



ioMemory VSL 3.2.6
USER GUIDE FOR LINUX

APRIL 04, 2014



Table of Contents

| | |
|--|-----------|
| Table of Contents | 2 |
| ioMemory VSL 3.2.6 User Guide for Linux | 5 |
| Legal Notices | 5 |
| Introduction | 6 |
| Overview | 6 |
| About the ioMemory Platform | 6 |
| Software Installation | 8 |
| Installation Overview | 8 |
| Installing RPM Packages | 9 |
| Installing DEB Packages | 12 |
| Loading the ioMemory VSL Driver | 15 |
| Upgrading the Firmware | 18 |
| Configuration | 20 |
| Setting the ioMemory VSL Options | 20 |
| Enabling PCIe Power Override | 20 |
| Virtual Controller Configuration | 23 |
| Using the Device as Swap | 25 |
| Using the Logical Volume Manager | 26 |
| Configuring RAID Using mdadm | 28 |
| Discard (TRIM) Support | 31 |
| Performance and Tuning | 32 |
| Disable CPU Frequency Scaling | 32 |
| Limiting ACPI C-States | 32 |



| | |
|--|-----------|
| Setting NUMA Affinity | 33 |
| Setting the Interrupt Handler Affinity | 33 |
| Monitoring and Managing Devices | 35 |
| Management Tools | 35 |
| Example Conditions to Monitor | 36 |
| Device LED Indicators | 38 |
| Maintenance | 40 |
| Uninstalling the Software | 40 |
| Unloading the Software Driver | 41 |
| Upgrading the Kernel | 41 |
| Disabling the ioMemory VSL Software | 41 |
| Disabling Auto-Attach | 42 |
| Unmanaged Shutdown Issues | 42 |
| Appendix A - Command-line Utilities Reference | 46 |
| fio-attach | 46 |
| fio-beacon | 47 |
| fio-bugreport | 48 |
| fio-detach | 49 |
| fio-format | 50 |
| fio-pci-check | 51 |
| fio-status | 52 |
| fio-sure-erase | 54 |
| fio-update-iodrive | 56 |
| Appendix B - Monitoring the Health of Devices | 59 |
| Health Metrics | 59 |
| Health Monitoring Techniques | 59 |
| Software RAID and Health Monitoring | 60 |
| Appendix C - Using Module Parameters | 61 |



| | |
|---|-----------|
| Appendix D - NUMA Configuration | 63 |
| About NUMA Architecture | 63 |
| Using the numa_node_forced_local Parameter | 63 |
| Using the numa_node_override Parameter | 63 |
| Appendix E - Upgrading Devices from VSL 2.x to 3.x | 66 |
| Upgrade Procedure | 66 |
| Fusion Powered Support | 71 |
| E-Mail | 71 |
| Warranty Support | 71 |
| Telephone Support | 71 |
| Web | 71 |



ioMemory VSL 3.2.6 User Guide for Linux

Legal Notices

The information contained in this document is subject to change without notice.

Copyright © 2014 Fusion-io, Inc. All rights reserved.

Fusion-io, the Fusion-io logo, ioMemory, Virtual Storage Layer, VSL, ioFX, ioSphere, ioScale, and ioDrive are registered trademarks, and Adaptive Flashback and Sure Erase are trademarks of Fusion-io, Inc. in the United States and other countries.

The names of other organizations and products referenced herein are the trademarks or service marks (as applicable) of their respective owners. Unless otherwise stated herein, no association with any other organization or product referenced herein is intended or should be inferred.

You may not use the ioMemory VSL software for any competitive benchmarking purpose without prior written consent from Fusion-io. You may not publish any performance data related to the ioMemory VSL software without prior written consent from Fusion-io.

Fusion-io
2855 E. Cottonwood Parkway, Box 100
Salt Lake City, UT 84121
USA

(801) 424-5500

Part Number: D0001657-025_2


Published: April 04, 2014




Introduction

Overview

Congratulations on your purchase of a Fusion-io solid-state storage device. This guide explains how to install, troubleshoot, and maintain the software for your ioMemory devices.

 Throughout this manual, when you see a reference to an **ioMemory device**, you may substitute your particular device(s), such as an ioDrive2 device, ioScale device, or each of the two ioMemory devices of an ioDrive Duo device.

 **Products with Multiple Devices**
Some products, such as an ioDrive Duo device, are actually comprised of **multiple ioMemory devices**. If your product consists of multiple ioMemory devices, you will manage each ioMemory device as an independent device.

For example, if you have an ioDrive Duo device, you can independently attach, detach, and/or format each of the two ioMemory devices. Each of the two devices will be presented as an individual device to your system.

About the ioMemory Platform

The ioMemory platform combines ioMemory VSL software (VSL stand for Virtual Storage Layer) with ioMemory hardware to take enterprise applications and databases to the next level.

Performance

The ioMemory platform provides consistent microsecond latency access for mixed workloads, multiple gigabytes per second access and hundreds of thousands of IOPS from a single product. The sophisticated ioMemory architecture allows for nearly symmetrical read and write performance with best-in-class low queue depth performance, making the ioMemory platform ideal across a wide variety of real world, high-performance enterprise environments.

The ioMemory platform integrates with host system CPUs as flash memory to give multiple (and mostly idle) processor cores, direct and parallel access to the flash. The platform's cut-through architecture gives systems more work per unit of processing, and continues to deliver performance increases as CPU power increases.

Endurance

The ioMemory platform offers best-in-class endurance in all capacities, which is crucial for caching and write-heavy databases and applications.

Reliability

The ioMemory platform eliminates concerns about reliability like NAND failures and excessive wear. The all-new intelligent, self-healing feature called Adaptive Flashback provides complete, chip-level fault tolerance. Adaptive



Flashback technology enables an ioMemory product to repair itself after a single chip or a multi-chip failure without interrupting business continuity.



Software Installation

Before continuing with the installation of this software, please read the following:

1. Ensure that your operating system is included in the list of **supported operating systems** contained in the *ioMemory VSL Release Notes* for this release.
2. Before installing the ioMemory VSL software, make sure you have properly installed the ioMemory device(s). Refer to the *ioMemory Hardware Installation Guide* for full details and hardware requirements.



Every ioMemory device in a system must be upgraded to the appropriate firmware.

For example, if you have a system running ioMemory VSL software version 2.2.3 with ioMemory devices previously installed, and you want to install new ioDrive2 devices (that require the latest version of the firmware), then you will need to upgrade all of the existing devices with firmware that supports this version of the ioMemory VSL software. Follow the upgrade path in the *ioMemory VSL Release Notes* to determine the upgrade sequence.



Upgrade Previous Devices First

If you have ioMemory devices configured for ioMemory VSL software version 2.x or earlier, you must upgrade their firmware before installing new devices in the system. See [Upgrading Devices from VSL 2.x to 3.x on page 66](#) for the upgrade instructions.



All commands require administrator privileges. Use `sudo` or log in as "root" to run the install.

Installation Overview

1. Download the latest version of the software at <http://support.fusionio.com>.
2. If you are installing this version of ioMemory VSL software on a system with ioDrive devices configured with firmware for ioMemory VSL software version 2.x, you must carefully follow the instructions in the [Upgrading Devices from VSL 2.x to 3.x on page 66](#). (Follow those instructions instead of the normal installation instructions.)
3. If you have a previous version of the ioMemory VSL software installed, you will need to uninstall the ioMemory VSL software and the utilities. See [Uninstalling the Software on page 40](#) for instructions. Once you have uninstalled the packages, return to this page.



Kernel Upgrades

If you ever plan to upgrade the kernel when the ioMemory VSL software is installed, you **must** follow the procedure for [Upgrading the Kernel on page 41](#).

4. Install the latest version of the ioMemory VSL software. Depending on the format your distribution uses for installation packages, you will need to follow the instructions on [Installing RPM Packages on page 9](#) or [Installing](#)



[DEB Packages on page 12](#). The installation instructions will help you determine which of these package types is supported by your kernel:

- A pre-compiled binary package
 - A source-to-build package
5. Install utilities and management software (included in driver installation instructions).
 6. Follow the instructions on [Loading the ioMemory VSL Driver on page 15](#) and consider [Setting the ioMemory VSL Options on page 20](#).
 7. Determine if you need to upgrade the firmware to the latest version, see [Upgrading the Firmware on page 18](#).

