



MSTFLINT Package -Firmware Burning and Diagnostics Tools Documentation v4.26

Table of Contents

Overview	4
Intended Audience	4
Document Revision History	4
Release Notes.....	5
Release Notes Update History.....	5
General Information	5
Package Tools	5
Dependencies.....	6
mstflint Supported Operating Systems and Platforms.....	6
Supported Flash Types.....	10
Supported Interface Cards (NICs).....	10
Supported Adapter Cards Firmware Versions	11
Supported Switch Systems Software	11
Changes and New Features	12
mstflint Bug Fixes in this Version.....	12
mstflint Known Issues	13
User Manual	16
Supported Operating Systems.....	16
Access to Hardware Devices.....	16
Compilation and Installation	17
Firmware Generation, Configuration, and Update Tools.....	17
mstfwmanager - Firmware Update and Query Tool.....	18
mstarchive - Binary Files Compression Tool.....	21
mstconfig - Changing Device Configuration Tool	22
mstflint - Firmware Burning Tool	31
mstfwreset - Loading Firmware on 5th Generation Devices Tool.....	61
mstcongestion - Tool for Setting Congestion Mode and Action.....	67
mstprivhost - NIC Configuration by the Host Restriction Tool.....	68
Debug Utilities.....	69
mstfwtrace Utility	69
mstregdump Utility	71
mstreg Utility.....	71

mstlink Utility	73
mstresourcedump Utility	90
mstresourceparse Utility.....	92
Troubleshooting	95
General Related Issues	95
mstconfig Related Issues.....	95
Installation Related Issues	96
Firmware Burning Related Issues	96
Secure Firmware Related Issues	98
Appendixes.....	99
mstflint - Updating Firmware Using ethtool/devlink and .mfa2 File	99
Document Revision History	101
Release Notes Revision History	101
mstflint Release Notes Change Log History	101
mstflint Bug Fixes History	116
mstflint User Manual Revision History	122
Legal Notices and 3rd Party Licenses	126

Overview

The NVIDIA® Firmware Tools (mstflint) package is a set of firmware management and debug tools for NVIDIA devices. The document describes mstflint features, tools content and configuration.

The documentation here relates to:

- [Release Notes](#)
- [User Manual](#)

Intended Audience

This manual is intended for system administrators responsible for managing and debugging firmware for NVIDIA devices.

See also [Document Conventions and Related Documents](#).

Document Revision History

A list of the changes made to the user manual are provided in [Document Revision History](#).

Release Notes

These are the release notes for mstflint. mstflint supports Linux operating system. Please see the supported platform table for further details.

The tools functionality is identical in all operating systems unless otherwise noted.

Release Notes Update History

Revision	Date	Description
4.26	November 5, 2023	Initial release of this Release Notes version.

General Information

Package Tools

The following is a list of the available tools in the package, together with a brief description of each tool. The tools apply to single switch systems or adapter cards.

The mstflint tools do not provide cluster wide functionality.

Category	Tool	Description	Operating System
Firmware Update and Configuration	mstflint	This tool burns a firmware binary image or an expansion ROM image to the Flash of a network adapter/switch device. It includes query functions to the burnt firmware image and to the binary image file.	Linux FreeBSD
	mstconfig	Allows the user to change some of the device configurations without having to create and burn a new firmware.	Linux FreeBSD
	mstfwmanager	The mstwmanager is a firmware update and query utility. It provides a simple 'single click' firmware update functionality. Note: The same tool with embedded firmware binaries is released separately and is named mxup.	Linux FreeBSD
	mstarchive	The mstarchive tool allows the user to create a file with the mfa2 extension. The new file contains several binary files of a given firmware for different adapter cards.	Linux FreeBSD
Debug and Diagnostics Utilities	mstregdump	Dumps device internal configuration data.	Linux FreeBSD
	mstmcr	Reads/writes a single word from/to a device configuration register space	Linux FreeBSD
	mstvpd	Reads PCI device VPD	Linux FreeBSD
	mstfwreset	Load Firmware after firmware update on ISFU capable devices.(5th generation devices)	Linux FreeBSD

Category	Tool	Description	Operating System
	mstfw trace	Extracts and prints trace messages generated by the firmware of 5th generation devices. This tool supports secure firmware flow only.	Linux FreeBSD
	mstreg	Exposes supported access registers, and allows users to obtain information regarding the registers fields and attributes, and to set and get data with specific register.	Linux FreeBSD
	mstlink	Displays and configures port related data at the physical layer.	Linux FreeBSD
	mstresource dump	Extracts and prints data segments generated by the firmware.	Linux
	mstresource parse	Parses and prints data segments content.	Linux

Detailed installation instructions along with complete descriptions of the various tools in the package can be found in the Firmware Tools User Manual.

Dependencies

Flags Dependencies

Configure Flag	Dependencies	Additional Tools Installed
enable-dc	zlib	
enable-fw-mgr	curl, zlib, lzma	mstarchive, mstfwmanager
enable-adb-generic-tools	expat	mstlink, mstreg
enable-xml2	libxml2, xml2	
enable-inband	infiniband-diags*	
enable-cs	Openssl >= 1.0.2K	

mstflint Supported Operating Systems and Platforms

mstflint is supported on the following platforms:

Table Legend:

+ (Green)	Supported and tested
** (Orange)	Supported but not tested
*** Blue	Partially tested

Supported Operating Systems and Platforms

OS	Platform	Status
RHEL 7.2	x86_64	+
RHEL 7.4	x86_64	+

OS	Platform	Status
RHEL 7.4	PPC64	+
RHEL 7.4	PPC64LE	+
RHEL 7.5 Alt - Community	ARM	**
RHEL 7.6 Alt - Community	ARM	+
RHEL 7.6	PPC64 (Power 9)	+
RHEL 7.6	x86_64	+
RHEL 7.7	x86_64	+
RHEL 7.8	x86_64	+
RHEL 7.9	x86_64	+
RHEL 7.9	PPC64	**
RHEL 7.9	PPC64LE	+
RHEL 8	ARM	**
RHEL 8	PPC64LE	**
RHEL 8	x86_64	+
RHEL 8.1	PPC64LE	**
RHEL 8.1	x86_64	+
RHEL 8.2	x86_64	+
RHEL 8.2	PPC64LE	**
RHEL 8.3	PPC64LE	**
RHEL 8.3	x86_64	+
RHEL 8.4	PPC64LE	**
RHEL 8.4	x86_64	+
RHEL 8.5	PPC64LE	**
RHEL 8.5	x86_64	+
RHEL 8.6	x86_64	+
RHEL 8.6	PPC64LE	**
RHEL 8.7	x86_64	**
RHEL 8.7	PPC64LE	+
RHEL 8.8	x86_64	+
RHEL 8.8	PPC64LE	**
RHEL 8.8	ARM64	+
RHEL 8.9 - Beta	x86_64	**
RHEL 8.9 - Beta	PPC64LE	+
RHEL 8.9 - Beta	ARM64	**
RHEL 9.0	x86_64	+
RHEL 9.0	PPC64LE	+
RHEL 9.1	x86_64	**
RHEL 9.1	PPC64LE	+
RHEL 9.2	x86_64	+
RHEL 9.2	PPC64LE	+
RHEL 9.2	ARM64	+

OS	Platform	Status
RHEL 9.3	x86_64	+
RHEL 9.3	PPC64LE	+
RHEL 9.3	ARM64	+
Centos Stream v8 - Community	x86_64	**
Centos Stream v8 - Community	PPC64LE	**
Centos Stream v9 - Community	x86_64	+
Centos Stream v9 - Community	PPC64LE	**
OEL 7.9	x86_64	+
OEL 8.4	x86_64	+
OEL 8.6	x86_64	+
OEL 8.7	x86_64	+
OEL 8.8	x86_64	**
OEL 9.0	x86_64	+
OEL 9.1	x86_64	**
OEL 9.2	x86_64	+
Fedora 32 - Community	X86_64	+
Fedora 35 - Community	X86_64	**
Sles12 SP2 - Community	PPC64LE	**
Sles12 SP2 - Community	x86_64	**
Sles12 SP3 - Community	x86_64	**
Sles12 SP3 - Community	ppc64LE	**
Sles12 SP4 - Community	ARM64	**
Sles12 SP4 - Community	PPC64LE	**
Sles12 SP4 - Community	x86_64	**
Sles12 SP5	ARM64	**
Sles12 SP5	x86_64	+
Sles12 SP5	PPC64LE	**
Sles15 SP2	x86_64	**
Sles15 SP2- Community	PPC64LE	**
Sles15 SP3	PPC64LE	**
Sles15 SP3	x86_64	+
Sles15 SP4	PPC64LE	+
Sles15 SP4	x86_64	+
Sles15 SP5	PPC64LE	+
Sles15 SP5	x86_64	+
EulerOS V2.0 SP9 - Community	x86_64	**
EulerOS V2.0 SP10 - Community	x86_64	+
EulerOS V2.0 SP11	x86_64	+
EulerOS V2.0 SP12	x86_64	**
EulerOS V2.0 SP12	ARM64	+
OpenEuler 20.3 SP1 - Community	x86_64	**

OS	Platform	Status
OpenEuler 20.3 SP3	x86_64	+
OpenEuler 22.3 LTS	x86_64	+
Ubuntu 16.04 - Community	x86_64	**
Ubuntu 16.04 - Community	PPC64LE	+
Ubuntu 18.04	x86_64	+
Ubuntu 18.04	PPC64LE	+
Ubuntu 18.04	ARM64	+
Ubuntu 20.04	PPC64LE	+
Ubuntu 20.04	ARM64	+
Ubuntu 20.04	x86_64	+
Ubuntu 22.04	x86_64	+
Ubuntu 23.04	x86_64	+
BCLinux 21.10 SP2	x86_64	+
BCLinux 21.10 SP2	ARM	+
Debian 9.13	x86_64	+
Debian 10.8	x86_64	+
Debian 10.9	x86_64	+
Debian 10.13	x86_64	+
Debian 10.13	ARM	**
Debian 11.3	Arm	+
Debian 11.3	x86_64	+
Debian 12.1	x86_64	+
Debian 12.1	ARM	+
Citrix server 8.2	x86_64	+
Anolis 8.4 - Community	x86_64	**
Anolis 8.6	x86_64	+
Anolis 8.6	ARM	+
Korg 6.5	x86_64	+
Korg 6.5	ARM	**
OpenSUSE 15.3 - Community	x86_64	**
Photon 3.0 - Community	x86_64	**
Xen 7.1.2	x86_64	+
CTYunOS3	x86_64	+
CTYunOS3	ppc64le	**
Alma 8.5	x86_64	**
KylinOS v10 SP2	x86_64	+
KylinOS v10 SP3	x86_64	+
KylinOS v10 SP3	ARM	**
Allinux 3.2	x86_64	+
Allinux 3.2	ppc64le	+
DriveOS 6.0.5.0	x86_64	+

OS	Platform	Status
DriveOS 6.0.5.0	ARM	+
UOS v20 1021e	x86_64	+
UOS v20 1021e	ARM	**
UOS v20 1040d	x86_64	+
FreeBSD 13.0-STABLE	x86_64	+
	aarch64	+
FreeBSD 14.0-STABLE	aarch64	+
FreeBSD 14-CURRENT	x86_64	+
SONiC 202211_1	64 Bit	+
MLNX-OS 3.11.2000	64 Bit	+
Cumulus 5.6	64 Bit	**
NV-OS 25.01.2500	64 Bit	**
DVS 4.6.2000	64 Bit	+

Supported Flash Types

mstflint supports the following Flash types.

Vendor	Flash Family	Tested P/N
Winbond	W25QxxBV	W25Q32FVSSIG
		W25Q32FVSSIGS
		W25Q32FVSSIGT
		W25Q128FVSSIGS
	W25Qxxx	W25Q256JVBIMT
		W25Q128JVSIQ
Macronix	MX25L16xxx	MX25L12845GM2I-08G
	MX25Lxxx	MX25L25645GXDI-08G
Micron	N25Q0xxx	MT25QL128ABA1ESE-OSIT
ISSI	IS25LPxxx	IS25LP128-JBLE SPA# U1323A
	IS25WPxxx	IS25WP256E-RHLE
Cypress	S25FL256L	S25FL256LDPBHV023
		S25FL128SAGMFIG00
Gigadevice		GD25LB256EBFRY
		GD25B256DFIGR

Supported Interface Cards (NICs)

With respect to mstflint, NVIDIA IC devices are divided into two groups: Group I and Group II (4th generation and 5th generation, respectively). The ICs are listed in the following table:

IC Group	IC Device
Group II/5th Generation	<ul style="list-style-type: none"> • Adapter Cards: <ul style="list-style-type: none"> • NVIDIA BlueField-3 • NVIDIA BlueField-2 • NVIDIA BlueField • NVIDIA ConnectX-7 • NVIDIA ConnectX-6 Lx • NVIDIA ConnectX-6 Dx • NVIDIA ConnectX-6 • NVIDIA ConnectX-5 • NVIDIA ConnectX-4 Lx • NVIDIA ConnectX-4 • NVIDIA Connect-IB • Switch Systems: <ul style="list-style-type: none"> • NVIDIA Quantum-2 • NVIDIA Quantum • NVIDIA Spectrum-3 • NVIDIA Spectrum-2 • NVIDIA Spectrum • NVIDIA Switch-IB 2 • NVIDIA Switch-IB
Group I/4th Generation	<ul style="list-style-type: none"> • Adapter Cards: <ul style="list-style-type: none"> • NVIDIA ConnectX-3 • NVIDIA ConnectX-3 Pro • Switch Systems: <ul style="list-style-type: none"> • NVIDIA SwitchX-2

Supported Adapter Cards Firmware Versions

MFT supports the following NVIDIA® network adapter cards:

NICs	Recommended Firmware Rev.	Additional Firmware Rev. Supported
BlueField-3	32.38.3056	32.38.1002
BlueField-2	24.38.1002	24.37.1014
BlueField	18.33.1048	18.33.1048
ConnectX-7	28.39.1002	28.38.1002
ConnectX-6 Lx	26.39.1002	26.38.1002
ConnectX-6 Dx	22.39.1002	22.38.1002
ConnectX-6	20.39.1002	20.38.1002
ConnectX-5 / ConnectX-5 Ex	16.35.3006	16.35.2000
ConnectX-4 Lx	14.32.1010	14.32.1010
ConnectX-4	12.28.2006	12.28.2006

To download the firmware binaries, please visit [Firmware Downloads](#).

Supported Switch Systems Software

The following are the Supported Switch Systems Software.

Switch Software	Version
MLNX-OS	3.11.1000
SONIC	202211_1
Cumulus	5.6
DVS	4.6.1000
NV_OS	25.01.2000

Changes and New Features

Component/ Tool	Description	Operating System
Rev. 4.26		
General	Added support for an additional flash type - Winbond Part No. W25Q256JVFIQ.	All
General	Removed dependency on Boost library.	All
General	Added support for VPATH builds (see https://www.gnu.org/software/automake/manual/html_node/VPATH-Builds.html).	All
General	The descriptors for DEB/RPM packages are now configured to include two additional tools: mstreg and mstlink. This results in the addition of build-time dependencies on libexpat1-dev and liblzma-dev.	All

mstflint Bug Fixes in this Version

For a list of old Bug Fixes, please see [mstflint Bug Fixes History](#).

Internal Ref.	Description
3582574	Description: Fixed an issue that prevented ConnectX-5 EX reset using the fastfwreset tool.
	Keywords: fastfwreset
	Discovered in Version: 4.25.0
	Fixed in Release: 4.26.0
3582575	Description: Fixed an issue that caused incorrect enumeration in NVIDIA devices.
	Keywords: incorrect enumeration
	Discovered in Version: 4.25.0
	Fixed in Release: 4.26.0
3613010	Description: Fixed an issue where mstdump did not work with Quantum-2 switches due to the absence of the Quantum2.csv file from the C:\Program Files\Mellanox\WinMFT\mstdump_dbs\ folder.
	Keywords: mstdump, Quantum-2
	Discovered in Version: 4.25.0
	Fixed in Release: 4.26.0

mstflint Known Issues

The following table provides a list of known issues and limitations of mstflint. For a list of old Known Issues, please see [Archived Known Issues](#) file.

Internal Ref. No.	Issue
3641618	Description: Running a command triggers the following error message: <code>/lib/libgcc_s.so.1: version GCC_4.5.0 required by /usr/local/lib/gcc12/libstdc++.so.6 not found</code>
	Workaround: Run the following command: <code>export LD_LIBRARY_PATH=/usr/local/lib/gcc12:\$LD_LIBRARY_PATH</code>
	Keywords: libstd, gcc, mft, libgcc
	Discovered in Version: 4.26.0
3446066	Description: When using ConnectX-7 and later cards, the link should be fully down (not in polling state) for the loopback configuration can be applied.
	Workaround: N/A
	Keywords: mstlink
Discovered in Version: 4.23.0	
3418112	Description: Loading a new firmware may require running <code>mlxfwreset</code> , and in some cases rebooting or initiating a power-cycle.
	Workaround: N/A
	Keywords: mstfwreset
Discovered in Version: 4.24.0	
3292150	Description: The <code>mstfwreset</code> tool does not support Cedar system.
	Workaround: N/A
	Keywords: mstfwreset, Cedar, ConnectX-7
Discovered in Version: 4.23.0	
3188577	Description: Some firmware scratchpad registers have been moved to a different location. Therefore, if you use your own utility to dump <code>mstregdumps</code> , you must update your CSV file with the latest CSV, CSV2 files that are included in the MFT package. Otherwise, the <code>mstregdumps</code> device will not retrieve the firmware version, and the FAEs will not be able to use NVIDIA internal tools to debug the error.
	Workaround: N/A
	Keywords: CSV, mstregdump
	Discovered in Version: 4.22.0

Internal Ref. No.	Issue
2829041	<p>Description: Running mstlink on LID devices when the OpenSM is not enabled, can cause the machine to hang.</p> <p>Workaround: To resolve the issue, run the following:</p> <ol style="list-style-type: none"> 1. opensm & (Press 'Enter') 2. mst restart; 3. mst ib add; 4. mst status; 5. Get the correct LID device from "InBand" devices <p>Keywords: mstlink, LID, InBand, OpenSM</p> <p>Discovered in Version: 4.20.0</p>
2823492	<p>Description: mstfwreset is not supported on DPU with GPU boards.</p> <p>Workaround: N/A</p> <p>Keywords: mstfwreset</p> <p>Discovered in Version: 4.18.0</p>
2715716	<p>Description: mstfwreset is not supported on secure-boot host devices.</p> <p>Workaround: N/A</p> <p>Keywords: mstfwreset</p> <p>Discovered in Version: 4.18.0</p>
2752916	<p>Description: The information of the IB/ETH protocols should not be stored on the same CSV file. Doing so will result in a mismatch on the columns of CSV file.</p> <p>Workaround: N/A</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.18.0</p>
2838222	<p>Description: mstfwreset is not supported on kernel 3.10.0-1062.el7.x86_64 due to a kernel bug that leads to 'rescan' PCI operation to take a few minutes.</p> <p>Workaround: N/A</p> <p>Keywords: mstfwreset</p> <p>Discovered in Version: 4.18.0</p>
2670833	<p>Description: Burning firmware using DMA might fail on virtual FreeBSD machines.</p> <p>Workaround: N/A</p> <p>Keywords: Firmware burning, DMA, FreeBS, VM</p> <p>Discovered in Version: 4.17.0</p>
2484780	<p>Description: Configuring TX/RX_rate to 200GbE in test mode fails.</p>

Internal Ref. No.	Issue
	<p>Workaround: To work with the new speeds specify the number of lanes as shown below:</p> <ul style="list-style-type: none"> • 100G_1X/200G_2X/400G_4X/800G_8X for NDR speeds • 50G_1X/100G_2X/200G_4X/400G_4X for HDR speeds <p>Keywords: 200GbE, Tx/Rx</p> <p>Discovered in Version: 4.17.0</p>
2208845/2099263	<p>Description: mstlink does not support test mode for 50GE-KR4 speed.</p> <p>Workaround: N/A</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.16.0</p>
2001890	<p>Description: The argparse module is installed by default in Python versions =>2.7 and >=3.2. In case an older Python version is used, the argparse module is not installed by default and therefore must be manually installed.</p> <p>Workaround: N/A</p> <p>Keywords: Python, argparse module</p> <p>Discovered in Version: 4.13.3-2</p>
1923665	<p>Description: Force Mode does not work when using mstlink in ConnectX-6 InfiniBand adapter cards.</p> <p>Workaround: N/A</p> <p>Keywords: mstlink, Force Mode, ConnectX-6 IB</p> <p>Discovered in Version: 4.13.3-2</p>
1431471	<p>Description: In ConnectX-5 adapter cards, the time-stamp capability using mstflint, is supported only on the device using the "-d" flag, and not on the binary using the "-i" flag.</p> <p>Workaround: Use the "-d" flag to set the time-stamp.</p> <p>Keywords: mstflint</p> <p>Discovered in Version: 4.11.0</p>
1442454	<p>Description: Occasionally, when running mstfwreset over a Multi-Host device, the driver remains down if the mstfwreset operation fails.</p> <p>Workaround: N/A</p> <p>Keywords: mstfwreset</p> <p>Discovered in Version: 4.11.0</p>

User Manual

The mstflint package is a set of firmware management and debug tools for NVIDIA® devices. mstflint can be used for:

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

The list of the available tools in the package can be found in the [Release Notes](#) document.

Supported Operating Systems


Please refer to the release notes of your version for supported platforms and kernels.

Access to Hardware Devices

The table below lists the NVIDIA® devices supported by mstflint, the supporting tools, and the access methods to these devices.

Device Type	Product Name	HW Access Method		
		PCI	I2C	In-Band
HCA (InfiniBand)	NVIDIA Connect-IB	V	V	V
IB/ETH Network Adapter	NVIDIA ConnectX-3 Pro	V	V	V
	NVIDIA ConnectX-4	V	V	V
	NVIDIA ConnectX-5	V	V	V
	NVIDIA ConnectX-5 Ex	V	V	V
	NVIDIA ConnectX-6	V	V	V
	NVIDIA ConnectX-6 Dx	V	V	V
	NVIDIA ConnectX-7	V	V	V
	NVIDIA BlueField-2	V	V	V
	NVIDIA BlueField-3	V	V	V
Ethernet Adapter (NIC)	NVIDIA ConnectX-4 Lx	V	V	
	NVIDIA ConnectX-6 Dx	V	V	
	NVIDIA ConnectX-6 Lx	V	V	
	NVIDIA ConnectX-7	V	V	
	NVIDIA BlueField-2	V	V	
Switch	NVIDIA Switch-IB®	V ¹	V	V
	NVIDIA Switch-IB 2	V ¹	V	V
	NVIDIA Spectrum™	V	V	
	NVIDIA Spectrum-2	V	V	
	NVIDIA Spectrum-3	V	V	
	NVIDIA Quantum	V	V	V

Note. V¹ indicates managed switch products only. V² In-band capability is only available for mstflint if mstflint is compiled with MLNX_OFED driver. mstflint tools access NVIDIA devices via the PCI Express interface, via a USB to I2C adapter (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 mstflint tools address the target hardware device using a PCI device. To list all available PCI devices, use the “lspci” command.

To see the NVIDIA PCI devices, run “lspci | grep -l “Mellanox””.

Ex:

```
# lspci | grep -i "Mellanox"
15:00.0 Infiniband controller: Mellanox Technologies MT27700 Family [ConnectX-4]
1a:00.0 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4 Lx]
1a:00.1 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4 Lx]
```

Local PCI devices may also be accessed using device aliases. Supported aliases are:

- PCI device “bus:dev.fn” (e.g. 03:00.0 (BDF format))
- OFED RDMA device (e.g. mlx4_0)
- Network interface with “net-” prefix, (e.g. net-eth2”)

Compilation and Installation

1. Compile mstflint.

```
./autogen.sh
mkdir build; cd build
./configure [options]
make -j [n]
```

Note: If using FreeBSD OS, run:

```
./autogen.sh
mkdir build; cd build
./configure [options] MAKE="gmake"
gmake -j [n]
```

To check all the possible values of [OPTION]... [VAR=VALUE]..., run “./configure --help”.

2. Install mstflint.

```
make install
```

Firmware Generation, Configuration, and Update Tools

This chapter contains the following sections:

- [mstfwmanager - Firmware Update and Query Tool](#)
- [mstarchive - Binary Files Compression Tool](#)
- [mstconfig - Changing Device Configuration Tool](#)

- [mstflint - Firmware Burning Tool](#)
- [mstfwreset - Loading Firmware on 5th Generation Devices Tool](#)
- [mstcongestion - Tool for Setting Congestion Mode and Action](#)
- [mstprivhost - NIC Configuration by the Host Restriction Tool](#)

mstfwmanager - Firmware Update and Query Tool

The mstfwmanager is a firmware update and query utility which scans the system for available NVIDIA devices (only mst PCI devices) and performs the necessary firmware updates. For further information on firmware update, please refer to [Booting HCA Device in Livefish Mode](#).



The examples throughout the document use pci “bus.dev.fn” format.

mstfwmanager Synopsis

```
# [-d|--dev DeviceName] [-h|--help] [-v|--version] [--query] [--query-format Format] [-u|--update] [-i|--image-file
FileName] [-D|--image-dir DirectoryName] [-f|--force] [-y|--yes] [--no] [--clear-semaphore] [--exe-rel-path] [-l|--
list-content] [--archive-names] [--nofs] [--log] [-L|--log-file LogFileName] [--no-progress] [-o|--outfile
OutputFileName] [--online] [--online query-psid PSIDs] [--key key] [--download DirectoryName] [--download-default]
[--get-downloadopt OPT] [--download-device Device] [--download-os OS] [--download-type Type] [--ssl-certificate
Certificate] [--no_fw_ctrl]
```

where:

-d --dev DeviceName	Perform operation for specified mst device(s). Multiple devices can be specified delimited by semicolons. A device list containing semicolons must be quoted.
-h --help	Show this message and exit.
-v --version	Show the executable version and exit.
--query	Query device(s) info
--query-format Format	(Query Onlinequery)outputformat,XML Text-defaultText
-u --update	Update firmware image(s) on the device(s).
-i --image-file FileName	Specified image file to use.
-D --image-dir DirectoryName	Specified directory instead of default to locate image files.
-f --force	Force image update
-y --yes	Answer is yes in prompts
--no	Answer is no in prompts
--clear-semaphore	Force clear the flash semaphore on the device, No command is allowed when this flag is used. NOTE: May result in system instability or flash corruption if the device or another application is currently using the flash. Exercise caution.
--exe-rel-path	Use paths relative to the location of the executable
-l --list-content	List file/Directory content, used with --image-dir and --image-file flags
--archive-names	Display archive names in listing
--nofs	Burn image in a non failsafe manner
--log	Create log file
-L --log-file LogFileName	Use specified log file
--no_fw_ctrl	Do not use firmware Ctrl update
--no-progress	Do not show progress
-o --outfile OutputFileName	Write to specified output file
--online	Fetch required FW images online from NVIDIA server
--online-query-psid PSIDs	Query FW info, PSID(s) are comma separated
--key key	Key for custom download/update
--download DirectoryName	Download files from server to a specified directory
--download-default	Use Default values for download
--get-download-opt OPT	Get download options for OS or Device Options are: OS, Device
--download-device Device	Use '--get-download-opt Device' option to view available devices for device specific downloads
--download-os OS	Only for self_extractor download: Use '--get-download-opt OS' option to view available OS for sfx download
--download-type Type	MFA self_extractor - default All
--ssl-certificate Certificate	SSL certificate for secure connection

Querying the Device

To query a specific device, use the following command line:

```
# mstfwmanager -d <device> --query
```

To query all the devices on the machine, use the following command line:

```
# mstfwmanager --query
```

Examples:

Query the device.

```
mstfwmanager -d 09:00.0 --query
Querying Mellanox devices firmware ...

Device #1:
-----
Device Type:      ConnectX3
Part Number:     MCX354A-FCA_A2-A4
Description:     ConnectX-3 VPI adapter card; dual-port QSFP; FDR IB (56Gb/s) and 40GigE; PCIe3.0 x8 8GT/s; RoHS R6
PSID:           MT_1020120019
PCI Device Name: 0000:09:00.0
Port1 GUID:     0002c9000100d051
Port2 MAC:      0002c9000002
Versions:       Current Available
FW             2.31.5050  2.32.5000
Status: Update required
-----
Found 1 device(s) requiring firmware update. Please use -u flag to perform the update.
```

Query all the devices.

```
Querying Mellanox devices firmware ...

Device #1:
-----

Device Type:      ConnectX3Pro
Part Number:     MCX354A-FCC_Ax
Description:     ConnectX-3 Pro VPI adapter card; dual-port QSFP; FDR IB (56Gb/s) and 40GigE; PCIe3.0 x8 8GT/s; RoHS R6
PSID:           MT_1090111019
PCI Device Name: 0000:84:00.0
Port1 GUID:     f452140300347881
Port2 GUID:     f452140300347882
Versions:       Current Available
FW             2.42.5016  N/A
PXE           3.4.0753  N/A

Status:         No matching image found

Device #2:
-----

Device Type:      ConnectX4LX
Part Number:     MCX4121A-ACA_Ax
Description:     ConnectX-4 Lx EN network interface card; 25GbE dual-port SFP28; PCIe3.0 x8; ROHS R6
PSID:           MT_2420110034
PCI Device Name: 0000:02:00.0
Base MAC:       e41d2dfd8b8a
Versions:       Current Available
FW             14.25.8306  N/A
PXE           3.5.0702  N/A
UEFI          14.18.0022  N/A

Status:         No matching image found
```

Query XML:

```
# mstfwmanager --query --query-format XML
<Devices>
  <Device pciName="0000:84:00.0" type="ConnectX3Pro" psid="MT_1090111019" partNumber="MCX354A-FCC_Ax">
    <Versions>
      <FW current="2.42.5016" available="N/A"/>
      <PXE current="3.4.0753" available="N/A"/>
    </Versions>
    <GUIDs port1="f452140300347881" port2="f452140300347882"/>
    <Status>No matching image found</Status>
    <Description>ConnectX-3 Pro VPI adapter card; dual-port QSFP; FDR IB (56Gb/s) and 40GigE; PCIe3.0 x8 8GT/s; RoHS R6</Description>
  </Device>
  <Device pciName="0000:02:00.0" type="ConnectX4LX" psid="MT_2420110034" partNumber="MCX4121A-ACA_Ax">
    <Versions>
      <FW current="14.25.8306" available="N/A"/>
      <PXE current="3.5.0702" available="N/A"/>
      <UEFI current="14.18.0022" available="N/A"/>
    </Versions>
    <MACs Base_Mac="e41d2dfd8b8a" />
  </Device>
</Devices>
```

```
<Status>No matching image found</Status>
<Description>ConnectX-4 Lx EN network interface card; 25GbE dual-port SFP28; PCIe3.0 x8; ROHS R6</
Description>
</Device>
</Devices>
```

Archived Images Content

Supports listing the contents of images archive.

- When running this command, the tool will list all firmware images within this PLDM package for each image it displays.

Usage:

```
mstfwmanager -i <pldm-path> --list-content
```

- When running this command, the tool will list all firmware images within this mfa package.

Usage:

```
mstfwmanager -i <mfa-file> --list-content
```

For each image, it displays the following: PSID, Part Number, firmware version, and device description.

mstarchive - Binary Files Compression Tool

The mstarchive tool allows the user to create a file with the MFA2 extension. The new file contains several binary files of a given firmware for different adapter cards.

mstarchive accepts the following attributes as its input:

- --bins-dir - The path to a folder with the binary files that will be included in the MFA2 file
- --version - The MFA2 file's version
- --out-file - The output of the mstarchive file (MFA2 file)
- -m|--mfa2-file mfa2_file - MFA2 file to parse

Example:


```
mstarchive --bins-dir /full/path/to/bin/directory/ --version 1.1.1 --out-file out.mfa2 mstarchive --mfa2-file
out.mfa2
Creation Time : 2019-09-18 08:35:43
Devices 2
PSID : <...>
Num of Images 1
Index 0
Version : 10.16.1200
Date : 2019-09-18 08:35:43 PSID : <...>
Num of Images 1
Index 1
Version : 10.16.1200
Date : 2019-09-18 08:35:43
```

mstarchive Synopsis

```
[--help] [--version version] [--out-file out_file] [--bins-dir bins_dir] [-m|--mfa2-file mfa2_file]
```

where:

--help	Shows the help message and exit
--version version	MFA2's version in the following format: x.x.x
--out-file out_file	The output file
-bins-dir bins_dir	The directory with the binaries files
-m --mfa2-file mfa2_file	Mfa2 file to parse

 The .mfa2 file can be used with ethtool to burn adapter cards firmware. The procedure is described in [mstflint - Updating Firmware Using ethtool/devlink and .mfa2 File](#) section.

mstconfig - Changing Device Configuration Tool


The mstconfig tool allows the user to change some of the device configurations without reburning the firmware. The configuration is also kept after reset.

By default, mstconfig shows the configurations that will be loaded in the next boot.

For 5th generation devices, it is also possible to query the default configurations and the configurations that are used by the current running firmware.

Tool Requirements

- OFED/UPStream driver to be installed and enabled (for ConnectX-3 and ConnectX-3 Pro)
- Access to the device through BDF format
- For the adapter cards below, the following firmware versions are required:
 - ConnectX[®]-3/ConnectX-3 Pro: v2.31.5000 or above
 - Connect-IB[®]: v10.10.6000 or above
- Supported devices:
 - Adapter cards: ConnectX-3/ConnectX-3 Pro/Connect-IB/ConnectX-4/ConnectX-4 Lx/ConnectX-5/ConnectX-5 Ex/BlueField[®]/ConnectX-6/ConnectX-6 Dx
 - Switches: Switch-IB/Switch-IB 2/Spectrum[™]/Spectrum-2/Quantum
- Changing device configurations enabled.

 For changes after a successful configuration to take effect, reboot the system.

mstconfig Synopsis

```
# mstconfig [Options] <commands> [Parameters]
```

where:

-d --dev <device>	Perform operation for a specified MST device.
-b --db <filename>	Use a specific database file.
-f --file <conf_file>	Raw configuration file.
-h --help	Display help message.
-v --version	Display version info.
-e --enable_verbosity	Show default and current configurations.
-y --yes	Answer yes in prompt.
-a --all_attrs	Show all attributes in the XML template
-p --private_key	pem file for private key
-u --key_uuid	Keypair uuid
clear_semaphore	Clear the tool semaphore.
i[show_confs]	Display information about all configurations.
q[query]	Query supported configurations.
r[reset]	Reset all configurations to their default value.
s[et]	Set configurations to a specific device.
set_raw	Set raw configuration file.(only Connect-IB/ConnectX-4/Lx)
get_raw	Gets raw configuration file (5th generation/Group II devices only)
backup	Backup configurations to a file (only Connect-IB/ConnectX-4/Lx). Use set_raw command to restore file.
gen_tlvs_file	Generate List of all TLVs. TLVs output file name must be specified. (*)
g[en_xml_template]	Generate XML template. TLVs input file name and XML output file name must be specified. (*)
xml2raw	Generate binary configuration dump file from XML file. XML input file name and bin output file name must be specified. (*)
raw2xml	Generate configuration file from XML file. XML input file name and bin output file name must be specified. (*)
xml2bin	Apply a configuration file, that was created with create_conf command. bin input file name must be specified. (*)
create_conf	Generate Configuration file from XML file. XML input file name and bin output file name must be specified. (*)
apply	Apply a Configuration file. bin input file name must be specified. (*)

Examples of mstconfig Usage

Querying the Device Configuration

To query the device's configuration, use the following command line:

```
# mstconfig -d <device> query
```

ConnectX-3 Example:

```
# mstconfig -d 41:00.0 q
Device type: ConnectX-3
PCI device: 41:00.0
41:00.0
Device 1:
-----
Configurations:      Next Boot
SRIOV_EN             True(1)
NUM_OF_VFS           16
WOL_MAGIC_EN_P1     False(0)
WOL_MAGIC_EN_P2     False(0)
```



N/A means that the device default configuration is set.



For Array type parameters, the query command will not show a value for it. It will only show you the word "Array" and the range of the array.

- For example: HOST_CHAINING_DESCRIPTORs Array[0..7]
- To query the fifth element in the array, run: `mstconfig -d <device> query HOST_CHAINING_DESCRIPTORs[5]`
- To specify a range: `mstconfig -d <device> query HOST_CHAINING_DESCRIPTORs[3..7]`
- To set the fifth element in the array, run: `mstconfig -d <device> set HOST_CHAINING_DESCRIPTORs[5]=3`
- Or you can set value for more than one element: `mstconfig -d <device> set HOST_CHAINING_DESCRIPTORs[3..7]=3`

ConnectX-4 Lx Example:

```
# mstconfig -d 41:00.0 --enable_verbosity q
Device #1:
-----
Device type: ConnectX4LX
PCI device: 41:00.0

Configurations:      Default      Current      Next Boot
* NUM_OF_VFS         8             5             5
SRIOV_EN             True(1)       True(1)       True(1)
The '*' shows parameters with next value different from default/current value.
```

Setting Device Configuration

To set the device configuration, use the following command line:


```
# mstconfig -d <device> set [Parameters....]
```

Example:

```
# mstconfig -d 41:00.0 set WOL_MAGIC_EN_P2=1 NUM_OF_VFS=24
Device type: ConnectX-3
PCI device: 41:00.0
Configurations:      Next Boot      New
NUM_OF_VFS           16             24
WOL_MAGIC_EN_P2     False(0)       True(1)

Apply new Configuration?(y/n) [n]: y
Applying... Done!

-I- Please reboot the system to load new configurations.
```

Resetting Device Configuration to Default

To reset the device configuration to default, use the following command line:

```
# mstconfig -d <device> reset
```

Example:

```
# mstconfig -d 41:00.0 reset
Reset configuration for device 41:00.0? ? (y/n) [n] : y
Applying... Done!

-I- Please power-cycle device to load new configurations.
>mstconfig 41:00.0 query

Device 1:
-----
Device type: ConnectX-3
PCI Device: 41:00.0
Configurations:      Next Boot
SRIOV_EN             True(1)
NUM_OF_VFS           8
WOL_MAGIC_EN_P1     False(0)
WOL_MAGIC_EN_P2     False(0)
```

Using mstconfig

Using mstconfig with PCI Device in Bus Device Function (BDF) Format

Example:

```
# mstconfig -d 41:00.0

Device 1:
-----
Device type:      ConnectX-3
PCI Device:      41:00.0
Configurations:  Next Boot
SRIOV_EN        True(1)
NUM_OF_VFS      16
WOL_MAGIC_EN_P1 False(0)
WOL_MAGIC_EN_P2 False(0)
```

Using mstconfig to Set VPI Parameters

In order to set VPI parameters through mstconfig, use the following command line:

```
# mstconfig -d <device> set [LINK_TYPE_P1=<link_type>] [LINK_TYPE_P2=<link_type>]
```

Example: Configuring both ports as InfiniBand:

```
# mstconfig 41:00.0 set LINK_TYPE_P1=1 LINK_TYPE_P2=1

Device #1:
-----
Device type:   ConnectX3Pro
PCI device:   41:00.0
Configurations:
  LINK_TYPE_P1  Next Boot  New
                ETH(2)    IB(1)
  LINK_TYPE_P2  ETH(2)    IB(1)

Apply new Configuration? ? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.
```

Using mstconfig to Set SR-IOV Parameters

In order to set SR-IOV parameters through mstconfig, use the following command line:

```
# mstconfig -d <device> set [SRIOV_EN=<0|1>] [NUM_OF_VFS=<NUM>]
```

Example: Turning on SR-IOV and enabling 8 Virtual Functions per Physical Function:

```
# mstconfig -d 41:00.0 set SRIOV_EN=1 NUM_OF_VFS=8

Device #1:
-----
Device type:   ConnectX4
PCI device:   41:00.0
Configurations:
  SRIOV_EN      Next Boot  New
                0          1
  NUM_OF_VFS    0          8

Apply new Configuration? ? (y/n) [n] : y Applying... Done!
-I- Please reboot machine to load new configurations.
```

Using mstconfig to Set Preboot Settings

For a full description of the preboot configurable parameters refer to [Supported Configurations and their Parameters](#) under "Preboot Settings".

Example: Enable boot option ROM on port 1, set boot retries to 3 and set the boot protocol to PXE.

```
# mstconfig -d 41:00.0 set BOOT_OPTION_ROM_EN_P1=1 BOOT_RETRY_CNT_P1=3 LEGACY_BOOT_PROTOCOL_P1=1

Device #1:
-----
Device type:   ConnectX3Pro
PCI device:   41:00.0
Configurations:
  BOOT_OPTION_ROM_EN_P1  Next Boot  New
                        False(0)  True(1)
  BOOT_RETRY_CNT_P1      0          3
  LEGACY_BOOT_PROTOCOL_P1 2          1

Apply new Configuration? ? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.
```

Example: Configure VLAN ID to 3 on port 2

```
# mstconfig -d 41:00.0 set BOOT_VLAN_P2=3

Device #1:
-----
Device type:   ConnectX3Pro
PCI device:   41:00.0
Configurations:
  BOOT_VLAN_P2  Next Boot  New
                1          3

Apply new Configuration? ? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.
```

mstconfig Raw Configuration Files

mstconfig allows applying raw configuration file for a pre-set configuration. Raw configuration files are intended for advanced users. This document does not cover the generation of such files.

Set the raw configuration file:

```
# mstconfig -f ./tlv_file.conf -d 41:00.0 set_raw

Raw TLV #1 Info:
Length: 0xc
Version: 0
OverrideEn: 1
Type: 0x00000080
Data: 0xa0000000 0xa0020010 0x00000000

Raw TLV #2 Info:
Length: 0x4
Version: 0
OverrideEn: 1
Type: 0x01020012
Data: 0x0000000b

Raw TLV #3 Info:
Length: 0x8
Version: 0
OverrideEn: 1
Type: 0x00000190
Data: 0x00000010 0x00007d00

Raw TLV #4 Info:
Length: 0x4
Version: 0
OverrideEn: 1
Type: 0x00000082
Data: 0x0000000c
Operation intended for advanced users.

Are you sure you want to apply raw TLV file? ? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.
```



Never apply files from an unreliable source.

mstconfig Commands

mstconfig Backup Command

The backup command is used to save the current non-volatile configurations (TLV) in the device into a file in raw TLV syntax so it can be restored anytime using the set_raw command.

mstconfig backup command allows backing up the all of the configurations which are related only to the PCI Physical Function associated with the given MST device. To back up all of the device configurations, perform the operation from every PCI Physical Function the device exposes. Restoring the configurations must be made from the matching PCI Physical Function.



In a MultiHost environment, these operations are required to be executed per host.

```
# mstconfig -d 41:00.0 -f /tmp/backup.conf backup
Collecting...
Saving output...
Done!
# cat /tmp/backup.conf
MLNX_RAW_TLV_FILE
% TLV Type: 0x00000400, Writer ID: ICMD mstCONFIG(0x09), Writer Host ID: 0x00 0x00000014 0x00000400 0x00000000
0x00000000 0x000e0000 0x001000f6 0x20160526 0x11250000
# mstconfig -d 41:00.0 -f /tmp/backup.conf set_raw
Raw TLV #1 Info:
```

```

Length: 0x14
Version: 0
OverrideEn: 0
Type: 0x00000400
Data: 0x00000000 0x000e0000 0x001000f6 0x20160526 0x11250000
Operation intended for advanced users.
Are you sure you want to apply raw TLV file? ? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.

```

Generating an XML Template for the Configurations

Users can generate an XML file that contains a template for the configurations. The template describes the configurations and their parameters. No values are included in the template.

To generate such a template, run the `gen_tlvs_file` command. This command will generate a file containing a list of all supported configurations by `mstconfig`, with a zero appearing in the end of each configuration. To choose a configuration, change the 0 to 1, then save the file and run the `gen_xml_template` command. An XML file containing the required configurations will be generated.

Example:

```

# mstconfig gen_tlvs_file /tmp/confs.txt
Saving output...
Done!
# cat /tmp/confs.txt
nv_host_to_bmc      0
nv_kdnet_data       0
nv_fpga_data        0
nv_packet_pacing    0
nv_debug_mode       0
nv_global_pci_conf  0

```

In order to include the `nv_kdnet_data` configuration in the template, change the 0 to 1, as demonstrated in the following example.

Example:

```

nv_kdnet_data      1

#mstconfig gen_xml_template /tmp/confs.txt /tmp/template.xml
Saving output...
Done!

#cat /tmp/template.xml
<?xml version="1.0" encoding="UTF-8"?>
<config xmlns="http://www.mellanox.com/config">
<nv_kdnet_data>
      <!-- Legal Values: False/True -->
      <kdnet_en></kdnet_en>
</nv_kdnet_data>
</config>

```

Advance Options

Add the `-a` flag in order to allow advance options in the XML generated file. When using this flag, each TLV in the XML file has additional attributes that must be filled.

by default, all TLVs will be at MLNX priority. Other possible values are OEM and USER.

mstconfig xml2raw Command

The `xml2raw` command is an easy way to generate a flawless raw configuration file that can be used in the `set_raw` command. The input for the command is an XML file that contains the data of the required configurations. To generate an XML file and fill it with the desired values, run

the commands from [Generating an XML Template for the Configurations](#), and then use the `xml2raw` command to generate a raw file.

Example:

mstconfig xml2raw Command

The `xml2raw` command is an easy way to generate a flawless raw configuration file that can be used in the `set_raw` command. The input for the command is an XML file that contains the data of the required configurations. To generate an XML file and fill it with the desired values, run the commands from [Generating an XML Template for the Configurations](#), and then use the `xml2raw` command to generate a raw file.

Example:

```
# cat /tmp/template.xml
<?xml version="1.0" encoding="UTF-8"?>
<config xmlns="http://www.mellanox.com/config">
  <nv_kdnet_data>
    <!-- Legal Values: False/True -->
    <kdnet_en>True</kdnet_en>
  </nv_kdnet_data>
</config>

#mstconfig xml2raw /tmp/template.xml /tmp/confs.raw
Saving output...
Done!

#cat /tmp/confs.raw
MLNX_RAW_TLV_FILE
0x03000004 0x00000085 0x00000000 0x80000000
```

mstconfig xml2bin Command


The `xml2bin` command is an easy way to generate a binary file that contains a binary dump of configurations. The input for the command is an XML file that contains the data of the required configurations. To generate an XML file and fill it with the desired values, run the commands from Section 2.4.10, and then use the `xml2bin` command to generate a binary file.


Example:

```
#mstconfig xml2bin /tmp/template.xml /tmp/confs.bin
Saving output...
Done!
# hexdump -C /tmp/confs.bin
00000000 80 00 00 00          |....|
00000004
```

mstconfig create_conf Command

The `create_conf` command assists in creating an NV configuration file that can be used for LifeCycle and Secure Firmware Updates purposes. The flow for creating a configuration file is the same as the flow for `xml2bin`. The user must provide the command with an XML file containing the required configurations and their values. The command can sign the configuration file, if the user provides a private key and UUID. The sign result will be appended to the end of the configuration file. If no private key and UUID are provided, the tool will compute an SHA256 digest and append it to the file. The generated signature will be used by the firmware for authentication purposes.

 Currently the only supported NV configurations types are CS tokens, Debug tokens and MLNX ID which are used for Secure Firmware Updates. Additionally, it supports two RSA keys of 2048 or 4096 bits length.

 The NV configurations files must have the applicable_to configuration.

Example:

```
#mstconfig create_conf --private_key privatekey.pem --key_uuid "29ee36ee-13b7-11e7-83de-0cc47a6d39d2" /tmp/  
template.xml /tmp/nvconf.bin  
Saving output...  
Done!
```


mstconfig apply Command


The apply command can be used to apply the NV configurations files to the firmware using the apply command. Only Firmware that supports applying configurations files can be used.

Example:


```
#mstconfig -d 41:00.0 apply /tmp/nvconf.bin  
Applying...  
Done!
```

mstconfig Supported Configurations and Parameters

 The list of MFT Supported Configurations and Parameters is available by running the “mstconfig -d <device> show_confs” command.

 Before setting the number of VFs in SR-IOV, make sure your system can support that number of VFs. If your hardware and software cannot support that number, this may cause your system to cease working. Therefore, mlxconfig protects the user by making sure that when setting SR-IOV parameters, for ConnectX-3 and ConnectX-3 Pro, the value of NUM_OF_VFS*PCI_BAR_SIZE⁽¹⁾ must not exceed 512. For 5th generation devices (Group II devices), however, the value is dependent on the firmware. Also, NUM_OF_VFS must not exceed the limit defined by the firmware (127 VFs upper bound). The same calculation applies to BAR size settings.

⁽¹⁾. PCI_BAR_SIZE refers to the PCI BAR size per function, either physical or virtual.

 In case there were no server booting after enabling SR-IOV, please refer to [Troubleshooting](#).



Support was added to set some of the parameters in mlxconfig in textual values in addition to the numerical values that are still supported. For example: LINK_TYPE_P1 can be set as follows: LINK_TYPE_P1=ETH, instead of: LINK_TYPE_P1=2
 Note that the textual values are case insensitive (either “True” or “true” are accepted).

mstflint - Firmware Burning Tool

The mstflint (Flash interface) utility performs the following functions:

- Burns a binary firmware image to the Flash device attached to an adapter or a switch device.
- Burns an Expansion ROM image to the Flash device attached to adapters.
- 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.

mstflint Synopsis

Switches Options

--allow_rom_change	Allows burning/removing a ROM to/from Firmware image when product version is present.
--hmac_key <hmac_key>	Path to the file containing the HMAC key (For FS4 image only).
--ignore_dev_data	Do not attempt to take device data sections from device (sections will be taken from the image. FS3 image only). Commands affected: burn
--key_uuid <uuid_file>	UUID matching the given private key to be used by the sign command
--key_uuid2 <uuid_file>	UUID matching the given private key to be used by the sign command
--no_fw_ctrl	Do not attempt to work with the firmware Ctrl update commands.
--private_key <key_file>	Path to PEM formatted private key to be used by the sign command
--private_key2 <key_file>	Path to PEM formatted private key to be used by the sign command
--use_fw	Access to flash using FW (ConnectX-3/ConnectX-3Pro device only). Commands affected: all
-banks <banks>	Set the number of attached flash devices (banks)
-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). Commands affected: burn
-clear_semaphore	Force clear the flash semaphore on the device. No command is allowed when this flag is used. NOTE: May result in system instability or flash corruption if the device or another application is currently using the flash. Exercise caution.
-d[evice] <device>	Device flash is connected to. Commands affected: all

-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). Commands affected: burn
-flash_params <type,log2size,num_of_ flashes>	Use the given parameters to access the flash instead of reading them from the flash. Supported parameters: <ul style="list-style-type: none"> • Type: The type of the flash, such as: M25Pxxx, M25Pxx, SST25VFxx, W25QxxBV, W25Xxx, AT25DFxxx, S25FLXXXP • log2size: The log2 of the flash size • num_of_flashes: the number of the flashes connected to the device
--flashed_version	When specified, only flashed fw version is fetched Commands affected: query
-guid <GUID>	GUID base value. 4 GUIDs are automatically assigned to the following values: guid -> node GUID guid+1 -> port1 guid+2 -> port2 guid+3 -> system image GUID NOTE: port2 guid will be assigned even for a single port HCA - The HCA ignores this value. Commands affected: burn, sg
-guids <GUIDs...>	4 GUIDs must be specified here. The specified GUIDs are assigned to the following fields, respectively: node, port1, port2 and system image GUID. NOTE: port2 guid must be specified even for a single port HCA. The HCA ignores this value. It can be set to 0x0. Commands affected: burn, sg
-h[elp]	Prints this message and exits
-hh	Prints extended command help
-i[image] <image>	Binary image file. Commands affected: burn, verify
-log <log_file>	Prints the burning status to the specified log file
--low_cpu	When specified, cpu usage will be reduced. Run time might be increased Commands affected: query
-mac <MAC>1	MAC address base value. 2 MACs are automatically assigned to the following values: mac -> port1 mac+1 -> port2 Commands affected: burn, sg
-macs <MACs...>1	2 MACs must be specified here. The specified MACs are assigned to port1, port2, respectively. Commands affected: burn, sg NOTE: -mac/-macs flags are applicable only for NVIDIA Ethernet products.
-no	Non interactive mode - assume answer "no" to all questions. Commands affected: all
-no_flash_verify	Do not verify each write on the flash.
-nofs	Burns image in a non failsafe manner.
-override_cache_replacement ²	Allow accessing the flash even if the cache replacement mode is enabled. NOTE: This flag is often referred to as -ocr NOTE: This flag is intended for advanced users only. Running in this mode may cause the firmware to hang.
-qq	Run a quick query. When specified, mstflint will not perform full image integrity checks during the query operation. This may shorten execution time when running over slow interfaces (e.g., I2C, MTUSB-1). Commands affected: query
-s[ilent]	Do not print burn progress flyer. Commands affected: burn
-striped_image	Use this flag to indicate that the given image file is in a "striped image" format. Commands affected: query verify

-uid <UID>	5th Generation (Group II) devices only. Derive and set the device's base UID. GUIDs and MACs are derived from the given base UID according to NVIDIA Methodologies. Commands affected: burn, sg
-use_dev_rom	Save the ROM which exists in the device (FS3 and FS4 image only). Commands affected: burn
-use_image_guids	Burn (guids/uids/macs) as appears in the given image. Commands affected: burn
-use_image_ps	Burn vsd as appears in the given image - do not keep existing VSD on flash. Commands affected: burn
-use_image_rom	Do not save the ROM which exists in the device. Commands affected: burn
-v	Version info.
-vsd <string>	Write this string, of up to 208 characters, to VSD when burn.
-y[es]	Non interactive mode - assume answer "yes" to all questions. Commands affected: all
--comid_file <template_file>	Path to CoMID template file to be filled with M1-M4 measurements by the get_measurements.

Note 1. The -mac and -macs options are applicable only to NVIDIA Ethernet adapter and switch devices.

Note 2. When accessing SwitchX via I2C or PCI, the -override_cache_replacement flag must be set.

Command Parameters

Common FW Update and Query

b[urn]	Burn flash
q[ue]ry [full]	Query misc. flash/firmware characteristics, use "full" to get more information.
verify v [showitoc]	Verify entire flash, use "showitoc" to see ITOC headers in FS3/FS4 image only.
swreset	SW reset the target un-managed switch device. This command is supported only in the In-Band access method.
sign_with_hmac	Sign image with HMAC.
get_measurements gm	Calculates M1-M4 measurements (For FS4 image only)

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.
qrom	Query ROM in a given image.

Initial Burn, Production:

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.
checksum cs	Perform MD5 checksum on firmware.
timestamp ts <set query reset> [timestamp] [FW version]	Set/query/reset firmware timestamp.
cache_image ci	Cache FW image (Windows only).
sign	Sign firmware image file
set_public_keys [public key binary file]	Set Public Keys (For FS3/FS4 image only).
set_forbidden_versions [forbidden versions]	Set Forbidden Versions (For FS3/FS4 image only).
binary_compare bc	Verify the firmware image on a device which operates in livefish mode by comparing it with an existing binary firmware file.


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:

hw query	Query HW info and flash attributes.
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
wb <data-file> <addr>	Write a data block to flash
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 commands are non-failsafe when performed on a 5th generation (Group II) device: sg, smg, sv and set_vpd.

 Manufacture GUIDs are similar to GUIDs. However, they are located in the protected area of the flash and set during production. By default, firmware will use GUIDs unless specified otherwise during production.

mstflint: Burning a Firmware Image

The mstflint 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:

```
# mstflint -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)	<p>Optional, for InfiniBand adapters and 4th generation (Group I) switches. One or four GUIDs.</p> <ul style="list-style-type: none"> • 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. <p>NOTE: For 4th generation (Group I), four GUIDs must be specified but Ports 1 and 2 GUIDs are ignored and should be set to 0.</p> <p>NOTE: A GUID is a 16-digit hexadecimal number. If less than 16 digits are provided, leading zeros will be inserted.</p>
MAC(s)	<p>Optional, for Ethernet and VPI adapters and switches.</p> <ul style="list-style-type: none"> • 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. <p>NOTE: A MAC is a 12-digit hexadecimal number. If less than 12 digits are provided, leading zeros will be inserted.</p>

To burn a firmware image:

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

```
# mstflint -d 41:00.0 -i fw-4099-2_42_5000-MCX354A-FCB_A2.bin burn
```

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

```
# mstflint -d 41:00.0 -i fw-4099-2_42_5000-MCX354A-FCB_A2.bin -guid 1234567deadbeef burn
```

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

```
# mstflint -d 41:00.0 -i fw-4099-2_42_5000-MCX354A-FCB_A2.bin -mac 1234567deadbeef 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.

```
burn# mstflint -d 41:00.0 -i fw-4099-2_42_5000-MCX354A-FCB_A2.bin -nofs -guid 12345678 burn
```

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

```
# mstflint -d 41:00:0 ri Flash_Image_Copy.bin
```


6. 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.

```
# mstflint -d 41:00:0 -i /tmp/fw-sx.bin -nofs -guids 000002c900002100 0 0 000002c900002100 -macs 0002c9002100 0002c9002101 b
```


Burning the MFA2 Images

Burning the MFA2 images enables the user to extract (i.e. unzip) 4MB images from MFA2 archive that matches the device type and device PSID. If there are more than one matching images, the user may use the `--latest_fw` flag and burn the latest firmware, or choose the required image from the user menu.


 The device flash **MUST** have all relevant device information (signatures, PSID, VPD, DEV_INFO, MFG_INFO, etc.) valid since MFA2 format does not have that information and without the burn process will fail.

```
#mstflint -d <device> -i <mfa2 file> --psid <PSID string> (optionally) --latest_fw (optionally) -silent (optionally) b (or burn)
```

- Burning the MFA2 Images when the Device Includes a Valid Image
In this scenario, the user *may* (optional) provide a “`-psid`” flag and extract from the MFA2 archive the image that matches this flag, and this way actually change the PSID on the device.
- Burning the MFA2 Images when in Live Fish Mode
In this scenario, the user *must* provide a “`-psid`” flag and extract from the MFA2 archive the image that matches this flag, and this way actually change the PSID on the device.

 Burning mfa2 requires installing mstflint with `--enable-fw-mgr` option.

Cable Firmware Update (In-Field-Firmware-Update)

 This capability is supported only hosts with NVIDIA ConnectX-6 adapter cards.

