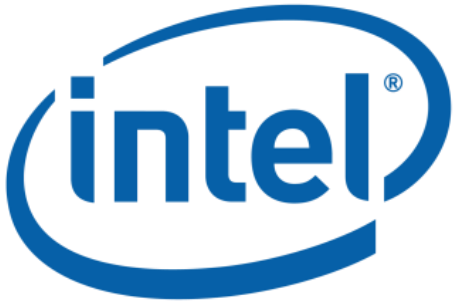

Intel® Xeon Phi™ Coprocessor

End User Software Configuration Tools User Guide

SKU# 328249-003
October, 2013

Table of Contents

1 Intel® Xeon Phi™ Coprocessor Check Utility: miccheck

1.1 Description

The Intel® Xeon Phi™ Coprocessor Check Utility (miccheck) checks the status of various aspects of one or more Intel® Xeon Phi™ Coprocessors on a single platform (node).

1.2 General Usage

This tool can be used by specifying various command line arguments, as follows:

```
/usr/bin/miccheck --help | -h
/usr/bin/miccheck --verbose | -v
/usr/bin/miccheck --device=0 | -d 0
/usr/bin/miccheck
```

1.3 Detailed Usage

1.3.1 General Commands

1.3.1.1 Help Command

Using this command displays the online tool help for miccheck.

1.3.1.1.1 Usage:

```
/usr/bin/miccheck -h
/usr/bin/miccheck --help
```

1.3.1.1.2 Arguments:

None.

1.3.1.2 Verbose Command

Using this command enables verbosity.

1.3.1.2.1 Usage:

```
/usr/bin/miccheck -v
/usr/bin/miccheck --verbose
```

1.3.1.2.2 Arguments:

None.

1.3.2 Available Tests

1.3.2.1 Default Tests

Table 1-1 lists the checks that are made if you run the miccheck tool without any arguments:

Table 1-1. miccheck Tests Run by Default

Test	Description
pci_numdev	Check whether Intel(R) Xeon Phi(TM) coprocessors are detected over PCI [enabled=True]
driver_loaded	Check whether Intel(R) Xeon Phi(TM) driver is loaded in the host [enabled=True]
driver_numdev	Check whether driver detected the same number of devices as PCI enumeration did [enabled=True]
mpssd_loaded	Check whether MPSS daemon is running [enabled=True]
driver_ver	Check whether loaded driver version is correct [enabled=False]
dev_state	Check whether device state is online and its postcode is FF [enabled=True]
dev_rasdaemon	Check whether RAS daemon is available in device [enabled=True]
flash_ver	Check whether running flash version of device is correct [enabled=True]

1.3.2.2 Additional / Optional Tests

Table 1-2 lists the tests that you can run individually.

Table 1-2. Additional and Optional miccheck Tests

Test	Description
Ping	Check whether network interface of device can be pinged [enabled=False]
ssh	Check whether network interface of device can be accessed through ssh [enabled=False]

2 Intel® Xeon Phi™ Coprocessor Flash Tool: micflash

This document is supplemental to the tool itself. Always check the tool's help for the latest and most accurate way to use this tool.

Note: This tool works only on the Intel® Xeon Phi™ Coprocessor.

2.1 Description

micflash is the Intel® Many Integrated Core flash tool. It is a simple to use command line tool that provides an interface for upgrading and viewing version information about the flash memory on the Intel® Xeon Phi™ coprocessors installed in a platform.

The primary purpose of this tool is to update both the flash and the system management control firmware; however, the tool can also be used to save and retrieve version information from the flash and system management control firmware.

Note : You must be logged in as root to use this tool. The examples presented may or may have not been taken from the same product that this document covers; therefore, the data displayed in this manual might not match the outputs the tool actually provides.

2.2 General Usage

The following commands show how the micflash utility is typically used. See the Detailed Usage sections below for specifics on each set of arguments.

- `sudo /usr/bin/micflash –help`
- `sudo /usr/bin/micflash –compatible <file> [{-device|-d} <deviceID>] [-silent] [-log <logfile>]`
- `sudo /usr/bin/micflash –devinfo [{-device|-d} <deviceID>] [-silent] [-log <logfile>]`
- `sudo /usr/bin/micflash –getversion {[<file>]|[{-device|-d} <deviceId>]} [-silent] [-log <logfile>]`
- `sudo /usr/bin/micflash –version`
- `sudo /usr/bin/micflash –save <file> [{-device|-d} <deviceID>] [-silent] [-log <logfile>]`
- `sudo /usr/bin/micflash –update [<path>] [{-device|-d} <deviceID>] [-v] [-vv] [-noDowngrade] [-noreboot] [-silent] [-log <logfile>] [-smcbootloader]`

If the deviceID is not specified, micflash will target device 0 by default on systems with a single Coprocessor, an error is reported on systems with multiple Corprocessors.

The –help directive can be used with any of the commands listed above to get help specific to the command and to limit the text output.

2.3 Detailed Usage

2.3.1 Compatible Command

The compatible command checks that a flash file is compatible for use on the specified coprocessor card.

2.3.1.1 Usage:

```
sudo /usr/bin/micflash –compatible <file> [{-device|-d} <deviceID>] [-silent]
[-log <logfile>]
```

2.3.1.2 Arguments:

file
Specifies the name and location of the file containing the flash for the coprocessor to be checked for compatibility.

deviceID
Specifies the ID number of the coprocessor to check compatibility of.

silent
Optional. If specified, the tool will suppress status messages. Errors will still be reported.

logfile
Optional. Specifies the name and location of the file into which the operation status is to be logged.

2.3.1.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micflash -compatible
/usr/share/mpss/flash/EXT_HP2_B0_0386-03.rom.smc -device 0
mic0: Image check: Valid image
```

2.3.2 DevInfo Command

The DevInfo command retrieves and displays the vendor ID and the device ID of a specific coprocessor on the platform.

To use this command, the Intel® Xeon Phi™ Coprocessor must be in ready mode. To do this, use the micctrl tool:

```
sudo micctrl –r –w
```

2.3.2.1 Usage:

```
/usr/bin/micflash –devinfo [{-device|-d} <deviceID>] [-silent] [-log
<logfile>]
```

2.3.2.2 Arguments:

deviceID
The zero-based ID of a coprocessor on the platform.

silent
Optional. If specified, the tool will suppress status messages. Errors will still be reported.

logfile
Optional. Specifies the name and location of the file into which the operation status is to be logged.

2.3.2.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micflash –devinfo –device 0
mic0: Flash: Atmel AT25DF161
mic0: Transitioning to ready state
```

2.3.3 GetVersion Command

The GetVersion retrieves the flash version information for a specified Intel® Xeon Phi™ Coprocessor on the platform.

To use this command, the Intel® Xeon Phi™ Coprocessor must be in ready mode. To do this, use the micctrl tool:

```
sudo micctrl –r -w
```

2.3.3.1 Usage:

```
micflash -getversion {[<file>]|[{-device|-d} <deviceId>]} [-silent] [-log
<logfile>]
```

2.3.3.2 Arguments:

file
Specifies the name of the image file. If the file is specified, the version is read from this file. If this file is not specified, the version is taken from the Intel® Xeon Phi™ Coprocessor specified in the deviceID argument.

deviceID
Specifies which device to operate on. This argument is optional if only one Mic device is present; however, it must be used with the –device directive.

silent
Optional. If specified, the tool will suppress status messages. Errors will still be reported.

logfile
Optional. Specifies the name and location of the file into which the operation status is to be logged.

2.3.3.3 Examples:

```
[mic@localhost bin]$ sudo /usr/bin/micflash –getversion –device 0
mic0: Flash read started
mic0: Read done
mic0: Version: 2.1.03.0386
mic0: Transitioning to ready state
```

2.3.4 Update Command

This command is used to update the specified Intel® Xeon Phi™ Coprocessor on the platform with the specified flash image file.

To use this command, the Intel® Xeon Phi™ Coprocessor must be in ready mode. To do this, use the micctrl tool:

```
sudo micctrl –r -w
```

2.3.4.1 Usage:

```
micflash –update [<path>] [{-device|-d} <deviceID>] [-v] [-vv] [-noDowngrade]
[-noreboot] [-silent] [-log <logfile>] [-smcbootloader]
```

2.3.4.2 Arguments and Commands:

path
Optional. Can be a regular file name or a directory path, in which case micflash looks for a compatible image inside the specified directory. If this argument is not specified, micflash will look for a compatible image in the /opt/intel/mic/flash directory.

deviceID
Specifies which device to operate on. This command is optional if only one Mic device is present; however, it must be specified with the –device directive.

v
Optional. If specified, the tool displays verbose status messages.

vv
Optional. If specified, the tool displays progress with a rotating-bar progress monitor.

noDownGrade
Optional. If specified, the tool will not allow downgrade of the flash software.

noreboot
Optional. If specified, the tool will not display a reboot message when the flash update has successfully completed.

silent
Optional. If specified, the tool will suppress status messages. Errors will still be reported.

logfile
Optional. Specifies the name and location of the file into which the operation status is to be logged.

smcbootloader

Optional. If specified, the tool will update the SMC bootloader before performing a flash update operation.

2.3.4.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micflash –update
No image path specified – Searching: /usr/share/mpss/flash
mic0: Flash image: /usr/share/mpss/flash/EXT_HP2_B0_0386-03.rom.smc
mic0: Flash update started
mic0: Flash update done
mic0: SMC update started
mic0: SMC update done
mic0: Transitioning to ready state

Please restart host for flash changes to take effect
```

2.3.5 Save Command

This command saves the Intel® Xeon Phi™ Coprocessor flash image to a specified file name in binary format.

To use this command, the Intel® Xeon Phi™ Coprocessor must be in ready mode. To do this, use the micctrl tool:

```
sudo micctrl –r -w
```

2.3.5.1 Usage:

```
/usr/bin/micflash –save <file> [{-device|-d} <deviceID>] [-silent] [-log
<logfile>]
```

2.3.5.2 Arguments:

file
The path and name of the image file to save.

deviceID
Specifies which device to operate on. This argument is optional if only one Mic device is present; however, it must be specified with the –device directive.

silent
Optional. If specified, the tool will suppress status messages. Errors will still be reported.

logfile
Optional. Specifies the name and location of the file into which the operation status is to be logged.

2.3.5.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micflash –save myFlashImage.rom -device 0
mic0: Flash read started
mic0: Read done
mic0: Transitioning to ready state
```

3 Intel® Xeon Phi™ Coprocessor Information Tool: micinfo

This documentation is specific to the tool features that are only available on the Intel® Xeon Phi™ Coprocessor.

This document is supplemental to the tool itself. Always check the tool's help for the latest and most accurate way to use this tool.

3.1 Description

The micinfo utility is the Many Integrated Core basic information tool. It is a simple to use tool that retrieves information from the host platform and the Intel® Xeon Phi™ Coprocessors installed in the platform.

NOTE: The level of this tool's output is tied to the privilege level of the user. Users with root privileges can view all information. Non-root users can only see a subset of all the data that is available.

The examples presented here may or may have not been taken from the same product; therefore, the data displayed in this manual may not match the outputs the tools actually provide.

3.2 General Usage

The micinfo tool can be used in the following ways. See the detailed usage sections for specifics.

```
/usr/bin/micinfo
/usr/bin/micinfo -help
/usr/bin/micinfo –listDevices
/usr/bin/micinfo –version
/usr/bin/micinfo –deviceInfo <deviceID> [-group] [groupname]
```

Executing /usr/bin/micinfo without any arguments will display all data for all command groups.

3.3 Detailed Usage

3.3.1 Running with no arguments

Running micinfo with no arguments will, in most cases, output all groups of data as described below. It will display information about the host platform (which does not have an individual group command).