Installing RPM Packages

To install the Linux ioMemory VSL software and utilities on SUSE, RHEL, Fedora, and CentOS:

1. You will need to install a version of the ioMemory VSL software that is built for your kernel. To determine what kernel version is running on your system, use the following command at a shell prompt:


```
$ uname -r
```

2. Compare your kernel version with the binary versions of the software available at <http://support.fusionio.com> .
 - If there is a binary version of the software that corresponds to your kernel version, download that. **For example:**

```
iomemory-vsl-<kernel-version>-<VSL-version>.x86_64.rpm
```

- If there is no binary version of the software corresponding to your kernel, download the source package. **For example:**

```
iomemory-vsl-<VSL-version>.src.rpm
```

 Exact package names may vary, depending on software and kernel version chosen.


Use the source package that is made for your distribution. Source packages from other distributions are not guaranteed to work.

3. Download the support RPM packages you need from <http://support.fusionio.com> . These packages provide utilities, firmware, and other files.


Examples:

| Package | What Is Installed |
|-----------------------------------|---|
| fio-util-<VSL-version>.x86_64.rpm | ioMemory VSL utilities – Recommended |



| | |
|--|---|
| fio-firmware- <version>.<date>- 1.0.noarch.rpm | Firmware archive package, installs the firmware archive file in a specific location (/usr/share/fio/firmware) – Optional |
| fusion_ <version>.<date> .fff | Firmware archive file (not an installer package) used to upgrade the firmware, make note of where you store this file – Recommended |
| libvsl- <version>.x86_ 64.rpm | SDK libraries needed for management tools – Recommended , see Monitoring and Managing Devices on page 35 for more information on available management tools. |
| lib32vsl- <version>.i386.rpm | SDK libraries needed for working with 32-bit management applications – Optional and only available for certain distributions. <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> You must have a full set of 32-bit system libraries installed (32-bit compatibility layer installed on your 64-bit system) before you can install this package.</div> |
| fio-common-<VSL- version>.x86_64.rpm | Files required for the init script – Recommended |
| fio-sysvinit-<VSL- version>.x86_64.rpm | Init script – Recommended , see Loading the ioMemory VSL Driver on page 15 for more information. |

4. Change to the directory to where you downloaded the installation packages.
5. If needed, build the ioMemory VSL software from source:
 - **If you downloaded a binary version of the software:** skip the **Building the Software from Source** instructions below and continue to the next step.
 - **If you downloaded the software source package:** follow these **Building the Software from Source** instructions:

| Building the Software from Source |
|--|
| <p>You only need to follow these additional instructions if you downloaded the source package.</p> <ol style="list-style-type: none"> a. Install the prerequisite files for your kernel version. <div style="border: 1px solid #007bff; padding: 10px; margin: 10px 0;"> <p> Some of the prerequisite packages may already be in the default OS installation. If your system is not configured to get packages over the network, then you may need to mount your install CD/DVD.</p> </div> <ul style="list-style-type: none"> ◦ On RHEL/CentOS 5/6, you need kernel-devel, kernel-headers, rpm-build, GCC4, and rsync. <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <pre>\$ yum install kernel-headers-`uname -r` kernel-devel-`uname -r` gcc rsync rpm-build make</pre> </div> <p>This command will force yum to download the exact versions for your kernel. If the</p> |



exact versions are no longer available in the repository, then you will have to manually search for them on the Internet.

- On SLES 10/11 you need `kernel-syms`, `make`, `rpm-build`, `GCC4`, and `rsync`

```
$ zypper install kernel-syms make rpm gcc rsync
```

- To build an RPM installation package for the current kernel, navigate to the directory with the downloaded source RPM file and run this command:

```
$ rpmbuild --rebuild iomemory-vsl-<VSL-version>.src.rpm
```



If your kernel is an Unbreakable Enterprise Kernel (UEK), you may also need to use the `--nodeps` option.

When using a `.rpm` source package for a non-running kernel, run this command:

```
$ rpmbuild --rebuild --define 'rpm_kernel_version <kernel-version>' iomemory-vsl-<VSL-version>.src.rpm
```

- The new RPM package is located in a directory that is indicated in the output from the `rpmbuild` command. To find it, look for the `Wrote` line. In the following example, the RPM packages are located in the `/usr/src/redhat/RPMS/x86_64/` directory.

```
...
Processing files: iomemory-vsl-source-<version>-1.0.x86_64.rpm
Requires(rpmlib): rpmlib(PayloadFilesHavePrefix) <= 4.0-1
rpmlib(CompressedFileNames) <= 3.0.4-1
Obsoletes: iodrive-driver-source
Checking for unpackaged file(s): /usr/lib/rpm/check-files
/var/tmp/iomemory-vsl-<version>-root
Wrote: /usr/src/redhat/RPMS/x86_64/iomemory-vsl-2.6.18-128.el5-<version>-1.0.x86_64.rpm
Wrote: /usr/src/redhat/RPMS/x86_64/iomemory-vsl-source-<version>-1.0.x86_64.rpm
```

In this example, `iomemory-vsl-2.6.18-128.el5-<version>-1.0.x86_64.rpm` is the package you will use.

- Copy your custom-built software installation RPM package into the directory where you downloaded the installation packages and navigate to that directory.
- Continue to the next step.

- Enter the following command to install the custom-built software package. Use the package name that you just copied/downloaded into that directory.



```
$ rpm -Uvh iomemory-vsl-<kernel-version>-<VSL-version>.x86_64.rpm
```

7. Enter the following commands to install the support files:

```
$ rpm -Uvh lib*.rpm
rpm -Uvh fio*.rpm
```



If the installation of the `fio-util-<VSL-version>.x86_64.rpm` package fails due to missing dependencies, you will need to install the missing packages before you run the install command again, for example:

```
$ yum install lsof pciutils
```

The ioMemory VSL software and utilities are installed to the following locations:

| Package Type | Installation Location |
|-----------------------|--|
| ioMemory VSL software | <code>/lib/modules/<kernel-version>/extra/fio/iomemory-vsl.ko</code> |
| Utilities | <code>/usr/bin</code> |



You may also install the ioSphere software (optional GUI management software). ioSphere software and documentation are available as a separate download in the ioSphere download folder.

Once the packages are installed, continue to [Loading the ioMemory VSL Driver on page 15](#).

Installing DEB Packages

To install the Linux ioMemory VSL software and utilities on Debian/Ubuntu

1. You will need to install a version of the ioMemory VSL software that is built for your kernel. To determine what kernel version is running on your system, use the following command at a shell prompt:

```
$ uname -r
```

2. Compare your kernel version with the binary versions of the software available at <http://support.fusionio.com> .
 - If there is a binary version of the software that corresponds to your kernel version, download that. **For example:**

```
iomemory-vsl-<kernel-version>-<VSL-version>.amd64.deb
```

- If there is no binary version of the software corresponding to your kernel, download the source package. **For example:**

```
iomemory-vsl-<VSL-version>.tar.gz
```



Exact package names may vary, depending on software and kernel version chosen.



Use the source package that is made for your distribution. Source packages from other distributions are not guaranteed to work.

3. Download the support RPM packages you need from <http://support.fusionio.com> . These packages provide utilities, firmware, and other files.

Examples:

| Package | What Is Installed |
|--|---|
| fio-util-<VSL-version>.x86_64.deb | ioMemory VSL utilities – Recommended |
| fio-firmware-<version>.<date>-<number>_all.deb | Firmware archive package, installs the firmware archive file in a specific location (/usr/share/fio/firmware) – Optional |
| fio-firmware-<version>.<date>.fff | Firmware archive file (not an installer package) used to upgrade the firmware, make note of where you store this file – Recommended |
| libvsl-<version>.x86_64.deb | SDK libraries needed for management tools – Recommended , see Monitoring and Managing Devices on page 35 for more information on available management tools. |
| fio-common-<VSL-version>.x86_64.deb | Files required for the init script – Recommended |
| fio-sysvinit-<VSL-version>.x86_64.deb | Init script – Recommended , see Loading the ioMemory VSL Driver on page 15 for more information. |

4. Change to the directory to where you downloaded the installation packages.
5. If needed, build the ioMemory VSL software from source:
 - **If you downloaded a binary version of the software:** skip the **Building the Software from Source** instructions below and continue to the next step.
 - **If you downloaded the software source package:** follow these **Building the Software from Source** instructions:

| Building the Software from Source |
|---|
| <p>You only need to follow these additional instructions if you downloaded the source package.</p> <ol style="list-style-type: none"> Install the prerequisites for building the Fusion-io source ioMemory VSL package. <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <pre>\$ sudo apt-get install gcc fakeroot build-essential debhelper linux-headers-\$(uname -r) rsync</pre> </div> <p>You can also install packages separately by running this command:</p> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <pre>\$ sudo apt-get install gcc rsync</pre> </div> Download the software source file for your target operating system at http://support.fusionio.com |



- c. Unpack the ioMemory VSL source file.

```
$ tar zxvf iomemory-vsl_<VSL-version>.tar.gz
```

- d. Change directory to the folder created in the previous step.

```
$ cd iomemory-vsl-source-<VSL-version>
```

- e. Begin the software rebuild.

```
$ sudo dpkg-buildpackage
```

You can also build the ioMemory VSL as a non-root user.

```
$ dpkg-buildpackage -rfakeroot -b
```

- f. The new DEB packages are located in the parent directory.



A failure of any of these builds may be due to required packages missing from the system. The output from the failed command informs you of which packages are missing. Try installing any missing packages and then repeating the build process.

- g. You have now built installation packages for your distribution and kernel.
- h. Copy your custom-built software installation packages into the directory where you downloaded the installation packages.
- i. Continue to the next step.

6. Use the following command to install each of the desired packages, adding package names as needed.




Your exact package names will depend on the file names you download. Use the ioMemory VSL software package name that is built for your kernel version.

```
$ dpkg -i iomemory-vsl-<kernel-version>_<VSL-version>_amd64.deb fio-firmware_<firmware-version>_all.deb fio-util_<VSL-version>_amd64.deb
```

The ioMemory VSL software and utilities are installed to the following locations:

| Package Type | Installation Location |
|-----------------------|--|
| ioMemory VSL software | /lib/modules/<kernel-version>/extra/fio/ |
| Utilities | /usr/bin |



 You may also install the ioSphere software (optional GUI management software). ioSphere software and documentation are available as a separate download in the ioSphere download folder.


Once the packages are installed, continue to [Loading the ioMemory VSL Driver on page 15](#).

Loading the ioMemory VSL Driver


To load the ioMemory VSL driver:

Run this command:

```
$ modprobe iomemory-vsl
```

 The ioMemory VSL driver automatically loads at system boot. The ioMemory device is now available to the OS as `/dev/fiox`, where *x* is a letter (i.e., a, b, c, etc.).

To confirm the ioMemory device is attached, run the `fio-status` utility from the command line. The output lists each drive and its status (attached or not attached).

 If the ioMemory device is not automatically attaching, check the `/etc/modprobe.d` files to ensure that the `auto_attach` option is turned on (set to 1).

- For this command to work on SLES 10 systems, you must edit the `/etc/init.d/iomemory-vsl` file's init info and change `udev` to `boot.udev`. The file should look like this:

```
### BEGIN INIT INFO
# Provides:          iomemory-vsl
# Required-Start:    boot.udev
```

- On SLES systems, you must also allow unsupported modules for this command to work.
 - **SLES 11 Update 2:** Modify the `/etc/modprobe.d/iomemory-vsl.conf` file and uncomment the appropriate line:

```
# To allow the ioMemory VSL driver to load on SLES11, uncomment
below
allow_unsupported_modules 1
```

- **SLES 10 SP4:** Modify the `/etc/sysconfig/hardware/config` file so the `LOAD_UNSUPPORTED_MODULES_AUTOMATICALLY` sysconfig variable is set to `yes`, for example:

```
LOAD_UNSUPPORTED_MODULES_AUTOMATICALLY=yes
```

Controlling Driver Loading

You can control driver loading either through the init script or through udev.



In newer Linux distributions, users can rely on the udev device manager to automatically find and load drivers for their installed hardware at boot time, though udev can be disabled and the init script used in nearly all cases. We recommend using the init script to load the ioMemory VSL driver if you are managing a RAID array using LVM, mdadm, or Veritas Storage Foundation.

For older Linux distributions (such as SLES 10) without udev functionality, users must rely on a boot-time init script to load needed drivers.

Using the init Script

On systems where udev loading of the driver doesn't work, or is disabled, the init script may be enabled to load the driver at boot. On some distros it may be enabled by default.

 The init Script is part of the `fio-sysvinit` package, which must be installed before you can enable it.

You can disable this loading of the driver with the following command:

```
$ chkconfig --del iomemory-vsl
```

To re-enable the driver loading in the init script, use the following command:

```
$ chkconfig --add iomemory-vsl
```

The ioMemory VSL software install process places an init script in `/etc/init.d/iomemory-vsl`. In turn, this script uses the setting options found in the options file in `/etc/sysconfig/iomemory-vsl`. The options file must have `ENABLED` set (non-zero) for the init script to be used:

```
ENABLED=1
```

The options file contains documentation for the various settings—two of which, `MOUNTS` and `KILL_PROCS_ON_UMOUNT`, are discussed further in the section [Handling Driver Unloads on page 18](#).


Mounting Filesystems when Using the init Script

Because the ioMemory VSL driver does not load by the standard means (in the `initrd`, or built into the kernel), using the standard method for mounting filesystems (`/etc/fstab`) for filesystems hosted on the ioMemory VSL software does not work. To set up auto-mounting of a filesystem hosted on an ioMemory device:

1. Add the filesystem mounting command to `/etc/fstab` as normal.
2. You must add the 'noauto' option and the '0 0' flag to `/etc/fstab` as in the two following sample entries.

```
/dev/fioa /mnt/fioa ext3 defaults,noauto 0 0  
/dev/fiob1 /mnt/ioDrive ext3 defaults,noauto 0 0
```

(where the `a` in `fioa` can be `a`, `b`, `c`, etc., depending on how many ioMemory devices you have installed in the system).

 Failure to add 'noauto 0 0' to `fstab` may cause a boot failure. {warning}



To have the init script mount these drives after the driver is loaded and unmounted and before the driver is unloaded, add a list of mount points to the options file using the procedure documented there.

For the filesystem mounts shown in the earlier example, the line in the options file would look like this:

```
MOUNTS="/mnt/fioa /mnt/iodrive"
```

Using udev

On systems that rely on udev to load drivers, users need to modify an ioMemory VSL software options file if they want to prevent udev from auto-loading the ioMemory VSL at boot time. To do this, locate and edit the `/etc/modprobe.d/iomemory-vsl.conf` file that already has the following line:

```
# blacklist iomemory-vsl
```

To disable loading, remove the `"#"` from the line and save the file.

With the blacklist command in place, restart Linux. The ioMemory VSL driver will not be loaded by udev.

To restore the udev-loading of the driver, replace the `"#"` to comment out the line.

On either udev or init script systems

Users can disable the loading of the driver at boot time, and thus prevent the auto-attach process for diagnostic or troubleshooting purposes on either udev or init script systems. Follow the steps in the section [Disabling Auto-Attach on page 42](#) to disable or re-enable the auto-attach functionality.

Alternatively, you can prevent the ioMemory VSL driver from loading by appending the following parameter at the kernel command line of your boot loader:

```
iodrive=0
```

However, this method is not preferred as it prevents the driver from functioning at all, thus limiting the amount of troubleshooting you can perform.

ioMemory devices and Multipath Storage

If you are using ioMemory devices along with multipath storage, you must blacklist the ioMemory devices to prevent device-mapper from trying to create a dm-device for each ioMemory device. **This must be done prior to activating dm-multipath and/or loading the driver.** If ioMemory devices are not blacklisted, they will appear busy and you will not be able to attach, detach, or update the firmware on the devices.

To blacklist ioMemory devices, edit the `/etc/multipath.conf` file and include the following:

```
blacklist {
devnode          "^fio[a-z] "
}
```





Handling Driver Unloads


Special consideration must be taken during ioMemory VSL driver unload time. By default, the init script searches for any processes holding open a mounted filesystem and kills them, thus allowing the filesystem to be unmounted. This behavior is controlled by the option `KILL_PROCS_ON_UMOUNT` in the options file. If these processes are not killed, the filesystem cannot be unmounted. This may keep the ioMemory VSL software from unloading cleanly, causing a significant delay on the subsequent boot.

Upgrading the Firmware


With the ioMemory VSL software loaded, you need to check to ensure that the ioMemory device's firmware is up-to-date and then update the firmware if needed. You can do this with either the command-line utilities or the optional ioSphere software (GUI).

 Make sure you have downloaded the firmware archive file that goes with this version of the ioMemory VSL software.

 There is a specific upgrade path that you must take when upgrading an ioMemory device. Consult the *ioMemory VSL Release Notes* for this ioMemory VSL software release before upgrading ioMemory devices.

 Do not attempt to downgrade the firmware on any ioMemory device, doing so may void your warranty.

When installing a new ioMemory device along with existing devices, you must upgrade all of the currently installed devices to the latest available versions of the firmware and ioMemory VSL software before installing the new devices. Consult the *ioMemory VSL Release Notes* for this ioMemory VSL software release for any upgrade considerations.

 **Upgrading VMware Guest OS**
If you are using your ioMemory device with a VMware guest OS (using VMDirectPathIO), you must cycle the power on the VMware host server after you upgrade the device(s). Just restarting the virtual machine won't apply the change.

Command-line Interface

More information on these command-line utilities is available in [Command-line Utilities Reference on page 46](#). All command-line utilities require root permission.

1. Run the `fio-status` utility and examine the output. See [fio-status on page 52](#) for usage information.
 - If any device is in minimal mode and the reason is outdated firmware.
 - If the a device is not in minimal mode, but the firmware listed for that device is a lower number than the latest firmware version available with this version of the ioMemory VSL software, then the firmware is old, but not outdated.
2. If the firmware is old or outdated, update it using the `fio-update-iodrive` utility. See [fio-update-iodrive on page 56](#) for complete information and warnings.



Optional GUI - ioSphere software

You can use the ioSphere software to check the status of your ioMemory devices. If the ioSphere software indicates that the device's firmware is outdated, you can also use the ioSphere software to upgrade the device firmware. Consult the ioSphere software documentation for more information on installing and using the software.



Configuration

Once you have your ioMemory device and ioMemory VSL software installed and loaded, and the firmware on the device is current, you may need to configure the device and/or software. This section outlines some of the common configurations that you may need to consider.

Setting the ioMemory VSL Options

This section explains how to set ioMemory VSL software options. For more information about setting specific options, see [Using Module Parameters on page 61](#).

One-Time Configuration

ioMemory VSL software options can be set at install time, on the command line of either `insmod` or `modprobe`. For example, to set the `auto_attach` option to 0, run the command:

```
$ modprobe iomemory-vsl auto_attach=0
```

This option takes effect only for this load of the ioMemory VSL software; subsequent calls to `modprobe` or `insmod` will not have this option set.

Persistent Configuration

To maintain a persistent setting for an option, add the desired option to `/etc/modprobe.d/iomemory-vsl.conf` or a similar file. To prevent the ioMemory devices from auto-attaching, add the following line to the `iomemory-vsl.conf` file:

```
options iomemory-vsl auto_attach=0
```

This option then takes effect for every subsequent ioMemory VSL software load, as well as on autoload of the ioMemory VSL software during boot time.

Enabling PCIe Power Override


If you have installed any products with multiple ioMemory devices, such as the ioDrive Duo device, the device may require additional power to properly function (beyond the minimum 25W provided by PCIe Gen2 slots). Even if additional power is not required for your device, all dual ioMemory devices that receive additional power may benefit with improved performance.

ioDrive2 Duo devices **must** have additional power in order to properly function. For more information on which devices require additional power, see the section on *Power Cables for Multi-device Products* in the *ioMemory Hardware Installation Guide*.





This additional power may be provided in two ways:

- **External Power Cable:** This cable ships with all dual ioMemory devices. See the *ioMemory Hardware Installation Guide* for information on installing this cable.

 When a power cable is used, all of the power is drawn from the cable and no power is drawn from the PCIe slot.

- **Enabling Full Slot Power Draw:** Some PCIe slots provide additional power (often up to 75W of power). If your slot is rated to provide at least 55W, you may allow the device to draw full power from the PCIe slot by setting an ioMemory VSL software module parameter. For more information on enabling this override parameter, see the instructions below in the next section.

 This parameter overrides the setting that prevents device(s) from drawing more than 25W from the PCIe slot. The parameter is enabled per device (using device serial numbers). Once the setting is overridden, each device may draw up to the full 55W needed for peak performance.

 **WARNING:** If the slot is not capable of providing the needed amount of power, enabling full power draw from the PCIe slot may result in malfunction or even damage server hardware. You are responsible for any damage to equipment due to improper use of this override parameter and Fusion-io expressly disclaims any liability for any damage arising from such improper use. Contact Customer Support if there are any questions or concerns about the override parameter use.

Before you enable this override parameter, ensure that each PCIe slot you will use is rated to provide enough power for all slots, devices, and server accessories. Consult the server documentation, BIOS interface, setup utility, and/or use `fio-pci-check` (if available) to determine the slot power limits.



Confirm with Server Manufacturer

Contact the server manufacturer to confirm the power limits and capabilities of each slot, as well as the entire system.

The following are important considerations:

- If you are installing more than one dual ioMemory device and enabling the override parameter for each device, make sure the motherboard is rated to provide 55W power to each slot that is used.



For example, some motherboards safely provide up to 75W to any one slot, but run into power constraints when multiple slots are used to provide that much power. Installing multiple devices in this situation may also result in server hardware damage. Consult with the manufacturer to determine the total PCIe slot power available.

- The override parameter, if enabled correctly, will persist in the system, and will enable full power draw on an enabled device even if the device is removed and then placed in a different slot within the same system. If the device is placed in a slot that is not rated to provide 55W of power, you may damage your server hardware.
- This override parameter is a setting for the ioMemory VSL software per server, and is not stored in the device.



When moved to a new server, the device will default to the 25W power limit until an external power cable is added or this override parameter is enabled for that device in the new server. Consult with the manufacturer to determine the total PCIe slot power available for the new server.

Enabling the Override Parameter

Determine Serial Number(s)

Before you enable this parameter, determine the **adapter serial number** for each device you will put in a compatible slot. Use the `fio-status` command-line utility to determine the adapter serial number(s).



Serial Number Label

You may also inspect the adapter serial number label(s) on the device(s) to determine the serial number(s). However, as a best practice, confirm that each serial number is an adapter serial number by running `fio-status`. The adapter serial number label resides on the back of all ioDrive Duo devices and ioDrive2 Duo devices. On ioDrive Duo devices, it is on the PCB component that is attached to the PCIe connector.

- **Using `fio-status`:** Run the `fio-status` command-line utility. Sample output:

```
fio-status
...
Adapter: Dual Controller Adapter
Fusion-io ioDrive2 DUO 2.41TB, Product Number:F01-001-2T41-CS-0001, FIO
SN:1149D0969
External Power: NOT connected
PCIe Power limit threshold: 24.75W
Connected ioMemory modules:
fct2:   SN:1149D0969-1121
fct3:   SN:1149D0969-1111
```

In this example, 1149D0969 is the adapter serial number.

- **Using `fio-beacon`:** If you have multiple devices installed, you may use the `fio-beacon` utility to verify where each device is physically located. Consult the utility documentation [fio-beacon on page 47](#) for more information.

Setting the Parameter

Set the module parameter by editing the `/etc/modprobe.d/iomemory-vsl.conf` file and changing the value for the `external_power_override` parameter. Example:

```
options iomemory-vsl external_power_override=<value>
```

Where the `<value>` for this parameter is a comma-separated list of adapter serial numbers. For example:
1149D0969,1159E0972,24589



You must reboot or unload/load the driver to enforce any parameter changes.



Virtual Controller Configuration

Depending on your use case and application, you may benefit from configuring supported devices to use Virtual Controller technology.

When configured, each physical ioMemory device is split into two (virtual) logical devices. Splitting the ioMemory device into two virtual devices has the following implications:

- **Latency:** There is no affect on latency.
- **Throughput:** The total peak I/O bandwidth of the device is approximately the same.
- **IOPS:** Depending on the use of the virtual devices (especially the average I/O size), the peak IOPS for each virtual device is about the same for a non-split device. In other words, the combined peak IOPS of the two virtual devices can be nearly double that of a non-split device. For details, see in the *Maintenance* section.
- **Capacity:** Due to virtualization overhead, the combined capacity of the two virtual devices is slightly less than that of a single-controller device. See the *ioMemory VSL Release Notes* for a list of compatible devices and their Virtual Controller capacities.

Converting your ioMemory device to a Virtual Controller configuration will split the ioMemory device into two logical devices.

For 512B I/Os, the combined IOPS performance of the two virtual devices is approximately double that of a single-controller device. For 4kB I/Os, there is more than an 80% improvement in IOPS performance with virtual devices. For 16kB and larger I/Os, there is no improvement of total IOPS performance over a non-Virtual Controller configuration.

Latency in the virtual devices is unaffected, and the combined bandwidth of the two virtual devices is the same as it would be without the split. Due to the overhead of an additional device, the combined capacity of the two virtual devices is slightly less than that of a single-controller device.

Splitting a single physical device into multiple virtualized devices, or merging multiple virtualized devices back to a single physical device, requires a low-level format, which will erase all of the data on the device. Be sure to back up all of your data.

Supported Devices

Only relatively new devices (with few writes performed) may be split or merged. Devices with too much wear are unsuitable for converting to or from a Virtual Controller configuration. Merging virtual devices may also result in additional wear (depending on the wear differences of the two virtual devices).

To be suitable for splitting or merging, devices (including Virtual Controller devices) must have 90% or more of their remaining rated endurance of Petabytes Written (PBW). This rating as well as the current percentage remaining is visible in `fio-status` with the `-a` option. For example:

```
fio-status /dev/fctl1 -a
...
Rated PBW: 17.00 PB, 99.95% remaining
```

In the above example, the device is suitable for conversion because it has more than 90% of the rated PBW remaining.

If you attempt to merge or split a device that does not support Virtual Controller technology or a device that has too much wear, the update utility will not allow the conversion and the firmware upgrade will not take place. See the Release Notes for a list of devices that support Virtual Controller technology and their capacities after the conversion.



Multi-device Products

For products with more than one ioMemory device, such as an ioDrive2 Duo device, you must configure all of the ioMemory devices to Virtual Controller technology at the same time. All of the devices must also be merged at the same time. For example, the two ioMemory devices in an ioDrive2 Duo device will be converted into four virtual devices. The utility will not allow a conversion if you attempt to split or merge only one physical device in a multi-device product.

Splitting Controllers

Be sure to use firmware that supports Virtual Controller technology. Consult the Release Notes to determine if the firmware for that release supports Virtual Controller technology.

1. Back up all of your data. Because a low-level format is needed to complete the conversion, all of the user data on your device will be erased.
2. Use the `fio-update-iodrive` command-line utility to configure an ioMemory device to use Virtual Controller technology:
 - Use the `--split` option to split the controller.
 - Use the `-d` option to specify a device, otherwise all installed devices that can be split will be split.
 - Specify the firmware path, and check the *ioMemory VSL Release Notes* to make sure the firmware supports Virtual Controller technology.

Example:

```
fio-update-iodrive --split -d /dev/fct0 <firmware-path>
```

After rebooting, each physical device will be split into two virtual devices. Each ioMemory device will therefore split into two logical devices, each with a unique device path. For example, `/dev/fct0` may become `/dev/fct0` and `/dev/fct1`. You will manage each device as a unique device.

3. Reboot.
4. Load the ioMemory VSL driver.
5. Run `fio-status` to determine which devices need to be formatted.
6. Low-level format the device(s). For example:

```
fio-format /dev/fct0 /dev/fct1
```

Formatting will erase all user data, be sure to back up your data. You can reverse the split by merging the controllers (without losing data) up until you format the virtual devices.

Merging Controllers

If your ioMemory device (including the two virtual devices) is suitable for merging, then you will be able to use the `fio-update-iodrive` utility to merge the virtual devices back into one physical device.



1. Back up all of your data. Because a low-level format is needed to complete the merge, all of the user data on your device will be erased.
2. Use the `fio-update-iodrive` command-line utility to configure the device for merging:
 - a. Use the `--merge` option to merge the virtual devices.
 - b. Use the `-d` option to specify a device.



The `fio-update-iodrive` utility only successfully works against one of the two virtual devices for each physical ioMemory device. Out of the two virtual devices, only the first virtual device (in terms of device numbering) is linked to the physical device (and the firmware). The second virtual device is not linked, and any firmware operation against that second virtual device will fail with this message:

```
Error: Device '/dev/fctx' had an error while updating.  
This device does not support firmware update.
```

This is expected, and the error will not affect the update/merge of the first (linked) virtual device. The update operation will complete on all devices that can merge and otherwise accept firmware changes.

- c. Specify the firmware path, and check the *ioMemory VSL Release Notes* to make sure the firmware supports Virtual Controller technology.

Example:

```
fio-update-iodrive --merge -d /dev/fct0 <firmware-path>
```

3. Reboot.
4. Load the ioMemory VSL driver.
5. Run `fio-status` to determine which devices need to be formatted.
6. Low-level format the device(s). For example:

```
fio-format /dev/fct0
```



Formatting will erase all user data, be sure to back up your data. You can reverse the merge by splitting the controllers (without losing data) up until you format the merged device.

Using the Device as Swap


To safely use the ioMemory device as swap space requires passing the `preallocate_memory` kernel module parameter. To set this parameter, use either the optional ioSphere software (see ioSphere software documentation), or add the following line to the `/etc/modprobe.d/iomemory-vsl.conf` file (see [Using Module Parameters on page 61](#) for more information on using parameters):


```
options iomemory-vsl preallocate_memory=1149D2717-1121,1149D2717-1111,10345
```




- Where 1149D2717-1111,1149D2717-1111,10345 are serial numbers obtained from `fio-status`, see [fio-format on page 50](#).

A 4K sector size format is required for swap—this reduces the ioMemory VSL software memory footprint. Use `fio-format` to format the ioMemory device with 4K sector sizes.

 Be sure to provide the serial numbers for the ioMemory device, not an adapter, when applicable.

 You must have enough RAM available to enable the ioMemory device with pre-allocation enabled for use as swap. Attaching an ioMemory device, with pre-allocation enabled, without sufficient RAM may result in the loss of user processes and system instability.

Consult the *ioMemory VSL Release Notes* for RAM requirements with this version of the ioMemory VSL software.

 The `preallocate_memory` parameter is recognized by the ioMemory VSL software at load time, but the requested memory is not actually allocated until the specified device is attached.


Using the Logical Volume Manager

The Logical Volume Manager (LVM) volume group management application handles mass storage devices like ioMemory devices if you add the ioMemory device as a supported type:

1. Locate and edit the `/etc/lvm/lvm.conf` configuration file.
2. Add an entry similar to the following to that file:

```
types = [ "fio", 16 ]
```

The parameter "16" represents the maximum number of partitions supported by the device.

 If using LVM or MD, do not use `udev` to load the ioMemory VSL driver. The init script will ensure that the LVM volumes and MD devices are detached before attempting to detach the ioMemory device.

Mounting Filesystems when Using the init Script

If you are using the init script to load the ioMemory VSL software, once you have configured your devices you will need to follow the instructions in the *Mounting Filesystems when Using the init Script* subsection of the section [Controlling Driver Loading on page 15](#).

Configuring RAID using the Logical Volume Manager

The simplest way to using your ioMemory devices with LVM is to use the entire block device with it (for example: `/dev/fioa` rather than `/dev/fioa1`, `/dev/fioa2`, etc.). This way the block device does not need to be partitioned ahead of time, though partitioning is also supported. The examples that follow assume the entire block device is used.



Whether you plan to stripe (RAID 0) or mirror (RAID 1) the devices, the first two steps are the same:

1. First, create physical volumes, for example:

```
$ pvcreate /dev/fioa /dev/fiob
```

2. Next add these physical volumes to a volume group, for example:

```
$ vgcreate iomemory_vg /dev/fioa /dev/fiob
```

Creating a Striped Volume (RAID 0) Using LVM

With the volume group created, you can create logical volumes within this volume group. In this instance, only one will be created and we name it `iomemory_lv`:

1. Create the striped volume using the `-i2` option for two stripes, for example:

```
$ lvcreate -l 100%VG -n iomemory_lv -i2 iomemory_vg
```

2. Create a file system on the newly created volume, for example:

```
$ mkfs.ext3 /dev/iomemory_vg/iomemory_lv
```

3. In order to make this persistently mount at boot time, edit the `/etc/sysconfig/iomemory-vsl` file by adding the volume group path under the example line:

```
Example: LVM_VGS="/dev/vg0 /dev/vg1"  
LVM_VGS="/dev/iomemory_vg"
```

Be sure to just add the volume group path and not the logical volume as well. For more information on using the init script and the `/etc/sysconfig/iomemory-vsl` file, see [Loading the ioMemory VSL Driver on page 15](#).

Creating a Mirrored Volume (RAID 1) Using LVM

With the volume group created, you can create logical volumes within this volume group. In this instance, only one will be created and we name it `iomemory_lv`:

1. Create the mirrored volume using the `-m1` option for an original linear volume plus one copy, for example:

```
$ lvcreate -l 50%VG -n iomemory_lv -m1 --corelog iomemory_vg
```

You can monitor the progress of the initial mirror synchronization using the `lvs` command. For example:

```
lvs -a -o +devices
```



With the `--corelog` option, LVM uses an in-memory region or log to track the state of the mirror legs. Since it is in-memory and lost at reboot, it must be regenerated at boot by scanning the mirror legs. For alternative configurations, consult the LVM documentation.



2. Create a file system on the newly created volume, for example:

```
$ mkfs.ext3 /dev/iomemory_vg/iomemory_lv
```


3. In order to make this persistently mount at boot time, edit the `/etc/sysconfig/iomemory-vsl` file by adding the volume group path under the example line:


```
# Example: LVM_VGS="/dev/vg0 /dev/vg1"
LVM_VGS="/dev/iomemory_vg"
```

Be sure to just add the volume group path and not the logical volume as well. For more information on using the init script and the `/etc/sysconfig/iomemory-vsl` file, see [Loading the ioMemory VSL Driver on page 15](#).

Configuring RAID Using mdadm

You can configure two or more ioMemory devices into a RAID array using software-based RAID solutions.

 If you are using RAID1/Mirrored and one device fails, be sure to run `fio-format` on the replacement device (not the existing, good device) before rebuilding the RAID. Following are some examples of some common RAID configurations using the `mdadm` utility.

 The Linux kernel RAID 5 implementation performs poorly at high data rates. This is an issue in the Linux kernel. Alternatives include using RAID 10, or possibly a third-party RAID stack.

Mounting Arrays

Once you are done making your array (by following one of the configuration samples below), you must edit the `/etc/sysconfig/iomemory-vsl` file and add the array path under the example line. This will make the array mount at boot time.

In all of the examples below, we create arrays named `md0`. Here is how you would edit the `/etc/sysconfig/iomemory-vsl` file to include the array using `md0` as an example (you add the array just below the example line in that file):

```
# Example: MD_ARRAYS="/dev/md0 /dev/md1"
MD_ARRAYS="/dev/md0"
```

If you are using the init script to load the ioMemory VSL software, once you have configured your devices you will need to follow the instructions in the *Mounting Filesystems when Using the init Script* subsection of the section [Controlling Driver Loading on page 15](#).

RAID 0

To create a striped set, where `fioa` and `fiob` are the two ioMemory devices you want to stripe, run this command:



```
$ mdadm --create /dev/md0 --chunk=256 --level=0 --raid-devices=2 /dev/fioa /dev/fiob
```

Making the Array Persistent (Existing after Restart)

On some versions of Linux, the configuration file is in `/etc/mdadm/mdadm.conf`, not `/etc/mdadm.conf`.

Inspect `/etc/mdadm.conf`. If there are one or more lines declaring the devices to inspect, make sure one of those lines specifies "partitions" as an option. If it does not, add a new `DEVICE` line to the file specifying "partitions" like this:

```
DEVICE partitions
```


Also add a device specifier for the fio ioMemory devices:

```
DEVICE /dev/fio*
```

To see if any updates are needed to `/etc/mdadm.conf`, issue the following command:

```
$ mdadm --examine --scan
```

Compare the output of this command to what currently exists in `mdadm.conf` and add any needed sections to `/etc/mdadm.conf`.


 For example, if the array consists of two devices, there will be three lines in the output of the command that are not present in the `mdadm.conf` file: one line for the array, and two device lines (one line for each device). Be sure to add those lines to the `mdadm.conf` so it matches the output of the command.

For further details please see the `mdadm` and `mdadm.conf` man pages for your distribution.

With these changes, on most systems the RAID 0 array will be created automatically upon restart. However, if you have problems accessing `/dev/md0` after restart, run the following command:

```
$ mdadm --assemble --scan
```

You may also want to disable udev loading of the ioMemory VSL driver, if needed, and use the init script provided for driver loading. Please see the *Using the init Script* subsection of the section [Controlling Driver Loading on page 15](#) for further details on how to use the init script.

 In SLES 11, you may need to run the following commands to make sure these services are run on boot:

```
chkconfig boot.md on  
chkconfig mdadm on
```

RAID 1

To create a mirrored set using the two ioMemory devices `fioa` and `fiob`, run this command:

```
$ mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/fioa /dev/fiob
```



RAID 10

To create a striped, mirrored array using four ioMemory devices (fioa, fiob, fioc, and fiod), run this command:

```
$ mdadm --create /dev/md0 -v --chunk=256 --level=raid10 --raid-devices=4
/dev/fioa /dev/fiob /dev/fioc /dev/fiod
```


Building a RAID10 Across Multiple Devices


In a RAID10 configuration, sets of two disks are mirrored, and then those mirrors are striped. When setting up a RAID10 across multiple ioMemory devices, it is best to make sure that no mirror resides solely on the two ioMemory devices that comprise an a single product (such as an ioDrive Duo device).

In order to get the data to lay out properly,

- Use the `--layout=n2` option when creating the RAID10 (though it should be the default)
- Ensure that no two ioMemory devices from the same device are listed side by side.

The following sample code shows some recommended configurations.

 The following commands assume that all ioMemory devices have been freshly formatted with the `fio-format` utility.

 The ordering of the fiox devices is critical.

```
# 2 Dual Device Products RAID10
$ mdadm --create --assume-clean --level=raid10 --layout=n2 -n 4 /dev/md0 \
/dev/fioa /dev/fioc \
/dev/fiob /dev/fiod
# Mirror groups are: fioa,fioc and fiob,fiod
```

```
# 3 Dual Device Products RAID10
$ mdadm --create --assume-clean --level=raid10 --layout=n2 -n 6 /dev/md0 \
/dev/fioa /dev/fiod \
/dev/fioc /dev/fiof \
/dev/fioe /dev/fiob
```

```
# 4 Dual Device Products RAID10
$ mdadm --create --assume-clean --level=raid10 --layout=n2 -n 8 /dev/md0 \
/dev/fioa /dev/fiod \
/dev/fioc /dev/fiof \
/dev/fioe /dev/fioh \
/dev/fiog /dev/fiob
```

```
# 8 Dual Device Products RAID10
$ mdadm --create --assume-clean --level=raid10 --layout=n2 -n 16 /dev/md0 \
/dev/fioa /dev/fiod \
```



```
/dev/fioc /dev/fiof \  
/dev/fioe /dev/fioh \  
/dev/fiog /dev/fioj \  
/dev/fioi /dev/fiol \  
/dev/fiok /dev/fion \  
/dev/fiom /dev/fiop \  
/dev/fioo /dev/fiob
```

Discard (TRIM) Support

With this version of the ioMemory VSL software, Discard (also known as TRIM) is enabled by default.

Discard addresses an issue unique to solid-state storage. When a user deletes a file, the device does not recognize that it can reclaim the space. Instead the device assumes the data is valid.

Discard is a feature on newer filesystem releases. It informs the device of logical sectors that no longer contain valid user data. This allows the wear-leveling software to reclaim that space (as reserve) to handle future write operations.

TRIM on Windows Server 2008 R1

Windows TRIM is not built into Windows Server 2008 R1, and it is therefore not supported.

Discard (TRIM) on Linux

Discard is enabled by default in the ioMemory VSL software release. However, for discard to be implemented, the Linux distribution must support this feature, and discard must be turned on.

In other words, if your Linux distribution supports discard, and discard is enabled on the system, then discard will be implemented on your ioMemory device.

Under Linux, discards are not limited to being created by the filesystem, discard requests can also be generated directly from userspace applications using the kernel's discard ioctl.



There is a known issue that ext4 in Kernel.org 2.6.33 or earlier may silently corrupt data when discard is enabled. This has been fixed in many kernels provided by distribution vendors. Please check with your kernel provider to be sure your kernel properly supports discard. For more information, see the Errata in the *ioMemory VSL Release Notes* for this version of the software



On Linux, MD and LVM do not currently pass discards to underlying devices. Thus any ioMemory device that is part of an MD or LVM array will not receive discards sent by the filesystem.

The LVM release included in Red Hat 6.1 supports passing discards for several targets, but not all ([RHEL 6.1 Documentation](#)). Please see your distribution's documents for exact details.



Performance and Tuning

ioMemory devices provide high bandwidth, high Input/Output per Second (IOPS), and are specifically designed to achieve low latency.

As ioMemory devices improve IOPS and low latency, the device performance may be limited by operating system settings and BIOS configuration. These settings may need to be tuned to take advantage of the revolutionary performance of ioMemory devices.

While ioMemory devices generally perform well out of the box, this section describes some of the common areas where tuning may help achieve optimal performance.

Disable CPU Frequency Scaling

Dynamic Voltage and Frequency Scaling (DVFS) are power management techniques that adjust the CPU voltage and/or frequency to reduce power consumption by the CPU. These techniques help conserve power and reduce the heat generated by the CPU, but they adversely affect performance while the CPU transitions between low-power and high-performance states.

These power-savings techniques are known to have a negative impact on I/O latency and IOPS. When tuning for performance, you may benefit from reducing or disabling DVFS completely, even though this may increase power consumption.

DVFS, if available, is often configurable as part of your operating systems power management features as well as within your system's BIOS interface. Within the operating system and BIOS, DVFS features are often found under the Advanced Configuration and Power Interface (ACPI) sections; consult your computer documentation for details.

Limiting ACPI C-States

Newer processors have the ability to go into lower power modes when they are not fully utilized. These idle states are known as ACPI C-states. The C0 state is the normal, full power, operating state. Higher C-states (C1, C2, C3, etc.) are lower power states.

While ACPI C-states save on power, they can have a negative impact on I/O latency and maximum IOPS. With each higher C-state, typically more processor functions are limited to save power, and it takes time to restore the processor to the C0 state.

When tuning for maximum performance you may benefit from limiting the C-states or turning them off completely, even though this may increase power consumption.

Setting ACPI C-State Options

If your processor has ACPI C-states available, you can typically limit or disable them in the BIOS interface (sometimes referred to as a Setup Utility). ACPI C-states may be part of the Advanced Configuration and Power Interface (ACPI) menu. Consult your computer documentation for details.



C-States Under Linux

Newer Linux kernels have drivers that may attempt to enable APCI C-states even if they are disabled in the BIOS. You can limit the C-state in Linux (with or without the BIOS setting) by adding the following to the kernel boot options:

```
intel_idle.max_cstate=0 processor.max_cstate=0
```

In this example, the maximum C-state allowed will be C0 (disabled).

Setting NUMA Affinity

Servers with a NUMA (Non-Uniform Memory Access) architecture may require special installation instructions in order to maximize ioMemory device performance. This includes most multi-socket servers.

On some servers with NUMA architecture, during system boot, the BIOS will not associate PCIe slots with the correct NUMA node. Incorrect mappings result in inefficient I/O handling that can significantly degrade performance. To prevent this, you must manually assign ioMemory devices optimally among the available NUMA nodes.

See [NUMA Configuration on page 63](#) for more information on setting this affinity.


Setting the Interrupt Handler Affinity

Device latency can be affected by placement of interrupts on NUMA systems. We recommend placing interrupts for a given device on the same NUMA node that the application is issuing I/O from. If the CPUs on this node are overwhelmed with user application tasks, in some cases it may benefit performance to move the the interrupts to a remote node to help load-balance the system.

Many operating systems will attempt to dynamically place interrupts across the nodes, and generally make good decisions.

Linux IRQ Balancing

In Linux this dynamic placement is called IRQ Balancing. You can check to see if the IRQ balancer is effective by checking `/proc/interrupts`. If the interrupts are unbalanced (too many device interrupts on one node) or on an overwhelmed node, you may need to stop the IRQ balancer and manually distribute the interrupts in order to balance the load and improve performance.

 Restarting the IRQ Balancer after the ioMemory VSL software loads (and the ioMemory devices are attached) may resolve interrupt affinity issues. For example, run one of the following commands (depending on your distribution):

```
/etc/init.d/irqbalance start
```

```
/etc/init.d/irq_balancer start
```

If that does not resolve the affinity issues, then we recommend manual pinning the device interrupts to specific nodes.



Hand-tuning interrupt placement in Linux is an advanced option that requires profiling of application performance on any given hardware. Please see your operating system documentation for information on how to pin specific device interrupts to specific nodes.



Monitoring and Managing Devices

Fusion-io provides many tools for managing your ioMemory devices. These tools will allow you to monitor the devices for errors, warnings, and potential problems. They will also allow you to manage the devices including performing the following functions:

- Firmware upgrades
- Low-level formatting
- Attach and detach actions
- Device status and performance information
- Configuring Swap and Paging
- Generating bug reports

Management Tools

Fusion-io has provided several tools for monitoring and managing ioMemory devices. These include stand-alone tools that require no additional software and data-source tools that can be integrated with other applications.

Consider the descriptions of each tool to decide which tool (or combination of tools) best fits your needs.



The ioMemory VSL software does print some error messages to the system logs, and while these messages are very useful for troubleshooting purposes, the ioMemory VSL software log messages are not designed for continual monitoring purposes (as each is based on a variety of factors that could produce different log messages depending on environment and use case). For best results, use the tools described in this section to regularly monitor your devices.

Stand-alone Tools

These stand-alone tools do not require any additional software.

- **Command-line Utilities:** These utilities are installed with the ioMemory VSL software and are run manually in a terminal. The `fio-status` utility provides status for all devices within a host. The other utilities allow you to perform other management functions. See [Command-line Utilities Reference on page 46](#) for full details.
- **ioSphere software:** The GUI browser-based ioSphere software allows you to monitor and manage every ioMemory device installed in multiple hosts across your network. It collects all of the alerts for all ioMemory devices and displays them in the Alert Tab. You may also set up the ioSphere software to send email or SMS messages for specific types of alerts or all alerts. The ioSphere software packages and documentation are available as separate downloads.

Data-source Tools

These data-source tools provide comprehensive data, just like the stand-alone tools, but they do require integration with additional software. At a minimum, some tools can interface with a browser. ***However, the benefit of these tools is that they can be integrated into existing management software that is customized for your organization.***



These tool packages and documentation are also available as separate downloads (separate from the ioMemory VSL software packages).

- **SNMP Subagent:** The Fusion-io SNMP AgentX subagent allows you to monitor and manage your ioMemory devices using the Simple Network Management Protocol. You can use a normal SNMP browser, or customize your existing application to interface with the subagent.
- **SMI-S CIM Provider:** The CIM provider allows you to monitor and manage your devices using the Common Information Model. You can use a normal CIM browser, or customize your existing application to interface with the CIM provider.
- **ioMemory VSL Management SDK:** This C programming API allows you to write customize applications for monitoring and managing ioMemory devices.

Example Conditions to Monitor

This section gives examples of conditions you can monitor. It is intended as an introduction and not as a comprehensive reference. These conditions will have slightly different names, states, and values, depending on the tool you choose. For example, an SNMP MIB may have a different name than a SMI-S object or an API function.

In order to properly monitor these conditions, you should become familiar with the tool you choose to implement and read the documentation for that tool. You may also discover additional conditions that you wish to frequently monitor.

For quick reference, the possible states/values of these conditions are described as Normal (**GREEN**), Caution/Alert (**YELLOW**), or Error/Warning (**RED**). You may implement your own ranges of acceptable states/values, especially if you use a data-source tool.

Device Status

All of the monitoring tools return information on the status of the ioMemory devices, including the following states:

| | |
|---------------|--|
| GREEN | Attached |
| YELLOW | Detached, Busy (including: Detaching, Attaching, Scanning, Formatting, and Updating) |
| RED | Minimal Mode, Powerloss Protect Disabled |

If the device is in Minimal Mode, the monitoring tool can display the reason for the Minimal Mode status.

Required Actions

If the device is in Minimal Mode, the action will depend on the reason. For example, if the reason is outdated firmware, then you will need to update the firmware.

Temperature

ioMemory devices require adequate cooling. In order to prevent thermal damage, the ioMemory VSL software will start throttling write performance once the on-board controller reaches a specified temperature. If the controller temperature continues to rise, the software will shut down the device once the controller temperature reaches the maximum operating temperature.



These temperatures depend on the device. Newer ioMemory devices have higher thermal tolerances. Consult the *ioMemory Hardware Installation Guide* to determine the thermal tolerances of all devices you will monitor. **This table uses the controller thermal tolerances for newer devices** (93°C throttling, 100°C shutdown).

| | |
|--------|---------|
| GREEN | <93°C |
| YELLOW | 93-99°C |
| RED | 100°C |

You may wish to shift the conditions by a few degrees so the **YELLOW** condition exists before throttling occurs. For example:

| | |
|--------|---------|
| GREEN | <90°C |
| YELLOW | 90-96°C |
| RED | 97°C |



NAND Board Temperature

Newer ioMemory devices also report the temperature of the NAND Boards. This is also a critical temperature to monitor. Consult the *ioMemory Hardware Installation Guide* to see if your device reports this temperature and to see the temperature thresholds.

Required Actions

If the temperature is at or approaching the **YELLOW** condition, you must increase the cooling for your system. This may include increasing the fan speed, bringing down the ambient temperature, reducing write load, or moving the device to a different slot.

Health Reserves Percentage

ioMemory devices are highly fault-tolerant storage subsystem with many levels of protection against component failure and the loss nature of solid-state storage. As in all storage subsystems, component failures may occur.

By pro-actively monitoring device age and health, you can ensure reliable performance over the intended product life. The following table describes the Health Reserve conditions.

| | |
|--------|-------|
| GREEN | >10% |
| YELLOW | 0-10% |
| RED | 0% |

At the 10% healthy threshold, a one-time warning is issued. At 0%, the device is considered unhealthy. It enters *write-reduced* mode. After the 0% threshold, the device will soon enter *read-only* mode.

For complete information on Health Reserve conditions and their impact on performance, see [Monitoring the Health of Devices on page 59](#).



Required Actions

The device needs close monitoring as it approaches 0% reserves and goes into write-reduced mode, which will result in reduced write performance. Prepare to replace the device soon.

Write (Health Reserves) Status

In correlation with the Health Reserves Percentage, the management tools will return write states similar to these:


| | |
|--------|--|
| GREEN | Device is healthy |
| YELLOW | Device is getting close to entering reduced write mode. |
| RED | Device has entered reduced-write or read-only mode to preserve the flash from further wearout. |

Required Actions

The device needs close monitoring as it approaches 0% reserves and goes into write-reduced mode, which will result in reduced write performance. Prepare to replace the device soon.

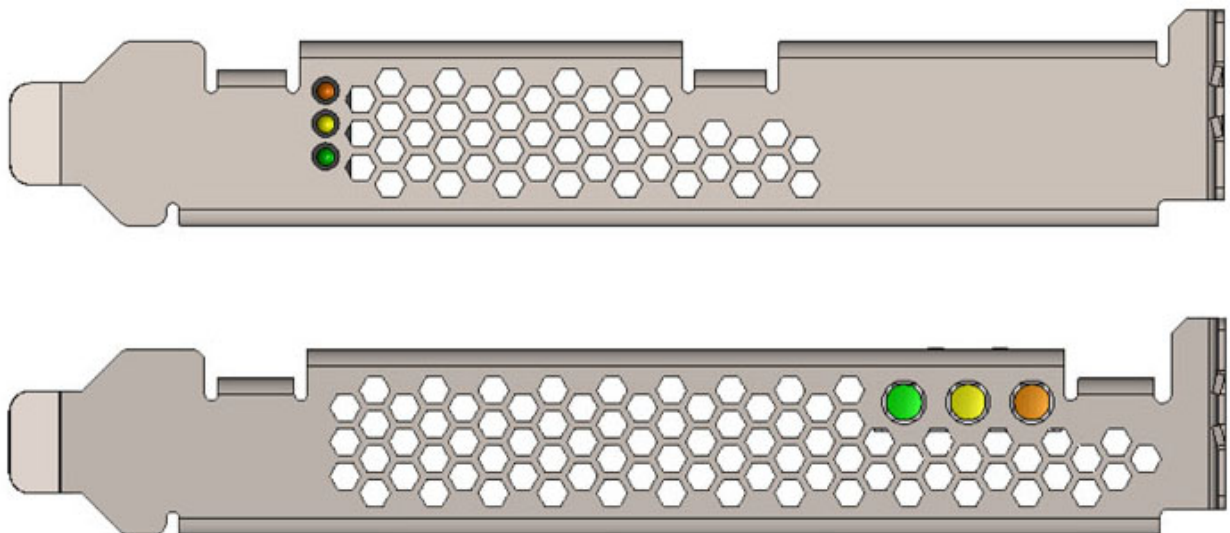
Device LED Indicators

If you have physical access to the devices and depending on your device configuration, you can use the LED indicator (s) on the bracket to monitor their status. Consult the subsections below to determine if your device uses one or more LED indicators and what behaviors are indicated.

 ioMemory devices may have an additional LEDs (that are not on the bracket, as shown below. You can ignore those other LEDs, as they are not meant for monitoring device and software functionality.

Three LEDs

The LEDs on your device should be similar to one of these configurations:



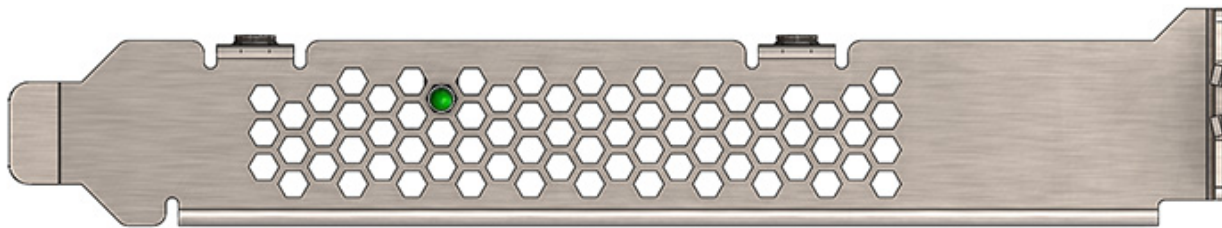


This table explains the information that these LEDs convey:

| Green | Yellow | Amber | Indication | Notes |
|----------|----------|-------|---|--|
| OFF | OFF | OFF | Power is off. | |
| OFF | OFF | LIT | Power is on. Problem with device, or driver not loaded (and device unattached). | Use <code>fio-status</code> to view problem, or load driver (and attach device). |
| LIT | OFF | OFF | Power is on. Driver loaded (device may or may not be attached). | You may need to attach the device. |
| LIT | FLASHING | OFF | Writing (Rate indicates volume of writes). | Can appear in combination with the Read LED indication. |
| FLASHING | OFF | OFF | Read (rate indicated volume of reads). | Can appear in combination with the Write LED indication. |
| LIT | LIT | LIT | Location Beacon. | Use the <code>fio-beacon</code> utility to initiate this behavior. |

One LED

If your device has one LED, it should be similar to this configuration:




This table explains the information that the LED conveys:

| LED | Indications | Notes |
|--------------------|--|--|
| LIT | Power is on and driver is working. | |
| FLASHING (Fast) | Read and/or write activity. | The faster flashing only indicates activity, it does not reflect the amount of data that is read or written. The flashing may not indicate reads from empty sectors (all zeros). |
| FLASHING (Slow) | Location beacon. | Use the <code>fio-beacon</code> utility to initiate this behavior. |
| OFF | This indicates one of the following: Power is off, driver is not loaded, or driver is not working. | Check <code>fio-status</code> to see if device is attached and there are no errors. |




Maintenance

This section explains additional software maintenance functions not covered in the sections [Configuration on page 20](#) and [Monitoring and Managing Devices on page 35](#).

 All commands require administrator privileges. Log in as "root" or use sudo to run the commands.

Uninstalling the Software

 If you came to this section from the *Installation Overview* section, return to [Installation Overview on page 8](#) after you uninstall previous versions of the ioMemory VSL software and utilities.

Uninstalling the ioMemory VSL Utilities and Other Support Packages

Uninstalling 2.x Support Packages

To uninstall the support RPM packages, run this command (adding or removing package names as needed):

```
$ rpm -e fio-util fio-snmp-agentx fio-common fio-firmware iomanager-gui  
iomanager-jre libfio libfio-doc libfusionjni fio-sysvinit fio-smis fio-snmp-  
mib libfio-dev
```

To uninstall the support DEB packages, run this command (adding or removing package names as needed):

```
$ dpkg -r fio-util fio-snmp-agentx fio-common fio-firmware iomanager-gui  
iomanager-jre libfio libfio-doc libfusionjni fio-sysvinit fio-smis fio-snmp-  
mib libfio-dev
```

Uninstalling 3.x Support Packages

To uninstall the support RPM packages, run this command (adding or removing package names as needed):

```
$ rpm -e fio-util fio-snmp-agentx fio-common fio-firmware libvsl libvsl-doc  
fio-sysvinit fio-smis fio-snmp-mib libvsl-dev
```

To uninstall the support DEB packages, run this command (adding or removing package names as needed):

```
$ dpkg -r fio-util fio-snmp-agentx fio-common fio-firmware libvsl libvsl-doc  
fio-sysvinit fio-smis fio-snmp-mib libvsl-dev
```

Uninstalling the ioMemory VSL RPM Package

With versions 2.x and later (including 3.x releases) of the ioMemory VSL software, you must specify the kernel version of the package you are uninstalling. Run this command to find the installed driver packages:



```
$ rpm -qa | grep -i iomemory
```

Sample output:

```
iomemory-vsl-2.6.18-194.el5-2.2.2.82-1.0
```

Uninstall the ioMemory VSL software by running a command similar to this example (specify the kernel version of the driver you wish to uninstall):

```
$ rpm -e iomemory-vsl-2.6.18-194.el5-2.2.0.82-1.0
```

Uninstalling the ioMemory VSL software DEB Package

With versions 2.x and later (including 3.x releases) of the ioMemory VSL software, you must specify the kernel version of the package you are uninstalling. Run this command to find the installed packages:

```
$ dpkg -l | grep -i iomemory
```

Sample output:

```
iomemory-vsl-2.6.32-24-server
```

Uninstall the ioMemory VSL software by running a command similar to this example (specify the kernel version of the software you wish to uninstall):

```
$ dpkg -r iomemory-vsl-2.6.32-24-server
```

Unloading the Software Driver

To unload the driver, run this command:

```
$ modprobe -r iomemory-vsl
```

Upgrading the Kernel

If you ever plan to upgrade the kernel when the ioMemory VSL software is installed, you **must**:

1. Unload the ioMemory VSL driver.
2. Uninstall the ioMemory VSL software.
3. Upgrade the kernel.
4. