For MT1016 PhyX devices

51000 - For MT58100 SwitchX based switch systems

4099 - For MT27500 ConnectX-3

2.3.1 Additional mlxburn Options

The following is a list of additional options. Please see Chapter 3, "flint – Firmware Burner" for the HCA options, and Chapter 4, "spark - Switch Firmware Burning" for the switch options.

For adapters, bridges and 4th generation switches:[1]

```
[-byte_mode] [-use_image_ps] [-skip_is] [-mac(s)] [-guid(s)] [-sysguid] [-vsd] [-qq] [-uid(s)] [-
log] [-blank_guids] [-flash_params] [-allow_psid_change] [-no_flash_verify] [-use_image_rom
-override_cache_replacement] [-use_image_guids] [-bank]
```

For InfiniScale III switches:[2]

```
   [-guid] [-sysguid] [-ndesc] [-bsn] [-pe_i2c] [-se_i2c] [-is3_i2c]
```

2.4 Examples of mlxburn Usage

2.4.1 Host Channel Adapter Examples

- To update firmware on an MT25408 ConnectX® adapter device with the configuration file (.ini) auto-detected, enter:

```
mlxburn -fw ./fw-25408-rel.mlx -dev /dev/mst/mt25418_pci_cr0
```

- To generate a failsafe image file for the same adapter above without burning, enter:

```
mlxburn -fw ./fw-25408-rel.mlx -conf ./MHEH28-XTC_A1.ini -wrimage ./fw-25418.bin
```

- To update firmware on the same adapter above with the configuration file (.ini) explicitly specified, enter:

```
mlxburn -fw ./fw-25408-rel.mlx -dev /dev/mst/mt25418_pci_cr0

-conf ./MHEH28-XTC_A1.ini
```

2.4.2 SwitchX™ Switch Examples

- Burn an MSX6025 switch system using the In-Band access method:

```
mlxburn -dev /dev/mst/SW_MT51000_000002c900002100_lid-0x000E -fw ./fw-sx.mlx
```

1. The arguments of the -guids and -macs flags must be provided within quotation marks; for example, mlxburn -macs "0002c900001 0002c900002"
2. The -guid flag is supported by all Mellanox Technologies' adapters and switch devices.

- Generate an MT48436 image and perform an In-Band update of the device with LID 0x18:

```
mlxburn -dev lid-0x000E -fw ./fw-sx.mlx
```

- Generate and burn a new MSX6025 via I2C:
  - Set the I2C network to access the InfiniScaleIV switch.

```
mlxi2c -d /dev/mst/mtusb-1 p /SX
```

- Burn the new image (the flash is still blank) specifying the Node GUID, system GUID, base MAC and Switch MAC. Note that 4 guids (in quotes) should be specified as an argument to the -guids flag. The 2 middle GUIDs are ignored by SwitchX and should be set to 0.

```
mlxburn -d /dev/mst/mtusb-1 -fw ./fw-sx.mlx -conf MSX6025F_A1.ini -guids "000002c900002100 0 0
000002c900002100" -macs "0002c9002100 0002c9002101" -nofs
```

2.4.3 InfiniScale IV Switch Examples

- Burn an MTS3600 switch system via I2C:
  a. Set the I2C network to access the InfiniScale IV switch.

```
mlxi2c -d /dev/mst/mtusb-1 p /IS4_PRIM
```

  b. Burn with quick query (-qq) to shorten burn time

```
mlxburn -dev /dev/mst/mtusb-1 -fw ./fw-IS4.mlx -qq
```

- Burn an MTS3600 switch system using the In-Band access method:

```
mlxburn -dev /dev/mst/SW_MT48438_lid-0x0003 -fw ./fw-IS4.mlx
```

- Generate and burn a new MTS3600 via I2C:
  a. Set the I2C network to access the InfiniScaleIV switch.

```
mlxi2c -d /dev/mst/mtusb-1 p /IS4_PRIM
```

  b. Burn the new image (the flash is still blank) specifying the Node and System GUIDs. Note that 4 guids (in quotes) should be specified as an argument to the -guids flag. The 2 middle GUIDs are ignored by the InfiniScaleIV and should be set to 0.

```
mlxburn -dev /dev/mst/mtusb-1 -fw ./fw-IS4.mlx -conf ./MTS3600Q-1UNC_A1.ini-guids
"0002c9000100d060 0 0 0002c9000100d060" -nofs
```

- Generate and Burn a new MT3600 switch system via I2C in 2 steps:
  a. Generate the image:

```
mlxburn -fw ./fw-IS4.mlx -conf ./MTS3600Q-1UNC_A1.ini -wrimage ./fw-is4.bin
```

  b. Burn using flint tool:

```
flint -d /dev/mst/mtusb-1 -i /tmp/fw-is4.bin -nofs -guids 0002c9000100d060 0 0 0002c9000100d060 b
```

- To generate an MT48436 image and perform an In-Band update of the device with LID 0x18, enter:

```
mlxburn -fw ./fw-IS4.mlx -dev lid-0x18
```

2.4.4 BridgeX Gateway Examples

- To update firmware on BridgeX™ device with the configuration file (.ini) auto-detected, enter:

```
mlxburn -d /dev/mst/mt64102_pci_cr1 -fw ./fw-BridgeX-rel.mlx
```

- To generate a failsafe image file for the same BridgeX above without burning, enter:

```
mlxburn -fw ./fw-BridgeX-rel.mlx -conf ./MTB4020-PC0_A1.ini -wrimage ./fw-BridgeX.bin
```

- To update firmware on the same BridgeX above with the configuration file (.ini) explicitly specified, enter:

```
mlxburn -fw ./fw-BridgeX-rel.mlx -dev /dev/mst/mt64102_pci_cr1-conf ./MTB4020-PC0_A1.ini
```

- To burn a firmware binary file for a BridgeX device, enter:

```
mlxburn -image ./fw-BridgeX.bin -dev /dev/mst/mt64102_pci_cr1
```

2.4.5 InfiniScale III Switch Examples

- To update firmware on an MT47396 InfiniScale III device, enter:

```
mlxburn -fw IS3FW.BIN -dev /dev/mst/mtusb-1
```

This firmware update cannot be performed before initializing the MST device (mtusb-1) to connect to the I2C-compatible bus of the InfiniScale III and its EEPROM.

- To generate an image for the InfiniScale III switch device, enter:

```
mlxburn -fw IS3FW.BIN -conf ./MTS2400-A00.INI -wrimage IS3FW.img
```

The generated firmware image to be burnt to a switch device must have a ‘.img’ file name extension.

- To update firmware on an MT43132 InfiniScale device in a switch system such as Flextronics' F-X430066 Stallion 8 4X IB port switch, enter:

```
mlxburn -image Stallion_5_5_0.eeprom -dev /dev/mst/mtusb-1 -dev_type 43132
```

This firmware update cannot be performed before initializing the MST device (mtusb-1) to connect to the I2C-compatible bus of the InfiniScale and its EEPROM.

- To generate an MT47396 image and perform an In-Band update of the device with LID 0x11, enter:

```
mlxburn -dev lid-0x11 -fw ./IS3FW.BIN
```

2.5 Exit Return Values

The following exit values are returned:

- 0 - successful completion
- >0 - an error occurred

3 flint – Firmware Burning Tools

3.1 Overview

The **flint** (Flash interface) utility performs the following functions:

- Burns a binary firmware image to the Flash device attached to an adapter, bridge or switch device
- Burns an Expansion ROM image to the Flash device attached to a ConnectX / ConnectX-2 adapter device
- Queries for firmware attributes (version, GUIDs, UIDs, MACs, PSID, etc.)
- Enables executing various operations on the Flash memory from the command line (for debug/production)
- Disables/enables the access to the device's hardware registers, and changes the key used for enabling. This feature is functional only if the burnt firmware supports it

3.2 flint Synopsis

flint

```
[switches...] <command> [parameters...]
```

3.2.1 Switch Descriptions

-d[evice] <device> - The device to which the Flash is connected.

Affected commands: All

-i[mage] <image> - Binary image file.