If Intel® MPSS is not started; then some data will be listed as not being available.

Any user can run micinfo; however, running it without root privileges, some items, such as PCI-e data (bus speed, link, width, etc.), will not be available.

3.3.2 listDevices Command

This command lists the Intel® Xeon Phi™ Coprocessors found on the platform.

3.3.2.1 Usage:

```
/usr/bin/micinfo –listDevices
```

3.3.2.2 Arguments:

None.

3.3.2.3 Example:

The following example lists out all of the Intel® Xeon Phi™ Coprocessors found on the platform (in this case, just one devices was found with a deviceID of 0)

```
[mic@localhost bin]$ sudo /usr/bin/micinfo -listdevices

MicInfo Utility Log

Created Sat Oct  5 19:46:01 2013


List of Available Devices

deviceId |  domain  | bus# | pciDev# | hardwareId
---------|----------|------|---------|-----------
       0 |        0 |    2 |       0 |   225C8086
-------------------------------------------------
```

3.3.3 version Command

This command lists the tool version information.

3.3.3.1 Usage:

```
/usr/bin/micinfo –version
```

3.3.3.2 Arguments:

None.

3.3.3.3 Example:

The following example lists out all of the Intel® Xeon Phi™ Coprocessors found on the platform (in this case, just one devices was found with a deviceID of 0)

```
[mic@localhost bin]$ sudo /usr/bin/micinfo –version
MicInfo Utility Log

Created Sat Oct  5 19:47:06 2013
```

```
VERSION: 3.1-0.1.build0
Copyright 2011-2013 Intel Corporation All Rights Reserved.
```

3.3.4 Groups

All group names must be used in conjunction with the –deviceInfo directive and the deviceID of the Intel® Xeon Phi™ Coprocessor in order to retrieve group data:

```
/usr/bin/micinfo –deviceInfo <deviceID> [-group] [groupname]
```

The desired deviceID value can be retrieved using the –listDevices directive:

```
/usr/bin/micinfo –listDevices
```

The following groups are available:

- Versions
- Board
- Core
- Thermal
- GDDR

3.3.4.1 Versions group

Use the Versions group to get version information from the specified Intel® Xeon Phi™ Coprocessors on the platform.

3.3.4.1.1 Usage:

```
/usr/bin/micinfo –deviceInfo <deviceID> -group Versions
```

3.3.4.1.2 Arguments:

deviceID
Specifies which Intel® Xeon Phi™ Coprocessor to use. This argument is optional in a single coprocessor setup, but is required in a multiple coprocessor setup.

- 0 = first Intel® Xeon Phi™ Coprocessor.
- 1 = second Intel® Xeon Phi™ Coprocessor, and so on.

The DeviceID will default to the first Intel® Xeon Phi™ Coprocessor if "-1" is passed on a single Intel® Xeon Phi™ Coprocessor setup.

3.3.4.1.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micinfo -deviceInfo 0 -group Versions
MicInfo Utility Log

Created Sat Oct  5 19:51:31 2013
```

```
        System Info
                HOST OS                 : Linux
                OS Version              : 2.6.32-131.0.15.el6.x86_64
                Driver Version          : 3.1-0.1.build0
                MPSS Version            : 3.1
                Host Physical Memory    : 1519 MB

Device No: 0, Device Name: mic0

        Version
                Flash Version           : 2.1.03.0386
                SMC Firmware Version    : 1.15.4830
                SMC Boot Loader Version : 1.8.4326
                uOS Version             : 2.6.38.8+mpss3.1
                Device Serial Number    : ADKC22900203
```

3.3.4.2 Board Group

Use the Board group to get board information from the specified Intel® Xeon Phi™ Coprocessor on the platform.

3.3.4.2.1 Usage:

```
/usr/bin/micinfo –deviceInfo <deviceID> -group Board
```

3.3.4.2.2 Arguments:

deviceID
Specifies which Intel® Xeon Phi™ Coprocessor to use. This argument is optional in a single coprocessor setup, but is required in a multiple coprocessor setup.

- 0 = first Intel® Xeon Phi™ Coprocessor.
- 1 = second Intel® Xeon Phi™ Coprocessor, and so on.

The DeviceID will default to the first Intel® Xeon Phi™ Coprocessor if "-1" is passed on a single Intel® Xeon Phi™ Coprocessor setup.

3.3.4.2.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micinfo -deviceInfo 0 -group Board
MicInfo Utility Log

Created Sat Oct  5 19:53:05 2013


        System Info
                HOST OS                 : Linux
                OS Version              : 2.6.32-131.0.15.el6.x86_64
                Driver Version          : 3.1-0.1.build0
                MPSS Version            : 3.1
                Host Physical Memory    : 1519 MB
```

```
Device No: 0, Device Name: mic0

        Board
                Vendor ID               : 0x8086
                Device ID               : 0x225c
                Subsystem ID            : 0x2500
                Coprocessor Stepping ID : 1
                PCIe Width              : x16
                PCIe Speed              : 5 GT/s
                PCIe Max payload size   : 128 bytes
                PCIe Max read req size  : 512 bytes
                Coprocessor Model       : 0x01
                Coprocessor Model Ext   : 0x00
                Coprocessor Type        : 0x00
                Coprocessor Family      : 0x0b
                Coprocessor Family Ext  : 0x00
                Coprocessor Stepping    : B0
                Board SKU               : ES2-P/A/X 1750
                ECC Mode                : Enabled
                SMC HW Revision         : Product 300W Active CS
```

3.3.4.3 Core Group

Use the Core group to get core information from the specified Intel® Xeon Phi™ Coprocessors on the platform.

3.3.4.3.1 Usage:

/usr/bin/micinfo –deviceInfo <deviceID> -group Core

3.3.4.3.2 Arguments:

deviceID
Specifies which Intel® Xeon Phi™ Coprocessor to use. This argument is optional in a single coprocessor setup, but is required in a multiple coprocessor setup.

- 0 = first Intel® Xeon Phi™ Coprocessor.
- 1 = second Intel® Xeon Phi™ Coprocessor, and so on.

The DeviceID will default to the first Intel® Xeon Phi™ Coprocessor if "-1" is passed on a single Intel® Xeon Phi™ Coprocessor setup.

3.3.4.3.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micinfo -deviceInfo 0 -group Core
MicInfo Utility Log

Created Sat Oct  5 19:54:25 2013


        System Info
                HOST OS                 : Linux
                OS Version              : 2.6.32-131.0.15.el6.x86_64
                Driver Version          : 3.1-0.1.build0
                MPSS Version            : 3.1
```

```
                Host Physical Memory   : 1519 MB

Device No: 0, Device Name: mic0

        Cores
                Total No of Active Cores : 61
                Voltage                  : 999000 uV
                Frequency                : 1090909 kHz
```

3.3.4.4 Thermal Group

Use the Thermal group to get thermal information from the specified Intel® Xeon Phi™ Coprocessors on the platform.

3.3.4.4.1 Usage:

```
/usr/bin/micinfo –deviceInfo <deviceID> -group Thermal
```

3.3.4.4.2 Arguments:

deviceID
Specifies which Intel® Xeon Phi™ Coprocessor to use. This argument is optional in a single coprocessor setup, but is required in a multiple coprocessor setup.

- 0 = first Intel® Xeon Phi™ Coprocessor.
- 1 = second Intel® Xeon Phi™ Coprocessor, and so on.

The DeviceID will default to the first Intel® Xeon Phi™ Coprocessor if "-1" is passed on a single Intel® Xeon Phi™ Coprocessor setup.

3.3.4.4.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micinfo -deviceInfo 0 -group Thermal
MicInfo Utility Log

Created Sat Oct  5 19:55:08 2013


        System Info
                HOST OS                 : Linux
                OS Version              : 2.6.32-131.0.15.el6.x86_64
                Driver Version          : 3.1-0.1.build0
                MPSS Version            : 3.1
                Host Physical Memory    : 1519 MB

Device No: 0, Device Name: mic0

        Thermal
                Fan Speed Control       : On
                Fan RPM                 : 3300
                Fan PWM                 : 60
                Die Temp                : 46 C
Device No: 0, Device Name: mic0

         Thermal
                 Fan Speed Control        : On
```

```
        Fan RPM                 : 900
        Fan PWM                 : 20
        Die Temp                : 46 C
```

3.3.4.5 GDDR Group

Use the GDDR group to get GDDR information from the specified Intel® Xeon Phi™ Coprocessors on the platform.

3.3.4.5.1 Usage:

```
/usr/bin/micinfo –deviceInfo <deviceID> -group GDDR
```

3.3.4.5.2 Arguments:

deviceID
Specifies which Intel® Xeon Phi™ Coprocessor to use. This argument is optional in a single coprocessor setup, but is required in a multiple coprocessor setup.

- 0 = first Intel® Xeon Phi™ Coprocessor.
- 1 = second Intel® Xeon Phi™ Coprocessor, and so on.

The DeviceID will default to the first Intel® Xeon Phi™ Coprocessor if "-1" is passed on a single Intel® Xeon Phi™ Coprocessor setup.

3.3.4.5.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micinfo -deviceInfo 0 -group GDDR
MicInfo Utility Log

Created Sat Oct  5 19:56:46 2013


        System Info
                HOST OS                 : Linux
                OS Version              : 2.6.32-131.0.15.el6.x86_64
                Driver Version          : 3.1-0.1.build0
                MPSS Version            : 3.1
                Host Physical Memory    : 1519 MB

Device No: 0, Device Name: mic0

        GDDR
                GDDR Vendor             : Elpida
                GDDR Version            : 0x1
                GDDR Density            : 2048 Mb
                GDDR Size               : 7936 MB
                GDDR Technology         : GDDR5
                GDDR Speed              : 5.500000 GT/s
                GDDR Frequency          : 2750000 kHz
                GDDR Voltage            : 1501000 uV
```

4 Intel® Xeon Phi™ Coprocessor RAS Daemon Tool: micrasd

This document is supplemental to the tool itself. Always check the tool's help for the latest and most accurate way to use this tool.

Note: This tool works only on the Intel® Xeon Phi™ Coprocessor.

4.1 Description

The micrasd tool is an application that runs on the host platform that handles and logs Intel® Xeon Phi™ Coprocessor errors. It can run in a single use mode or in a daemon mode.

The micrasd tool can also be started and stopped as a Linux system service, as shown below:

```
sudo service micras start
sudo service micras stop
```

The micras service is dependent on the Intel® MPSS service running; micras cannot be started until after Intel® MPSS successfully starts, and micras must be stopped before Intel® MPSS is stopped.

The Linux chkconfig command specifies which services are automatically started on system boot. To automatically start the micras service, use the following command:

```
sudo chkconfig micras on
```

Or, you can use "off" as an argument if you do not want to start micras automatically when the host boots. The errors will be logged into Linux syslog. You can locate the errors under syslog */var/log/messages* with the "micras" tag.

Notes: You must have root privileges to run this tool.
Only one instance of micrasd is permitted to execute at any given time on the node.

4.2 General Usage

The micrasd tool can be used in the following ways. See the subsequent detailed usage sections for specifics.

```
/usr/bin/micrasd –help
/usr/bin/micrasd -daemon
/usr/bin/micrasd –loglevel <level>
/usr/bin/micrasd -maint
```

4.3 Detailed Usage

4.3.1 Running micras Without Arguments

Running micrasd with no arguments will display a listing of the current issues that are present on any Intel® Xeon Phi™ Coprocessor s in the host platform, and will continue to print out events as they happen.

4.3.2 -daemon Command

This command is used to instruct micrasd to run in daemon mode. In this mode, the utility will run continuously in the backgroud and monitor/log errors.

4.3.2.1 Usage:

/usr/bin/micrasd -daemon

4.3.2.2 Arguments:

None.

4.3.2.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micrasd -daemon
Enable daemon mode.
Finish parsing options.
Sat Oct  5 20:19:59 2013 MICRAS INFO    : ./micrasd: Running in Daemon Mode.
```

4.3.3 -loglevel Command

This command sets the level of detail that gets logged with the micrasd tool.

4.3.3.1 Usage:

```
/usr/bin/micrasd –loglevel <level>
```

4.3.3.2 Arguments:

level
Sets the level of logging to perform. The accepted levels are from 1 to 15. It is a 3 bit representation, where bits 0 – 3 mean the following:

- Bit 0 – Enables Informational Messages
- Bit 1 – Enables Warning Messages
- Bit 2 – Enables Error Message
- Bit 3 – Enables Critical Message

4.3.3.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micrasd -loglevel 7
Log level set to be 7.
Finish parsing options.
```

```
MicRAS already running!
```

4.3.4 Maintenance Mode

With Maintenance mode enabled, upon detection of any uncorrectable errors, micrasd forces the card into Maintenance test and repair mode.

4.3.4.1 Usage:

```
/usr/bin/micrasd –maint
```

4.3.4.2 Arguments:

None.

4.3.4.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/micrasd –maint
Enable Maintenance mode test and repair.
Finish parsing options.
MicRAS already running!
```

5 Intel(R) Xeon Phi(TM) Products Reliability Monitor: crashmgr

This documentation is specific to the tool features that are only available on the Intel® Xeon Phi™ Coprocessor.

This document is supplemental to the tool itself. Always check the tool's help for the latest and most accurate way to use this tool.

5.1 Description

The Intel® Xeon Phi™ Products Reliability Monitor (refered to as crash manager in the rest of this doc) is designed to monitor the overall health of Intel® Xeon Phi™ on a cluster level. It runs on the head node, or management node and monitors the status of the compute nodes as well as collects error information including hardware errors (crash symptoms will be reported to and logged by crashmgr). This utility can also run as a Linux* service in the background and the user can issue commands to query node status and error information.

5.2 Intel(R) Xeon Phi(TM) Products Reliability Monitor as a Service

To have this utility run as a Linux* service daemon in the background, the mpss and micras services need to be up and running on each compute node.

The following commands will start the mpss and micras services:

```
sudo service mpss start
sudo service micras start
```

After installing the Intel® MPSS Crash Manager RPM on the head node, start the service as follows:

```
sudo service crashmgr start
```

This service can be stopped at any time by issuing the following command on the head node:

```
sudo service crashmgr stop
```

To check the status of this service from the head node, use:

```
sudo service crashmgr status
```

5.3 General Usage

The crashmgr utility can be used in the following ways. See the detailed usage sections for specifics.

```
/usr/bin/crashmgr --help
/usr/bin/crashmgr –-daemon
```

```
/usr/bin/crashmgr --cmd <command options> [--node] [node_number] [--type]
[node_type] [--file] [file_name] [--loglevel] [log_level]
```

5.4 Detailed Usage

5.4.1 daemon Command

This command instructs the utility to run in daemon mode. The user must have root privileges to use this command.

5.4.1.1 Usage:

```
/usr/bin/crashmgr --daemon
```

5.4.1.2 Arguments:

None.

5.4.1.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/crashmgr --daemon

Enable daemon mode.
Finish parsing options.
Mon Oct  7 12:23:53 2013 CM INFO    : Parsing Config File
Mon Oct  7 12:23:53 2013 CM INFO    : Parsing config file: SUCCESS! Total
Available Nodes : 3
Mon Oct  7 12:23:54 2013 CM INFO    : MCA filter log file not existed, create
a new filter log file.
Mon Oct  7 12:23:54 2013 CM INFO    : ./crashmgr: Running in Daemon Mode.
Crash manager is now running in the background!
```

5.4.2 cmd Command

Once the crash manager daemon or service is up and running, the '--cmd' command can be used to send commands to the crash manager daemon to peek at the information or control the daemon.

5.4.2.1 shownode Option:

The shownode command is used to peek at the current status of the compute node.

KNOWN ISSUE: Currently all the MIC card on a same host and the host machine share the same status as they are bound together. This means that if mic0 on a Host machine got an MCA error and entered into Maintenance state, the Host machine and all cards installed on the Host will show the status as "Diagnostics". This will be changed in a future release.

5.4.2.1.1 Usage:

```
/usr/bin/crashmgr --cmd shownode
```

5.4.2.1.2 Arguments:

None.

5.4.2.1.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/ crashmgr --cmd shownode
Finish parsing options.
Mon Oct  7 12:25:03 2013 CM INFO    : Output File located at : /usr/bin
MESSAGE FROM DAEMON:
The List of Nodes:
DevId: 0 ParentId: 0 Mic_Num: 1 Port: 898 IpAddr(Node):   192.168.0.100 IpAddr(Host):
192.168.0.100 Status: Spare
DevId: 1 ParentId: 0 Mic_Num: 1 Port: 898 IpAddr(Node):     192.168.0.2 IpAddr(Host):
192.168.0.100 Status: Spare
DevId: 2 ParentId: 2 Mic_Num: 1 Port: 898 IpAddr(Node):   192.168.0.101 IpAddr(Host):
192.168.0.101 Status: Spare
DevId: 3 ParentId: 3 Mic_Num: 1 Port: 898 IpAddr(Node):   192.168.0.102 IpAddr(Host):
192.168.0.102 Status: Spare
DevId: 4 ParentId: 3 Mic_Num: 1 Port: 898 IpAddr(Node):     192.168.0.4 IpAddr(Host):
192.168.0.102 Status: Spare
```

Where node status is one of the following:

Spare: node is up and running, this node is not allocated to any job launched through Crash Manager

Running: node is up and running, this node is executing a job launched through Crash Manager

Link Lost: Crash Manger losts communication with the node, specially when the MicRAS daemon is not running on the host

Resetting: the node is in resetting state

MCA: the node gets a machine check error

Diagnostics: the node is in diagnostics state

Unreachable: when the Xeon Phi card is in the ready state

5.4.2.2 showerr Option:

The showerr command is used to show the errors reported from the nodes.

5.4.2.2.1 Usage:

```
/usr/bin/crashmgr --cmd showerr
```

5.4.2.2.2 Arguments:

None.

5.4.2.2.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/ crashmgr --cmd showerr
Finish parsing options.
Mon Oct  7 12:29:18 2013 CM INFO    : Output File located at : /usr/bin
MESSAGE FROM DAEMON:
-------------------------------------------------------------------------------------------
-----------------------
HostId MicId Org Id  Flag       Ctrl            Status          Addr            Misc
Timestamp
```

```
--------------------------------------------------------------------------------
------------------------
No error found!
```

5.4.2.3 loglevel Option:

This command sets the level of detail that gets logged with the crashmgr tool.

5.4.2.3.1 Usage:

```
/usr/bin/crashmgr –loglevel <level>
```

5.4.2.3.2 Arguments:

level
Sets the level of logging to perform. The accepted levels are from 1 to 7. It is a 2 bit representation, where bits 0 – 2 mean the following:

- Bit 0 – Enables Informational Messages
- Bit 1 – Enables Warning Messages
- Bit 2 – Enables Error Message

5.4.2.3.3 Example:

```
[mic@localhost bin]$ sudo /usr/bin/ crashmgr –loglevel 7
Log level set to be 7.
Finish parsing options.
Thu Oct 10 08:01:44 2013 CM INFO    : Parsing Config File
Thu Oct 10 08:01:44 2013 CM INFO    : Parsing config file: SUCCESS! Total
Available Nodes : 3
Thu Oct 10 08:01:44 2013 CM INFO    : Open CrashMgr MCA filter log history.
CrashMgr daemon already running! please use --cmd to send commands to the
daemon.
```