The In-Field-Firmware-Update (IFFU) tool works via the HCAs in the datacenters and is intended for remote control. The tool is used to update cables transceivers' firmware.

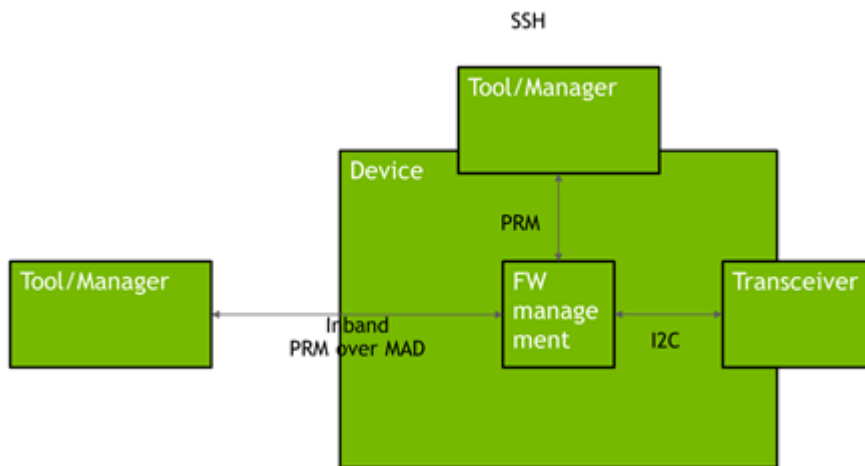
Optical Cables and Transceivers are active network components which run firmware, and as any component running firmware, the ability to update firmware is mandatory. Transceiver firmware update is a system flow which requires the following elements:

- Tool/Manager which will perform the firmware update

- HCA firmware management used as a middleman between the Manager and the cable transceiver
- Transceiver firmware: target for upgrade

The figure below shows the tool/manager which runs on a remotely controlled HCA on host shown as 'Device'.

The manager can query the transceivers type and the current running firmware to understand if an update is required. When an update is required, the manager can apply a set of commands that will send the remote host device a new firmware images for the specific transceiver(s) and activate a firmware update flow. The set of commands is defined with low level primitives to support full flexibility for the user. High level script can be applied on top of the manager and allow system wide update.



The Tool/Manager host must have MST rev. 4.16.00 or later installed. Remote control from outside the cluster (data center) requires access to the host being used as Tool/Manager. When the cluster has many HCAs, multiple hosts may be engaged in the upgrade process. The host(s) can be remotely controlled via VNC access.

Firmware Burning Across a Cluster (Data Center)

The IFFU function described below works on one switch. Cluster-wide firmware updating is done by use of a script which initiates the update procedure in multiple switches in parallel by initiating an instance of the flint command for each switch. In large clusters the script can be executed on multiple hosts, each handling a different part of the cluster.

Cable Burn Command

```
# mstflint -d <device> --linkx <flags> <commands>
```

where:

Flags:

<device>	The name of the target switch (one only).
--downstream_device_id_start_index <downstream_device_id_start_index>	The port number of the first LinkX cable/transceiver. (min. port number = 1)

--num_of_downstream_devices <num_of_downstream_devices>	The number of cables/transceivers to burn. They are burnt sequentially.
--linkx_auto_update	Use this flag to burn all supported cables/transceivers connected to the HCA.
--download_transfer	Use this flag to perform download and transfer of all cable data for cables. Download and transfer are not performed by default. This flag is only relevant for cable components.
--activate	Use this flag to apply the activation of the new firmware in the updated devices. Activation is not performed by default.
--activate_delay_sec <timeout in seconds>	Use this flag to activate all cable devices connected to host with delay, acceptable values are between 0 and 255 (default - 1, immediately). Important: 'activate' flag must be set. This flag is relevant only for cable components.
-i <image>	'i' indicates 'binary Image' followed by the path and file name of the bin file to download into the cable/transceiver.
--downstream_device_ids <list of ports>	Use this flag to specify the LNKX ports to perform query. List must be only comma-separated numbers, without spaces

Commands:

b[urn]	Burn flash
q[uey]	Query misc. flash/firmware characteristics.

Updating the Firmware

Burning a firmware cable transceiver connected to the host is done using the "mstflint" tool. To do so, the user should use the "-linkx" flag.

Firmware can be burnt in follow one of the methods:

- Burn with Auto-update:

1. Transfer the data from the host.

```
# mstflint -d <device> --linkx --linkx_auto_update --download_transfer -i <image> b
```

Example:

```
# mstflint -d /dev/mst/mt4123_pciconf0 --linkx --linkx_auto_update --download_transfer -i image.bin b
```

2. Activate the firmware.

```
# mstflint -d <device> --linkx --linkx_auto_update --activate b
```

Example:

```
# mstflint -d /dev/mst/mt4123_pciconf0 --linkx --linkx_auto_update --activate b
```

Transfer and Activate Example:

```
# mstflint -d /dev/mst/mt4123_pciconf0 --linkx --linkx_auto_update --download_transfer --activate -i image.bin b
```

- Burning a specific port in the HCA using the 'Range':

1. Transfer the data from the host.

```
# mstflint -d <device> --linkx --downstream_device_id_start_index <port_number> --num_of_downstream_devices 1 --download_transfer -i <image> b
```


2. Activate the firmware.

```
# mstflint -d <device> --linkx --downstream_device_id_start_index <port_number> --num_of_downstream_devices 1 --activate b
```

Example of Download Transfer with Activation, range is 1 to 2:


```
# mstflint -d /dev/mst/mt4123_pciconf0 --linkx --downstream_device_id_start_index 1 --num_of_downstream_devices 2 download_transfer --activate -i image.bin b
```

This will update 2 AOCs/Transceivers starting from port 1, i.e. all ports in the range 1...2.

 You cannot 'overburn' the same firmware version into a transceiver/AOC as the one already installed. This is to prevent wasting time re-burning transceivers in a large cluster.


Example of successful update of 1 AOC:

```
-I- Downloading FW ...  
FSMST_INITIALIZE - OK  
Writing COMPID_LINKX component - OK  
FSMST_LOCKED - OK  
FSMST_DOWNSTREAM_DEVICE_TRANSFER - OK  
FSMST_LOCKED - OK  
Please wait while activating the transceiver(s) FW ...  
FSMST_ACTIVATE - OK..]  
-I- Cable burn finished successfully.
```

 Downloading and burning takes approx. 1½ minute + activation ½ minute for one cable.

Querying Vendor Specific Firmware Information from a NVIDIA AOC / Transceiver

Querying a firmware cable transceiver is done using the "mstflint" tool.

 In case the Vendor Specific query command is not support by the firmware, it will run the CMIS standard query implemented by the firmware.

```
# mstflint -d <cable device> q
```

Querying Firmware Information from an AOC / Transceiver

Querying a firmware cable transceiver connected to the host is done using the "mstflint" tool. To do so, the user should use the "-linkx" flag.

```
# mstflint -d <device> --linkx --downstream_device_ids <ids> [--output_file <file_name>] q
```

Query ports 1,2,5 Example:

```
# mstflint -d <device> --linkx --downstream_device_ids 1,2 q
```

The system responds with information about the firmware version loaded into the transceivers.

Checking successful burning and operation - Example:

It is essential to check that the links come up AFTER the cable FW is updated and reactivated. This can be done as follows:

```
# for i in {1..2}; do echo $i; mstlink -d /dev/mst/mt4123_pciconf0 -p $i -m | grep 'Part\|FW\|State'; done
```

The 'State' parameter was added to the query. The response has the following format (example):

```
# 1
State           : Active
Vendor Part Number : MFS1S00-H010
FW Version      : 38.100.59

2
State           : Active
Vendor Part Number : MFS1S00-H010
FW Version      : 38.100.59
```

mstflint: Managing an Expansion ROM Image

To burn an Expansion ROM image, run the following command:

```
# mstflint -d <mst device> brom <image name>.mrom
```

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

Example:

```
# mstflint -d 41:00.0 brom example.mrom
Current ROM info on flash: N/A

New ROM info: type=PXE version=3.5.305 cpu=AMD64
Burning ROM image - OK
Restoring signature - OK
#
```

To read an expansion ROM image to a file, run the following command:

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

Example:

```
# mstflint -d 41:00.0 rromexample.mrom
# mstflint -d 41:00.0 q
Image type:      FS2
FW Version:     2.42.5000
FW Release Date: 4.5.2017
Rom Info: type=PXE version=3.5.305 cpu=AMD64
Device ID:     4099

Description:  Node          Port1          Port2          Sys image
GUIDs:       f45214030001b8a0 f45214030001b8a1 f45214030001b8a2 f45214030001b8a3
MACs:       f4521401b8a1      f4521401b8a2
```



```
VSD:
PSID:      MT_1090120019
#
```

To remove the expansion ROM, run the following command:

```
# mstflint -d <mst device> drom
```

Examples:

```
# mstflint -d 41:00.0 drom
Removing ROM image - OK
Restoring signature - OK
```

mstflint: Performing Checksum Calculation on Image/Device

The flint utility allows performing an MD5 checksum on the non-persistent sections of the firmware image. For example: the sections that are changed when performing a firmware upgrade.

To perform a checksum on the flash, run the following command line:
mstflint -d <mst device> checksum
To perform a checksum on a firmware image, run the following command line:

```
# mstflint -i <image file> checksumwhere:
```

device	Flash device to verify.
image file	Image file to verify.

Examples:
mstflint -i fw-ConnectX4Lx.bin checksum -I- Calculating Checksum ... Checksum:
68ddae6bfe42f87f09084f3f468a35c6

```
mstflint -d 41:00.0
-I- Calculating Checksum ...
Checksum: 68ddae6bfe42f87f09084f3f468a35c6
```

mstflint: Querying the Firmware Image

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

```
# mstflint -d <device> q
```

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

```
# mstflint -i <image file> q
```

where:

device	Device on which the query is run.
image file	Image file on which the query is run.

Examples:

- Query the FW on the device.
mstflint -d 41:00.0 query
- Query the FW image file.
mstflint -i 25408-2_42_5000-MCX354A-FCB_A2.bin query
- Security Attributes field in Query output:
This field lists the security attributes of the device's firmware, where:

- Secure-fw: This attribute indicates that this binary/device supports secure-firmware-updates. It means that only officially signed binaries can be loaded to the device from the host, and that the current binary is signed.
- Signed-fw: This attribute indicates that that this binary is signed and that the device can verify digital signatures of new updates. However, unlike, secure-fw, there might still be methods to upload unsigned binaries to the device from the host.
- debug: This attribute indicate that this binary is (or this device runs) a debug-version. Debug versions are custom made for specific data-centers or labs, and can only be installed after a corresponding debug-fw token is pushed to the device. The debug-fw-token, which is digitally signed, includes a list of the target devices MAC addresses.
- dev: This attribute indicates that the firmware is signed with development (test) key.
- Default Update Method" field in Query Full output:{This field reflect the method which mstflint will use in order to update the device. The user can enforce a different method using the -no_fw_ctrl or the -ocr flags.The default methods are:
 - Legacy: mstflint will use the low level flash access registers.
 - fw_ctrl: mstflint will operate the ‘firmware component update’ state machine.
- Secure-boot attributes
 - Secure-boot : This attribute indicates if the device supports secure-boot
 - Life-cycle : This attribute indicates the current status of secure-boot
 - Security-version:
 - For query on image: This attribute indicates the security-version of the image.
 - For query on device:
 - “EFUSE security version”: Indicates the security version of the device
 - “Image security version”: Indicates the security version of the image on the flash
 - Programming method: Indicates when the boot will program the “EFUSE security version” to be aligned with the “image security version”.

Querying the MFA2 File

This capability enables the user to query the MFA2 file using a PSID.

To query, run:

```
flint -i <mfa2_file> --psid <PSID> q
```

mstflint: Setting GUIDs and MACs

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

4th Generation (Group I) Devices

On 4th generation/Group I 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 mstflint will re-burn the current image using the given GUIDs/MACs/UIDs.

1. Change the GUIDs/MACs on a device:

```
# mstflint -d 41:00.0 q
-W- Running quick query - Skipping full image integrity checks.
Image type:      FS2
FW Version:      2.42.5000
FW Release Date: 4.5.2017
Device ID:       4099
Description:     Node          Port1          Port2          Sys image
GUIDs:          f45214030001b8a0 f45214030001b8a1 f45214030001b8a2 f45214030001b8a3
MACs:           f4521401b8a1     f4521401b8a2
VSD:
PSID:           MT_1090120019

# mstflint -d 41:00.0 -guid 0x452140300abadaba -mac 0x300abadaba sg
-W- GUIDs are already set, re-burning image with the new GUIDs ...
You are about to change the Guids/Macs/Uids on the device:

      New Values      Current Values
Node GUID:           452140300abadaba f45214030001b8a0
Port1 GUID:          452140300abadabb f45214030001b8a1
Port2 GUID:          452140300abadabc f45214030001b8a2
Sys.Image GUID:     452140300abadabd f45214030001b8a3
Port1 MAC:           00300abadaba     f4521401b8a1
Port2 MAC:           00300abadabb     f4521401b8a2

Do you want to continue ? (y/n) [n] : y
Burning FS2 FW image without signatures - OK
Restoring signature - OK

# mstflint -d 41:00.0 q
Image type:      FS2
FW Version:      2.31.5050
FW Release Date: 4.5.2014
Device ID:       4099
Description:     Node          Port1          Port2          Sys image
GUIDs:          452140300abadaba 452140300abadabb 452140300abadabc 452140300abadabd
MACs:           00300abadaba     00300abadabb
VSD:
PSID:           MT_1090120019
```

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

```
# mstflint -i /tmp/image.bin q
Image type:      fs2
FW Version:      2.31.5050
FW Release Date: 4.5.2014
Device ID:       4099
Description:     Node          Port1          Port2          Sys image
GUIDs:          f45214030001b8a0 f45214030001b8a1 f45214030001b8a2 f45214030001b8a3
MACs:           00300abadaba     00300abadabb
VSD:
PSID:           MT_1090120019

# mstflint -i /tmp/image.bin -guid 0002c9000abcdef0 -mac 02c90abcdef0 sg
You are about to change the Guids/Macs/Uids on the device:

      New Values      Current Values
Node GUID:           0002c9000abcdef0 f45214030001b8a0
Port1 GUID:          0002c9000abcdef1 f45214030001b8a1
Port2 GUID:          0002c9000abcdef2 f45214030001b8a2
Sys.Image GUID:     0002c9000abcdef3 f45214030001b8a3
Port1 MAC:           02c90abcdef0     00300abadaba
Port2 MAC:           02c90abcdef1     00300abadabb

Do you want to continue ? (y/n) [n] : y
Restoring signature - OK
# mstflint -i /tmp/image.bin q
Image type:      FS2
FW Version:      2.31.5050
FW Release Date: 4.5.2014
Device ID:       4099
Description:     Node          Port1          Port2          Sys image
GUIDs:          0002c9000abcdef0 0002c9000abcdef1 0002c9000abcdef2 0002c9000abcdef3
MACs:           02c90abcdef0     02c90abcdef1
```

VSD:
PSID: MT_1090120019

5th Generation (Group II) Devices

On 5th Generation (Group II) 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, -uid flag must be specified. For ConnectX-4, -guid/-mac flags can be specified. By default, 8 GUIDs will be assigned for each port starting from base, base+1 up until base+7 for port 1 and base+8 up until base+15 for port 2.

To change the step size and the number of GUIDs per port, specify `guids_num=<num> step_size=<size>` to the `sg` command.

1. Change GUIDs for device:

```
# mstflint -d 41:00.0 q
Image type:      FS3
FW Version:      10.10.3000
FW Release Date: 29.4.2014
Description:     UID
Base GUID1:      0002c903002ef500  8      1
Base GUID2:      0002c903002ef508  8      1
Base MAC1:       0002c92ef500      8      1
Base MAC2:       0002c92ef508      8      1
Image VSD:
Device VSD:     VSD
PSID:           MT_1240110019

# mstflint -d 41:00.0 -uid 0002c123456abcd -ocr sg
-W- Firmware flash cache access is enabled. Running in this mode may cause the firmware to hang.
Updating GUID section - OK
Updating ITOC section - OK
Restoring signature - OK

# mstflint -d 41:00.0 q
Image type:      FS3
FW Version:      10.10.3000
FW Release Date: 29.4.2014
Description:     UID
Base GUID1:      00002c123456abcd  8      1
Orig Base GUID1: 0002c903002ef500  8      1
Base GUID2:      00002c123456abd5  8      1
Orig Base GUID2: 0002c903002ef508  8      1
Base MAC1:       00002c56abcd      8      1
Orig Base MAC1:  0002c92ef500      8      1
Base MAC2:       00002c56abd5     8      1
Orig Base MAC2:  0002c92ef508     8      1
Image VSD:
Device VSD:     VSD
PSID:           MT_1240110019
```

⚠ Orig Base GUID/MAC refers to the GUIDs/MACs located in the MFG(manufacture guides) section of the flash/image.

2. Change GUIDS for device (specifying `guids_num` and `step_size`):

```
# mstflint -d 41:00.0 q
Image type:      FS3
FW Version:      10.10.3000
FW Release Date: 29.4.2014
Description:     UID
Base GUID1:      0002c903002ef500  8      1
Base GUID2:      0002c903002ef508  8      1
Base MAC1:       0002c92ef500      8      1
```

```

Base MAC2:      0002c92ef508      8      1
Image VSD:
Device VSD:    VSD
PSID:          MT_1240110019

# mstflint -d 41:00.0 -uid 0000000000000001 -ocr sg guids_num=2 step_size=1
-W- Firmware flash cache access is enabled. Running in this mode may cause the firmware to hang.
Updating GUID section - OK
Updating ITOC section - OK
Restoring signature - OK

# mstflint -d 41:00.0 q
Image type:    FS3
FW Version:    10.10.3000
FW Release Date: 29.4.2014
Description:    UID      GuidsNumber  Step
Base GUID1:    0000000000000001  2      1
Orig Base GUID1: 0002c903002ef500  8      1
Base GUID2:    0000000000000003  2      1
Orig Base GUID2: 0002c903002ef508  8      1
Base MAC1:     000000000001      2      1
Orig Base MAC1: 0002c92ef500      8      1
Base MAC2:     000000000003      2      1
Orig Base MAC2: 0002c92ef508      8      1
Image VSD:
Device VSD:    VSD
PSID:          MT_1240110019

```

3. Change GUIDs for image:

```

# mstflint -i /tmp/connect-ib.bin q
Image type:    FS3
FW Version:    10.10.3000
FW Release Date: 29.4.2014
Description:    UID      GuidsNumber  Step
Base GUID1:    0002c903002ef500  8      1
Base GUID2:    0002c903002ef508  8      1
Base MAC1:     0002c92ef500      8      1
Base MAC2:     0002c92ef508      8      1
Image VSD:
Device VSD:    VSD
PSID:          MT_1240110019

# mstflint -i /tmp/connect-ib.bin -uid 000123456abcd sg
Updating GUID section - OK
Updating ITOC section - OK
Restoring signature - OK

# mstflint -i /tmp/connect-ib.bin q
Image type:    FS3
FW Version:    10.10.3000
FW Release Date: 29.4.2014
Description:    UID      GuidsNumber  Step
Base GUID1:    000000123456abcd  8      1
Orig Base GUID1: 0002c903002ef500  8      1
Base GUID2:    000000123456abd5  8      1
Orig Base GUID2: 0002c903002ef508  8      1
Base MAC1:     00000056abcd      8      1
Orig Base MAC1: 0002c92ef500      8      1
Base MAC2:     00000056abd5      8      1
Orig Base MAC2: 0002c92ef508      8      1
Image VSD:
Device VSD:    VSD
PSID:          MT_1240110019

```

4. Change GUIDs and MACs for the ConnectX-4 device:

```

# mstflint -d 41:00.0 -guid e41d2d0300570fc0 -mac 0000e41d2d570fc0 -ocr sg
-W- Firmware flash cache access is enabled. Running in this mode may cause the firmware to hang.
Updating GUID section - OK
Updating ITOC section - OK
Restoring signature - OK

# mstflint -d 41:00.0 q
Image type:    FS3
FW Version:    12.0100.5630
FW Release Date: 23.3.2015

```

```

Description:      UID                GuidsNumber
Base GUID:       e41d2d0300570fc0    4
Base MAC:        e41d2d570fc0    4
Image VSD:
Device VSD:
PSID:           MT_2190110032
add note:
GUIDs and MACs can be changed separately on ConnectX4

```

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. 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 mstflint "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.

To pre-burn an image with blank GUIDs/MACs:

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

```

# mstflint -d 41:00.0 q
Image type:      FS2
FW Version:     2.31.5050
FW Release Date: 4.5.2014
Device ID:      4099
Description:    Node      Port1      Port2      Sys image
GUIDs:         ffffffff ffffffff ffffffff ffffffff
MACs:          ffffffff ffffffff ffffffff ffffffff
VSD:           n/a
PSID:          MT_1090120019

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

```

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

```
# mstflint -d 41:00.0 -guid 0x0002c9030abcdef0 -mac 0x0002c9bcdef1 sg
```

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

```

sg# mstflint -d 41:00.0 q
Image type:      FS2
FW Version:     2.31.5050
FW Release Date: 4.5.2014
Device ID:      4099
Description:    Node      Port1      Port2      Sys image
GUIDs:         0002c9030abcdef0 0002c9030abcdef1 0002c9030abcdef2 0002c9030abcdef3
MACs:          0002c9bcdef1      0002c9bcdef2
VSD:           n/a
PSID:          MT_1090120019

```

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.

To pre-burn an image with specific GUIDs/MACs for each device:

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

```
mstflint -i ./fw-ConnectX3-rel.bin -guid 0x0002c9030abcdef0 -mac 0x0002c9bcdef1 -striped_image sg
```

2. (Optional) After assembly, query the image on flash to verify there are no GUIDs on the device.

```
sg# mstflint -i ./fw-ConnectX3-rel.bin -striped_image q
Image type:      FS2
FW Version:      2.31.5050
FW Release Date: 4.5.2014
Device ID:       4099
Description:
GUIDs:           Node          Port1          Port2          Sys image
MACs:            0002c9030abcdef0 0002c9030abcdef1 0002c9030abcdef2 0002c9030abcdef3
VSD:             n/a
PSID:            MT_1090120019
```

Now the fw-ConnectX3-rel.bin image can be pre-burned to the flash. After the assembly, the device would be fully functional.

mstflint: Verifying the Firmware Image

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

```
# mstflint -d <device> verify
```

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

```
# mstflint -i <image file> v
```

where:

device	Flash device to verify.
image file	Image file to verify.

Examples:

```
# mstflint -d 41:00.0 v
# mstflint -i ./image_file.bin verify
```

Comparing the Binary Image

Binary comparison of the firmware image enables the user to verify that a given firmware image contains the image that matches the given device.



Since ConnectX-4/ConnectX-4 Lx devices have iTOC (image specific) and dTOC (device specific) sections at the beginning of the device flash, and the MFA2 archive does not have the dTOC information by its definition, the binary comparison will ignore the device specific

sections on the device.

```
#mstflint -d <device> -i <fw image> --silent (optional) bc (or  
binary_compare)
```

The Verify Command on Encrypted Flash/Image



The `verify` command on encrypted flash/image is applicable on adapter cards starting from ConnectX-7.

The flint tool supports the `verify` command on encrypted flash/image, as follows:

- When both the device and the image are given the `verify` command: verifying encrypted flash, the flint tool will execute `binary-compare` between the flash and the given image (the image is expected to be the one burnt on the device). In case of a device in recovery mode, the `verify` action is applicable before transcoding.
- When an encrypted device/image is given the `verify` command: only DTOC CRCs will be verified. In case a device is given, the `verify` is applicable in recovery mode only.

Verifying MFA2 Archive

Binary verifying of MFA2 archive enables the user to verify that a given MFA2 archive contains the image that matches the given device.



Since ConnectX-4/ConnectX-4 Lx devices have iTOC (image specific) and dTOC (device specific) sections at the beginning of the device flash, and the MFA2 archive does not have the dTOC information by its definition, the binary comparison will ignore the device specific sections on the device.

```
#mstflint -d <device> -i <mfa2 file> --silent (optional) bc (or  
binary_compare)
```

mstflint: Setting the VSD

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

Example:

```
# mstflint -d 41:00.0 -vsd "MELLANOX" sv  
Setting the VSD - OK  
Restoring signature - OK  
  
# mstflint -d 41:00.0 q  
Image type: FS2  
FW Version: 2.31.5050  
FW Release Date: 4.5.2014  
Device ID: 4099  
Description: Node Port1 Port2 Sys image  
GUIDs: f45214030001b8a0 f45214030001b8a1 f45214030001b8a2 f45214030001b8a3  
MACs: 00300abadaba 00300abadabb  
VSD: MELLANOX  
PSID: MT_1090120019
```


mstflint: 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.


 The secure host feature requires a MLNX_OFED driver installed on the machine.

4th Generation Devices

To disable/enable access to the hardware:

1. Set the key:

```
# mstflint -d 41:00.0 set_key 22062011
Setting the HW Key - OK
Restoring signature - OK
```

 A driver restart is required to activate the new key.

2. Access the HW while HW access is disabled:


```
# mstflint -d 41:00.0 q
E- Cannot open 41:00.0: HW access is disabled on the device.
E- Run "mstflint -d 41:00.0 hw_access enable" in order to enable HW access.
```

3. Enable HW access:

```
# mstflint -d 41:00.0 hw_access enable
Enter Key: *****
```

4. Disable HW access:

```
# mstflint -d 41:00.0 hw_access disable
```

 **WARNING:**

1. Once a hardware access key is set, the hardware can be accessed only after the correct key is provided.
2. If a key is lost, there is no way to recover it using the tool. The only way to recover from a lost key is to:
 - Connect the flash-not-present jumper on the card
 - Boot in "flash recovery" mode
 - Re-burn FW
 - Re-set the HW access key

5th Generation Devices

Secure Host can be enabled on 5th generation devices in one of the following manners:

1. Set the key:

```
# mstflint -d 41:00.0 set_key 18022018
-I- Secure Host was enabled successfully on the device.
```

2. Disable HW access:

```
# mstflint -d 41:00.0 hw_access disable 18022018
-I- Secure Host was enabled successfully on the device.
```

If the key was not provided in the command line, an interactive shell will ask for it, and verifying it:

```
# mstflint -d 41:00.0 set_key
Enter Key : *****
Verify Key : *****
-I- Secure Host was enabled successfully on the device.
```

Or

1. Disable the Secure Host (Enable HW access):

```
# mstflint -d 41:00.0 hw_access enable 18022018
-I- The Secure Host was disabled successfully on the device.
And the same as previous, providing the key can be done in interactive shell:
# mstflint -d 41:00.0 hw_access enable
Enter Key : *****
-I- The Secure Host was disabled successfully on the device.
```

mstflint: Flash Operations

Reading a Word from Flash

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

```
# mstflint -d <device> rw addr
```

where:

device	The device the dword is read from.
addr	The address of the word to read.

Example:

```
# mstflint -d 41:00.0 rw 0x20
```

Writing a dword to Flash

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

```
# mstflint -d <device> ww 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:

```
# mstflint -d 41:00.0 ww 0x10008 0x5a445a44
```

Writing a dword to Flash Without Sector Erase

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

```
# mstflint -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:

```
# mstflint -d 41:00.0 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.

Erasing a Sector

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

```
# mstflint -d <device> e addr
```

where:

device	The device the dword is erased from.
addr	The address of a word in the sector that you want to erase.

Example:

```
# mstflint -d 41:00.0/mtusb-1 e 0x1000
```

Querying Flash Parameters

To query flash parameters use the following command line:

```
# mstflint -d <device> [-ocr] hw query
```

where:

device	The device to query.
--------	----------------------

Example:

```
# mstflint -d 41:00.0 hw query
```

mstflint: Comparing the Binary Image



This capability is applicable to ConnectX-5 onwards adapter cards.

This enables the user to verify a firmware image on a device which operates in livefish mode by comparing it with an existing binary firmware file.

Example:

```
mstflint -d <device BDF, like 04:00.0 > -i <binary file *.BIN> <-silent>/<-s> bc
```



The <-silent>/<-s> parameter is optional. However, if the silent mode is chosen, the percentage progress is not displayed.

mstflint: Firmware Timestamping for Multi-Host Environment

In a multi-host environment, every host can upgrade the NIC firmware. All hosts are treated equally and there is no designated host. Hence, there can be situations where one host will try to upgrade the firmware and another will try to downgrade; which may lead to two or more unnecessary server reboots. In order to avoid such situations, the administrator can add a timestamp to the firmware they want to upgrade to. Attempts to burn a firmware image with a timestamp value that is lower than the current firmware timestamp will fail.



Firmware timestamping can be used on Connect-IB/ConnectX-4/ConnectX-4 Lx HCAs for controlling the firmware upgrade/downgrade flow.

Setting a Timestamp on Image

In order to set a timestamp on an image, run:

```
# mstflint -i ./fw-4115.bin timestamp set [UTC time]
```



The user can either specify a combined date and time timestamp in UTC which conforms to ISO 8601, or let the tool use the machine's time for the timestamp.

Querying a Timestamp on Image

To view the timestamp that was set on the device, run:

```
# mstflint -d /41:00.0/mt4115_pciconf0 timestamp query
Current timestamp : N/A. No valid timestamp found
Next timestamp   : 2015-12-21T10:58:23Z 12.15.0005
```

- “Current timestamp” represents the current running firmware timestamp. If “N/A” is visible, then the timestamp entry is invalid (example: first use of the feature or after resetting the timestamp).
- “Next timestamp” represents the next firmware that is allowed to be burnt on the HCA. Updating the “Next timestamp” requires an equal or newer timestamp to be provided.

Resetting a Timestamp on Device

To reset the timestamp that was set on the device, run:

```
# mstflint -d /41:00.0/mt4115_pciconf0 timestamp reset
```

Resetting the timestamp on device causes invalidation of both “Current timestamp” and “Next timestamp” fields.

Setting a Timestamp on Device

In case it is not possible to modify the firmware image, it is possible to set the timestamp directly on the device by specifying the timestamp and firmware version tied to it.

```
# mstflint -d /41:00.0/mt4115_pciconf0 timestamp set <UTC time> <Firmware version>
```

Querying a Timestamp on Device

To view the timestamp that was set on the device, run:

```
# mstflint -d /41:00.0/mt4115_pciconf0 timestamp query
Current timestamp : N/A. No valid timestamp found
Next timestamp   : 2015-12-21T10:58:23Z 12.15.0005
```

- “Current timestamp” represents the current running firmware timestamp. If N/A is visible, then the timestamp entry is invalid (example: first use of the feature or after resetting the timestamp).
- “Next timestamp” represents the next firmware that is allowed to be burnt on the HCA. Updating the “Next timestamp” requires an equal or newer timestamp to be provided.

Resetting a Timestamp on Device

To reset the timestamp that were set on the device, run:

```
# mstflint -d 41:00.0 timestamp reset
```

Resetting the timestamp on device causes invalidation of both “Current timestamp” and “Next timestamp” fields.

Important Notes

Please note the following:

- If a firmware image contains a timestamp, the burning tool will automatically attempt to set it on the device. If the operation succeeds, the firmware will be burnt.
- If a timestamp was only set on the device, the burning tool will prevent the burning of any firmware version different than the one set in the timestamp set operation.
- Lack of timestamp in both image and device will cause no checks to be performed.

mstflint Limitations

- When running mstflint via an MTUSB-1 device, a burn/query command may take up to 45 minutes to complete.
 - To accelerate the burn process add the flag `-no_flash_verify` to the command line which skips the flash verification step. This flag, however, does not verify if the image is burnt correctly.
- Burning an image to a ConnectX-3 adapter in Flash recovery mode may fail on some server types (that use PCIe spread spectrum). The tool may not be able to recognize the device’s PCI CONF0 or the image burn may not complete successfully.
 - To burn the device, use the MTUSB-1 connection.
- To load the newly burnt firmware image, a driver restart is required for ConnectX-3/ ConnectX-3 Pro cards.
 - For fifth generation (Group II) devices, run the `mstfwreset` tool or reboot the system.

mstflint: Secure Host

Secure host is the general term for the capability of a device to protect itself and the subnet from malicious software through mechanisms such as blocking access of untrusted entities to the device configuration registers.



WARNING:

- Once a hardware access key is set, the hardware can be accessed only after the correct key is provided.
- If a key is lost, please refer to [Key Loss Recovery](#).



- The hardware access in this mode is allowed only if a correct 64 bits key is provided.
- The secure host feature for ConnectX-3/ConnectX-3 Pro HCAs requires a MLNX_OFED driver installed on the machine.

Using Secure Host

Secure Host feature is supported for all NVIDIA network adapters (listed in Group 1 and group 2). For group 1 network adapters, the user is required to generate and burn a firmware image that supports the feature (see “Generating/Burning a Firmware Supporting Secure Host” below).

For Group 2 network adapters, the feature is supported on firmware version 1x.22.1002 or newer.

Generating/Burning a Firmware Supporting Secure Host

1. Make sure you have INI and mlx files suitable for the device. Both files are available for download at: http://www.mellanox.com/page/custom_firmware_table
 - a. Add `cr_protection_en=true` under [HCA] section in the INI file.
2. Burn the image on the device using `mstflint`:

```
# mstflint -d 41:00.0 -i fw-4099.secure.bin b
```

3. For changes to take effect, reboot is required.

Setting the Secure Host Key

To set the key, run:

```
# mstflint -d 41:00.0 set_key 22062011
Setting the HW Key - OK
Restoring signature - OK
```



A driver restart is required to activate the new key.

Disabling/Enabling Access to the Hardware

1. Access the hardware while hardware access is disabled:

```
# mstflint -d 41:00.0 q
E- Cannot open 41:00.0: HW access is disabled on the device.
E- Run "mstflint -d 41:00.0 hw_access enable" in order to enable HW access.
```


2. Enable hardware access:

```
# mstflint -d 41:00.0 hw_access enable
Enter Key: *****
```

3. Disable hardware access:

```
# mstflint -d 41:00:0 hw_access disable
```

Removing the Secure Host

 This section is applicable to Group 1 network adapters only.

To remove the secure host feature:

1. Make sure you have INI and MLX file suitable for the device.
 - a. Remove `cr_protection_en=true` from the INI (if present)
2. Burn the firmware on the device (make sure hardware access is enabled prior to burning):

```
# mstflint -d 41:00:0 -i fw-4099.unsecure.bin b
```

3. Execute a driver restart in order to load the unsecure firmware:


```
# service openibd restart
```

Key Loss Recovery

If a key is lost, there is no way to recover it using the tool. The only way to recover is to:

1. Connect the flash-not-present jumper on the card.
2. Reboot the machine.
3. Re-burn firmware
4. Remove the flash-not-present jumper.
5. Reboot the machine
6. Re-set the hardware access key

mstflint: Secure Firmware Update

 Secure Firmware Update is supported only on ConnectX-4 onwards adapter cards and as of mstflint v4.10.0-3.

A “Secure firmware update” is the ability of a device to verify digital signatures of new firmware binaries, in order to assure that only officially approved versions can be installed from the host, the network[1] or a Board Management Controller (BMC).

The firmware of devices with “secure firmware up date” functionality (secure FW), restricts access to specific commands and registers that can be used to modify the firmware binary image on the flash, as well as commands that can jeopardize security in general. Most notably, the commands and registers for random flash access are disabled.

Secure FW verifies new binaries before activating them, compared to legacy devices where this task is done by the update tool using direct flash access commands. In addition to signature verification, secure FW also checks that the binary is designated to the same device model, that the new

firmware is also secured, and that the new FW version is not included in a forbidden versions blacklist. The firmware rejects binaries that do not match the verification criteria.

Secure FW utilizes the same ‘fail safe’ upgrade procedures, so events like power failure during update should not leave the device in an unstable state. The table below lists the impact of secure FW update on mstflint tools.

Tool	Flow	Secure FW	With CS Token	Blocked Commands
mstflint	Burn FW	Working with controlled fw update	Working with controlled fw update	
	Query	Working with MCC commands	Working with MCC commands	
	Set GUIDs	Working with controlled fw update	Working with controlled fw update	
	Verify	Working partially (BOOT image)	Working partially (BOOT image)	
	Set DV INFO: SET MFG, SET VSD, VPD	Not supported in Secure FW	Not supported in Secure FW	MFBA
	ROM OPS: BROM, DROM	Not supported, BOOT image modification is not supported (MFBA)	Not supported, BOOT image modification is not supported (MFBA)	MFBA
	"-ocr" override cache replacement (Direct flash GW access)	Not supported in Secure FW	Not supported in Secure FW	Flash GW is blocked
	HW SET (Set flash parameters)	Flash GW is blocked	Flash GW is blocked	Flash GW is blocked
	"--no_fw_ctrl" (Legacy Flow)	Not supported in Secure FW	Not supported in Secure FW	MFBA
mstmcr	Read	working	working	working
	Write	Read Only CR- Space	working	Read Only CR- Space
mstregdump	Read	working	working	working
mstconfig	working	working	working	working
mstfwreset	working	working	working	working

The following sections describe how Secure FW updates are performed.

Signing Binary Image Files

For firmware Secure purposes, you may sign the image file using the sign command. If you do not provide the sign command with a private key and UUID, the command will only compute SHA256 digest and add it to the image signature section. The sign command supports RSA keys with lengths of 2048 and 4096 bits.

- If you provide a private key with the length of 2048 bits, the command will compute SHA256 digest and encrypt it with the private key and add the result with the provided UUID to the appropriate image signature section.

- If you provide a private key with the length of 4096 bits, the command will compute SHA512 digest and encrypt it with the provided key and add the result with the provided UUID to the appropriate image signature.

You can sign with two keys in the same command by providing keys with lengths of 2048 and 4096 bits. The flags to be used for the first private key and uuid are “--private_key” and “--key_uuid”, and for the second private and uuid use “--private_key2” and “--key_uuid2”.

The motivation for signing with two keys is to allow a firmware update from both firmwares, the one that supports only 2048bit keys and the one that supports 4096bit keys.

Examples:

```
# mstflint -i /tmp/image.bin sign --private_key privatekey.pem --key_uuid "e0129552-13ba-11e7-a990-0cc47a6d39d2"
```

```
# mstflint -i /tmp/image.bin sign --private_key privatekey_2048.pem --key_uuid "e0129552-13ba-11e7-a990-0cc47a6d39d2" --private_key2 privatekey_4096.pem --key_uuid2 "a0b43568-17cb-16e9-a990-0ff47a6d39e4"
```

Setting a “Public Keys” Section in a Binary Image File

To override the public keys section in a given binary image file, use `set_public_key`.

```
# mstflint -i /tmp/image.bin set_public_keys public_key.bin
```

Setting a "Forbidden Versions" Section in a Binary Image File

To override the forbidden versions section in a given binary image file, use `set_forbidden_versions`.

```
# mstflint -i /tmp/image.bin set_forbidden_versions forbidden_versions.bin
```

Secure Firmware Implications on Burning Tools

When Secure Firmware is enabled, the `mstflint` output slightly changes due to the differences in the underlying NIC accessing methods. Some functionalities may be restricted according to the device security level.

`mstflint query` under secure mode:

```
# mstflint -d 41:00.0 q
Image type:      FS3
FW Version:     12.19.2278
FW Release Date: 7.6.2017
Description:    UID                               GuidNumber
Base GUID:      7efe90030029205e 4
Base MAC:       00007cfe9029205e 4
Image VSD:
Device VSD:
PSID:           MT_2190110032
Security Attributes: secure-fw, dev
```



Unavailable information is reported as N/A.

In secure firmware, a firmware update will be successful if an image is signed with a valid key that is recognized by the running firmware on the chip. For more information, please refer to [Signing Binary Image Files](#). If the security type permits legacy flash access commands, the `--no_fw_ctrl` flag can be used to command the `mstflint` to work in the non firmware controlled mode. This means that all the non-secure functionality will be supported using this flag, and the burn flow will work without requiring a signed image. Example:

```
# mstflint -d 41:00.0 --no_fw_ctrl q
Image type:      FS3
FW Version:      12.19.2096
FW Release Date: 26.3.2017
Description:     UID          CuidsNumber
Base GUID:       248a07030094050c   4
Base MAC:        0000248a0794050c 4
Image VSD:
Device VSD:
PSID:            MT_2170110021
```

Re-Signing a Binary Image File

The following procedure is intended to be implemented by customers who want to use their keys to sign a secured firmware.

1. Set the public keys in a given firmware image:
 - a. Generate a binary file that contains 8 public keys.
You can use `mstconfig` command `xml2bin` to generate the file:
 - i. To generate 2048 bits public keys:
 1. Run: `mstconfig gen_tlvs_file output.txt`.
 2. Open the `output.txt`.
 3. Go to the line starting with `"file_public_key"` and change the 0 to 1.
 4. Save the file and exit.
 5. Run: `mstconfig gen_xml_template output.txt output.xml`
 6. Open the `output.xml`.
 7. Duplicate the xml node `"file_public_key"` so the file has 8 copies, for each node fill it as follows:
 - `cs_token_en = 0`
 - `fw_en = 1`
 - `mlnx_nvconf_en = 1`
 - `vendor_nvconf_en = 1`
 - `auth_type: 0x3` for 2048 bits keys and `0x4` for 4096 bits keys.

```
<public_key_exp>4083403379</public_key_exp>
<keypair_uuid>5A7A2B2A87DB7416</keypair_uuid>
<key>
f800000300000000000000000000000010001c459afea005911e797dc00000000000b8168ba624e5cac81d4
91f48c6a3b8f1a816cb7dea789d770893b0fb5abeb67f7a8d19ad8d4203dd8b85b3faaf96187b116eb
1c5d3f3517c3ce8b4422395f2e43ccb286d4bc4474c8385e857349f35be3094f25ccbd71c209c6531f0
d8bcaacd8bbf14af58809e8937e4db424b3d0c48e0cae7b89f53f797b9e24335900448466b0e5182e3a9
4c31e18487f8fe367862c8a70e8c7007d2400760461bbb36470a26d6db13d2e63d137d67cd449c0788c
307ce2dbc3f580ec7207cdb856472520ee956912cfa77e6e793f620d6e362fa13da036003f85ae8dbb
22d4b314ceb64c
</key>
```

⚠ You can have spaces between the bytes: f8 00 00 03, or you can have multiple lines.
The order of the bytes is the same as the output of openssl file, Therefore, you can take the key as is from the openssl file.

8. Save and Exit.
 9. Run: `mstconfig xml2bin output.xml output.bin`
 - ii. To generate 4096 bits public keys, please follow the same steps as above, but use "file_public_key_4096" instead of "file_public_key".
For further information, see [mstconfig xml2bin Command](#).
 - b. Set the key's binary file in the firmware image using the `mstflint set_public_keys` command.
For further information, see [Setting a "Public Keys" Section in a Binary Image File](#).
2. If there is need to modify the definition for the forbidden_versions in a given firmware image then:
- a. Generate a binary file that contains the forbidden versions.
You can use the `mstconfig xml2bin` command to generate it according to the steps described in Step a above (Generate a binary file that contains 8 public keys).
An example for forbidden versions xml node:

```
<nv_forbidden_versions>  
<creation_time_day>18</creation_time_day>  
<creation_time_month>6</creation_time_month>  
<creation_time_year>7e2</creation_time_year>  
<creation_time_second>d</creation_time_second>  
<creation_time_minute>19</creation_time_minute>  
<creation_time_hour>12</creation_time_hour>  
<min_allowed_fw_version>0</min_allowed_fw_version>  
<forbidden_fw_version index="0">53:1f:0d06</forbidden_fw_version>  
<forbidden_fw_version index="1..31">0</forbidden_fw_version>  
</nv_forbidden_versions>
```

- b. Set the key's binary file in the firmware image using the `mstflint set_forbidden_versions` command.
For further information, see [Setting a "Forbidden Versions" Section in a Binary Image File](#).
3. Sign the firmware image with a private key.

⚠ Please notice that signing the image must be after setting the public keys, and the forbidden versions. For further information, see [Secure Firmware Update](#).

- a. Run the `mstflint sign` command.

⚠ To sign with a 2048 bits private key only, make sure that the firmware image does not contain a 4096 bits key signature.
Run the `mstflint set_public_keys` command with a 4096 bits keys section filled with zeros and then sign with a 2048 bits private key.
For further information, see [Signing Binary Image Files](#).

⚠ To sign with 4096 bits private key only, run the `mstflint set_public_keys` command with a 2048 bits keys section filled with zeros and then sign with the a 4096 bits private key.
For further information, see [Signing Binary Image Files](#).

Burning/Querying a Component in mstflint

Burning a Component Firmware Image

Clock Synchronizer Images

The `mstflint` utility enables the user to burn the Clock Synchronizer firmware from a binary image.

```
# mstflint --device <41:00.0> --image <clock synchronizer image> burn
```

Where:

<code>-d --device</code>	41:00.0
<code>-i --image</code>	Specified component firmwared image file to use.

Querying the Component Firmware Image

Clock Synchronizer Images

To query the Clock Synchronizer image on a device, use the following command line:

```
# mstflint --device <41:00.0> --component_type sync_clock query_components
```

To query the Clock Synchronizer image in a file, use the following command line:

```
# mstflint --image <image file> query
```

mstfwreset - Loading Firmware on 5th Generation Devices Tool

`mstfwreset` tool enables the user to load updated firmware on a NIC/switch without having to reboot the machine. `mstfwreset` supports 5th Generation (Group II) HCAs and allows a smooth firmware upgrade.

Tool Requirements

- Access to device through BDF format
- Firmware supporting ISFU
 - Connect-IB: v10.10.3000 or above

- ConnectX-4: v12.0100.0000 or above
- ConnectX-4 Lx: v14.0100.0000 or above
- Device's firmware updated with latest mstflint burning tools (mstflint)
- Supported devices: Connect-IB / ConnectX-4 / ConnectX-4 Lx / ConnectX-5 / BlueField / ConnectX-6
- Supported OSs: FreeBSD, Linux

Query Command

```
mstfwreset -d <device> query
```

Reset Command

```
mstfwreset -d <device> reset-[y] [--level <0,3,4>] [--type <0..2>] [--sync <0,1>] [-s] [-m]
```

mstfwreset Synopsis

where:

-d --device <device>	Device to work with	-
-l --level <0..5>	Run reset with the specified reset-level	N / A f o r s w i t c h d e v i c e s .

-t --type <0,1>	Run reset with the specified reset-type	N / A f o r s w i t c h d e v i c e s .
-m --mst_flags MST_FLAGS	Provide mst flags to be used when invoking mst restart step. For example: --mst_flags="--with_fpga"	N / A f o r s w i t c h d e v i c e s .
-y --yes	Answer "yes" on prompt	-
--skip_driver -s	Skip driver start/stop stage (driver must be stopped manually)	N / A f o r s w i t c h d e v i c e s .
-v --version	Print tool version	-


-h --help	Show help message and exit	-
--skip_fsm_sync	Skip fsm syncing	-
q query	Query for reset level required to load new firmware	N / A f o r s w i t c h d e v i c e s .
r reset	Execute reset Level	-
reset_fsm_register	Reset the fsm sync register to idle state	-
--sync	Run reset with the specified reset-sync	N / A f o r s w i t c h d e v i c e s .

Reset Levels and Types

Reset levels and types depend on the extent of the changes introduced when updating the device's firmware. The tool will display the supported reset levels and types that will ensure the loading of the new firmware. Those reset levels and types are:

- Reset-levels:
 - 0: Driver, PCI link, network link will remain up ("live-Patch")
 - 3: Driver restart and PCI reset
 - 4: Warm Reboot
 - 5: Cold Reboot


- Reset-types (relevant only for reset-levels 3,4):
 - 0: Full chip reset
 - 1: Phy-less reset ("port-alive" - network link will remain up)

 Exact reset level and types needed to load new firmware may differ, as it depends on the difference between the running firmware and the firmware we are upgrading to.

mstfwreset for Multi-Host NICs

mstfwreset supports a Multi-Host setup. To reset the firmware for a device in a Multi-Host setup, you have to run the tool on all the hosts simultaneously when in legacy mode. The tool utilizes a synchronization mechanism supported by the firmware in order to synchronize between the different running instances of the tool on the hosts.

For debugging purposes, it is possible to avoid the synchronization by running the tool with the flag `--skip_fsm_sync`.

 When running mstfwreset on a Multi-Host setup, a time-out of 3 minutes is expected for all the hosts until they join the firmware reset process.

mstfwreset for Socket-Direct NICs

To reset the firmware on a socket-direct NIC, run the tool on all PCI devices related to the same NIC with function 0 simultaneously.

See the following example on a Linux OSs:

```
$ lspci -d 15b3:
08:00.0 Infiniband controller: Mellanox Technologies MT27800 Family [ConnectX-5]
08:00.1 Infiniband controller: Mellanox Technologies MT27800 Family [ConnectX-5]
0e:00.0 Infiniband controller: Mellanox Technologies MT27800 Family [ConnectX-5]
0e:00.1 Infiniband controller: Mellanox Technologies MT27800 Family [ConnectX-5]
* All PCI devices above are related to the same NIC

* Run mstfwreset on all the PCI devices with function 0 (08:00.0, 0e:00.0)
$ mstfwreset -d 08:00.0 reset -y &
$ mstfwreset -d 0e:00.0 reset -y &
```

mstfwreset for SmartNICs

To reset the firmware on a SmartNIC, run the tool simultaneously on the host and on the NIC's integrated Arm processor.

 Firmware reset will trigger the adapter card's reset which will reboot the Arm processor.

mstfwreset for Switch Devices

Running mstfwreset on a switch device is done in the same form as running mstfwreset on a NIC. The only difference is that there are no level, types or sync parameters.

Examples of mstfwreset Usage

To query device reset level after firmware update use the following command line:

```
# mstfwreset -d 41:00.0 query
```

Supported reset levels for loading firmware on device, 41:00.0

Example:

```
Reset-levels:
0: Driver, PCI link, network link will remain up ("live-Patch") -Not Supported
3: Driver restart and PCI reset -Supported (default)
4: Warm Reboot -Supported
5: Cold Reboot -Supported

Reset-types (relevant only for reset-levels 3,4):
0: Full chip reset -Supported (default)
1: Phy-less reset ("port-alive" - network link will remain up) -Not Supported

Reset-sync (relevant only for reset-level 3):
0: Tool is the owner -Supported (default)
1: Driver is the owner -Supported
In the new mlxfwreset for BF2 and BF3, sync1 is the default. In order to switch to sync0, provide --sync 0 manually.
```

To reset device in order to load new firmware, use the following command line:

```
# mstfwreset -d 41:00.0 reset
```

Example

```
3: Driver restart and PCI reset
Continue with reset?[y/N] y
-I- Stopping Driver -Done
-I- Sending Reset Command To Fw -Done
-I- Resetting PCI -Done
-I- Starting Driver -Done
-I- Restarting MST -Done
-I- FW was loaded successfully.
```



- When running the reset command without specifying a reset level the minimal reset level will be performed.
- When running the reset command without specifying a reset type the default reset type would be 0 (Full chip reset).

To reset a device with a specific reset level to load new firmware, use the following command line:

```
# mstfwreset -d 41:00.0 -l 4 reset
```

Example

```
Requested reset level for device, 41:00.0:
4: Warm Reboot
Continue with reset?[y/N] y
-I- Sending reboot command to machine
```

mstfwreset Limitations


The following are the limitations of mstfwreset:

- Executing a reset level that is lower than the minimal level (as shown in query command) will yield an error
- When burning firmware with mstfwreset at the end of the burn the following message is displayed:
-l- To load new FW run mstfwreset or reboot machine.
If this message is not displayed, a reboot is required to load a new firmware.
- On an old firmware, after a successful reset execution, attempting to query or reset again will yield an error as the load new firmware command was already sent to the firmware.
- In case mstfwreset exits with error after the “Stopping driver” step and before the “Starting driver” step, the driver will remain down. The user should start the driver manually in this case.
- The new mstfwreset sync capability (-sync) is available only if supported by the firmware and all the drivers on all the hosts. To check if this is supported, run the "query" command.
- mstfwreset for switch devices does not work over InfiniBand.

mstcongestion - Tool for Setting Congestion Mode and Action

mstcongestion is a tool used to configure device’s behavior in case of excessive ingress traffic where the ingress traffic is higher than the PCIe capability. The excessive traffic can either be dropped (drop action) or marked as CE (Congestion Encountered) in the IP header.

The tool can work in either aggressive mode where traffic is dropped/marked in an aggressive way, or in dynamic mode where the drop/mark is more relaxed.

 mstcongestion is not supported in ESXi 7.0.

 mstcongestion is supported on ConnectX-4 Lx onwards Multi-Host devices only.

Tool Requirements

- Firmware version ConnectX-4 Lx: 14.23.1020 or later

mstcongestion Synopsis

```
# mstcongestion [option] [-d|--device <PCI DEVICE>] [--mode <MODE>] [--action <ACTION>] [-q|--query] [-h|--help] [-v|--version]
```

where:

-d --device <PCI DEVICE>	NVIDIA PCI device address
--mode <MODE>	Set Mode, options are: [aggressive dynamic]

--action <ACTION>	Set Action, options are: [disabled drop mark] Note: The “mark” option is available only if the driver supports such capability.
-q --query	Query congestion
-h --help	Show help message and exit
-v --version	Show version and exit


mstprivhost - NIC Configuration by the Host Restriction Tool

mstprivhost enables the user to restrict the hosts from configuring the NIC. Meaning, only the Arm side will have the privilege to configure the NIC.

 This utility is only supported in BlueField devices.

mstprivhost Synopsis

```
mstprivhost [-h] [-v] --device DEVICE {r,restrict,p,privilege,q,query}
```

- 
- New configurations take effect immediately.
 - A restricted host is not allowed to be port_owner, to own the tracer and to read physical port counters.
 - Without performing privilege, the host can be re-restricted from the Arm side with new disable parameters [disable_rshim, disable_tracer, etc].

where:

-h, --help	Shows this help message and exit
-v, --version	Shows program's version number and exit
--device DEVICE, -d DEVICE	Device to work with.
--disable_rshim	When TRUE, the host does not have an RSHIM function to access the embedded CPU registers
--disable_tracer	When TRUE, the host will not be allowed to own the Tracer
--disable_counter_rd	When TRUE, the host will not be allowed to read Physical port counters
--disable_port_owner	When TRUE, the host will not be allowed to be Port Owner
r,restrict	Set host 1 (Arm) privileged, host 0 (x86_64) restricted.
p,privilege	Set host 1 (Arm) privileged, host 0 (x86_64) privileged (back to default).
q,query	Query current host configuration
-f, --full	Run with query command for high verbosity level - valid from embedded ARM CPU only.

Example of mstprivhost:

- Enabling Full Host Restriction (Embedded ARM CPU Only):

```
mstprivhost -d 03:00.0 r --disable_rshim --disable_tracer --disable_counter_rd --disable_port_owner
```

- Disabling Host Restriction (Embedded ARM CPU Only): :

```
mstprivhost -d 03:00.0 p
```

- Query the status of the host\hosts (the full flag valid for embedded ARM CPU Only):

```
mstprivhost -d 03:00.0 q --full
Host configurations
-----
host index      : 0          1          2          3
level          : PRIVILEGED  PRIVILEGED  PRIVILEGED  PRIVILEGED

Port functions status:
-----
disable_rshim   : FALSE     FALSE     FALSE     FALSE
disable_tracer  : FALSE     FALSE     FALSE     FALSE
disable_port_owner : FALSE     FALSE     FALSE     FALSE
disable_counter_rd : FALSE     FALSE     FALSE     FALSE
```

Debug Utilities

This section contains:

- [mstfwtrace Utility](#)
- [mstregdump Utility](#)
- [mstreg Utility](#)
- [mstlink Utility](#)
- [mstresourcedump Utility](#)
- [mstresourceparse Utility](#)

mstfwtrace Utility

The mstfwtrace utility extracts and prints trace messages generated by the firmware running on 5th generation (Group II) devices iRISCs.

These trace messages inform developers of software drivers about internal status, events, critical errors, etc. Trace messages generated by iRISCs are stored in the trace buffer. The trace buffer is located in host memory.



When using secure firmware, the user needs to validate that tracer's value is set to "1" to enable it.

- For MLNX_OFED up to 4.6: /sys/kernel/debug/tracing/events/mlx5/fw_tracer/enable
- For MLNX_OFED 4.6 and above: /sys/kernel/debug/tracing/events/mlx5/mlx5_fw/enable
- For linux kernel tracer (when MLNX_OFED is not available): /sys/kernel/debug/tracing/events/enable

By default, the firmware does not print trace messages. Please contact your FAE for more details on how to enable firmware tracing.

mstfwtrace Usage

1. Enter the following command:

```
# mstfwtrace [options...]
```

where:

-h --help	Print this help message and exit
-d --device	PCI device (BDF)
--tracer_mode	Tracer mode [MEM]
--real_ts	Print real timestamps in [hh:mm:ss:nsec] format
-i --irisc	iRISC name (use the "all" option after -i)
-s --stream	Run in streaming mode
-m --mask	Trace class mask, use "+" to enable multiple classes or use integer format, e.g: -m class1+class2+... or 0xff00ff00
-l --level	Trace level
-v --version	Print tool's version and exit
--ignore_old_events	Ignore collecting old events

Device-Specific Information:

- Connect-IB, ConnectX-4, ConnectX-4 Lx, ConnectX-5, ConnectX-6, ConnectX-6 Dx, ConnectX-6 Lx, ConnectX-7, BlueField, BlueField-2, Switch-IB, Switch-IB 2, Quantum, Quantum-2, Spectrum, Spectrum-2, Spectrum-3:
iRISC names: [all]
- Trace classes:
DEBUG_INIT, INIT, ICM, ICM_FREE_LIST, HOST_MNG, CMD_IF, PHY_IB, PHY_RX_ADAP, PHY_EYE_OPN, PHY_COMMON, PHY_MANAGER, PWR, FLR, ICM_ACCESS, MAD, RXT_CHECKS, I2C, TRANSPORT, FW_LL, RX_ERRORS, CMD_DRIVER, PROFILING, MANAGEMENT, FLASH, STEERING, IFARM, ICMD, PCI, DC_CLEANUP, PHY_ETH, VIRT

Example:

```
# mstfwtrace -d 41:00.1 -i all
Read old events:
[0x28ed5b22771a5] 0 [0xa1] IRON populate_local_dbase end entry_ix=0x0,
<state_31_28,cmd_ix_20_16,gvmi_15_0>=0x10000001, cause_been_set=0
[0x28ed5b227b160] 0 [0xa5] access_reg: register_id=0x9043
[0x28ed5b228297f] 0 [0xa1] IRON populate_local_dbase end entry_ix=0x0,
<state_31_28,cmd_ix_20_16,gvmi_15_0>=0x10000001, cause_been_set=0
[0x28ed5b2284365] 0 [0xa3] access_reg: register_id=0x9043
Read new events:
[0x28edf8a8a3788] 0 [0xa1] IRON populate_local_dbase end entry_ix=0x0,
<state_31_28,cmd_ix_20_16,gvmi_15_0>=0x10000001, cause_been_set=0
[0x28edf8a8a5bc0] 0 [0xa5] access_reg: register_id=0x9043
[0x28edf92960db4] 0 [0xa1] IRON populate_local_dbase end entry_ix=0x0,
<state_31_28,cmd_ix_20_16,gvmi_15_0>=0x10000001, cause_been_set=0
[0x28edf929623dd] 0 [0xa2] access_reg: register_id=0x9043
```

mstregdump Utility

The mstdump utility dumps device internal configuration registers. The dump file is used by NVIDIA Support for hardware troubleshooting purposes. It can be applied on all NVIDIA devices.

mstdregump Usage

To run mstregdump:

```
mstregdump [-full] <device> [i2c-slave] [-v[ersion] [-h[elp]]]
```

where

-full	Dump an expanded list of addresses. Note: Use this flag carefully. Non-safe addresses might be read.
<device>	The device name
-v --version	Display version info
-h --help	Print this help message
i2c_slave	I2C slave [0-127]

Example:


```
[root@mymach]# mstregdump 41:00.0 > mt4099.dmp
```

This dumps the internal configuration data of the device into the mt4099.dmp file.

mstreg Utility

The mstreg utility allows users to obtain information regarding supported access registers, such as their fields and attributes. It also allows getting access to register data from firmware and setting access register data on firmware.

Registers can be get/set in unknown (RAW) mode by providing register ID and length.

 Unknown (RAW) mode is risky as no checks are performed, please consult with NVIDIA support before using it.

mstreg Usage

mst driver must be started prior to running mstreg tool.

Some access registers depend on setup configuration such as link up/down. Invalid setup may cause failures.

To run mstreg, use the following line:

```
mstreg [options]
```

where:

-h --help	Displays help message.
-v --version	Displays version info.
-d --device <device>	Performs operation for a specified mst device.
-a --adb_file <adb_file>	An external ADB file
--reg_name <reg_name>	Known access register name
--reg_id <reg_ID>	Access register ID
--reg_len <reg_length>	Access register layout length (bytes)
-i --indexes <idxs_vals>	Register indexes
-g --get	Register access GET
-s --set <reg_dataStr>	Register access SET
--show_reg <reg_name>	Prints the fields of a given reg access (must have reg_name)
--show_regs	Prints all available access registers
--yes	Non-interactive mode, answer yes to all questions

Examples:

Show all available access registers:

```
mstreg -d 41:00.0 --show_regs
Available Access Registers
-----
CWTP
CWTPM
MCIA
MLCR
MPCNT
MPEIN
NCFG
PAOS
PDDR
PMDR
PMLP
PPAOS
PPCNT
PPLM
PPLR
PPRT
PPTT
PTAS
PTYS
ROCE_ACCL
SBCM
SBDCR
SBPM
SBPR
SBSR
SLRG
SLRP
SLTP
....
```

Query a single access register (PAOS):

```
mstreg -d 41:00.0 --show_reg PAOS
Field Name | Address (Bytes) | Offset (Bits) | Size (Bits) | Access
-----
oper_status | 0x00000000 | 0 | 4 | RO
admin_status | 0x00000000 | 8 | 4 | RW
local_port | 0x00000000 | 16 | 8 | INDEX
swid | 0x00000000 | 24 | 8 | INDEX
e | 0x00000004 | 0 | 2 | RW
ee | 0x00000004 | 30 | 1 | WO
ase | 0x00000004 | 31 | 1 | WO
-----
```


Note: There might be indexes in access register fields that must be provided when setting or getting data.

Get access register data (PAOS with indexes: local_port 1, swid 0):

```
mstreg -d 41:00.0 --reg_name PAOS --get --indexes "local_port=0x1,swid=0x0"
Sending access register...