Affected commands: burn, verify

-qq - Quick query the device for firmware information. This operation can be specified when executing either a query or burn operation. When specified, flint will not perform full image integrity checks. This may shorten execution time when running over slow interfaces (e.g., I2C, MTUSB-1).

Affected commands: burn query

-guid <GUID> - Base value for up to 4 GUIDs which are automatically assigned the following values:

guid -> node GUID

guid+1 -> port1

guid+2 -> port2

guid+3 -> system image GUID.

Affected commands: burn sg

-guids <GUIDs...> - 4 GUIDs must be specified here. These GUIDs will be assigned to: node, port1, port2 and system image GUID respectively.

Affected commands: burn sg

-mac <MAC> [1]	- MAC base address value. Each of the two ConnectX EN ports is assigned a MAC address as follows: MAC is assigned to Port 1 MAC+1 is assigned to Port 2 Affected commands: burn sg
-macs <MACs...> [1]	- 2 MAC addresses must be specified here—one for each ConnectX EN port. The first MAC address is assigned to Port 1 and the second MAC address to Port 2. Affected commands: burn sg
-uid <UID>	- BridgeX only. Derive and set the device UIDs (GUIDs, MACs, WWNs). UIDs are derived from the given base UID as per Mellanox's methodology. Affected commands: burn, sg
-uids <UIDs...>	- BridgeX only. 29 space-separated UIDs must be specified here, in the following order: G0-MAC-PI0 G0-MAC-PI1 G0-MAC-PI2 G0-MAC-PE0 G0-MAC-PE1 G0-MAC-PE2 G0-MAC-PE3 G0-FC-WWPN-P0 G0-FC-WWPN-P1 G0-FC-WWPN-P2 G0-FC-WWPN-P3 G0-IB-NODE-GUID G0-IB-PORT-GUID G0-FC-WWNN G1-MAC-PI0 G1-MAC-PI1 G1-MAC-PI2 G1-MAC-PE0 G1-MAC-PE1 G1-MAC-PE2 G1-MAC-PE3 G1-FC-WWPN-P0 G1-FC-WWPN-P1 G1-FC-WWPN-P2 G1-FC-WWPN-P3 G1-IB-NODE-GUID G1-IB-PORT-GUID G1-FC-WWNN IB-SYSTEM-GUID Affected commands: burn, sg
-blank_guids	- Burn the image with blank GUIDs and MACs (where applicable). These values can be set later using the "sg" command (see details below). NOTE: An image that is burnt with blank GUIDs/MACs will fail to boot the machine as long as the GUIDs/MACs are *not* set. Affected commands: burn
-clear_semaphore	- Force the clearing of the Flash semaphore on the device. This flag should come BEFORE the -d[evice] flag in the command line. No command is allowed when this flag is used. NOTE: Using this flag may result in an unstable behavior and flash image corruption if the device or another flash application is currently using the flash. Handle with care.
-byte_mode	- Shift the address when accessing Flash internal registers. May be required for burn/write commands when accessing certain Flash types on InfiniHost®IIIEx HCAs.
-no_flash_verify	- Do not verify each write on the Flash device.

1. The -mac and -macs options are applicable only to Mellanox Technologies Ethenet adapter and switch devices.

-striped_image - Use this flag to indicate that the given image file is in a "striped image" format. See “mlxburn Synopsis” -striped_image flag for details. Affected commands: query verify

-h[elp] - Print this message and exit.

-hh - Print extended command help.

-nofs - Do not burn image in failsafe mode.

-allow_psid_change - Allow burning a FW image with a different PSID (Parameter Set ID)than the one currently on flash. Note that changing a PSID may cause the device to malfunction. Use only if you know what you are doing

-skip_is - Allow burning the FW image without updating the invariant sector to ensure failsafe burning even if the invariant sector of the image is different from the one burnt on the Flash.

Note: Some FW releases may include important changes to the invariant sector that must be included in the FW update process. In these cases, the -skip_is flag should *not* be used. Please refer to the specific FW release notes for details.

-s[ilent] - Print errors only.

Affected commands: burn

-y[es] - Non-interactive mode. Assume the answer "yes" to all questions.

Affected commands: all

-no - Non-interactive mode. Assume the answer "no" to all questions.

Affected commands: all

-vsd <vendor-specific-data> -A VSD string, composed of up to 208 characters, will be written to the VSD

section in the flash. If not specified, the current VSD will be preserved.

-use_image_ps - Burn vsd as it appears in the given image. (Default: Retain current vsd on Flash.)

Affected commands: burn

-use_image_guids - Burn (guids/uids/macs) as appears in the given image.

Affected commands: burn

-use_image_rom - Do not save the ROM which exists in the device.

Affected commands: burn

-dual_image - Make the burn process burn two images on flash (previously default algorithm). Current default failsafe burn process burns a single image (in alternating locations).

Affected commands: burn

-banks <banks> - Set the number of attached Flash devices (banks)

-log <log_file> - Print the burning status to the specified log file

-flash_params <type,log2size,num_of_flashes>

- Use the given parameters to access the flash instead of reading them from the flash.

Supported parameters:

Type: The flash’s type: M25PXxx, M25Pxx, SST25VFxx, W25QxxBV.

log2size: The log2 of the flash size.

num_of_flashes: The number of the flashes connected to the device

-v - Print version information

3.2.2 Command Descriptions

The **flint** utility commands are:

Common FW Update and Query:

b[urn] - Burn flash

q[uery] [full] - Query misc. flash/firmware characteristics, use "full" to get more information.

v[erify] - Verify entire flash

swreset - SW reset the target un-managed switch device. This command is supported only in the In-Band access method.

Expansion ROM Update:

brom <ROM-file> - Burn the specified ROM file on the flash.

drom - Remove the ROM section from the flash.

rrom <out-file> - Read the ROM section from the flash.

Initial Burn, Production:

bb - Burn Block - Burns the given image as is. No checks are done.

sg [nocrc] - Set GUIDs.

sv - Set the VSD.

Misc FW Image operations:

ri <out-file> - Read the fw image on the flash.

dc [out-file] - Dump Configuration: print fw configuration file for the given image.

dh [out-file] - Dump Hash: print hash file for the given image.

HW Access Key:

set_key [key] - Set/Update the HW access key which is used to enable/disable access to HW. The key can be provided in the command line or interactively typed after the command is given.

NOTE: The new key is activated only after the device is reset.

hw_access <enable|disable> [key] - Enable/disable the access to the HW. The key can be provided in the command line or interactively typed after the command is given

Low Level Flash Operations:

cfi - Query flash device parameters

e[rase] <addr> - Erase sector

rw <addr> - Read one dword from flash

ww <addr> < data> - Write one dword to flash

wwne <addr> - Write one dword to flash without sector erase

wbne <addr> <size> <data ...> - Write a data block to flash without sector erase

rb <addr> <size> [out-file] - Read a data block from flash

The following sections provide the command line syntax for the following **flint** utility commands, together with examples of usage.

3.2.2.1 Burning a FW Image

The FLINT utility enables you to burn the Flash from a binary image.

To burn the entire Flash from a raw binary image, use the following command line:

```
flint -d <device> -i <fw-file> [-guid <GUID> | -guids <4 GUIDS> | -mac <MAC> | -macs <2 MACs>]
burn
```

where:

device – Device on which the flash is burned.

fw-file – Binary firmware file.

GUID(s) (optional, for InfiniBand adapters and 4th generation switches) – One or four GUIDs.

If 4 GUIDS are provided (-guids flag), they will be assigned as node, Port 1, Port 2 and system image GUIDs, respectively.
If only one GUID is provided (-guid flag), it will be assigned as node GUID. Its values +1, +2 and +3 will be assigned as Port 1, Port 2 and system image GUID, respectively.
If no -guid/-guids flag is provided, the current GUIDs will be preserved on the device.

For 4th generation, four GUIDs must be specified but Ports 1 and 2 GUIDs are ignored and should be set to 0.

A GUID is a 16-digit hexadecimal number. If less than 16 digits are provided, leading zeros will be inserted.

MAC(s) (optional, for Ethernet and VPI adapters and switches).

If 2 MACs are provided (-macs flag), they will be assigned to Port 1 and Port 2, respectively.
If only one MAC is provided (-mac flag), it will be assigned to Port 1; MAC+1 will be assigned to Port 2.
If no -mac/-macs flag is provided, the current LIDs will be preserved on the device.

A MAC is a 12-digit hexadecimal number. If less than 12 digits are provided, leading zeros will be inserted.

Examples:

1. Update the FW on the device, keeping the current GUIDs and VSD. (Note: This is the common way to use flint.)

```
> flint -d /dev/mst/mt25418_pci_cr0 -i fw-25408-2_1_000-MHGH28-XSC_A1.bin burn
```

2. Update the FW on the device, specifying the GUIDs to burn.

```
> flint -d /dev/mst/mt25418_pci_cr0 -i 25408-2_1_000-MHGH28-XSC_A1.bin-guid 12345678deadbeef
burn
```

3. Update the FW on the device, specifying the MACs to burn.

```
> flint -d /dev/mst/mt25448_pci_cr0 -i fw-25448-6_0_111-MNEH28-XTC_A1.bin-mac 12345678beef burn
```

4. Burn the image on a blank Flash device. This means that no GUIDs are currently burnt on the device, therefore they must be supplied (with -guid/-guids) by the burning command. Moreover, the burn process cannot be failsafe when burning a blank Flash, therefore the -nofs flag must be specified.

```
> flint -d /dev/mst/mt25418_pci_cr0 -i 25408-2_1_000-MHGH28-XSC_A1.bin -nofs-guid
1234567812345678 burn
```

5. Read FW from the device and save it as an image file.

```
> flint -d /dev/mst/mt25418_pci_cr0 ri Flash_Image_Copy.bin
```

6. MT48436 InfiniScale IV switch:

   Burn the image on a blank Flash device. This means that no GUIDs are currently burnt on the device, therefore they must be supplied (with -guid/-guids) by the burning command. Moreover, the burn process cannot be failsafe when burning a blank Flash, therefore the -nofs flag must be specified.

```
> flint -d /dev/mst/mtusb-1 -i /tmp/fw-is4.bin -nofs -guids 0002c9000100d060 0 0 0002c9000100d060
b
```

7. MT48436 InfiniScale IV switch inband firmware update:

```
> flint -d lid-0x18 -i /tmp/fw-is4.bin b
```

8. MT58100 SwitchX switch:

   Burn the image on a blank Flash device. Meaning, no GUIDs/MACs are currently burnt on the device, therefore they must be supplied (with -guid/-guids and -mac/-macs) by the burning command. Moreover, the burn process cannot be failsafe when burning a blank Flash, therefore the -nofs flag must be specified.

```
> flint -d /dev/mst/mtusb-1 -i /tmp/fw-sx.bin -nofs  -guids 000002c900002100 0 0 000002c900002100
-macs 0002c9002100 0002c9002101 b
```

9. MT58100 SwitchX switch inband firmware update:

```
> flint -d lid-0x18 -i /tmp/fw-sx.bin b
```

3.2.2.2 Querying the FW Image

- To query the FW image on a device, use the following command line:

```
flint -d <device> q
```

- To query the FW image in a file, use the following command line:

```
flint -i <image file> q
```

where:

device – Device on which the query is run.

image file – Image file on which the query is run.

Examples:

a. Query the FW on the device.

```
> flint -d /dev/mst/mt25418_pciconf0 query
```

b. Query the FW image file.

```
> flint -i 25408-2_1_000-MHGH28-XSC_A1.bin query
```

3.2.2.3 Verifying the FW Image

- To verify the FW image on the Flash, use the following command line:

```
flint -d <device> v
```

- To verify the FW image in a file, use the following command line:

```
flint -i <image file> v
```

where:

device is the Flash device to verify, and

image file is the image file to verify

Examples:

```
flint -d /dev/mst/mt25418_pci_cr0 v

flint -i ./image_file.bin v
```

3.2.2.4 Managing an Expansion ROM Image

1. To burn an Expansion ROM image, use the following command:

```
 flint -d <mst device> brom <image name>.rom
```

The "brom" command installs the ROM image on the Flash device or replaces an already existing one.

Example:

```
# flint -d /dev/mst/mt25418_pci_cr0 brom example.rom

   Current ROM info on flash: N/A
   New ROM info: type=GPXE version=2.0.100 devid=25418 proto=IB

Burning ROM image    - OK
Restoring signature  - OK
#
```

2. To read an Expansion ROM image to a file, use the following command:

```
flint -d <mst device> rrom <image name>.rom
```

Example:

```
# flint -d /dev/mst/mt25418_pci_cr0 rrom example.rom
# flint -d /dev/mst/mt25418_pci_cr0 q
Image type:      ConnectX
FW Version:      2.6.1410
Rom Info:        type=GPXE version=2.0.100 devid=25418 proto=IB
Device ID:       25418
Chip Revision:   A0
Description:     Node           Port1          Port2          Sys image
GUIDs:           0002c9000100d050 0002c9000100d051 0002c9000100d052 0002c9000100d050
MACs:                               0002c9000001     0002c9000002
Board ID:         (MT_04C0110002)
VSD:
PSID:            MT_04C0110002
#
```

3. To remove the Expansion ROM, use the following command:

```
flint -d <mst device> drom
```

Example:

```
# flint -d /dev/mst/mt25418_pci_cr0 drom

 Removing ROM image   - OK

 Restoring signature  - OK

#
```

3.2.3 Additional Debug / Production Commands

3.2.3.1 Setting GUIDs and MACs

To set GUIDs/MACs/UIDs for the given device, use the ‘sg’ (set guids) command with the -guid(s) -uid(s) and/or -mac(s) flags.

On pre-ConnectX Devices

On pre-ConnectX devices, the ‘sg’ command is applicable only for images with blank (0xff) GUID values and a blank CRC. In other words, the firmware image was generated using the ‘-blank_guids’ flag.

The following is an example of a complete flow, where a blank-GUIDs image is first generated and burnt to the Flash. Then the GUIDs are set using the ‘sg’ command.

1. Generate the image with blank guids.

```
> mlxburn -fw ./fw-25408-rel.mlx -c ./MHGH28-XTC_A1.ini -wrimage \  /tmp/fw-25408-rel-2_3_000-
MHGH28-XTC_A1-noguids.bin -striped_image -blank_guids


-I- Generating image ...

-I- Image generation completed successfully.
```

2. Query the image to verify that the GUIDs and MACs are blank.

```
> flint -i /tmp/fw-25408-rel-2_3_000-MHGH28-XTC_A1-noguids.bin -striped_image q

Image type:      ConnectX

FW Version:      2.3.0

Device ID:       25418

Chip Revision:   A0

Description:     Node             Port1            Port2            Sys image

GUIDs:           ffffffffffffffff ffffffffffffffff ffffffffffffffff ffffffffffffffff

MACs:                             ffffffffffffffff ffffffffffffffff

Board ID:        n/a (MT_04A0110002)

VSD:             n/a

PSID:            MT_04A0110002


Warning: GUIDs/MACs values and their CRC are not set.

>
```

3. Burn the blank-GUIDs image. This image can be pre-burnt on the flash in an early production phase.
   In this example, the ‘flint bb’ command is used in order to burn the image as is.

```
> flint -d /dev/mst/mt25418_pci_cr0 -i /tmp/fw-25408-rel-2_3_000-MHGH28-XTC_A1-noguids.bin bb


Block burn: The given image will be burnt as is. No fields (such
```

```
as GUIDS,VSD) are taken from current image on flash.

Burn process will not be failsafe. No checks will be performed.

ALL flash, including the Invariant Sector will be overwritten.

If this process fails, computer may remain in an inoperable state.


Do you want to continue ? (y/n) [n] : y

100%
```

4. Set the GUIDs and MACs using the ‘flint sg’ command.

```
> flint -d /dev/mst/mt25418_pci_cr0 -guid 0x0002c90001777050 -mac 0x0002c9777051 sg
```

5. Query the image on the Flash to verify that the GUIDs and MACs were set correctly.

```
> flint -d /dev/mst/mt25418_pci_cr0 q

Image type:      ConnectX

FW Version:      2.3.0

Device ID:       25418

Chip Revision:   A0

Description:     Node             Port1            Port2            Sys image

GUIDs:           0002c90001777050 0002c90001777051 0002c90001777052 0002c90001777053

MACs:                             0002c9777051     0002c9777052

Board ID:        n/a (MT_04A0110002)

VSD:             n/a

PSID:            MT_04A0110002
```

On 4th Generation Devices

On 4th generation devices, the “sg” command can operate on both the image file and the image on the flash. When running the “sg” command on an image on the flash, if the GUIDs/MACs/UIDs in the image are non-blank, the flint will re-burn the current image using the given GUIDs/MACs/UIDs.

1. Change the GUIDs/MACs on a device

```
> flint -d /dev/mst/mt25418_pci_cr0 -qq q

-W- Running quick query - Skipping full image integrity checks.


Image type:      ConnectX

FW Version:      2.7.9450

Rom Info:        type=PXE  version=3.0.0 devid=25418 proto=VPI
```

```
Device ID:       25418
Chip Revision:   A0
Description:     Node             Port1            Port2            Sys image
GUIDs:           0002c9000120d050 0002c9000120d051 0002c9000120d052 0002c9000120d053
MACs:                                 02c90120d050     02c90120d051
Board ID:        VSD (MT_04C0110002)
VSD:             VSD
PSID:            MT_04C0110002

> flint -d /dev/mst/mt25418_pci_cr0 -guid 0002c9000abcdef0 -mac 02c90abcdef0 sg

-W- GUIDs are already set, re-burining image with the new GUIDs ...

Setting the GUIDs    - OK
Restoring signature  - OK



> flint -d /dev/mst/mt25418_pci_cr0 -qq q

-W- Running quick query - Skipping full image integrity checks.

Image type:      ConnectX
FW Version:      2.7.9450
Rom Info:        type=PXE  version=3.0.0 devid=25418 proto=VPI
Device ID:       25418
Chip Revision:   A0
Description:     Node             Port1            Port2            Sys image
GUIDs:           0002c9000abcdef0 0002c9000abcdef1 0002c9000abcdef2 0002c9000abcdef3
MACs:                                 02c90abcdef0     02c90abcdef1
Board ID:        VSD (MT_04C0110002)
VSD:             VSD
PSID:            MT_04C0110002
```

2. Change the GUIDs/MACs on an image file:

```
> flint -i /tmp/image.bin -qq q
```

```
-W- Running quick query - Skipping full image integrity checks.

Image type:      ConnectX
FW Version:      2.7.9450
Rom Info:        type=PXE  version=3.0.0 devid=25418 proto=VPI
Device ID:       25418
Chip Revision:   A0
Description:     Node             Port1            Port2            Sys image
GUIDs:           0002c9000120d050 0002c9000120d051 0002c9000120d052 0002c9000120d053
MACs:                                 02c90120d050     02c90120d051
Board ID:        VSD (MT_04C0110002)
VSD:             VSD
PSID:            MT_04C0110002



> flint -i /tmp/image.bin -guid 0002c9000abcdef0 -mac 02c90abcdef0 sg




>                  flint -i /tmp/image.bin -qq q

-W- Running quick query - Skipping full image integrity checks.

Image type:      ConnectX
FW Version:      2.7.9450
Rom Info:        type=PXE  version=3.0.0 devid=25418 proto=VPI
Device ID:       25418
Chip Revision:   A0
Description:     Node             Port1            Port2            Sys image
GUIDs:           0002c9000abcdef0 0002c9000abcdef1 0002c9000abcdef2 0002c9000abcdef3
MACs:                                 02c90abcdef0     02c90abcdef1
Board ID:        VSD (MT_04C0110002)
VSD:             VSD
```

```
PSID:           MT_04C0110002
```

3.2.3.2 Preparing a Binary Firmware Image for Pre-assembly Burning

In some cases, OEMs may prefer to pre-burn the flash before it is assembled on board. To generate an image for pre-burning for 4th generation devices (ConnectX® and newer), use the mlxburn "-striped_image" flag. The "striped image" file layout is identical to the image layout on the flash, hence making it suitable for burning verbatim.

When pre-burning, the GUIDs/MACs inside the image should be unique per device. The following are two methods to pre-burn an image. You can choose the best method suitable for your needs.

Method 1: Pre-burn an Image with Blank GUIDs/MACs:

In this method, the image is generated with blank GUIDs and CRCs. The GUIDs are set after the device is assembled using the flint "sg" command. To set GUIDs take less than 1 second when running on an image with blank GUIDs (through a PCI device).

A device that is burnt with blank GUIDs/MACs will not boot as a functional network device as long as the GUIDs/MACs are not set.

Flow Example:

1. Generate a striped image with blank guids.

```
> mlxburn -fw ./fw-ConnectX2-rel.mlx -./MHQH29B-XTR_A1.ini -wrimage
                                    ./fw-ConnectX2-rel.bin -striped_image -blank_guids


-I- Generating image ...
-I- Image generation completed successfully.
```

2. Burn the image to a flash using an external burner.
3. (Optional) After assembly, query the image on flash to verify there are no guids on the device.

```
> flint -d /dev/mst/mt26428_pci_cr0 q


Image type:      ConnectX
FW Version:      2.8.0
Device ID:       26428
Chip Revision:   B0
Description:     Node             Port1            Port2            Sys image
GUIDs:           ffffffffffffffff ffffffffffffffff ffffffffffffffff ffffffffffffffff
```

```
MACs:                                ffffffffffffffff     ffffffffffffffff

Board ID:        n/a (MT_0D80110009)

VSD:             n/a

PSID:            MT_0D80110009


-W- GUIDs/MACs values and their CRC are not set.
```

4. Set the correct GUIDs. Since the image is with blank GUIDs, this operation takes less than 1 second

```
> flint -d /dev/mst/mt26428_pci_cr0 -guid 0x0002c9030abcdef0 -mac 0x0002c9bcdef1 sg
```

5. Query the image on flash to verify that the GUIDs are set correctly.

```
> flint -d /dev/mst/mt26428_pci_cr0 q


Image type:      ConnectX

FW Version:      2.8.0

Device ID:       26428

Chip Revision:   B0

Description:     Node             Port1            Port2            Sys image

GUIDs:           0002c9030abcdef0 0002c9030abcdef1 0002c9030abcdef2 0002c9030abcdef3

MACs:                                0002c9bcdef1     0002c9bcdef2

Board ID:        n/a (MT_0D80110009)

VSD:             n/a

PSID:            MT_0D80110009
```

Method 2: Pre-burn an Image with Specific GUIDs/MACs for Each Device:

In this method, a "base" image is generated with arbitrary default GUIDs and then updated with the correct guids for each device.

Example Flow:

1. Generate the base image with arbitrary default GUIDs

```
> mlxburn -fw ./fw-ConnectX2-rel.mlx -c ./MHQH29B-XTR_A1.ini -wrimage  ./fw-ConnectX2-rel.bin -
striped_image
```

2. Per device, set the device specific GUIDs in the image.

```
>  flint -i ./fw-ConnectX2-rel.bin -guid 0x0002c9030abcdef0 -mac 0x0002c9bcdef1 -striped_image sg
```

3. (Optional) Query the image to verify the GUIDs are set. The "-striped_image" flag must be specified when querying a striped image.

```
>  flint -i ./fw-ConnectX2-rel.bin -striped_image q


Image type:      ConnectX
FW Version:      2.8.0
Device ID:       26428
Chip Revision:   B0
Description:     Node             Port1            Port2            Sys image
GUIDs:           0002c9030abcdef0 0002c9030abcdef1 0002c9030abcdef2 0002c9030abcdef3
MACs:                                 0002c9bcdef1     0002c9bcdef2
Board ID:        n/a (MT_0D80110009)
VSD:             n/a
PSID:            MT_0D80110009
```

Now the fw-ConnectX2-rel.bin image can be pre-burned to the flash. After the assembly the device would be fully functional.