Field Name      | Data
=====
oper_status     | 0x00000001
admin_status    | 0x00000001
local_port      | 0x00000001
swid            | 0x00000000
e               | 0x00000000
ee              | 0x00000000
ase             | 0x00000000
=====
```

Set access register data (PAOS with indexes: local_port 1 swid 0x0 and data: e 1):

```
mstreg -d 41:00.0 --reg_name PAOS --indexes "local_port=0x1,swid=0x0" --yes --set "e=0x1"
You are about to send access register: PAOS with the following data:
Field Name      | Data
=====
oper_status     | 0x00000001
admin_status    | 0x00000001
local_port      | 0x00000001
swid            | 0x00000000
e               | 0x00000001
ee              | 0x00000000
ase             | 0x00000000
=====

Do you want to continue ? (y/n) [n] : y
Sending access register...
```

Get access register data (PAOS (0x5006) in unknown mode (RAW) with indexes: local_port=0x1 swid=0x0):

```
mstreg -d 41:00.0 --reg_id 0x5006 --reg_len 0x10 --indexes "0x0.16:8=0x1,0x0.24:8=0x0" --get
Sending access register...

Address         | Data
=====
0x00000000     | 0x00010101
0x00000004     | 0x00000000
0x00000008     | 0x00000000
0x0000000c     | 0x00000000
=====
```

Set access register data (PAOS in unknown mode (RAW) with indexes: local_port=0x1 swid=0x0 and data e 1):

```
mstreg -d 41:00.0 --reg_id 0x5006 --reg_len 0x10 --indexes "0x0.16:8=0x1,0x0.24:8=0x0" --yes --set "0x4.0:2=0x1"
You are about to send access register id: 0x5006 with the following data:
Address         | Data
=====
0x00000000     | 0x00010101
0x00000004     | 0x00000001
0x00000008     | 0x00000000
0x0000000c     | 0x00000000
=====

Do you want to continue ? (y/n) [n] : y
Sending access register...
```

mstlink Utility

The mstlink tool is used to check and debug link status and related issues. The tool can be used on different links and cables (passive, active, transceiver, and backplane).



- mstlink is intended for advanced users with appropriate technical background.
- When using mstlink to disable the port state ("--port_state dn" flag) on a NIC connected through a Socket-Direct connection, the port must be disabled on both mst devices representing the physical port.
- When running mstlink to show SLTP for 16nm technology, the "--advanced" flag should be added to the run command.

mstlink Usage

To run mstlink:

```
mstlink [OPTIONS]
```

where:

Options:

-h --help	Display help message.
-v --version	Display version info.
-d --device <device>	Perform operation for a specified mst device
-p --port <port_number>	Port Number
--port_type <port_type>	Port Type [NETWORK(Default)/PCIE/OOB]
--depth <depth>	Depth level of the DUT of some hierarchy (valid for PCIe port type only)
--pcie_index <pcie_index>	PCIe index number (Internal domain index) (valid for PCIe port type only)
--node <node>	The node within each depth (valid for PCIe port type only)
--json	Print the output in json format

Queries:

-m --show_module	Show Module Info
-c --show_counters	Show Physical Counters and BER Info
-e --show_eye	Show Eye Opening Info
--show_fec	Show FEC Capabilities
--show_serdes_tx	Show Transmitter Info
--show_tx_group_map <group_num>	Display all label ports mapped to group <group_num> (for NVIDIA Spectrum-2 and NVIDIA Quantum devices).
--show_device	General Device Info
--show_ber_monitor	Show BER Monitor Info
--show_external_phy	Show External PHY Info

Commands:

-a --port_state <port_state>	Configure Port State [UP(up)/DN(down)/TG(toggle)]	
-s --speeds <speeds>	Configure Speeds [speed1,speed2,...]	
	--link_mode_force	Configure Link Mode Force (Disable AN)
-l --loopback <loopback>	Configure Loopback Mode [NO(no loopback)/RM(phy remote Rx-to-Tx loopback)/PH(internal phy Tx-to-Rx loopback)/EX(external loopback connector needed)/LL(link layer local loopback)]	
-k --fec <fec_override>	Configure FEC [AU(Auto)/NF(No-FEC)/FC(FireCode FEC)/RS(RS-FEC)/LL(LL-RS-FEC)/DF-RS(Interleaved_RS-FEC)/DF-LL(Interleaved_LL_RS-FEC)]	
	--fec_speed <fec_speed>	Speed to Configure FEC [100G/50G/25G/...] (Default is Active Speed)
--serdes_tx <params>	Configure Transmitter Parameters [polarity,ob_tap0,...]	
	--serdes_tx_lane <transmitter_lane>	Transmitter Lane to Set (Optional - Default All Lanes)
	--database	Save Transmitter Configuration for Current Speed Permanently (Optional)
	--tx_params_override	Set the parameters according to Data Base only, otherwise it will be set according to the best possible configuration chosen by the system (e.g. KR-startup) (Optional)
--tx_group_map <group_num>	Map ports to group <group_num> (for NVIDIA Spectrum-2 and NVIDIA Quantum devices)	
	--ports <ports>	Ports to be mapped [1,2,3,4..]
--test_mode <prbs_mode>	Physical Test Mode Configuration [EN(enable)/DS(disable)/TU(perform tuning)]	
	--rx_prbs <rx_prbs_mode>	RX PRBS Mode [PRBS31(Default)/PRBS7/...] (Optional - Default PRBS31)
	--tx_prbs <tx_prbs_mode>	TX PRBS Mode [PRBS31(Default)/PRBS7/...] (Optional - Default PRBS31)
	--rx_rate <rx_lane_rate>	RX Lane Rate [EDR(Default)/25G/10G/...] (Optional - Default 25G)
	--tx_rate <tx_lane_rate>	TX Lane Rate [EDR(Default)/25G/10G/...] (Optional - Default 25G)
	--invert_tx_polarity	PRBS TX polarity inversion (Optional - Default No Inversion)
	--invert_rx_polarity	PRBS RX polarity inversion (Optional - Default No Inversion)
	--lanes	PRBS lanes to set (one or more lane separated by comma)[0,1,2,...] Optional: Default all lanes
-b --ber_collect <csv_file>	Port Extended Information Collection [CSV File]	
--amber_collect <csv_file>	AmBER Port Extended Information Collection For 16nm Products and Later [CSV File]	
--ber_limit <limit_criteria>	BER Limit Criteria [Nominal(Default)/Corner/Drift] (Optional - Default Nominal)	
	--iteration <iteration>	Iteration Number of BER Collection
--pc	Clear Counters	
--set_external_phy	Set External PHY Note: The flag is supported in NVIDIA Spectrum switch systems only.	

	--twisted_pair_force_mode <twisted_pair_force_mode>	Twisted Pair Force Mode [MA(Master)/SL(Slave)]
--cable		Perform operations on the cables
	--dump	Dump cable pages in raw format
	--ddm	Get cable Digital Diagnostic Monitoring information
	--read	Perform read operation from specific page
	--length <length>	Length of data to read in bytes (Optional - Default 1 byte)
	--page <pageNum>	Specific page number to read/write
	--offset <offset>	Specific page offset to read/write
	--write <bytes>	Perform write operation with specific data (list of bytes, separated by ',')
	--prbs_select <side>	Module PRBS test mode side selector [MEDIA, HOST]
	--prbs_mode <cmd>	Perform PRBS test mode on the Module [EN(Enable),DS(Disable)]
	--generator_pattern <pattern>	Set PRBS generator pattern [PRBS31(default),PRBS23,PRBS7,PRBS11,PRBS9,PRBS13,SPR,SSPRQ]
	--swap_generator	Enable PAM4 MSB <-> LSB generator swapping (Optional)
	--invert_generator	Enable PRBS generator inversion (Optional)
	--generator_lanes <lanes>	PRBS generator lanes to set (one or more lane separated by comma)[0,1,2,3,4,5,6,7] (Optional - Default all lanes)
	--checker_pattern <pattern>	Set PRBS checker pattern [PRBS31(default),PRBS23,PRBS7,PRBS11,PRBS9,PRBS13,SPR,SSPRQ]
	--swap_checker	Enable PAM4 MSB <-> LSB checker swapping (Optional)
	--invert_checker	Enable PRBS checker inversion (Optional)
	--checker_lanes <lanes>	PRBS checker lanes to set (one or more lane separated by comma)[0,1,2,3,4,5,6,7] (Optional - Default all lanes)
	--lane_rate <rate>	Set PRBS checker and generator lane rate [HDR(50G)(default),1.25G,SDR(2.5G),10.3125G,FDR(14G),EDR(25G),NDR(100G)]
	--show_diagnostic_info	Show PRBS diagnostic counters information
	--clear_diagnostic_info	Clear PRBS diagnostic counters
	--control_parameters	Show Module Control Parameters
	--tx_equalization <value>	Set Module Tx Input Equalization in dB [NE(No Equalization),1,2,3,4,5,6,7,8,9,10,11,12]

	<code>--rx_emphasis <value></code>	Set Module RX Output Emphasis in dB. for CMIS, pre-emphasis value will be set [NE(No Equalization),0.5,1,1.5,2,2.5,3,3.5,4,5,6,7]
	<code>--rx_post_emphasis <value></code>	Set Module Rx Post Emphasis in dB [NE(No Equalization),1,2,3,4,5,6,7]
	<code>--rx_amplitude <value></code>	Set Module Rx Output Amplitude [0(100-400mV),1(300-600mV),2(400-800mV),3(600-1200 mV)]
<code>--margin</code>		Read the SerDes eye margins per lane
	<code>--measure_time <time></code>	Measure time in seconds for single eye [10, 30, 60, 90, 120, 240, 480, 600 and 900] (Optional - Default 60 for PCIe and 30 for Network ports)
	<code>--eye_select <eye_sel></code>	Eye selection for PAM4 [UP, MID, DOWN, ALL] (Default ALL)
	<code>--lane <lane_index></code>	Run eye for specific lane index (Default all lanes)
<code>--rx_error_injection</code>		Enable the RX link deterioration
	<code>--mixer_offset0 <value></code>	Fine change to the center of the eye [0x0 to 0x7ff]
	<code>--mixer_offset1 <value></code>	Coarse change to the center of the eye [0x0 to 0x3ff]
	<code>--show_mixers_offset</code>	Show mixer offset 0 and mixer offset 1
<code>--rx_fec_histogram</code>		Provide histogram of FEC errors. The result is divided to bins. Each bin is holding different number of errored bit within FEC protected block
	<code>--show_histogram</code>	Show FEC errors histogram
	<code>--clear_histogram</code>	Clears FEC errors histograms

Examples:

Get info of <device>, <port_number>:

```
mstlink -d <device> -p <port_number>
```

Get info of <device>, <port_number> and BER Counters:

```
mstlink -d <device> -p <port_number> -c
```

Get info of <device>, <port_number> and Transmitter Parameters:

```
mstlink -d <device> -p <port_number> --show_serdes_tx
```

Configure Port State:

```
mstlink -d <device> -p <port_number> --port_state UP
```

Configure Port Speeds:

```
mstlink -d <device> -p <port_number> --speeds 25G,50G,100G
```

Configure FEC:

```
mstlink -d <device> -p <port_number> --fec RS
```

Configure Port for Physical Test Mode:

```
mstlink -d <device> -p <port_number> --test_mode EN (--rx_prbs PRBS31 --rx_rate 25G --tx_prbs PRBS7 --tx_rate 10G --invert_rx_polarity --invert_tx_polarity)
```

Perform PRBS Tuning:

```
mstlink -d <device> -p <port_number> --test_mode TU
```



RX and TX lane rates for new devices include the PAM4 speeds (50G_1X and 100G_2X).

eg: mstlink -d <device> --test_mode EN --rx_rate [normal speeds | 50G_1X | 100G_2X] --tx_rate [normal speeds | 50G_1X | 100G_2X]



The PRBS pattern configured in PAM4 rates is PRBSQ.

Cable operations:

```
mstlink -d <device> --cable [Options]
```

Dump cable EEPROM pages:

```
mstlink -d <device> --cable --dump
```

Get cable DDM information:

```
mstlink -d <device> --cable --ddm
```

Read from cable:

```
mstlink -d <device> --cable --read --page <page number> --offset <bytes offset> --length <number of bytes>
```

Write to cable:

```
mstlink -d <device> --cable --write <bytes separated by comma> --page <page number> --offset <bytes offset>
```

Configure Transmitter Parameters (on lane, to database):

```
mstlink -d <device> -p <port_number> --serdes_tx <polarity>,<ob_tap0>,<ob_tap1>,<ob_tap2>,<ob_bias>,<ob_preemp_mode>,<ob_reg>,<ob_leva> (--serdes_tx_lane <lane number>) (--database)
```

Configure Transmitter Parameters for 16nm devices:

```
mstlink -d <device> -p <port_number> --serdes_tx <pre_2_tap>,<pre_tap>,<main_tap>,<post_tap>,<ob_m2lp>,<ob_amp>
```

Getting PCIe links info:

```
mstlink -d /dev/mst/mt41682_pciconf0 --port_type PCIE --show_links
Valid PCIe Links
-----
: depth, pcie_index, node, port
Link 1 : 3, 0, 0, 60
Link 2 : 3, 0, 1, 61
Link 3 : 3, 0, 2, 62
..
```

To query information for a specific link, the depth, pcie_index and node for the link must be specified:

```
mstlink -d /dev/mst/mt41682_pciconf0 --port_type PCIE --depth 3 --pcie_index 0 --node 1 --show_serdes_tx --show_eye
PCIe Operational (Enabled) Info
-----
Depth, pcie index, node : 3, 0, 1
Link Speed Active (Enabled) : 8G-Gen 3 (16G-Gen 4)
Link Width Active (Enabled) : 2X (16X)

EYE Opening Info (PCIe)
-----
Physical Grade : 84, 84
Height Eye Opening [mV] : 1194, 1194
Phase Eye Opening [psec] : 84, 84

Serdes Tuning Transmitter Info (PCIe)
-----
Serdes TX parameters      : Pol ,tap0 ,tap1 ,tap2 ,bias ,preemp_mode ,reg ,leva
Lane 0                    : 0 ,21 ,92 ,7 ,15 ,1 ,10 ,9
Lane 1                    : 1 ,21 ,92 ,7 ,15 ,1 ,10 ,9
Lane 2                    : 0 ,21 ,92 ,7 ,15 ,1 ,10 ,9
Lane 3                    : 1 ,21 ,92 ,7 ,15 ,1 ,10 ,9
Lane 4                    : 0 ,21 ,92 ,7 ,15 ,1 ,10 ,9
Lane 5                    : 1 ,21 ,92 ,7 ,15 ,1 ,10 ,9
Lane 6                    : 0 ,21 ,92 ,7 ,15 ,1 ,10 ,9
Lane 7                    : 1 ,21 ,92 ,7 ,15 ,1 ,10 ,9
```

To print the output in JSON format:

```
mstlink -d <device> --show_module --json
```

To show ports group map (for NVIDIA Quantum and NVIDIA Spectrum-2):

```
mstlink -d<device> --show_tx_group_map 0
```

To assign ports to a specific group on NVIDIA Quantum and NVIDIA Spectrum-2:

```
mstlink -d <device> --tx_group_map 1 -ports 1,2,3,5,4,8,7,8,9,10,11
```

To show histogram of FEC errors:


```
mstlink -d /dev/mst/mt4125_pciconf0 --rx_fec_histogram --show_histogram
```

To clear histogram:

```
mstlink -d /dev/mst/mt4125_pciconf0 --rx_fec_histogram --clear_histogram
```

Margin Scan Tool

The margin scan tool is used for scanning PCIe [Gen4 speed] or Network ports [EDR\25G or HDR\PAM4 speeds].

 If the margin scan fails with this message (Eye scan not completed), perform a reboot and run the scan again.

To enable the margin scan with measure time 10 seconds:

```
mstlink -d <device> --port_type PCIE -margin -measure_time 10
```

To enable the margin scan for Multi-host or Socket Direct systems through:

- depth, pcie_index and node:

```
mstlink -d <device> --port_type PCIE -depth 0 -pcie_index 1 -node 0 -margin -measure_time 30
```

- The local port (it can be shown by the `-show_links` command):

```
mstlink -d <device> --port_type PCIE -port 1 -margin -measure_time 10
```


RX Error Injection

Allows modifying the Eye Center capability by changing the `mixer_offset0` (fine change) and `mixer_offset1` (coarse change) flags for 28nm products to produce RX errors.

Flags Usage

- To change the mixers values:

```
mstlink -d /dev/mst/mt4117_pciconf0 --rx_error_injection --mixer_offset0 0x200 --mixer_offset1 0x305
```

 **Modifying `mixer_offset0` and `mixer_offset1` flags can change the Eye Center and might cause link degradation.**

- To query the mixers values:

```
mstlink -d /dev/mst/mt4117_pciconf0 --rx_err r_injection --show_mixers_offset
```

Rx-to-Tx Loopback Mode Activation

This capability enables Rx-to-Tx remote loopback mode.

Prerequisite

- The port should be disabled and configured with Force mode [Disable the Auto-negotiation]
- Remote loopback is supported for 25G/50G per lane speed
- If the NIC has 2 ports, both ports should be configured with the same speed

To enable Rx-to-Tx remote loopback mode:

1. Disable the port.

```
mstlink -d /dev/mst/mt4123_pciconf0 -port_state DN
```

2. Configure the link mode to Force [Disable the Auto-negotiation].

```
mstlink -d /dev/mst/mt4123_pciconf0 --speeds 100G --link_mode_force
```

3. Configure loopback with remote loopback [RM].

```
mstlink -d /dev/mst/mt4123_pciconf0 -loopback RM
```

4. Enable the port.

```
mstlink -d /dev/mst/mt4123_pciconf0 -port_state UP
```

To return to the normal link operation:

1. Disable the port.

```
mstlink -d /dev/mst/mt4123_pciconf0 -port_state DN
```

2. Clear the loopback configuration using the "NO loopback" option.

```
mstlink -d /dev/mst/mt4123_pciconf0 -loopback NO
```

3. Enable the port.

```
mstlink -d /dev/mst/mt4123_pciconf0 -port_state UP
```

Module PRBS Test Mode

The module PRBS test mode can be performed by using the new flags under the `--cable` command.

Notes

- This feature supports Active/Optical CMIS modules only.
- Either the media or host side can run with PRBS mode.
- To enable the PRBS test mode, the module should be plugged in and active.

Enabling\Disabling The Module PRBS Test Mode

To enable the module PRBS test mode, the side of the module should be selected using the `--prbs_select` flag. After providing the `--cable` flag, either the HOST or the MEDIA side should be selected, so the `--prbs_mode <EN\DS>` can be used to enable or disable the PRBS test mode process.

E.g.: the following command will enable the PRBS test mode on the HOST side of the module:

```
mstlink -d /dev/mst/mt53104_pciconf0 --port 3 --cable --prbs_select HOST --prbs_mode EN
```

The command above will put the HOST side of the module in PRBS test mode with default Checker and Generator parameters.

The Checker and Generator parameters can be overridden while enabling the PRBS test mode according to their related flags in the help menu:

```
mstlink -d /dev/mst/mt53104_pciconf0 --port 3 --cable --prbs_select HOST --prbs_mode EN --checker_pattern PRBS13 --invert_checker --generator_pattern PRBS31 --swap_generator --lane_rate HDR
```

To disable the PRBS test mode, the following command can be executed:

```
mstlink -d /dev/mst/mt53104_pciconf0 --port 3 --cable --prbs_select HOST --prbs_mode DS
```

PRBS Diagnostic Counters Information

After performing the PRBS test mode, the module counters can be queried by using the following command:

```
mstlink -d /dev/mst/mt53104_pciconf0 --port 3 --cable --prbs_select HOST --show_diagnostic_info
```

The module PRBS test mode counters can be cleared by using the following command, which will clear the diagnostic counters on the HOST side only:

```
mstlink -d /dev/mst/mt53104_pciconf0 --port 3 --cable --prbs_select HOST --clear_diagnostic_info
```

Module Control Parameters

Some of the module parameters can be controlled by mstlink after providing the `--control_parameter` flag, which can be executed under the `--cable` flag.

The possible parameters can be controlled as follows:

- Reading and configuring Tx Equalization
- Reading and configuring Rx Emphasis (PreCursor & PostCursor)
- Reading and configuring Rx Amplitude

Notes

- To apply the changes, the link should be disabled first.
- After configuring a new parameter, the link should be raised again to allow the firmware to load the new configuration.
- Cable control parameters are valid for active\optical modules only.

Querying and Configuring The Module Control Parameters

To query the currently configured module control parameters, the `--control_parameters` flag can be used under the `--cable` flag as follows:

```
mstlink -d /dev/mst/mt53104_pciconf0 --cable --control_parameters
...
Module Control Parameters
-----
TX Equalization           : 1dB
RX Emphasis (pre)        : 2.5dB
RX Post Emphasis         : No Equalization
RX Amplitude              : 600-1200 mV (P-P)
```

To configure the module control parameters, the following command can be executed:

```
mstlink -d /dev/mst/mt53104_pciconf0 --cable --control_parameters --tx_equalization 2 --rx_amplitude 1
```

Tool Usage with NIC vs. Switch (-p Flag)

When using the mstlink tool with an adapter card, notice that the "label_port" -p flag should not be used. To address different ports, please use different MST devices.

For example:

To address port 1 when using ConnectX-4:

```
mstlink -d /dev/mst/mt4115_pciconf0
```

To address port 2, use:

```
mstlink -d /dev/mst/mt4115_pciconf0.1
```



- Any mstlink command for a switch should include the "-p" flag to address the specific port in the switch.
- When working with an adapter card, if an MTUSB is used for communication with the NVIDIA NIC, to address port 2, use `mstlink -d /dev/mst/mt4115_pciconf0 --gvmi_address <0xAddress>`.

Tool Usage on NVIDIA Quantum HDR Switch Systems with Split Ports

If the split port number is not provided by the ibdiagnet tool, to use mstlink on NVIDIA Quantum HDR based switch systems split ports, run:

```
mstlink -d lid-<LID> -p <formula>
```

Formula:

In case of 2X port:

- 1- $\text{port_num} = \text{round_down}[(\text{Iblinkinfo_port_num} + 1) * 0.5]$
- 2- if $(\text{Iblinkinfo_port_num} + 1) \text{ modulo } 2 = 1$ then append '/2' to port_num

In case of 4X port, use only item #1 above.

Example:

```
43 23[ ] == ( 2X 53.125 Gbps Active/ LinkUp) ==> mstlink -d lid-43 -p 12
43 24[ ] == ( 2X 53.125 Gbps Active/ LinkUp) ==> mstlink -d lid-43 -p 12/2
```

Tool Usage on NVIDIA Quantum-2 NDR Switch Systems

In NVIDIA Quantum-2 NDR switch systems, there are 32-OSFP cages (8x), where each one holds 2 (4x) ports instead of 1, and each port can be accessed by providing the cage number and the port in the cage - “Cage/Port”.

```
mstlink -d <mst device> -p <Cage>/<Port>
```

If the split profile is ready, it is possible to access the split ports by providing the number of split to the port flag, e.g.:

- To access the main port of 15/2:

```
mstlink -d <mst device> -p 15/2
```

- To access the split port of 15/2:

```
mstlink -d <mst device> -p 15/2/2
```

PCIe

Link Speed and Width

For PCIe link speed and width, use the following flag: `--port_type PCIE`.

```
PCIe Operational (Enabled) Info
-----
Depth, pcie index, node      : [Depth, pcie index, node]
Link Speed Active (Enabled) : [Freq - Gen]
Link Width Active (Enabled) : [Width]
```

PCIe Switch

When using NVIDIA ConnectX-5 and newer devices, the PCIe interface can be configured for a PCIe switch. When the PCIe switch is enabled, the depth, pcie_index and node parameters are needed in order to specify the PCIe port from which the requested information (such as counters or eye info) is gathered.

Parameters	Description
Depth	This parameter defines the number of layers from the Root Complex to the specific port. <ul style="list-style-type: none"> • For NVIDIA ConnectX adapter cards multi-host mode, the depth should be set to 0. • For NVIDIA BlueField/BlueField-2 JBoF, the depth should be set to 3.
Pcie_index	This parameter defines the root complex ID or host index. <ul style="list-style-type: none"> • For NVIDIA ConnectX adapter cards multi-host mode, the pcie_index is the host index (0-3). • For NVIDIA BlueField/BlueField-2 JBoF, the pcie_index is always 0.

Parameters	Description
Node	<p>This parameter defines the specific PCIe port.</p> <ul style="list-style-type: none"> For NVIDIA ConnectX adapter cards multi-host mode, the node is always 0 for each host_index. For NVIDIA BlueField JBoF mode, this parameter range is 0x0-0xF, which amounts for up to 16 possible ports for BlueField JBoF. For NVIDIA BlueField-2, this parameter's range is 0x0-0x7. <p>Note: For NVIDIA BlueField/BlueField-2 SmartNIC mode, the PCIe link information can only be gathered from the external host. The PCIe interface status cannot be retrieved from the Arm side. When retrieving the PCIe link information from the external host, there is no need to specify the depth, pcie_index and node.</p>

Example: NVIDIA BlueField JBoF Mode

```
# mstlink -d /dev/mst/mt41682_pciconf0 --port_type pcie --depth 3 --pcie_index 0 --node 4 -c

PCIe Operational (Enabled) Info
-----
Depth, pcie index, node      : 3, 0, 4
Link Speed Active (Enabled)  : 8G-Gen 3 (16G-Gen 4)
Link Width Active (Enabled)  : 2X (2X)

Management PCIe Timers Counters Info
-----
dl down                      : 0

Management PCIe Performance Counters Info
-----
RX Errors                    : 0
TX Errors                    : 0
CRC Error dllp               : 0
CRC Error tlp                : 0
```

Link Counters

For PCIe counters information, use the `--port_type PCIE -c` flag.

```
Management PCIe Timers Counters Info
-----
dl down                      : [link down counter]

Management PCIe Performance Counters Info
-----
RX Errors                    : [Rx Errors]
TX Errors                    : [Tx Errors]
CRC Error dllp               : [CRC Errors dllp]
CRC Error tlp                : [CRC Errors tlp]
```

- **RX Errors:** indicate the number of transitions to recovery required due to framing errors and CRC (dllp and tlp) errors.
- **TX Errors:** indicate the number of transitions to recovery required due to EIEOS and TS errors.
- **CRC Error dllp:** indicate CRC error in Data Link Layer Packets.
- **CRC Error tlp:** indicate CRC error in Transaction Layer Packet.

Example:

```
# mstlink -d /dev/mst/mt4123_pciconf0 --port_type PCIE -c

PCIe Operational (Enabled) Info
-----
Depth, pcie index, node      : 0, 0, 0
Link Speed Active (Enabled)  : 16G-Gen 4 (16G-Gen 4)
Link Width Active (Enabled)  : 16X (16X)

Management PCIe Timers Counters Info
-----
dl down                      : 3
```

```
Management PCIe Performance Counters Info
-----
RX Errors           : 0
TX Errors           : 16
CRC Error dllp      : 0
CRC Error tlp       : 0
```

Link Eye Opening and Grade

For PCIe link physical grade and eye opening information, use the `--port_type PCIE -e` flag.

```
EYE Opening Info (PCIe)
-----
Physical Grade : [Grade0, Grade1, Grade2, Grade3, Grade4, Grade5, Grade6, Grade7, Grade8, Grade9, Grade10, Grade11,
Grade12, Grade13, Grade14, Grade15]
Height Eye Opening [mV] : [Height0, Height1, Height2, Height3, Height4, Height5, Height6, Height7, Height8,
Height9, Height10, Height11, Height12, Height13, Height14, Height15]
Phase Eye Opening [psec] : [Phase0, Phase1, Phase2, Phase3, Phase4, Phase5, Phase6, Phase7, Phase8, Phase9,
Phase10, Phase11, Phase12, Phase13, Phase14, Phase15]
```

Example:

```
# mstlink -d /dev/mst/mt4123_pciconf0 --port_type PCIE -e
PCIe Operational (Enabled) Info
-----
Depth, pcie index, node      : 0, 0, 0
Link Speed Active (Enabled)  : 16G-Gen 4 (16G-Gen 4)
Link Width Active (Enabled)  : 16X (16X)
EYE Opening Info (PCIe)
-----
Physical Grade                : 57279, 56340, 59340, 61824, 55140, 60501, 61530, 57392, 61573, 58930, 62752,
60421, 57188, 59796, 60066, 60847
Height Eye Opening [mV]      : 292, 288, 314, 325, 278, 310, 319, 299, 316, 318, 343,
323, 310, 311, 335, 318
Phase Eye Opening [psec]     : 30, 30, 30, 30, 30, 30, 30, 30, 30, 30, 28, 28,
28, 28, 30, 28, 30
```

Pass/Fail Criteria

SLRED (ConnectX-6/ConnectX-6 Dx/ConnectX-6 Lx)

```
mstlink -d [device] --port_type PCIE --margin
```

Gen3

Gen3	
Eye Grade	Figure of Merit (FOM)
0 < Eye Grade < 700	FAIL
700 < Eye Grade < 2300	Gray area
2300 < Eye Grade	PASS

Gen4

Gen4	
Eye Margin	FOM
0 < Eye Grade < 150	FAIL
150 < Eye Grade < 400	Gray area
400 < Eye Grade	PASS

PCIE Error Injection

This test feature allows errors injection over the PCI links. It is used to verify that the system can handle the PCIe errors, which rarely occur in regular usage.

The ConnectX-7 device includes testability features that can be configured to act as an error injection ‘exerciser’ in order to test other components in the system. This is supported when the ConnectX-7 is used as a PCIe switch.



- This is a PCIe related feature that should be run over PCIe links only (`--port_type PCIE`) with specific depth, PCIe index and node (DPN).
- If the DPN is not provided, the tool will take the default values - 0,0 and 0, respectively.
- The mapping between the BDF and its DPN can be found by executing the `show_links` command (see example below).

Error Types

ID	Error Type	Description	Unit	Additional Parameters (<code>--errors_parameters</code>)	Advanced Error Reporting Flag Set by This Error
0	ABORT	Cancels the current pending error, if exists.	NA	NA	NA
1	BAD_DLLP_LCRC	Flips a bit in the LCRC of the next “ error_duration ” DLLPs that are transmitted through the port.	Packets	NA	Bad DLLP Status
2	BAD_TLP_LCRC	Flips a bit in the LCRC of the next “ error_duration ” TLPs that are transmitted through the port. The packets are VDM TLPs that are sent by the port to the destination BDF - “ dbdf ”.	Packets	NA	Bad TLP Status
3	BAD_TLP_ECRC	Flips a bit in the ECRC of the next “ error_duration ” TLPs that are transmitted through the port. The packets are VDM TLPs that are sent by the port to the destination BDF - “ dbdf ”.	Packets	NA	ECRC Error Status
4	ERR_MSG	Sends an error signaling message to the RC.	Packets	Parameter 0: message type 0 - Correctable 1 - Nonfatal 2 - Fatal	ERR_COR Received / Non-Fatal Error Messages Received / Fatal Error Messages Received

5	MALFORMED_TLP	Sends an “ error_duration ” PM_ACTIVE_STATE_NACK message to the destination BDF - “ dbdf ” with TC=1 instead of 0.	Packets	NA	Malformed TLP Status
6	POISONED_TLP	Sends an “ error_duration ” VDMs with data to the destination BDF - “ dbdf ” with EP = 1.	Packets	NA	Poisoned TLP Received
7	UNEXPECTED_CPL	Sends “ error_duration ” completions to the destination BDF - “ dbdf ” with 0xff tag.	Packets	NA	Unexpected Completion Status
8	ACS_VIOLATION	Sends “ error_duration ” VDMs to the destination BDF - “ dbdf ” with source_bdf=0.	Packets	NA	ACS Violation Status
100	SURPRISE_LINK_DOWN	Sets a port state to DETECT.	NA	NA	Surprise Down Error Status
101	RECEIVER_ERROR	Sends a clock instead of data for “ error_duration ” usecs. A value of 0 in ‘ error_duration ’ means that this error must be toggled by the firmware as fast as possible.	uSec	NA	Receiver Error Status

PCIe Error Injection Inputs

The following values should be provided in the error injection command line. Some values may be optional according to the error type.

Input	Command Line Flag	Description	Obliga tory	Defa ult
Error Type	-- error_type	Error type according to the table above.	Yes	-
Error Duration	-- duration	The minimal number of packets with this error that will be sent, or the minimal amount of time that this error state would be applied.	No	1
Injection Delay	-- injection_delay	Delay in microseconds before the error is applied. This allows time for the completion to return to the tool caller correctly. A higher value can be used to allow the system to get to a lower power state.	No	0
Destination BDF	--dbdf	Destination BDF. Relevant for some of the errors that require packet generation. See error table above.	No	0:00.0
Additional Parameters	-- errors_parameters	Additional parameters according to the error type. See error table above.	No	0

mstlink will trigger the firmware to start the error injection process by providing the --pcie_error_injection flag with the requested configuration parameters.

Note that the command returns immediately, but the error injection can take longer to complete (according to the error duration and injection delay inputs).

When the tool is run without the parameters above, it will query the error injection state - Whether it is ready to start a new error injection, or it is in the middle of the previous injection.

Usage Example

Start the process by performing error injection with error type UNEXPECTED_CPL.

This example shows how to start the error injection by sending 5 unexpected completion packets. The packets (of error type UNEXPECTED_CPL (id 7)) are directed from BDF 05:00.0 to BDF 06:00.0 after 500µs of sending the command in the following environment:

PCIe Component	BDF
Root port	00:01.0
(exerciser) PCIe Switch Upstream port	01:00.0
(exerciser) PCIe Switch Downstream port	05:00.0
Endpoint	06:00.0

To get the related depth, pcie_index and node for the specific BDF 05:00.0, the `show_links` command should be executed as follows:

```
mstlink -d /dev/mst/mt4129_pciconf0 --port_type PCIE --show_links

Valid PCIe Links
-----
Legend          : depth ,pcie_index ,node ,port ,bdf
Link 1          : 0      ,0      ,0   ,0   ,01:00.0
Link 2          : 3      ,0      ,0   ,60  ,05:00.0
```

In this case, the depth, pcie_index, and node flags for the downstream port should be 3, 0, and 0, respectively.

Then, the following command can be executed to start the process:

```
mstlink -d /dev/mst/mt4129_pciconf0 --port_type PCIE --depth 3 --pcie_index 0 --node 0 --pcie_error_injection --
error_type UNEXPECTED_CPL --error_duration 5 --dbdf 06:00.0 --injection_delay 500

PCIe Operational (Enabled) Info
-----
Depth, pcie index, node      : 3, 0, 0
Link Speed Active (Enabled)  : 16G-Gen 4 (16G-Gen 4)
Link Width Active (Enabled)  : 16X (16X)

PCIe error injection might cause a PCIe bus failures or a system hang
Do you want to continue? yes
Starting PCIe Error Injection...
```

Query Error Injection Status

After sending the configuration command, the progress of the process can be checked by executing the tool with the `pcie_error_injection` flag only:

```
mstlink -d /dev/mst/mt4129_pciconf0 --port_type PCIE --depth 3 --pcie_index 0 --node 0 --pcie_error_injection

PCIe Operational (Enabled) Info
-----
Depth, pcie index, node      : 3, 0, 0
Link Speed Active (Enabled)  : 16G-Gen 4 (16G-Gen 4)
Link Width Active (Enabled)  : 16X (16X)

PCIe Error Injection Info
-----
Error Injection Status      : In progress
Error Injection Type        : UNEXPECTED_CPL
Error Injection Duration    : 5 Packets
```

Once the process is complete, the output will be changed to "ready". This means that another error injection request can be submitted:


```


mstlink -d /dev/mst/mt4129_pciconf0 --port_type PCIE --depth 3 --pcie_index 0 --node 0 --pcie_error_injection
PCIe Operational (Enabled) Info
-----
Depth, pcie index, node      : 3, 0, 0
Link Speed Active (Enabled)  : 16G-Gen 4 (16G-Gen 4)
Link Width Active (Enabled)  : 16X (16X)
PCIe Error Injection Info
-----
Error Injection Status       : Ready
Error Injection Type         : N/A
Error Injection Duration     : N/A


```


mstresourcedump Utility

The mstresourcedump tool extracts and prints data segments generated by the firmware. It is supported in 5th generation NIC devices. The dump output is used by NVIDIA for debug and troubleshooting.

 Scapy and Pyverbs are no longer resourcedump dependencies. The same functionality has been changed with a C code that is based on the RDMA core included in OFED or Upstream.

 mstresourcedump can be used only if Python 3.x is installed. Using lower versions will result in the tool's failure.

 It is important for the user to generate a bin file for debugging and troubleshooting cases when needed by NVIDIA support team.

 If the firmware version used is not supported, the tool will generate the following error message:
 “Error: Failed to fetch query data with exception: Failed to send Register RESOURCE DUMP with rc: 515. Exiting...”.

mstresourcedump Usage

```
mstresourcedump [-h] [-v] {dump,query}
```

where

dump	Dump command
query	Query command
-h, --help	Show help message and exit
-v, --version	Shows tool version and exit

mstresourcedump Query Usage

```
mstresourcedump query [-h] [--virtual-hca-id VIRTUAL_HCA_ID] --device DEVICE
```

where

-h, --help	Show help message and exit
-v, --virtual-hca-id	The virtual HCA (host channel adapter, NIC) ID
-d, --device	The device name
-m, --mem	Perform the dump through memory (OFED with rdma-core dependency). Accepts: [ibv device (for example "mlx5_4")]

An example of how to run the query command:

```
# mstresourcedump query --device 04:00.0
```

```
Segment Type - 0x1300 (FULL_EQC)
```

Dump Params	Applicability	Special Values
index1 (EQN)	Mandatory	N/A
num-of-obj1	N/A	N/A
index2 (N/A)	N/A	N/A
num-of-obj2	N/A	N/A

```
Segment Type - 0x1000 (FULL_QPC)
```

Dump Params	Applicability	Special Values
index1 (QPn)	Mandatory	N/A
num-of-obj1	N/A	N/A
index2 (N/A)	N/A	N/A
num-of-obj2	N/A	N/A
...		
...		
...		

mstresourcedump Dump Usage

```
mstresourcedump dump [-h] -d DEVICE -s SEGMENT [-v VIRTUAL_HCA_ID] [-i1 INDEX1] [-i2 INDEX2] [-n1 NUM_OF_OBJ1] [-n2 NUM_OF_OBJ2] [-de DEPTH] [-b BIN] [-m]
```

where

-h, --help	Show help message and exit
-v, --virtual-hca-id	The virtual HCA (host channel adapter, NIC) ID
-i1, --index1	The first context index to dump (if supported for this segment)
-i2, --index2	The second context index to dump (if supported for this segment)
-n1, --num-of-obj1	The number of objects to be dumped (if supported for this segment). accepts: ["all", "active", number, depends on the capabilities]
-n2, --num-of-obj2	The number of objects to be dumped (if supported for this segment). accepts: ["all", "active", number, depends on the capabilities]
-de, --depth	The depth of walking through reference segments. 0 stands for flat, 1 allows crawling of a single layer down the struct, etc. "inf" for all

-b, --bin	The output to a binary file that replaces the default print in hexadecimal, a readable format
-d, --device	The device name
-s, --segment	The segment to dump
-m, --mem	Perform the dump through memory (OFED with rdma-core dependency). Accepts: [ibv device (for example "mlx5_4")]

Examples of how to:

- run the dump command:


```
# mstresourcedump dump --device 04:00.0 --segment 0x1200 --index1 0x404 --depth 0
Found 10 segments:
-----
Segment Type: 0xffffe
Segment Size: 16 Bytes
Segment Data:
0x0004FFFE 0x00000000 0x00000000 0x101A0111
-----
Segment Type: 0xffffa
Segment Size: 20 Bytes
Segment Data:
0x0005FFFA 0x12000000 0x00000404 0x00000000
0x00000000
-----
```

- run the Dump command and save it in bin file:

```
# mstresourcedump dump --device 04:00.0 --segment 0x1200 --index1 0x404 --depth 0 -bin segment_1200.bin
write to file: segment_1200.bin
```

mstresourceparse Utility

The mstresourceparse tool parses and prints data segments content. The parser's output is used by NVIDIA representatives for debugging and troubleshooting.

 The tool's applicable parsing inputs can be the mstresourcedump outputs (bin file or "human readable" format), or the devlink JSON format output.

 To parse the segments data in the most efficient way, you must use the most suitable ADB file. For the ADB file, please contact [NVIDIA Support](#).

mstresourceparse Usage

```
mstresourceparse -d DUMP_FILE -a ADB_FILE [-h] [--version] [-o OUT] [-r] [-v]
```

where

-d, --dump-file	Location of the dump file used for parsing
-a, --adb-file	Location of the ADB file
-h, --help	Shows this help message and exit

<code>--version</code>	Shows the tool's version and exit
<code>-o, --out</code>	Location of the output file
<code>-r, --raw</code>	Prints the raw data in addition to the parsed data
<code>-v</code>	Verbosity notice

Examples:

- How to run basic parsing:

```
# mstresourceparse --dump-file notice.txt --adb-file fw-4119-rel-16_23_2008.adb
Parse 4 segments:
-----

                Segment - segment_info (0xffffe)
segment_header.segment_type = 0xffffe
segment_header.length_dw = 0x4
dump_version = 0x0
hw_version = 0x0
fw_version = 0x1063232c
-----

                Segment - segment_command (0xffffa)
segment_header.segment_type = 0xffffa
segment_header.length_dw = 0x5
vhca_id = 0x0
segment_called = 0x2000
index1 = 0x21
index2 = 0x0
num_of_obj1 = 0x0
num_of_obj2 = 0x0
-----

                Segment - segment_notice (0xffff9)
segment_header.segment_type = 0xffff9
segment_header.length_dw = 0xc
syndrome_id = 0x211
notice[0] = 0x2000
notice[1] = 0x21
notice[2] = 0x0
notice[3] = 0x0
notice[4] = 0x496e7661
notice[5] = 0x6c696420
notice[6] = 0x52657300
notice[7] = 0x0
notice msg = !Invalid Res
-----

                Segment - segment_terminate (0xffffb)
segment_header.segment_type = 0xffffb
segment_header.length_dw = 0x1
-----
```

- How to run parsing with 'raw' and 'verbosity' options:

```
# mstresourceparse --dump-file notice.txt --adb-file fw-4119-rel-16_23_2008.adb -raw -v
```

Notice - adb fw version 16.23.2008 is used for parsing while dump fw version is 16.99.9004

Parse 4 segments:

```
-----  
Segment - segment_info (0xffffe)  
segment_header.segment_type = 0xffffe  
segment_header.length_dw = 0x4  
dump_version = 0x0  
hw_version = 0x0  
fw_version = 0x1063232c  
RAW DATA:  
DWORD [0-3] :0x0004FFFE 0x00000000 0x00000000 0x1063232C  
-----
```

```
Segment - segment_command (0xffffa)  
segment_header.segment_type = 0xffffa  
segment_header.length_dw = 0x5  
vhca_id = 0x0  
segment_called = 0x2000  
index1 = 0x21  
index2 = 0x0  
num_of_obj1 = 0x0  
num_of_obj2 = 0x0  
RAW DATA:  
DWORD [0-3] :0x0005FFFA 0x20000000 0x00000021 0x00000000  
DWORD [4] :0x00000000  
-----
```

```
Segment - segment_notice (0xffff9)  
segment_header.segment_type = 0xffff9  
segment_header.length_dw = 0xc  
syndrome_id = 0x211  
notice[0] = 0x2000  
notice[1] = 0x21  
notice[2] = 0x0  
notice[3] = 0x0  
notice[4] = 0x496e7661  
notice[5] = 0x6c696420  
notice[6] = 0x52657300  
notice[7] = 0x0  
RAW DATA:  
DWORD [0-3] :0x000CFFF9 0x00000211 0x00000000 0x00000000  
DWORD [4-7] :0x00002000 0x00000021 0x00000000 0x00000000  
DWORD [8-11] :0x496E7661 0x6C696420 0x52657300 0x00000000  
notice msg = !Invalid Res  
-----
```

```
Segment - segment_terminate (0xffffb)  
segment_header.segment_type = 0xffffb  
segment_header.length_dw = 0x1  
RAW DATA:  
DWORD [0] :0x0001FFFB  
-----
```

- How to run parsing and save it into a file:

```
# mstresourceparse --dump-file notice.txt --adb-file fw-4119-rel-16_23_2008.adb -out out_flie.txt
write to file: out_flie.txt
```

Troubleshooting

You may be able to easily resolve the issues described in this section. If a problem persists and you are unable to resolve it yourself please contact your NVIDIA representative or [Support](#).

General Related Issues

Issue	Cause	Solution
Adapter is no longer identified by the operating system after firmware upgrade	Happens due to burning the wrong firmware on the adapter, firmware corruption or adapter's hardware failure.	Power cycle the server. If the issue persists, extract the adapter and contact Support
Server is booting in loop/not completing boot after performing adapter firmware upgrade	Happens due to burning the wrong firmware on the adapter, firmware corruption or adapter's hardware failure.	Extract the adapter and contact Support
Enabling hardware access after configuring new secure host key, fails	The new configuration of the secure host key was not loaded by the driver	Restart the driver before enabling the hardware access again
mstflint tools fail on PCI device with the following errors: <ul style="list-style-type: none"> • Operation not permitted • Failed to identify device • Failed to detect device ID • Unknown device • No such device • Failed to open device 	Tools PCI semaphore might be locked due to unexpected process shutdown.	Run the following command: # mcra -c <mst_pci_device> *Supported on mstflint-4.4.0 and newer versions.

mstconfig Related Issues

Issue	Cause	Solution
Server not booting after enabling SRIOV with high number of VFs	Setting number of VFs larger than what the Hardware and Software can support may cause the system to cease working	To solve this issue: <ol style="list-style-type: none"> 1. Disable SRIOV in bios 2. Reboot server 3. Change num of VFs 4. Enable SRIOV in bios
When Querying for current configuration on ConnectX-3/ ConnectX-3Pro, some of the parameters are shown as "N/A"	The current firmware on the device does not support showing the device's default configuration	Update to the latest firmware

Issue	Cause	Solution
After resetting configuration using the tool on 5th generation (Group II) devices, the configuration's value does not change	Firmware loads the default configuration only upon reboot	Reboot the server

Installation Related Issues

Issue	Cause	Solution
Unable to install the tool package on ESXi platform and the following message is printed on the screen: Got no data from process	Insufficient privileges	<ol style="list-style-type: none"> 1. Copy the tool's package to /tmp/vmware and continue with the installation. If the issue persists, reboot the ESX server and try again 2. Use full file path of the tool's package Note: an additional reboot will be required after completing the installation
Unable to install kernel-mft in Linux due to compilation error that contains the following message: 'error: conflicting types for 'compat_sigset_t'	CONFIG_COMPAT might not be enabled in the kernel configuration.	Set the CONFIG_COMPAT to "y" in the kernel .config file, and rebuild the kernel.

Firmware Burning Related Issues

Issue	Cause	Solution
The following message is printed on the screen when performing firmware update: An update is needed for the flash layout. The operation is not failsafe and terminating the process is not allowed.	A flash alignment operation is required.	Approve the alignment, avoid process interrupt.
Firmware update fails with the following message: -E- Burning FS4 image failed: Bad parameter Note: This is a rare scenario.	Firmware compatibility issue.	Re-run the burn command with --no_fw_ctrl flag.

Issue	Cause	Solution
<p>The following message is printed on the screen when performing firmware update: Shifting between different image partition sizes requires a current image to be re-programmed on the flash. Once the operation is done, reload FW and run the command again Note: This is a rare scenario.</p>	<p>Firmware compatibility issue.</p>	<p>Re-load firmware and re-run the burn command.</p>
<p>The following message is printed on the screen when trying to query/burn a Connect-IB device: -E- Cannot open Device: 41:00.0. B14 Operation not permitted MFE_CMDIF_GO_BIT_BUSY</p>	<p>Using an outdated firmware version with the Connect-IB adapter.</p>	<p>1. Unload MLNX_OFED driver: / etc/init.d/openibd stop. 2. Add “-ocr” option to the 'mstflint' command. For example: mstflint -d 41:00.0 -ocr q</p>
<p>The following message is reported on screen when trying to remove the expansion ROM using the 'drom' option: -E- Remove ROM failed: The device FW contains common FW/ROM Product Version - The ROM cannot be removed separately.B9</p>	<p>Updating only the EXP_ROM (FlexBoot) for recent firmware images which requires adding the 'allow_rom_change' option.</p>	<p>Allow “-allow_rom_change” option to the “mstflint” command. For example: mstflint -d <mst_device> -allow_rom_change drom</p>
<p>Generating a firmware image file on Windows fails and the following message is printed on screen: -E- File: C:/Users/Administrator/ps.ini, Line: 1 - Invalid syntax -E- Image generation failed: child process exited abnormally</p>	<p>Using a firmware configuration file (.ini) which was generated by PowerShell text redirection: mstflint -d <mst_device> dc > <fw_conf_file>.ini</p>	<p>Generate the firmware configuration file (.ini) using CMD edit and continue with generating the firmware image file.</p>
<p>Burning command fails and the following message is printed on screen: -E- Can not open 06:00.0: Can not obtain Flash semaphore (63). You can run "mstflint -clear_semaphore - d <device>" to force semaphore unlock. See help for details.</p>	<p>Semaphore can be locked for any of the following reasons:</p> <ul style="list-style-type: none"> • Another process is burning the firmware at the same time • Failure in the firmware boot • Burning process was forcefully killed • In a Multi-Host environment, another Host is currently burning the firmware 	<p>If no other process is taking place at the same time run the following command: mstflint -d <device> --clear_semaphore OR Reboot the machine.</p>
<p>Burning tool fails with the following message: -E- Unsupported binary version (2.0) please update to latest mstflint package.</p>	<p>The binary version is incompatible with the burning tool.</p>	<p>Update mstflint to the latest package.</p>
<p>Burning tool fails with an error mentioning Firmware time stamping e.g -E- Burning FS3 image failed: Stamped FW version mismatch: 12.16.0212 differs from 12.16.0230</p>	<p>The device was set with a timestamp for a different firmware version than the one being burnt or the image is stamped with an older timestamp</p>	<p>Either set a newer timestamp on the image than there is on the device, or reset the timestamp completely. mstflint -d <device> ts reset mstflint -i <image> ts reset</p>

Issue	Cause	Solution
Burning the image on Controlled FW (default update method: fw_ctrl in 'mstflint -d <device> query full' output), fails with: -E- Burning FS3 image failed: The Digest in the signature is wrong.	The image was changed without calculating the new digest on it with 'mstflint -i <img.bin> sign'.	Run 'mstflint -i <img.bin> sign', and retry.

Secure Firmware Related Issues

Issue	Cause	Solution
Changing device setting such as ROM/ GUIDS using the relevant mstflint commands result in failure with the following error: -E- <Operation> failed: Unsupported operation under Secure FW	Secure Firmware does not allow changes to the device data unless burning new Secure Firmware image.	N/A
Burning tool fails with the following error: -E- Burning FS3 image failed: The component is not signed.	The image is not signed with an RSA authentication.	Contact Support to receive a signed firmware image.
Burning tool fails with the following error: -E- Burning FS3 image failed: Rejected authentication.	The image authentication is rejected.	Contact Support to receive a signed firmware image.
Burning tool fails with the following error: -E- Burning FS3 image failed: Component is not applicable.	The image does not match the device (Wrong ID).	Contact Support to receive the firmware image for the device.
Burning tool fails with the following error: -E- Burning FS3 image failed: The FW image is not secured.	The image is not secured and is not accepted by the device.	Contact Support to receive a signed firmware image.
Burning tool fails with the following error: -E- Burning FS3 image failed: There is no Debug Token installed.	The debug firmware was burnt before the debug token was installed on the device.	Install the debug token using mstconfig and then re-burn the firmware.
Burning firmware on a secure device fails with one of the following messages: <ul style="list-style-type: none"> -E- Burning FS3 image failed: Rejected authentication The FW image is not secured The key is not applicable 	The image was not secured in a the proper way.	Ask for a secure image with the right keys that match the device.
Secure Firmware fails when using mstflint brom and drom commands.	mstflint brom and drom commands are not supported.	N/A
When the CR space is in read only mode, the tracers may demonstrate an unexpected behavior.	A writing permission is required for them to work properly.	N/A

Issue	Cause	Solution
Applying token on the device fails with one of the following messages: <ul style="list-style-type: none"> • Component is not applicable • The manufacturing base MAC was not listed • Mismatch FW version • Mismatch user timestamp • Rejected forbidden version 	The token was not generated or signed in the proper way.	Generate and sign tokens.
Burning the firmware using the "--use_dev_rom" flag has no effect and the ROM is replaced with the one on the image.	Controlled firmware does not support changing boot image component.	Use "--no_fw_ctrl".

Appendixes

- [mstflint - Updating Firmware Using ethtool/devlink and .mfa2 File](#)

mstflint - Updating Firmware Using ethtool/devlink and .mfa2 File

In order to flash the firmware on the device using ethtool, you need to prepare a .mfa2 firmware file using the mstarchive tool - see [mstarchive - Binary Files Compression Tool](#). Note that mstarchive requires installing mstflint with `--enable-fw-mgr` option.

➤ To perform firmware upgrade using ethtool/devlink, follow the steps below:

1. Run the mstarchive tool to generate the .mfa2 file (the following example assumes MFA2 v1.1.1).

```
# mstarchive -v 1.1.1 --bins-dir <source binaries directory> --out-file /lib/firmware/<file_name>.mfa2
```

2. Obtain the interface name of the adapter for which you wish to update firmware. For example, you can use `ifconfig -a`.

```
# ifconfig -a
...
p5p1: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether ec:0d:9a:48:af:2a txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

p5p2: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether ec:0d:9a:48:af:2b txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
...
```

3. Burn the firmware using the .mfa2 image with ethtool/devlink. Please use the .mfa2 file path relative to `/lib/firmware`.
ethtool command:

```
# ethtool -f <interface name> <file_name>.mfa2
```

devlink command:

```
$ devlink dev flash <dev> file <file_name>.mfa2
```

4. Query the adapter to verify that the new firmware version has been loaded following.