3.2.3.3 Setting the VSD

To set the vsd for the given image/device(4th generation), use the 'sv' command with -vsd flag.

Example:

```
> flint -d /dev/mst/mt25418_pci_cr0 -vsd  "VSD" sv
Setting the VSD      - OK
Restoring signature  - OK


> flint -d /dev/mst/mt25418_pci_cr0 q
Image type:      ConnectX
FW Version:      2.7.9450
Rom Info:        type=PXE  version=3.0.0 devid=25418 proto=VPI
Device ID:       25418
Chip Revision:   A0
Description:     Node             Port1            Port2            Sys image
GUIDs:           0002c9000120d050 0002c9000120d051 0002c9000120d052 0002c9000120d053
MACs:                                 02c90120d050     02c90120d051
Board ID:        VSD (MT_04C0110002)
```

```
VSD:            VSD

PSID:           MT_04C0110002
```

3.2.3.4 Disabling/enabling Access to the Hardware

The secure host feature enables ConnectX® family devices to block access to its internal hardware registers. The hardware access in this mode is allowed only if a correct 64 bits key is provided

Examples:

1. Set the key:

```
flint  -d /dev/mst/mt26428_pci_cr0 set_key 22062011

Setting the HW Key   - OK

Restoring signature  - OK
```

A driver restart is required to activate the new key

2. Access the FW while it's disable:

```
flint  -d /dev/mst/mt26428_pci_cr0 -qq q
E- Cannot open /dev/mst/mt26428_pci_cr0: HW access is disabled on the device.
E- Run "flint -d /dev/mst/mt26428_pci_cr0 hw_access enable" in order to enable HW access.
```

3. Enable the FW:

```
flint  -d /dev/mst/mt26428_pci_cr0 hw_access enable

Enter Key: ********
```

4. Disable the FW:

```
flint  -d /dev/mst/mt26428_pci_cr0 hw_access disable
```

3.2.3.5 Reading a Word from Flash

To read one dword from Flash memory, use the following command line:

```
flint -d <device> rw addr
```

where:

device is the device the dword is read from.

addr is the address of the word to read.

Example:

```
flint -d /dev/mst/mt23108_pci_cr0 rw 0x20
```

3.2.3.6 Writing a Dword to Flash

To write one dword to Flash memory, use the following command line:

```
flint -d <device> ww addr data
```

where:

device is the device the dword is written to.

addr is the address of the word to write.

data is the value of the word.

Example:

```
flint -d /dev/mst/mt23108_pci_conf01 ww 0x10008 0x5a445a44
```

3.2.3.7 Writing a dword to Flash Without Sector Erase

To write one dword to Flash memory without sector erase , use the following command line:

```
flint -d <device> wwne addr data
```

where:

device – the device the dword is written to.

addr – the address of the word to write.

data – the value of the word.

Example:

```
flint -d /dev/mst/mt23108_pci_cr0 wwne 0x10008 0x5a445a44
```

Note that the result may be dependent on the Flash type. Usually, bitwise and between the specified word and the previous Flash contents will be written to the specified address.

3.2.3.8 Erasing a Sector

To erase a sector that contains a specified address, use the following command line:

```
flint -d <device> e addr
```

where:

device is the device the sector is erased from, and

addr is the address of a word in the sector that you want to erase.

Example:

```
flint -d /dev/mst/mtusb-1 e 0x1000
```

4 spark - InfiniScale® III Firmware Burning Tool

4.1 Overview

The **spark** tool enables burning a binary firmware image to the EEPROM device attached to an InfiniScale (MT43132) or InfiniScale III (MT47396) switch device via a direct I2C connection or via vendor-specific MADs over the InfiniBand fabric. This tool is described in Section 4.2.

4.2 spark

4.2.1 spark Synopsis

spark

```
[switches...] <command> [parameters...]
```

where switches are:

-d[evice] <device> - defines the Mellanox device to which the EEPROM is connected.
Affected commands: All (see the commands below)

-i[mage] <image> - Image file is in ".img" format.
Affected commands: burn, verify, query

-guid <GUID> - Uses the given guid as the node guid of the burnt image. By default, the guid is taken from the image on the EEPROM.
Affected commands: burn

-sysguid <GUID> - Use the given guid as the system image guid of the burnt image. By default, this value is taken from the current image on the EEPROM.
Affected commands: burn

-bsn <BSN> - Mellanox Board Serial Number (BSN). The vallid BSN format is:
MTxxxxx[-]R[xx]ddmmyy-nnn[-cc]. By default, this value is taken from the current image on the EEPROM.
Affected commands: burn

-ndesc <Descr> - Use the given string (max 64 characters) as the node description. By default, this value is taken from the current image on the EEPROM.
Affected commands: burn

-is1 - Indicate that the target EEPROM to be burnt is for an InfiniScale (MT43132) device. If this option is not specified, the target EEPROM is for an InfiniScale III (MT47396) device. Not supported for In-band access.
Affected commands: burn

-is3_i2c <i2c_addr> - Provides the I2C address of the switch device. If this flag is not specified, then the the default address for Mellanox switch devices is: 0x6c.

-pe_i2c <i2c_addr> - Provides the I2C address of the primary EEPROM. By default, this address is read from the Mellanox switch device. Use this flag only if the switch device is not accessible.

-se_i2c <i2c_addr> - Provides the I2C address of the secondary EEPROM. By default, this address is read from the Mellanox switch device. Use this flag only if the switch device is not accessible.

-h[elp] - Prints this help message and exits.

-hh - Prints an extended command help

-nofs - Do *not* burn the firmware image in failsafe mode.

-s[ilent] - Print errors only.

Affected commands: burn

-sim - Simulates an EEPROM burn without actually writing the EEPROM. Use this flag to compare the image currently on the EEPROM with the given image file.

Affected commands: burn

-y[es] - Non-interactive mode. Assume the answer to all questions is "yes".

Affected commands: All

-v - Version information.

The commands of **spark** are:

b[urn] - Burns the binary image to the EEPROM.

Parameters: None

Examples:

spark -d /dev/mst/mtusb-1 -i image1.img burn

spark -d /dev/mst/mtusb-1 -guid 0x2c9000100d050 -i image1.img b

q[uery] - Queries miscellaneous EEPROM and firmware characteristics

Parameters: None

Example:

spark -d /dev/mst/mtusb-1 query

v[erify] - Verifies the entire EEPROM

Parameters: None

Example:

spark -d /dev/mst/mtusb-1 v

bb - Burns the given image as is (Burn Block). No checks are performed on EEPROM or on the given image. Also no fields (e.g., BSN or GUIDs) are read from the EEPROM.

Parameters: None

Example:

spark -d /dev/mst/mtusb-1 -i image1.img bb

ri - Reads the firmware image on the EEPROM and writes it to a file.

Parameters: filename to write the image to (in .img format)

Example:

spark -d /dev/mst/mtusb-1 ri file.img

rb - Reads a block of data from a single eeprom to the given file.
Parameters: <eeprom address> <start offset> <data size> <output file name>

Example:
spark -d /dev/mst/mtusb-1 rb 0x56 0x0 0x1000 out.img

swreset - SW-resets the target switch device. In response to this command, the target switch device:

continues to transport packets to their destinations

ignores management packets that are destined to the target switch device itself

resets itself after about 15 seconds from receiving the command. The 15-second delay is intended to allow for spark to reset other switches in the fabric.

Example:

spark -d 0x13 swreset // this command SW-resets the switch device with LID 0x13

Appendix A: PSID Assignment

In some cases, OEMs or board manufacturers may wish to use a specific FW configuration not supplied by Mellanox. After setting the new FW parameters in an INI file, the user should assign a unique PSID (Parameter Set ID) to this new configuration. The PSID is kept as part of the FW image on the device NVMEM. The firmware burning tools use this field to retain FW settings while updating FW versions.

This appendix explains how to assign a new PSID for a user customized FW, and how to indicate to the burning tools that a new PSID exists.

Please change FW parameters with caution. A faulty setting of FW parameters may result in undefined behavior of the burnt device.

A.1 PSID Field Structure

The PSID field is a 16-ascii (byte) character string. If the assigned PSID length is less then 16 characters, the remaining characters are filled with binary 0s by the burning tool.

Table 6 provides the format of a PSID.

Table 6 - PSID format

Vendor symbol	Board Type Symbol	Board Version Symbol	Parameter Set Number	Reserved
3 characters	3 characters	3 characters	4 characters	3 characters (filled with '\0')

Example:

A PSID for Mellanox's MHXL-CF128-T HCA board is MT_0030000001, where:

MT_	Mellanox vendor symbol
003	MHXL-CF128-T board symbol
000	Board version symbol
0001	Parameter Set Number

A.2 PSID Assignment and Integration Flow

To assign and integrate the new PSID to produce the new FW

1. Write the new FW configuration file (in .INI format).
2. Assign it with a PSID in the format described above. Use your own vendor symbol to assure PSID uniqueness.
   If you do not know your vendor symbol, please contact your local Mellanox FAE.
3. Set the PSID parameter in the new FW configuration file.

Appendix B: Flow Examples - mlxburn

To update an MT47396 InfiniScale® III and MT48436 InfiniScale® IV switch devices having a specific GUID (for example, 0x00000006660abcd0) or LID, the following are the recommended steps to update the device firmware.

For Linux device names should be listed with the /dev/mst prefix. For Windows, no prefix is required.

1. Make sure all subnet ports are in the active state. One way to check this is to run *opensm*, the Subnet Manager.

```
[root@mymach]> /etc/init.d/opensmd start

opensm start      [  OK  ]
```

2. Make sure the local ports are active by running 'ibv_devinfo'.
3. Obtain the device LID. There are two ways to do that:
   a. Using the "mst ib add" command:

      The "mst ib add"runs the ibdiagnet/ibnetdiscover tool to discover the InfiniBand fabric and then lists the discovered IB nodes as an mst device. These devices can be used for access by other MFT tools.

```
[root@mymach]> mst ib add

-I- Discovering the fabric - Running: /opt/bin/ibdiagnet -skip all

-I- Added 3 in-band devices
```

      To list the discovered mst inband devices run "mst status".

```
[root@mymach]> mst status

MST modules:

------------

    MST PCI module loaded

    MST PCI configuration module loaded


...


Inband devices:

-------------------

/dev/mst/CA_MT25418_sw005_HCA-1_lid-0x0001

/dev/mst/SW_MT47396_lid-0x0011
```

```
/dev/mst/SW_MT48438_lid-0x0003

[root@mymach]>
```

b. Using the ibnetdiscover tool, run:

```
[root@mymach]# ibnetdiscover | grep 00000006660abcd0 | grep -w Switch

Switch  24 "S-00000006660abcd0"   "MT47396 Infiniscale-III Mellanox Technologies" base port 0 lid
17 lmc 0
```

The resulting LID is given as a decimal number.

1. Run *mlxburn* with the LID retrieved in step #3 above to perform the In-Band burning operation.

   Burn the InfiniScale III switch:

```
[root@mymach]> mlxburn -dev /dev/mst/SW_MT47396_lid-0x0011 -fw ./IS3FW.BIN
-I- Reading PSID from board
-I- Using auto detected configuration file ./MT34000LE-A00.INI (PSID =  MT_0070000001)
-I- Generating image ...
- Checking primary image     -  OK
   Current FW Version:  1.0.0
   New FW Version:      1.0.0
- Burning secondary image    -  OK
- Verifying secondary image  -  OK
- Burning primary image      -  OK
- Verifying primary image    -  OK
-I- Image burn completed successfully.
```

   Burn the InfiniScale IV switch:

```
[root@mymach]> mlxburn -d /dev/mst/SW_MT48438_lid-0x0003 -fw ./fw-IS4.mlx -qq
-I- Querying device ...
-I- Using auto detected configuration file: ./MTS3600Q-1UNC_A1.ini (PSID = MT_0C20110003)
-I- Generating image ...

*** WARNING *** Running quick query - Skipping full image integrity checks.

Current FW version on flash: 7.0.135
New FW version:              7.0.138

Burning second FW image without signatures - OK
Restoring second signature                 - OK
-I- Image burn completed successfully.
```

Appendix C: Debug Utilities

C.1 itrace Utility

C.1.1 Overview

The itrace utility extracts and prints trace messages generated by the embedded iRISC processors of Mellanox Technologies' InfiniBand or Ethernet adapter devices. These trace messages inform developers of software drivers about internal status, events, critical errors, etc., for each iRISC. Trace messages generated by iRISCs are stored in the trace buffer. The trace buffer is located in host memory for MemFree adapter cards (i.e., without on-board memory), and in adapter memory for adapter cards with on-board memory.

The utility is a command line application controlled by command line parameters. It prints trace messages in text format to the console.

For more details, see the QUERY_DEBUG_MESSAGE command interface description in the device's *Programmer's Reference Manual* available via https://docs.mellanox.com (requires a customer login account).

C.1.2 Operation

In order to print the firmware traces, it is required that

- Debug firmware is burnt and loaded to the device
- The driver is up. Specifically, this means that
  - For adapters with on-board memory: The SYS_ENABLE command has been executed
  - For adapters without on-board memory (MemFree): The RUN_FW command has been executed
- The desired trace mask is set (see the -m flag below)

The MST driver must be started prior to running itrace tool. To start itrace:

1. Start the MST driver (mst start[1] or mst restart[1]).
2. Enter the following command:

```
itrace [options...] IRISC_NAME
```

where:

IRISC_NAME – is the iRISC for which traces are to be printed. This can be specified once anywhere in the command line as a special option without the leading hyphen. Run 'itrace -h' to get a list of iRISC names for each adapter device.

[options] can specify any of the following:

-h, --help – Displays help about itrace usage.

-m, --mask=*TRACE_MASK*– Sets the Trace Mask.

1. This step in not required in Windows.

To enable generating trace messages for an iRISC, the trace_mask register must be set according to the specifications in the device's *Programmer's Reference Manual*. Setting or clearing bits of the trace_mask register enables or disables, respectively, the generation of specific types of trace messages. The TRACE_MASK parameter must be a hexadecimal or decimal number and its value will be written into the trace_mask register. Changing trace_mask will not change or remove messages previously stored in the trace buffer, so disabled types of messages can still be displayed by itrace if they were previously generated.

Example:

For Linux device names should be listed with the /dev/mst prefix. For Windows, no prefix is required.

```
itrace -d /dev/mst/mt25204_pci_cr0 --nomap -m 0xffffffff tpt


This generates output regarding the sequence numbers, timestamps, and records of operations, such
as the following:

IRISC Trace Viewer (Mellanox InfiniHost, V4.4.2, Jul 31 2007 16:56:59)

FW Version: 1.2.922 09/07/2007 20:36:53


(00000001 c1b59bd1) SCHD: SQP:0x000402 exes_super_scheduler:busy_done

(00000003 c1b59e4e) SCHD: exeqpc_valid2freed(0x0) vec_busy_valid=0x00000010

(00000004 dda895e4) SCHD: SQP:0x000400 exes_super_scheduler:busy_done

(00000005 dda89760) SCHD: writing QpState SQPSTATE_GOOD_IDLE!!!!

(00000006 dda89868) SCHD: exeqpc_valid2freed(0x0) vec_busy_valid=0x00000010

(00000007 dda97ccf) SCHD: SQP:0x000400 exes_super_scheduler:busy_

(00000008 dda97e47) SCHD: writing QpState SQPSTATE_GOOD_IDLE!!!!

(00000009 dda97f4f) SCHD: exeqpc_valid2freed(0x0) vec_busy_valid=0x00000010

(0000000a dda9a8f6) SCHD: SQP:0x000400 exes_super_scheduler:busy

(0000000b dda9aa6e) SCHD: writing QpState SQPSTATE_GOOD_IDLE!!!!

(0000000c dda9ab79) SCHD: exeqpc_valid2freed(0x0) vec_busy_valid=0x00000010

(0000000d ddaaadc1) SCHD: SQP:0x000400 exes_super_scheduler:busy_

(00000029 ddaee521) INFO: IPCdata[00]=0x01abcd0a

(0000002a ddaee60c) INFO: IPCdata[01]=0x00000014

(0000002b ddaee8ce) MAD: exes_mad: QPN=0x000000, nda_nds=0x7c58d014

(0000002c ddaee9f2) SCHD: SQP:0x000000 sqpc_access_db_algorithm: INC
```

```
(0000002d ddaef0d5) SCHD:              exes_scheduler: try to insert

(0000002e ddaef2d9) SCHD: SQP:0x000000 exes_scheduler chosen

(0000002f ddaef6aa) SCHD:            EXES_GO(0x0)

.

.

.
```

-w, **--wait** - Runs itrace in wait mode. itrace will exit only if you press <Ctrl-C>. This is not the default behavior of itrace. Without the -w option, itrace will exit if there have been no new traces in the last 0.5 seconds.

-d, --d=DEVICE - Specifies the name of the MST device driver for accessing the cr-space. The default value is:/dev/mst/mt23108_pci_cr0.

To run itrace via the I2C interface, use this option to specify the following:

--d=device, where the device is an I2C device (such as mtusb-1)

--nomap - Sets itrace not to directly access memory (via memory mapping) for reading the trace buffer, but to use the adapter memory access Gateway instead. By default, itrace accesses the memory directly. If the cr-space device specified by the -d parameter is one of the I2C devices, -nomap is switched on.

--no-propel - Sets itrace not to animate the propeller in wait mode (-w option). By default, animation is enabled.

-v, --version - Prints the MFT version and exits

-c, --color - Enables color in trace output

-D, --dump - Dumps the trace buffer and exits. This option is useful for debugging itrace; it dumps the contents of the trace buffer in row format.

Typing **`--help`** at the command line displays manual pages describing the syntax of the itrace utility.

C.2 mstdump Utility

The mstdump utility dumps device internal configuration data. The data can be used by for hardware troubleshooting. It can be applied to all Mellanox adapter devices, BridgeX device, InfiniScale III switch devices and 4th generation switch devices.

C.2.1 Operation

The MST driver must be started prior to running mstdump tool. To start mstdump:

1. Start the MST driver (mst start[1] or mst restart[1]).

1. This step in not required in Windows.

2. Enter an mstdump command that complies with the following command syntax:

```
mstdump [-full] <mst device>  >  <dump file>
```

where the **-full** flag dumps all internal registers.

On BridgeX devices, using the -full flag may have undesired side-effects and require resetting the device.

Example:

```
[root@mymach]# mstdump /dev/mst/mt25408_pci_cr0 > mt25408.dmp
```

This dumps the internal configuration data of the device into the file mt25408.dmp.

C.3 mlxi2c Utility

The mlxi2c utility provides access, via the I2C MST device, to Mellanox 4th generation switch and bridge (BridgeX) devices.

C.3.1 Operation

The MST driver must be started prior to running mlxi2c. **To start mlxi2c**:

1. Start the MST driver (mst start[1] or mst restart).
2. Run mlxi2c with the following command line syntax:

```
mlxi2c [switches...] <command> [parameters...]
```

[switches...] summary:

-d <device>	- MST i2c device name default: "/dev/mst/mtusb-1".
	Affected commands: all
-s <system>	- System type default: Auto Detection.
	Supported systems: MTS3600, FJ_SWITCH, BX4020, BRIDGEX_EVB, BX4010, MTS3610, MTPDK24, IS5025, IS5030, IS5035.
	Affected commands: all
-m	- Subsystem module name.
	Affected commands: show
-h	- Print this help information.
-v	- Print version and exit.

<command> summary:

q <i2c_component>	- Query an i2c component. Supported components: Adm1024, LM075, PCA9555, PCA9505, ISMIC, PCF8591.
p <i2c_component>	- Route the i2c path to the indicated i2c component.
init_gpio <gpio_name>	- Configure the GPIO pins.

set_gpio <gpio_name> <pin_name> <value> - Set the output value of the given GPIO pin.

set_fan <fan_name> <Tach_num> <power_persent> - Set the power percent of the given fan.

set_power <module_name> <off/on> - Power on/off the specified module. Supported modules: MTS3610_LEAF, MTS3610_SPINE.

set_led <led_name> <color> - Change the color of the specify led to the given color.

scan - Scan the i2c slave addresses

show <info_type> - Show information about the system or specific subsystem module (related to -m flag). The information types are:

devs [-v]: List the i2c components. (-v) adds detailed information to the list.

modules: List the detachable subsystems modules.

temp: The temperatures of the components: ADM1024, LM075, PCF8591.

volt: The voltage values of: ADM1024, PCF8591.

ports: The present qsfp ports on the system.

fans: The speed of fans.

power: The power/voltage/current values of the power supplies.

slot_num: The slot number of the system (only for: FJ_SWITCH).

leds: The current colors of the LEDs.

inventory: The inventory of the present modules.

Examples:

1. Point to an InfiniScale IV device to enable accessing it directly by firmware utilities.

```
mlxi2c -d /dev/mst/mtusb-1 p /IS4_PRIM
```

2. Display the temperature values of InfiniScale IV or BridgeX temperature sensors in the system.

```
mlxi2c -d /dev/mst/mtusb-1 show temp
```

3. Display the addresses of all I2C-accessible devices.

```
mlxi2c -d /dev/mst/mtusb-1 scan
```

C.4 i2c Utility

The i2c utility provides low level access to the I2C bus on any Mellanox switch platform, enabling the user to read or write data.

C.4.1 Operation

The MST driver must be started prior to running i2c tool. To start i2c:

1. Start the MST driver (mst start[1] or mst restart[1]).

1. This step in not required in Windows.

2. Run i2c with the following command line syntax:

```
i2c [OPTIONS] <device> <cmd> <i2c_addr> <addr> [<data>]
```

where `[OPTIONS]` can be the following:

`-h`	– Prints this message.
`-a` <addr_width>	– Sets address width (in bytes) to the specified value. May be 0, 1, 2 or 4. Default: 1.
`-d` <data_width>	– Sets data width (in bytes) to the specified value. May be 1, 2 or 4. Default is 1.
`-x <data_len>`	– Presents each byte of data as two hexadecimal digits (such as 013C20343B). Note that this option is mutually exclusive with the "-d" option.

The remaining parameters are:

<device>	– Valid MST device.
`<cmd>`	– Command. May be "r[ead]" or "w[rite]".
`<i2c_addr>`	– I2C slave address.
<addr>	– Address (of length addr_width) inside I2C target device to read/write operation. Note that the <addr> value is ignored if <addr_witdh> = 0.
<data>	– Data (bytes of length data_width) to write to target device.

All parameters are interpreted as hexadecimal values.

Examples:

1. Read two bytes from address 0 of target I2C device at address 0x56:

```
> i2c -a 1 -d 2 /dev/mst/mtusb-1 r 0x56 0x00

0000
```

2. Write two bytes to the address above then read them:

```
> i2c -a 1 -d 2 /dev/mst/mtusb-1 w 0x56 0x00 0x1234

> i2c -a 1 -d 2 /dev/mst/mtusb-1 r 0x56 0x00

3412
```

3. Read (as separate) 16 bytes in hexadecimal format starting from address 0 of the target device above:

```
> i2c -a 1 -x 16 /dev/mst/mtusb-1 r 0x56 0x00

12340000000000000000000000000000
```

C.4.2 Exit Return Values

The following exit values are returned:

- 0 - successful completion
- >0 - an error occurred

C.5 mget_temp Utility

The mget_temp utility reads the hardware temperature from Mellanox Technologies devices with temperature sensors (ConnectX®, ConnectX®-2, ConnectX®-3, BridgeX devices, and 4th generation switches), and prints the reading in Celsius degrees.

C.5.1 Operation

The MST driver must be started prior to running mget_temp. To start mget_temp:

1. Start the MST driver (mst start[1] or mst restart).
2. Run mget_temp with the following command line syntax:

```
mget_temp [OPTIONS]
```

where [OPTIONS] are:

-h	- Print this message.
-d <dev>	- mst device name
-i	- activate and initiate thermal diodes for temperature measurement (intrusive)

On some systems, the thermal sensors are not activated by default. To initiate the thermal sensors, run "mget_temp -i"once after device power up. Then wait for at least half a second before running "mget_temp" for reading the temperature.

Examples:

The following steps show how to initialize and then read temperature sensors.

1. Initiate thermal sensors:

```
mget_temp -d /dev/mst/SW_MT48438_lid-0x000F -i
```

2. Wait half a second and then read the temperature by running:

```
mget_temp -d /dev/mst/SW_MT48438_lid-0x000F
```

1. This step in not required in Windows.

Appendix D: In-Band Access to Multiple IB Subnets

In most cases, an adapter is connected to a single InfiniBand subnet. The LIDs (Infiniband Local IDs) on this subnet are unique. In this state, the device access MADs are sent (to the target LID) from the first active port on the first adapter on the machine.

In case that the different IB ports are connected to different IB subnets, source IB port on the local host should be specified explicitly.

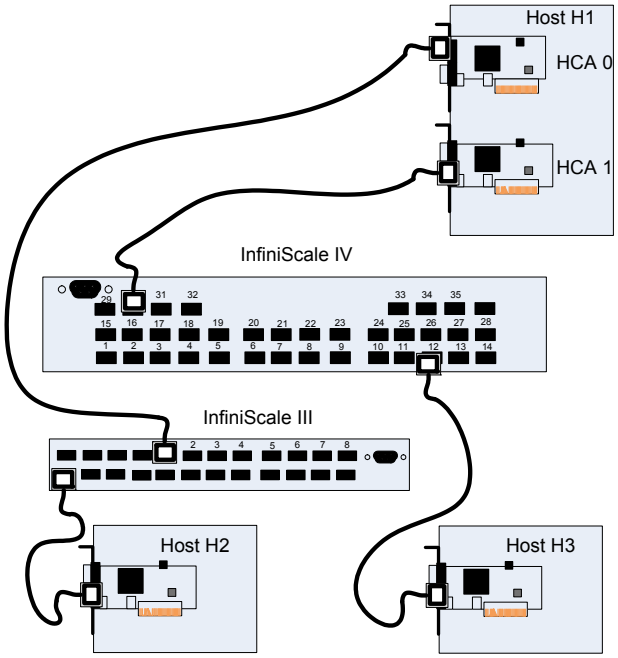
The device name would be in the format:

```
<any-string>lid-<lid-number>[,source adapter name][,source IB port number]
```

For example:

- On Linux: lid-3,mlx4_0,1
- On Windows: lid-3,0,1

Say we have the following setup:



H1 host has 2 adapters. Port 1 of the first adapter is conected to the InfiniScale III switch, and port 2 of the second adapter is connected to the InfiniScale IV switch. Since the 2 adapters on the H1 are not connected to the each other, there are 2 seperate IB subnets in this setup.

Subnet1 nodes: H1 IS3 switch and H2

Subnet2 nodes: H1 IS4 switch and H3

Running "ibv_devinfo" command on H1 would list the 2 adapter names. For ConnectX adapters, the names would be mlx4_0 and mlx4_1.

Running "mst ib add" would add ib devices from the default port (first active port on the first adapter) - only Subnet1 nodes would be listed.

To add the nodes of the second subnet, the source adapter and port should be specified to the "mst ib add" command in the following format:

```
mst ib add <hca_name> <hca_port>
```

Example:

1. Add nodes of both subnets, Run:

```
> mst ib add mlx4_0 1

> mst ib add mlx4_1 2
```

2. List the devices:

```
> mst status

...

/dev/mst/CA_MT25418_H1_HCA-1_lid-0x0001,mlx4_0,1

/dev/mst/CA_MT25418_H2_HCA-1_lid-0x0005,mlx4_0,1

/dev/mst/SW_MT47396_lid-0x0003,mlx4_0,1


/dev/mst/CA_MT25418_H1_HCA-1_lid-0x0010,mlx4_1,2

/dev/mst/CA_MT25418_H3_HCA-1_lid-0x0012,mlx4_1,2

/dev/mst/SW_MT48436_lid-0x0005,mlx4_1,2
```

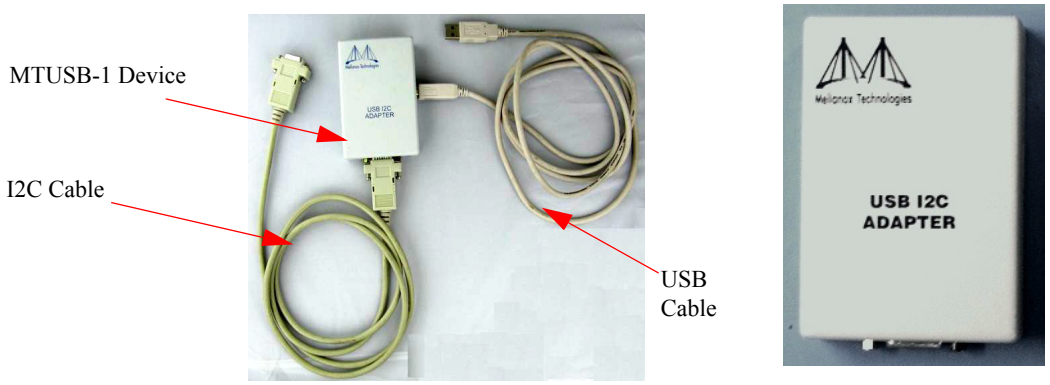
You can use the above device names with the MFT tools.

Appendix E: MTUSB-1 USB to I2C Adapter

E.1 Overview

The MTUSB-1 is a USB to I^2C bus adapter. This chapter provides the user with hardware and software installation instructions on machines running Linux or Windows operating systems.

Figure 3: MTUSB-1 Device



E.1.1 Package Contents

Please make sure that your package contains the items listed in Table 7 and that they are in good condition.

Table 7 - MTUSB-1 Package Contents

Item	Quantity	Description
MTUSB-1 device	1	USB to I^2C bus adapter
USB cable	1	USB_A to USB_B (1.8m)
I2C cable	1	9-pin male-to-male cable (1.5m)
Converter cable	2	9-pin female to 3-pin (small/large) (0.3m)

E.1.2 System Requirements

The MTUSB-1 is a USB device which may be connected to any Personal Computer with a USB Host Adapter (USB Standard 1.1 or later) and having at least one USB connection port.

E.1.3 Supported Platforms

MTUSB-1 supports the same platforms that are supported by the MFT tools package.

E.2 Hardware Installation

To install the MTUSB-1 hardware, please execute the following steps in the *exact* order:

1. Connect one end of the USB cable to the MTUSB-1 and the other end to the PC.
2. Connect one end of the I2C cable to the MTUSB-1 and the other end to the system/board you wish to control via the I2C interface. If the system/board uses a 3-pin connector instead of a 9-pin connector, connect the appropriate converter cable as an extension to the I2C cable on the 9-pin end, then connect its 3-pin end to the system/board.

E.3 Software Installation

The MTUSB-1 device requires that the Mellanox Firmware Tools (MFT) package be installed on the machine to which MTUSB-1 is connected – see Section 1.4,"MFT Installation," on page 10 of this manual for installation instructions.

For a Windows machine, it is also required to install the MTUSB-1 driver – visit http://www.dio-lan.com to download this driver. This driver is required for the first use of the MTUSB-1 device.

Once you have the requirements installed, you may verify that your MTUSB-1 device is detected by MFT software as described below.

1. Start the *mst*[1] driver. Enter:

```
mst start          (or mst restart if mst start was run earlier)
```

2. To obtain the list of *mst* devices, enter:

```
mst status
```

   If MTUSB-1 has been correctly installed, **"mst status"** should include the following device in the device list it generates:

   - On Linux: /dev/mst/mtusb-1
   - On Windows: mtusb-1

1. This step in not required in Windows.