```
# lspci -n | grep 15b3
04:00.0 0207: 15b3:1017
04:00.1 0207: 15b3:1017

# mstflint -d 04:00.0 q
Image type: FS4
FW Version: 16.26.0292
FW Version(Running): 16.25.1020
FW Release Date: 9.9.2019
Product Version: 16.26.0292
Rom Info: type=UEFI version=14.19.13 cpu=AMD64
          type=PXE version=3.5.802 cpu=AMD64
Description: UID GuidesNumber
Base GUID: 248a0703009e950c 4
Base MAC: 248a079e950c 4
Image VSD: N/A
Device VSD: N/A
PSID: MT_0000000008
Security Attributes: N/A
```

5. For the firmware update to take effect, you need to either reboot the server or run:

```
# mstfwreset -d 04:00.0 -y r
```

6. Validate the firmware update by a query.

Using mst:

```
# mstflint -d 04:00.0 q Image type: Image type: FS4
FW Version: 16.26.0292
FW Release Date: 15.5.2019
Product Version: 16.25.1042
Rom Info: type=UEFI version=14.18.19
          cpu=AMD64 type=PXE
          version=3.5.701 cpu=AMD64
Description: UID
Base GUID: ec0d9a030048af2a
Base MAC: ec0d9a48af2a
Image VSD: N/A
Device VSD: N/A
PSID: MT_00000000080
Security Attributes: N/A
#
```

Using devlink:

```
$ devlink dev info <dev>
pci/0000:05:00.0:
driver mlx5_core
versions:
  fixed: fw.psid MT_00000000080
  running: fw.version 16.23.1000
  stored: fw.version 16.25.1042
```

Document Revision History

Release Notes Revision History

mstflint Release Notes Change Log History

Component / Tool	Description	Operating System
Rev. 4.25.0		
mstconfig	Added --read_only as a new flag to the mlxconfig tool. When a query with this flag is enabled, the user is able to see read_only parameters. These parameters are marked with 'RO' in the query.	All
mstfwreset	Added support for DGX H100 device reset. All DGX H100 devices reset requests are handled by the mlxfwreset tool. When one of the DGX H100 devices is provided, the tool proceeds to reset all the devices accordingly. As the final step in the reset process, a reboot command is executed, resulting in a reboot of the entire setup.	All
mstlink	Added SNR (signal-noise ratio) for the media and host sides of active\optical NDR modules.	All
Rev. 4.24.0		
N/A	There were no mstflint changes in this version.	
Rev. 4.23.0		
General	This release contains reliability improvements and security hardening enhancements. NVIDIA recommends upgrading your software and tools to this release to improve the security and reliability.	All
Flint	Flint now supports CDB firmware update procedure for CMIS compliant cables.	All
General	NVIDIA firmware tools package now supports discovery and communication of InfiniBand devices on FreeBSD Operating Systems. Please note that this feature is dependent on OFED ibutils package.	All
General	Updated the supported firmware versions. For the updated version see Supported Adapter Cards Firmware Versions.	All
Bug Fixes	See Bug Fixes .	All
Rev. 4.22.0		
mstlink	Added support for error injection over PCI links.	All
NNT driver	Created a new NNT (NVIDIA Networking Tools) driver for MFT and MSTflint. The driver's source code is exposed in GitHub: https://github.com/Mellanox/NNT-Linux-driver/tree/main_devel	Linux
Cables	Added a setting that allows cable burning only via the primary ConnectX-7 adapter card in a setup with multiple ConnectX-7 cards. The error message "LinkX burn is not supported by secondary" will appear accordingly.	All
Cables	Added DDM information support for QSFP_CMIS cables.	All

mstlink	Removed the "Link Down" field from the BER collect in mstlink for EDR devices.	All
mstlink	Added "show eye" information with the (--show_eye) command for the PCIe links in Gen-1 and Gen-2 PCIe setups.	All
Bug Fixes	See Bug Fixes .	All
Rev. 4.20.1		
All	This version does not include changes related to MFT. The MFT version was changed to support a new ConnectX-7 firmware version.	All
Rev. 4.20.0		
All	Added support for NVIDIA ConnectX-7 adapter cards.	All
mstflint	mstflint cable/transceivers burning commands provide now validation and extract version from the image files.	All
mtsconfig	mlxconfig enables the users to apply token via MTUSB connected device.	All
mstresourcedump	Memory-Mode, data field is now transferred via memory instead of the resource-dump register. For further information, see mstresourcedump Utility	All
mstprivhost	Added a new flag to query all hosts status from the embedded Arm side for Multi-host systems. For further information, see mstprivhost - NIC Configuration by the Host Restriction Tool .	All
mstlink	Added support for the PRBS test mode of Active/Optical CMIS modules. For further information, see the Module PRBS test mode section .	All
mstlink	Added support for additional configuration flags of the module control parameters. For further information, see the Module control parameters section .	All
mstlink	Modified the output of the PCIe link information, removed the device status filed.	All
I ² C Access	Now the user can determine the I2C address to use for debug tools based on DevID.	All
Flash Support	Added additional Flash support for ConnectX-4 Lx and ConnectX-5 adapter cards	All
Bug Fixes	See Bug Fixes .	All
Rev. 4.17.0		
Anti-rollback Protection	Enabled Anti-rollback protection to prevent old vulnerable firmware versions from being flashed to the device.	All
DSFP Modules	Added support for DSFP modules in mstlink.	All
Parallel Firmware Burning (DMA Burning)	Added support for parallel firmware burning. Although DMA burning is supported in Virtual Machines as well, burning in such scenarios might be slower than on Physical Machines. Note: If the NIC driver is unloaded, burning via DMA is unsupported (due to BME is unset) and regular burn flow will be executed instead. Note: This capability is supported in 5th Generation devices only. Note: This capability requires mstflint kernel module to work.	Linux / FreeBSD

Bug Fixes	See Bug Fixes .	All
Rev. 4.14.0-1		
mstresourcedump	Added support for "--virtual-hca-id" command. Now the tool can provide info on the virtual HCA (host channel adapter, NIC) ID. For further information see resourcedump Utility	Linux
mstlink	HDR lane rate is now supported when in Pseudorandom Binary Sequence (PRBS) mode.	All
mstreg	Increased the registry keys the tool supports and now it exposes the full PRM. For additional information, refer to the PRM.	All
mstconfig	BOOT_INTERRUPT_DIS parameter was added to mlxconfig. When TRUE, legacy interrupts should not be used for receive/transmit indication. Polling should be used instead. Note: This is supported only if boot_legacy_interrupt_disable_supported is set to TRUE.	All
mstflint	Enables the user to to insert information manually to the flash on components such as MFG/DEV GUID/MAC when no information exists after the burn process using the command "mstflint -d <device> sg <guid>". If the information is not inserted manually, the existing GUID/MAC information will be used instead.	All
mstflint (Querying the MFA2 File)	Enables the user to query the MFA2 file using a PSID. For additional information, refer to Querying the MFA2 File .	All
mstlink	Added supported for switching between NRZ/PAM4 speeds for new devices that support HDR/200G speeds (ConnectX-6, ConnectX-6 Dx, Quantum, Spectrum-2).	All
Rev. 4.13.3-2		
Burning MFA2 Images	Enables the user to extract (i.e. unzip) 4MB images from MFA2 archive, that matches the device type and device PSIB. If there are more than one matching image, the user may use -latest flag and burn the latest firmware, or choose the required image from the user menu. Note: This feature is enabled when the "enable-fw-mgr" flag is set at the configure steps (see Dependencies) For further information see Burning the MFA2 Images .	All
Verifying MFA2 Archive	Enables the user to verify that a given MFA2 archive contains the image that matches the given device. Note: This feature is enabled when the "enable-fw-mgr" flag is set at the configure steps (see Dependencies) For further information see Verifying MFA2 Archive .	All
Binary Image Comparison	Enables the user to verify a firmware image on a device which operates in livefish mode by comparing it with an existing binary firmware file. For further information see Comparing the Binary Image .	Linux/FreeBSD
SDK	Added two new libraries to the WinMFT package for developing software that interacts with Mellanox devices The new SDK includes the mtrc and fastfwreset libs and headers.	Windows
mstresourcedump	Extracts and prints data segments generated by the firmware. It is supported in 5th generation NIC's devices. The dump output is used by Mellanox for debug and troubleshooting. For further information see mstresourcedump Utility . Note: This utility is supported only on Python 3.0 and up.	Linux Windows

mstlink	Added the option of using "--json" flag. The output of the tool when using this flag includes all options & commands in JSON format and prints it.	
mstreg	Added a new registry key: NCFG. This register is used to enable/disable device features and it is supported when <code>ICMD_QUERY_CAPABILITY.ncfg_reg==1</code> . For further information see mstreg Utility	All
Rev. 4.13.0		
Dynamic MSI-X Allocation	Dynamic MSI-X allocation capability allows users to control the number of MSI-X vectors allocated to a Virtual Function, thus, improve performance in guests systems. For further information of how to set this capability, see the "DYNAMIC_VF_MSIX_TABLE" parameter, in section MFT Supported Configurations and Parameters .	Windows
Fast Firmware Reset	Added support for a fast firmware reset (< 1 second) to ConnectX-5 adapter cards.	Windows
mstlink	Added support for reading the "Link Downed Counter" and "Link Error Recovery Counter" in the mlxlink utility when using InfiniBand protocol only.	All
mstlink	Added support for HDR PCIe grades in the EYE Opening Info in the mlxlink utility.	All
mstlink	Added a new flag (show links) to define the valid PCIe links. For further information, refer to mstlink Utility examples.	All
mstconfig	Added the ATS_ENABLED TLV param. When set to TRUE, the device will support Address Translation Service (ATS).	All
mstfwreset	Added save/restore ATS PCIE capability.	All
mstarchive	Added support for MFA2 query using the mstarchive tool. For further information refer to .mstarchive - Binary Files Compression Tool v4.13.0 .	All
fwtrace	fwtrace is now supported in non-secured mode as well.	Linux
Preboot Boot Settings	Updated the LEGACY_BOOT_PROTOCOL settings, added an NVME option. For further information refer to mstconfig Supported Configurations and Parameters .	All
mstfwreset	Added a new reset option (reset-type) to the reset command of mlxfwreset. The user can see the supported reset-types by using the query command. For further information refer to Copy of mstfwreset - Loading Firmware on 5th Generation Devices Tool .	All
mstconfig	Added the VF_VPD_ENABLE parameter to mstconfig. When set, the VPD capability is exposed to Virtual Functions.	All
Rev. 4.12.0		
mstfwmanager	Supports listing the contents of images archive. When running this command the tool will list all firmware images within this PLDM package for each image it displays.	All
mstlink	Displays and configures port related data at the physical layer.	All
mstfwtrace	Extracts and prints trace messages generated by the firmware of 5th generation devices. This tool supports secure firmware flow only.	Linux
mstreg	Exposes supported access registers, and allows users to obtain information regarding the registers fields and attributes, and to set and get data with specific register.	All

mstflint	Added image-reactivation feature which re-assigns the image signature to the previous image (in case the new image is faulty) enabling "fwreset" functionality or to burn a new image.	All
.deb Package Name	Changed the name of *.deb files from "mst-<version>.amd64.deb" to "mst-<version>_amd64.deb" e.g., from mst-4.11.0-34.amd64.deb to mst_4.11.0-34_amd64.deb	Linux
General	Added support for Spectrum-2 based switch systems.	All
mstfwmanager	Enabled the option to query PLDM images in mstfwmanager.	All
Switch Firmware	Enabled the option to extract the firmware ISSU version from the switches' firmware image.	Linux/MLNX-OS
Zero Touch RoCE	Added support for Zero Touch RoCE. It enables RoCE to operate on fabrics where no PFC nor ECN are configured. This makes RoCE configuration a breeze while still maintaining its superior high performance.	All
mstflint	Enabled setting VSD when Memory Chip Controller (MCC) capability is enabled.	All
mstflint	Added an option to reduce CPU utilization with "--low_cpu" flag.	All
General	Removed the COMFIG COMPACT definition.	Linux
General	Added support for libibmad 12.	Linux
mstconfig	Renamed the BOOT_RETRY_CNT1 parameter to BOOT_RETRY_CNT.	All
Bug Fixes	See Bug Fixes .	All
Rev. 4.11.0-5		
mstconfig	Added the below new params: <ul style="list-style-type: none"> • pci_atomics_disabled_ext_atomics_enabled • pci_atomics_enabled_ext_atomics_enabled • pci_atomics_enabled_ext_atomics_enabled_nocoherent • pci_atomics_enabled_ext_atomics_enabled_serialized • vf_nodnic_enable • vf_nodnic_supported • uctx_en (alpha support) • uctx_supported (alpha support) • prio_tag_required_en (alpha support) • prio_tag_required_supported (alpha support) • strict_vf_msix_num_enabled (alpha support) • strict_vf_msix_num_supported (alpha support) • nvme_emu_class_code (alpha support) • nvme_emu_device_id (alpha support) • nvme_emu_enable (alpha support) • nvme_emu_max_num_pf (alpha support) • nvme_emu_max_total_vf (alpha support) • nvme_emu_num_pf (alpha support) • nvme_emu_supported (alpha support) • nvme_emu_total_vf (alpha support) • nvme_emu_vendor_id (alpha support) 	All
General build	Fixed multiple builds and compilation issues.	All
mstfwtrace	Added support for the fwtracer tool. In this release, this tool supports secure FW flow only.	All
libibmad	Added support for libibmad 12 in addition to 5.	All
Python tools	Fixed Python wrapper to work properly in multiple OSes and multiple Python versions.	All
adb generic tools	Added support for the mstreg tool. To enable this option please use "--enable-adb-generic-tools" flag.	All

mstarchive	Added return values.	All
	Fixed an issue in the MFA2's header minor and subminor versions.	
Rev. 4.11.0-4		
Build related changes	Disabled the "-Werror" compiler flag.	All
	Added the option to choose local system libraries if available in the machine.	
	Enhanced python 3 compatibility.	
	Removed python shebangs from the scripts.	
	Fixed multiple rpmbuild issues.	
	Added the "--enable-all-static" configure flag that allows compilation of static executables, when the the option is supported by the compiler.	
	Aligned all python scripts to unix encoding.	
mstprivhost	Added mstprivhost tool to set host privilege configurations.	All
mstflint	Added missing Rom info attribute for legacy FW qq query.	All
	Renamed "--next_boot_fw_ver" to "--flashed_version" flag. This flag queries the flashed version. Note that the previous flag is supported to keep compatibility.	
	Fixed query timeout when device is locked.	
	Fixed a firmware upgrade issue on ConnectX-5 EN adapter cards.	
mstregdup	Fixed segfault	All
Rev. 4.11.0-3		
mstflint	Added the option to query only a flashed FW version, which reduces the CPU usage during the query. To use this option run the "--next_boot_fw_ver" flag.	All
	Fixed an issue that resulted in missing information during query.	
	Fixed a crash scenario when compiling with 'CXXFLAGS=-Wp,-D_GLIBCXX_ASSERTIONS'.	
Rev. 4.11.0-2		
Python 3	Added support for python 3.	All
mstflint	Added an option to reduce CPU utilization with "--low_cpu" flag.	All
OpenSSL Compatibility	Added compatibility to 1.1.X version, in addition to previously supported 1.0.2 version.	All
Rev. 4.11.0		
.deb Package Name	As of mstflint v4.12.0, the name of *.deb files will be changed from "mstflint-<version>.amd64.deb" to "mstflint_<version>.amd64.deb" e.g., from mstflint-4.10.0-104.amd64.deb to mstflint_4.10.0-104_amd64.deb	All
Supported Devices	Added support for Mellanox Quantum switch systems and ConnectX-6 Ready adapter cards. For further information on the ConnectX-6 adapter cards, please contact Mellanox Support.	All
mstarchive tool	The mstarchive tool allows the user to create a file with the mfa2 extension. The new file contains several binary files of a given firmware for different adapter cards. For further information, refer to section mstarchive .	All

mstprivhost	The ability to restrict the hosts from configuring the NIC. Meaning, only the Arm side will have the privilege to configure the NIC. Note: This utility is supported in BlueField devices only.	All
mstconfig in BlueField	Enables the user to manage (grant/restrict) mstconfig configuration privileges for BlueField Arm systems.	All
Bug Fixes	See MFT Bug Fixes History	All
Rev. 4.10.0		
ESXi	Added support for ESXi 6.7.	ESXi
FreeBSD	Added support for verbose output when running "mst status" in FreeBSD.	FreeBSD
mstfwreset	Enabled mstfwreset loading/unloading of the driver per a specific device in Linux OSes. Note: On Multi Host devices with firmware version lower than 1x.23.xxxx, the flag "--pci_link_downtime 2.5" must be added to mstfwreset	Linux
Secure Firmware	mstflint now handles all the burn parameters when MCC is enabled and displays the secure-FW CS tokens.	All
Supported Devices	[Beta] Added support for BlueField SmartNIC.	All
mstconfig	Added the option to query partial parameters	All
mstconfig	Added the following new parameters: <ul style="list-style-type: none"> • FLEX_PARSER_PROFILE_ENABLE • ECPF_ESWITCH_MANAGER • ECPF_PAGE_SUPPLIER • SAFE_MODE_ENABLE • SAFE_MODE_THERSHOLD • BOOT_UNDI_NETWORK_WAIT 	All
Bug Fixes	See MFT Bug Fixes History	Linux
Rev. 4.9.1		
mstfwreset	Added support for mstfwreset in Power9 platforms.	Linux
Rev. 4.9.0		
mstfwreset	Added support for a hot swap (or hot plug) of the PCIe slot.	Linux
Secure Firmware Update	Added support for Secure Firmware Update to ConnectX-4 adapter cards.	All
	Enabled signing the package with an RSA 4096 bit keys.	All
	Added support for setting the GUIDs when Secure Firmware Update is enabled.	All
mstconfig	Added the following mstconfig configuration parameters: <ul style="list-style-type: none"> • AUTO_RELOAD • DRIVER_SETTINGS • EXP_ROM_PXE_ENABLE • EXP_ROM_UEFI_ARM_ENABLE • EXP_ROM_UEFI_X86_ENABLE • INTERNAL_CPU_MODEL • IPV4 • IPV6 • PCI_DATA_WR_ORDERING_MODE • PXE_UNDI • STATUS_UPDATE • TCP • TCPIP • TRACER_ENABLE 	All
mstlink	Added support for force speed configuration.	All

	Added support for the PEPC (show_external_phy) register.	All
mstdump	Added support for nvlog dump.	All
Rev. 4.8.0		
mstconfig	Added support for hardware timestamp in ConnectX-3/ConnectX-3 Pro devices.	All
	Added the following mstconfig configuration parameters: <ul style="list-style-type: none"> MULTI_PORT_VHCA_EN BOOT_LACP_DIS IP_OVER_VXLAN_PORT IP_OVER_VXLAN_EN UEFI_HII_EN IB_ROUTING_MODE_P1 IB_ROUTING_MODE_P2 SRIOV_IB_ROUTING_MODE_P1 SRIOV_IB_ROUTING_MODE_P2 	All
Secure Firmware Update	Added support for Secure Firmware Update in ConnectX-5/ConnectX-5 Ex.	All
	Added support for setting forbidden versions.	All
FPGA management for JTAG Programming	Added the option to enable/disable FPGA management by the firmware for JTAG programming.	Linux
Rev. 4.7.0		
MST driver Microsoft certification	MST driver Microsoft certification allows running tools in extended secure boot environment.	Windows
Secure Firmware Update	Added support for Secure Firmware Update in ConnectX-4 and ConnectX-4 Lx.	Linux
mstflint	Added sign command for secured images. .	Linux
	Added a flag to enforce working in a non-secure mode, if available (according to security type).	
	Added expansion ROM CPU architecture to the mstflint query when the expansion ROM is available.	All
mstlink	Added a new tool that displays and configures port related data at the physical layer.	All
mstconfig	Added new mstconfig TLVs.	All
	Added support for generating and applying TLV configuration files.	All
mstdump	Added a new dump type “fsdump” to support dumping flow steering tables.	All
mst	Added support for adding remote devices in mst remote when the target machine does not have an MST kernel loaded.	Linux
mstcables	Added the option to dump the data from all readable pages.	All
	Added support for burning cable firmware on In Service Firmware Update (ISFU) supporting cables.	All
	Added support to access the cable via the MTUSB, when the cable is connected to a compatible board.	All
mstfwreset	Added support for MultiHost platforms.	All
Rev. 4.6.0		
Adapter Cards	Added support for ConnectX-5/ConnectX-5 Ex adapter cards. Note: ConnectX-5/ConnectX-5 Ex adapter cards are currently at Beta level.	All
mstconfig	Added an option to query active (current) configurations in mstconfig.	All

	Added new parameters in VPI settings configuration: XFI_MODE, PHY_TYPE, FORCE_MODE	
	Added a new parameter to the PCI configuration NON_PREFETCHABLE_PF_BAR	
mstburn	Added the ability to use mstvpd to read the device VPD when using mstburn.	Linux, Windows
fwreset	Added support for fwreset in PPC64 and PPC64LE platforms.	Linux
Rev. 4.5.0		
General	Added support for Innova IPsec 4 Lx EN /Innova Flex 4 Lx EN	Linux
	mstflint package size has been reduced in Linux by separating the architecture specific RPMs, and in ESXi, by moving relevant tools to the OEM package.	Linux / ESXi
mstcables	Enhanced cable query capabilities. Added the additional registers below for debug purposes when running the query (-q) flag: <ul style="list-style-type: none"> • device technology • identifier • wavelength/attenuation • speed/compliance 	All
	Added a new query to read thresholds and monitor the cable's properties: <ul style="list-style-type: none"> • Temperature • Voltage • RX/TX powers • TX Bias 	All
	Added a new RAW format for printing the data of the cable's pages using the "--raw/--format raw" flags.	All
mstconfig	Enabled mstconfig to work with a database that describes the meta data of the TLVs configuration of fifth generation devices.	All
	Added the following configuration TLVs to mstconfig: <ul style="list-style-type: none"> • MPFS • KEEP LINK UP • SW OFFLOAD CONF 	All
mstreg	Added support for PPTT, PPRT and PPAOS access registers in switches.	All
mstflint	Added support for viewing and changing OEMs' device flash parameters using an IB device when using mstflint.	All
Rev. 4.4.0		
mstfwreset	Added support for mstfwreset in PowerPC	Linux
mstconfig	Added the following new configurations: <ul style="list-style-type: none"> • Number of TCs • Number of VLS • Enable DCBX in CEE mode • Enable DCBX in IEEE mode • Allow the NIC to accept DCBX configuration from the remote peer • Enable DCBX • Enable the internal LLDP client • Select which LLDP TLV will be generated by the NIC 	All
General	Added support for all tools to work when the MST driver is not installed	Linux
mstcables	Added support for dumping Mellanox cables EEPROM by mstdump/mstdump tools	Linux Windows FreeBSD

	Added a new tool (mstcables) that reads/writes Mellanox cable registers and queries the cables info	Linux Windows FreeBSD
Build	Created one mstflint package for all 64 bits FreeBSD OSs	FreeBSD
mstfwmanager_pci	Removed support for mstfwmanager_pci tool (it is deprecated), since all the Linux tools can work without a kernel now. When required, use mstfwmanager instead.	Linux
mcra	Added support for clearing VSEC PCI semaphore by the mcra tool. The new capability can be used after killing a tool forcefully without clearing the semaphores. Supported devices: ConnectX-4, ConnectX-4 Lx and Connect-IB	All
mstreg	Added support for Switch-IB, Switch-IB 2, and Spectrum in the mstreg tool	All
mstconfig	Added the mstconfig tool to the mstflint package for WinPE	Windows
mstconfig	Added a backup command in mstconfig which allows user to save backup of the non-volatile configurations in a RAW file. This file can be set on the device by using the set_raw command	All
Build	Added support for running wrapped python tools (like fwtrace) in PPC64, PPC64LE and Arm platforms	Linux
mstreg	Added support for PPRT and PPTT registers in ConnectX-4 and ConnectX-4 Lx	All
Rev. 4.3.0		
General	Added support for Spectrum device.	All
	Added support for Switch-IB 2 device.	All
	4th generation and 5th generation IC devices are now also named Group I ICs and Group II ICs, respectively.	N/A
mstconfig	Added support for setting some of the parameters in textual values in addition to numerical values.	All
	Added new configurations: <ul style="list-style-type: none"> The PF log bar size The VF log bar size The number of PF MSIX The number of VF MSIX port owner Allow RD counters IP protocol used by flexboot 	All
	Added the option to display the configuration's default values.	All
mstflint	Added support to calculate checksum on selected sections in the firmware image.	All
	Added the option to attach a timestamp to the firmware image.	All
Burning Tools	Improved firmware burn performance in livefish mode on 5th generation devices.	All
	Added the ability to show the running firmware version in case it does not match with the burnt firmware version on the flash. This case generally occurs after firmware upgrade and before firmware reload.	All
mstreg	Added support for mstreg tool which can be used to modify access registers or to query them.	All
mst	Created an mst device per physical function. It can be seen by running 'mst status -v'.	All
fwtrace	Added support for the fwtrace tool in FreeBSD.	FreeBSD
mstfwreset	Added support for mstfwreset in Windows and FreeBSD.	Windows FreeBSD

Rev. 4.1.0		
General	Added support for ConnectX-4 Lx	Linux/Windows/ FreeBSD
	Added support for ConnectX-4	FreeBSD
mstconfig	Added support for the following configurations in ConnectX-4, ConnectX-4 Lx and Connect-IB: <ul style="list-style-type: none"> • IB Dynamically Connect • Internal Settings • RoCE Congestion Control ECN • RoCE Congestion Control Parameters • Wake on LAN 	Linux/Windows/ FreeBSD
	Added support for the following configurations in ConnectX-3 and ConnectX-3 Pro: <ul style="list-style-type: none"> • InfiniBand Boot Settings • Preboot Boot Settings 	Linux/Windows/ FreeBSD
msttrace	Added support for MEM mode in ConnectX-4	Windows
cpld_update	Added the cpld_update tool to the OEM package	Linux
mstfwreset	Added support for resetting the firmware	Windows/ FreeBSD
fwtrace	Added support in FreeBSD	FreeBSD
Burning Tools	This version supports new ConnectX-4/Connect-IB firmware version format (MM.mm.ssss). It also enables upgrade of older firmware version format: MM.mmmm.ssss	All
Rev. 4.0.0		
General	Added support for ConnectX-4 device	Linux/Windows
	Removed support for ConnectX and ConnectX-2	All
mst_fpga	Added a new tool that dumps registers and burns hardware for FPGA	Linux
mstconfig	Added support for ConnectX-4 and Connect-IB (Beta level)	Linux/Windows
Rev. 3.8.0		
General	Added support for Switch-IB device (at beta level)	Linux/Windows
	Added support for Debian/Ubuntu in PPC64 platform	Linux
mstphyburn	Added support for burning Aquantia external PHY	Linux
mstconfig	Added support for changing BAR size parameter	Linux/Windows
Rev. 3.7.1		
Bug Fixes	See MFT Bug Fixes History	Linux/Windows/ FreeBSD
Rev. 3.7.0		
mstfwmanager	Added online firmware update	Linux/Windows
mstburn	Added concurrency support to VPD read	Linux/Windows
	Added mstburn to mstflint	FreeBSD
mstflint	Added concurrency support to query firmware	Linux/Windows/ FreeBSD
General	Added support for Arm platform and Power8	Linux
	Removed support for x86	Windows
mstfwreset	Firmware reset for Connect-IB	Linux
fwtrace	Added fwtrace tool	Windows
Rev. 3.6.1		

mstconfig	Added mstconfig tool for changing non volatile configuration on device	Windows
Burning Tools	Added support for micron flash in mstflint and updated production burn flow on Connect-IB	Windows
Rev. 3.6.0		
mstconfig	Added mstconfig tool for changing non volatile configuration on device	Linux
Burning Tools	Added support for micron flash in mstflint and updated production burn flow on Connect-IB	Linux
mtserver	Added support for mtserver	FreeBSD
Rev. 3.5.0		
mstflint/wqdump	Redesigned the mstflint and wqdump utility to make their look and feel more user friendly	Linux/Windows
mstflint	Added support for brom in Connect-IB	Linux/Windows
mstmtdio	Added support for the mstmtdio utility	Linux
mstfwmanager	Added support for Connect-IB	Linux/Windows
FreeBSD	Added support for FreeBSD operating system (at beta level)	FreeBSD
Rev. 3.1.0		
General	The mstflint package now has 2 installation flavors - standard (default mode) and 'OEM'. The OEM mode provides the following extra functionality: <ul style="list-style-type: none"> • Tools for creating mstfwmanager package • Several features for mstflint that are used in Connect-IB™ production 	Linux
mstflint	Added support for burning Connect-IB via firmware interface. The '-override_cache_replacement' flag is not needed. This provides a 'safe' firmware update flow, without the risk of firmware or driver hanging	Linux
mstfwmanager	Added support for the mstfwmanager utility (at Beta level)	Linux
mstuptime	Added support for the mstuptime utility (at Beta level)	Linux
Rev. 3.0.0		
General	Added support for Connect-IB device (at beta level)	Linux/Windows
	Added support for ConnectX-3 Pro device (at beta level)	Linux/Windows
	Added support for Ubuntu operating system	Linux
	Added support for running tools against PCI device [domain]:bus:dev.fn like: 0000:1a:00.0 or 1a:00.0 and devices used by OFED driver like: mst4_0	Linux
	The package contains only the mstflint firmware update tool. Other debug tools were removed	Windows
mstflint	Added support for new flash types: N25Q0XX (Micron) and W25Xxx (Winbond)	Linux/Windows
mstdump	Added support for the mstdump utility (at beta level)	Linux/Windows
mstmcg	Renamed mcg to mstmcg	Linux/Windows
spark	spark was removed from mstflint version 3.0.0	Linux/Windows

Supported Devices	The following adapter cards and switch systems are no longer supported in mstflint version 3.0.0: <ul style="list-style-type: none"> • InfiniHost 4X • InfiniHost III Ex • InfiniHost III Lx 4X • InfiniScale • InfiniScale III 	Linux/Windows
Rev. 2.7.2b		
All	Added support for WinPE 4.0 OS	Windows
Rev. 2.7.2		
General	It is no longer required to run mst start/stop when using Winmstflint tools. The service is automatically loaded/unloaded when an mstflint tool is running. The mst service installation was removed from the setup	Windows
	Added support for SwitchX silicon devices	Windows
mstflint	Added support for Atmel AT25DFxx flash family	Windows
	Added support for burning firmware via Command Line Interface (CLI) on SwitchX devices	Windows
Rev. 2.7.1a		
Added the mcg tool (Beta level)	The mcg tool displays the current multicast groups and flow steering rules configured in the device. Target users: Developers of Flow Steering aware applications. This tool dumps the internal steering table which is used by the device to steer Ethernet packets and Multicast IB packets to the correct destination QPs. Each line in the table shows a single filter and a list of destination QPs. Packets that match the filter are steered to the list of destination QPs	Linux
Removed support for In-band access on OFED 1.4 InfiniBand driver	In-band access is supported using OFED 1.5.X and higher	Linux
Rev. 2.7.1		
General	Added mstconfig tool. This tool sets firmware configurations for Mellanox adapters. These configurations are nonvolatile they apply over device reboots. For further details, please run "mstconfig -h". The tool is at beta level	Linux
	Added support for ConnectX-3 silicon device	Windows
	Added the I2CBridge (Dimax's Driver for USB to I2C Adapter) as part of the Winmstflint installation package. However, the I2CBridge is not installed by default	Windows
mstflint installation change	Removed the isw tool. The isw tool functionality was replaced by the "msti2c" tool. For example, to scan the devices on the i2c bus, run: > msti2c -d <dev> scan instead of > isw -d <dev>	Windows
mstflint	Added support for Atmel AT25DFxx flash family	Linux
	Cleared error messages displayed when trying to burn firmware image of a different device. For example when burning ConnectX-2 firmware image on ConnectX-3 device	Linux
	Added support for flash type SST25VF016B	Windows
	Added support for flash type M25PX16	Windows

	<ul style="list-style-type: none"> The ROM section in the image now contains multiple boot images. Therefore mstflint was modified to display information for all of the images in the ROM section. Added support to display/burn UEFI ROM/ 	Windows
	Added an option to set the VSD and GUIDs in a binary image file. This is useful for production to prepare images for pre-assembly flash burning. These new commands are supported by Mellanox 4th generation devices	Windows
	Added an option to set the VSD and GUIDs on an already burnt device. These commands (“sg” and “sv”) re-burn the existing image with the given GUIDs or VSD. When the 'sg' command is applied on a device with blank (0xff) GUIDs, it updates the GUIDs without re-burning the image	Windows
mst	Added support for using ibnetdiscover in the 'mst ib add' command	Windows
mstburn	Added support for VPD read/write	Windows
Rev. 2.7.0a		
Bug Fixes	See MFT Bug Fixes History	Linux
Rev. 2.7.0		
General	Added support for Mellanox ConnectX-3 and SwitchX silicon devices	Linux
	Added Secure host feature which 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 (see mstflint changes). mstflint tools cannot run on a device with hardware access disabled. This feature is enabled only with supporting firmware	Linux
	Removed support for Itanium (ia64)	Linux
mstflint	Added the following commands: <ul style="list-style-type: none"> enable/disable access to the hardware set/change the key used to enable access to the hardware 	Linux
	The ROM section in the image now contains multiple boot images. Therefore the mstflint was modified to display information for all of the images in the ROM section	Linux
	Added support to display/burn UEFI ROM	Linux
	Added support for burning firmware via Command Line interface on SwitchX devices	Linux
mstburn	Added option to add or replace a single keyword in the VPD writable section (-vpd_set_keyword flag)	Linux
	Added the option to set a binary VPD field data	Linux
mstflint installation	Added the option --without-kernel which allows user to install mstflint without the mst kernel	Linux
Rev. 2.6.2		
mstflint installation change	RPM based installation: <ul style="list-style-type: none"> Applications are installed using a pre-compiled binary RPM Kernel modules are distributed as a source RPM and compiled by the installation script Fast installation process 	Linux
	Removed prerequisite libraries: expat and zlib-devel	Linux

	The package tools, libraries and headers are now installed under: { prefix }/bin or { prefix }/lib and { prefix }/include dirs. Directory /usr/mst is not created. For example, the “mread”, “mwrite” and “mcra” tools that were previously installed by default under /usr/mst/bin, now are installed under /usr/bin	Linux
	Removed the InfiniScale and InfiniBridge tools	Linux
	Removed the Infinivision tool set	Linux
	Removed the isw tool. The isw tool functionality was replaced by the “msti2c” tool. For example, to scan the devices on the i2c bus, run: > msti2c -d <dev> scan instead of > isw -d <dev>	Linux
mstflint	Added support for flash type SST25VF016B	Linux
	Added support for flash type M25PX16	Linux
	Added an option to set the VSD and GUIDs in a binary image file. This is useful for production to prepare images for pre-assembly flash burning. These new commands are supported by Mellanox 4th generation devices	Linux
	Added an option to set the VSD and GUIDs on an already burnt device. These commands (“sg” and “sv”) re-burn the existing image with the given GUIDs or VSD. When the 'sg' command is applied on a device with blank (0xff) GUIDs, it updates the GUIDs without re-burning the image	Linux
mst	Added support for using ibutils2/ibdiagnet and ibnetdiscover in the 'mst ib add' command	Linux
	Removed the _uar, _msix and _ddr devices from the mst device list	Linux
Debug tools	Added support for routing I2C bus to the IS4 device on IS50XX systems	Linux
Rev. 2.6.1		
Bug Fixes	See MFT Bug Fixes History	Linux
Rev. 2.6.0		
mstflint installation change	Added the options: --without-image-generation, --disable-dc, and --without-kernel which allow for a partial installation in order to avoid problems with SW dependencies	Linux
	Now allows a non-root user to prepare mstflint RPMs	Linux
All	Added Mellanox ConnectX-2 and BridgeX support	Linux/Windows
mstflint	Added a CRC check for the full image	Linux
	Support for query/burn of clp-gpxe ROM	Linux
	Prevents burning a ConnectX-2 image onto a ConnectX device and vice versa	Linux
	Added a logging option to mstflint	Linux
	For the ConnectX device family only: Added commands for an independent burn/read/remove of an Expansion ROM image. <i>For firmware versions earlier than 2.7.000:</i> It is possible to read the ROM image, or to replace an already existing ROM image (by the burn command). However, burning a new ROM image in case a previous image did not exist is not possible, nor is it possible to remove an existing ROM image	Linux
mstburn	Added the -fw_dir option which looks for a suitable firmware file in the given directory	Linux

	Support for generating a non-fail-safe image for ConnectX/ ConnectX-2, InfiniScale IV, and BridgeX devices	Linux
Debug tools	Updated the msti2c utility	Linux

mstflint Bug Fixes History

The table below lists the history of bugs fixed.

Internal Ref. No.	Issue
3471307	Description: Fixed an issue where incorrect eye information was displayed for 10G speed over ConnectX-7 devices.
	Keywords: Eye information, ConnectX-7
	Discovered in Version: 4.23.0
	Fixed in Release: 4.25.0
3272703	Description: Fixed an issue that prevented the RM loopback for ConnectX6/Dx from being applicable over 50G\lane link speeds.
	Keywords: mftlink
	Discovered in Version: 4.20.0
	Fixed in Release: 4.23.0
3255683	Description: Fixed an issue that caused multiple InfiniBand devices to be accessed in the same run.
	Keywords: IB devices
	Discovered in Version: 4.21.0
	Fixed in Release: 4.23.0
3236623	Description: Fixed an issue that prevented the "mlxlink_ext" tool from properly running when an ADB file had Windows-style line endings.
	Keywords: mftlink
	Discovered in Version: 4.16.3
	Fixed in Release: 4.23.0
3179769	Description: Removed dl_down counter information from the PCIe show_counter command.
	Keywords: mftlink, PCIe
	Discovered in Version: 4.21.0
	Fixed in Release: 4.22.0
2834389	Description: Limited the SET operations on IB devices to registers of up to 240 bytes.
	Keywords: mlxreg
	Discovered in Version: 4.18.0
	Fixed in Release: 4.20.0
2274123	Description: mstfwreset is supported on SmartNic devices on Windows OS only if the device's name format is "mt*_pciconf*" and not "***.**.*".
	Keywords: mstfwreset
	Discovered in Version: 4.16.0
	Fixed in Release: 4.20.0

Internal Ref. No.	Issue
2871042	<p>Description: mstfwmanager default query on switches will take pci_cr0 instead of pciconf0, which is expected to fail in secure-fw switches.</p> <p>Keywords: mstfwmanager, pci_cr0, pciconf</p> <p>Discovered in Version: 4.18.0</p> <p>Fixed in Release: 4.20.0</p>
2578580	<p>Description: Fixed an issue that resulted in getting MVPD read errors from the mstfwmanager during fast reboot.</p> <p>Keywords: mstfwmanager, MVPD_READ4 failed, fast reboot</p> <p>Discovered in Version: 4.16.0</p> <p>Fixed in Release: 4.17.0</p>
2628490	<p>Description: Fixed inconsistent flashing of the firmware when using the IPMB service.</p> <p>Keywords: mstflint</p> <p>Discovered in Version: 4.16.0</p> <p>Fixed in Release: 4.17.0</p>
2395589	<p>Description: Changed the mstflint "--activate" flag behavior to include a minimal delay of 1 second to avoid disconnections if the connected port is being activated. To use the "legacy" activation flow, use the "--activate_delay_sec 0" command.</p> <p>Keywords: "--activate" flag, mstflint</p> <p>Discovered in Version: 4.16.0</p> <p>Fixed in Release: 4.17.0</p>
2494596	<p>Description: mstflint now supports the "--activate_delay_sec" flag which performs the activation on the newly burned firmware after the specified delay.</p> <p>Note: The burn flow will be locked after this command has been sent for a couple of minutes, until activation flow is done.</p> <p>Keywords: "--activate_delay_sec" flag, mstflint</p> <p>Discovered in Version: 4.16.0</p> <p>Fixed in Release: 4.17.0</p>
2443427	<p>Description: Fixed an issue that resulted in "--json" flag not working with features that require a user confirmation.</p> <p>Note: Despite the fix, it is recommended to use the "--json" flag with the force flag set to yes.</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.16.0</p> <p>Fixed in Release: 4.17.0</p>
2071210	<p>Description: mstconfig query for the BOOT_INTERRUPT_DIS TLV shows a wrong value in the "current value" field.</p> <p>Keywords: mstconfig</p> <p>Discovered in Version: 4.14.0-1</p> <p>Fixed in Release: 4.17.0</p>
2224507	<p>Description: mstflint is currently not in ConnectX-6 Lx adapter cards.</p> <p>Keywords: mstflint</p> <p>Discovered in Version: 4.15.0</p> <p>Fixed in Release: 4.17.0</p>

Internal Ref. No.	Issue
2183083	<p>Description: mstflint does not support using combined short flags without a separation between them. For example:</p> <ul style="list-style-type: none"> • Not recommended: -emc • Recommended: -e -m -c <p>Keywords: Short flags</p> <p>Discovered in Version: 4.16.0</p> <p>Fixed in Release: 4.17.0</p>
2391274	<p>Description: mstfwreset is not supported in SmartNIC devices.</p> <p>Keywords: mstfwreset, SmartNIC devices.</p> <p>Discovered in Version: 4.16.3</p> <p>Fixed in Release: 4.17.0</p>
2060223	<p>Description: Performing a driver restart while burning the firmware results in firmware burning failure, and occasionally in device being inaccessible.</p> <p>Keywords: Firmware burning, driver restart</p> <p>Discovered in Version: 4.15.0</p> <p>Fixed in Release: 4.17.0</p>
1918749	<p>Description: mstlink tool displays a wrong speed when using ETH cables on ConnectX-6 adapter cards.</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.13.0</p> <p>Fixed in Release: 4.16.0</p>
1755062	<p>Description: To execute firmware reset on a multi-host card, mstfwreset must be run simultaneously on each one of the hosts. Running mstfwreset simultaneously on the same host is incorrect and may result in server hanging.</p> <p>Keywords: mstfwreset</p> <p>Discovered in Version: 4.12.0</p> <p>Fixed in Release: 4.16.0</p>
1747607	<p>Description: When using the mstfwreset tool to reset the firmware on the BlueField card, the firmware is not synchronized between the host (SmartNIC device) and the Arm side.</p> <p>Keywords: mstfwreset, BlueField</p> <p>Discovered in Version: 4.11.0</p> <p>Fixed in Release: 4.16.0</p>
2125012	<p>Description: In case a device enters the livefish mode and all the information on the flash including write-protected manufacturing information is lost, mstflint might not be able to recover the device.</p> <p>Keywords: mstflint</p> <p>Discovered in Version: 4.14.0-1</p> <p>Fixed in Release: 4.16.0</p>
2110890	<p>Description: If there is a json-c library installed on the machine, some of mstflint tools like mstflint and mstfwmanager will not be compiled successfully unless the installed library is removed.</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.14.0-1</p> <p>Fixed in Release: 4.16.0</p>

Internal Ref. No.	Issue
2084837	<p>Description: Setting the speeds (50GbE and 100GbE) for the new devices (Connect-X 6 and above, Quantum switches and above) requires specifying the number of lanes for the speed: <code>mstlink -d <dev> --speeds [50G_2X 50G_1X 100G_2X 100G_4X]</code> For PRBS mode, to work with PAM4 speeds, use the same speed naming for (50GbE, and 100GbE).</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.14.0-1</p> <p>Fixed in Release: 4.16.0</p>
2183083	<p>Description: MFT tools do not support using combined short flags without a separation between them. For example:</p> <ul style="list-style-type: none"> • Not recommended: <code>-emc</code> • Recommended: <code>-e -m -c</code> <p>Keywords: Short flags</p> <p>Discovered in Version: 4.15.0</p> <p>Fixed in Release: 4.16.0</p>
2297524	<p>Description: Fixed an issue that caused lifecycle to be wrongly reported in ConnectX-6 adapter cards.</p> <p>Keywords: Lifecycle, ConnectX-6</p> <p>Discovered in Version: 4.14.0-105</p> <p>Fixed in Release: 4.16.0</p>
2319179	<p>Description: Fixed an issue that caused HMAC not to be written in livefish. Note: HMAC is now supported only from the Arm side and only if not in secure mode.</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.14.0-105</p> <p>Fixed in Release: 4.16.0</p>
2084837	<p>Description: Setting the speeds (50GbE and 100GbE) for the new devices (Connect-X 6 and above, Quantum switches and above) requires specifying the number of lanes for the speed: <code>mstlink -d <dev> --speeds [50G_2X 50G_1X 100G_2X 100G_4X]</code> For PRBS mode, to work with PAM4 speeds, use the same speed naming for (50GbE, and 100GbE).</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.14.0-105</p> <p>Fixed in Release: 4.16.0</p>
2125012	<p>Description: In case a device enters the livefish mode and all the information on the flash including write-protected manufacturing information is lost, flint might not be able to recover the device.</p> <p>Keywords: mstflint</p> <p>Discovered in Version: 4.14.0-105</p> <p>Fixed in Release: 4.16.0</p>
2151018	<p>Description: Occasionally, when burning MFA2 using mstflint, it might get stuck if in the middle of the process mstfwreset is executed.</p> <p>Keywords: MFA2, mstflint</p> <p>Discovered in Version: 4.15.0</p> <p>Fixed in Release: 4.16.0</p>

Internal Ref. No.	Issue
2193807	<p>Description: Cable firmware burning capability is not supported.</p> <p>Keywords: mlx cables</p> <p>Discovered in Version: 4.15.0</p> <p>Fixed in Release: 4.16.0</p>
2319984	<p>Description: Fixed an issue that caused the margin scan to fail with the following message: Eye scan not completed.</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.15.0</p> <p>Fixed in Release: 4.16.0</p>
2259628	<p>Description: Wrong supported cable speed is displayed when using cable with P/N MCP2M00-A01A on a BlueField device.</p> <p>Keywords: BlueField, cables</p> <p>Discovered in Version: 4.15.0</p> <p>Fixed in Release: 4.16.0</p>
2288076	<p>Description: Fixed an issue that caused the device to be inaccessible for 3 minutes when applied bad tokens.</p> <p>Keywords: mstconfig</p> <p>Discovered in Version: 4.15.1</p> <p>Fixed in Release: 4.16.0</p>
1918749	<p>Description: mstlink tool displays a wrong speed when using ETH cables on ConnectX-6 adapter cards.</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.13.0</p> <p>Fixed in Release: 4.14.0-1</p>
1797470	<p>Description: Fixed an issue that prevented the mstflint tools from recognizing the second port on dual port devices.</p> <p>Keywords: mstflint, dual port devices</p> <p>Discovered in Version: 4.12.0</p> <p>Fixed in Release: 4.13.0</p>
1791107	<p>Description: In mstlink, the option of showing SLTP for 16nm technology is supported only when in Advanced mode.</p> <p>Keywords: mstlink</p> <p>Discovered in Version: 4.12.0</p> <p>Fixed in Release: 4.13.0</p>
1572590	<p>Description: swreset is currently not supported on the managed Mellanox Quantum switch systems.</p> <p>Keywords: swreset, Mellanox Quantum</p> <p>Discovered in Version: 4.11.0</p> <p>Fixed in Release: 4.13.0</p>
1608671/1523443	<p>Description: mstfwmanager "--download" command is currently not functional on PPC64/PPC64le and aarch64 platforms.</p> <p>Keywords: mstfwmanager, PPC64/PPC64le/aarch64</p> <p>Discovered in Version: 4.11.0</p>

Internal Ref. No.	Issue
	Fixed in Release: 4.13.0
1655224	<p>Description: Decreased mstflint query timeout from 80 seconds to 8 seconds. In case the tool does not get a response from the device after 8 seconds, the following error message is displayed: <i>"Cannot open Device: /dev/mst/mt4117_pciconf0. Resource unavailable".</i></p> <p>Keywords: mstflint query</p> <p>Discovered in Version: 4.11.0</p> <p>Fixed in Release: 4.12.0</p>
1307423	<p>Description: Execution of the mstfwreset utility on a device with VFs configured may take longer than expected to be completed.</p> <p>Keywords: mstfwreset</p> <p>Discovered in Version: 4.9.0</p> <p>Fixed in Release: 4.11.0</p>
1406842	<p>Description: mst tools run slower on Bluefield devices. Firmware burning may take up to 20 minutes.</p> <p>Keywords: BlueField, firmware burn</p> <p>Discovered in Version: 4.10.0</p> <p>Fixed in Release: 4.11.0</p>
1336170	<p>Description: mstfwreset is not supported in Secure Boot systems.</p> <p>Keywords: mstfwreset</p> <p>Discovered in Version: 4.10.0</p> <p>Fixed in Release: 4.11.0</p>
1213983	<p>Description: Connect-IB function per port (FPP_EB) is not exposed at mstconfig.</p> <p>Keywords: mstfwreset, Connect-IB</p> <p>Discovered in Version: 4.7.0</p> <p>Fixed in Release: 4.9.0</p>
1064918/ 1069102	<p>Description: mstfwreset does not load the firmware properly on a Socket-Direct card.</p> <p>Keywords: mstfwreset</p> <p>Discovered in Release: 4.7.0</p> <p>Fixed in Release: 4.8.0</p>
1097425	<p>Description: mstfwmanager does not handle Socket Direct adapters correctly.</p> <p>Keywords: mstfwmanager</p> <p>Discovered in Release: 4.7.0</p> <p>Fixed in Release: 4.8.0</p>
907531	<p>Description: mstfwreset is not functional on MultiHost and Socket Direct NICs.</p> <p>Keywords: mstfwreset</p> <p>Discovered in Release: 4.6.0</p> <p>Fixed in Release: 4.7.0</p>
969322/ 969566	<p>Description: mstfwreset may fail to reset the device on Ubuntu PPC64LE systems when multiple kernels are installed.</p> <p>Keywords: kernel module, mstfwreset, Ubuntu PPC64LE</p> <p>Discovered in Release: 4.6.0</p>

Internal Ref. No.	Issue
	Fixed in Release: 4.7.0
795226/ 795657/ 862607	Description: Occasionally, mstfwreset (driver mode) do not function after running mstfwreset in PowerPC machines. Keywords: mstfwreset Discovered in Release: 4.4.0 Fixed in Release: 4.6.0
795756/ 795916	Description: mstfwreset disables and enables all Mellanox devices' Network Interfaces when resetting the firmware on a device that at least one of its network interfaces is up. Keywords: mstfwreset Discovered in Release: 4.4.0 Fixed in Release: 4.5.0
795479/ 795521	Description: Running mstfwreset against OEM devices may enter the device to a undefined state. Keywords: mstfwreset Discovered in Release: 4.4.0 Fixed in Release: 4.5.0

mstflint User Manual Revision History

Revision	Date	Description
4.26.0	November 9, 2023	Updated: <ul style="list-style-type: none"> Compilation and Installation
4.25.1	October 22, 2023	Added: <ul style="list-style-type: none"> Advance Options to "Generating an XML Template for the Configurations" under mstconfig Commands.
4.24.0	May 4, 2023	Updated: <ul style="list-style-type: none"> mstregdump Utility Examples of mstfwreset Usage mstflint - Firmware Burning Tool mstlink Utility Added: <ul style="list-style-type: none"> Burning/Querying a Component in mstflint
4.23.0	January 31, 2023	Updated: <ul style="list-style-type: none"> User Manual - Added "MFTshell" mstfwreset - Loading Firmware on 5th Generation Devices Tool

Revision	Date	Description
4.22.0	October 31, 2022	<p>In the mstlink Utility chapter:</p> <ul style="list-style-type: none"> Updated: <ul style="list-style-type: none"> Margin Scan Tool RX Error Injection Module PRBS Test Mode Tool Usage with NIC vs. Switch (-p Flag) Added: <ul style="list-style-type: none"> PCIE Error Injection <p>In mstresourcedump Utility, added a note.</p>
4.21.0	July 30, 2022	<p>Updated mstflint: Burning a Firmware Image, added "Querying Vendor Specific Firmware Information from a NVIDIA AOC / Transceiver"</p> <p>Updated the "-l --loopback <loopback>" description in mstlink Utility</p>
4.20.0	April 30, 2022	<ul style="list-style-type: none"> Updated mstlink commands, added cable operations. Updated mstprivhost - NIC Configuration by the Host Restriction Tool, added Embedded Arm CPU options. Updated mstresourcedump Utility, added the "-mem" key.
4.18.0	November 30, 2021	<ul style="list-style-type: none"> Updated mstlink Utility <ul style="list-style-type: none"> Added "--tx_prbs <tx_prbs_mode>" & "--rx_prbs <rx_prbs_mode>" flags Added "--amber_collect" flag
4.17.0	June 30, 2021	<ul style="list-style-type: none"> Updated section mstflint: Querying the Firmware Image, added Secure-boot attributes. Updated section mstflint: Burning a Firmware Image, added the "--activate_delay_sec <timeout in seconds>" flag. Updated section mstlink, added "--fom_measurement <eye>" flag.
4.16.0	January 31, 2021	<ul style="list-style-type: none"> Added Cable Firmware Update (In-Field-Firmware-Update) Added Rx-to-Tx Loopback Mode Activation Added Margin Scan for PCIe Link Updated mstprivhost - NIC Configuration by the Host Restriction Tool. Added the "q query" flag
4.15.0	September 15, 2020	<ul style="list-style-type: none"> Added mstresourceparse Utility Updated mstlink Utility
4.14.0-1	March 20, 2020	Added Querying the MFA2 File .
4.13.3	December 12, 2019	<ul style="list-style-type: none"> Added a new registry key to mstreg Utility Added section Comparing the Binary Image
4.13.0	September 26, 2019	<ul style="list-style-type: none"> Updated the following sections: <ul style="list-style-type: none"> mstlink Utility. Added a new subsection Tool Usage on Quantum HDR Switch Systems with Split Ports Copy of mstfwreset - Loading Firmware on 5th Generation Devices Tool

Revision	Date	Description
4.12.0	April 30, 2019	<ul style="list-style-type: none"> Updated the following sections: <ul style="list-style-type: none"> mstfwmanager - Firmware Update and Query Tool: Added the following options: <ul style="list-style-type: none"> "--download-type Type" and "--ssl-certificate Certificate" mstflint - Firmware Burning Tool: Added the following options: <ul style="list-style-type: none"> "-qq", "-low_cpu" and "-flashed_version" mstlink Utility: Added the following options: <ul style="list-style-type: none"> "--depth <depth>", "--pcie_index <pcie_index>", and "--node <node>" mstfwtrace Utility: Added the following options: <ul style="list-style-type: none"> "-gvm" and "-ignore_old_events" Using mstconfig to Split a Port in a Remotely Managed Switch
3.4	Mar. 2019	Converted to online html format; some reorganization.
	Dec. 2, 2018	<ul style="list-style-type: none"> Added the following sections: <ul style="list-style-type: none"> mstarchive - Binary Files Compression Tool Using mstconfig to Split a Port in a Remotely Managed Switch mstcongestion - Tool for Setting Congestion Mode and Action mstprivhost
3.3	Sept. 26, 2018	<ul style="list-style-type: none"> Updated section "Command Parameters" on page 80. Added the following note to the "bb" option: Note: The MFT 'bb' option is an advanced flag used ONLY for production flows. It is NOT recommend to use it as it can cause unrecoverable firmware burning failures.
	July 4, 2018	<ul style="list-style-type: none"> Updated the following sections: <ul style="list-style-type: none"> Examples of mst Usage - FreeBSD Supported Configurations and their Parameters: Added the following parameters: BOOT_UNDI_NETWORK_WAIT, MKEY_BY_NAME, ECPF_ESWITCH_MANAGER, ECPF_PAGE_SUPPLIER, FLEX_PARSER_PROFILE_ENABLE, FLEX_IPV4_OVER_VXLAN_PORT, SAFE_MODE_THRESHOLD and SAFE_MODE_ENABLE Updated mstfwreset Synopsis Added table Impact of Secure Firmware on MFT to section Secure Firmware Update Added the following sections: <ul style="list-style-type: none"> Re-Signing a Binary Image File mstfwreset for SmartNICs Added items to section mstfwreset Limitations

Revision	Date	Description
3.2	Mar. 1, 2018	<ul style="list-style-type: none"> • Added the following sections: <ul style="list-style-type: none"> • For 5th Generation Devices • Updated the following sections: <ul style="list-style-type: none"> • Querying the Device Configuration • mstconfig create_conf Command • Signing Binary Image Files • mstlink Usage • Supported Configurations and their Parameters: <ul style="list-style-type: none"> Added the following parameters: SW_RECOVERY_ON_ERRORS RESET_WITH_HOST_ON_ERRORS IBM_TUNNELED_ATOMIC_EN EXP_ROM_PXE_ENABLE EXP_ROM_UEFI_x86_ENABLE EXP_ROM_UEFI_ARM_ENABLE HOST_CHAINING_MODE HOST_CHAINING_DESCRIPTOR HOST_CHAINING_TOTAL_BUFFER_SIZE • Using Secure Host

Legal Notices and 3rd Party Licenses

The following are the drivers' software, tools and HCA firmware legal notices and 3rd party licenses.

Product	Version	Legal Notices and 3rd Party Licenses
Firmware	xx.39.1002	<ul style="list-style-type: none">• HCA Firmware EULA• 3rd Party Notice
MLNX_OFED	23.10	<ul style="list-style-type: none">• License• 3rd Part Notice
MFT FreeBSD	4.26.0	<ul style="list-style-type: none">• 3rd Party Notice• License
MFT Linux		<ul style="list-style-type: none">• 3rd Party Notice• License
MFT VMware		<ul style="list-style-type: none">• 3rd Party Notice• License
MFT Windows		<ul style="list-style-type: none">• 3rd Party Notice• License
msfflint	4.26.0	<ul style="list-style-type: none">• 3rd Party Notice• License

Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. Neither NVIDIA Corporation nor any of its direct or indirect subsidiaries and affiliates (collectively: "NVIDIA") make any representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice. Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer ("Terms of Sale"). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer's own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer's sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer's product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, "MATERIALS") ARE BEING PROVIDED "AS IS." NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks

NVIDIA, the NVIDIA logo, and Mellanox are trademarks and/or registered trademarks of NVIDIA Corporation and/or Mellanox Technologies Ltd. in the U.S. and in other countries. Other company and product names may be trademarks of the respective companies with which they are associated.



Copyright

© 2023 NVIDIA Corporation & affiliates. All Rights Reserved.

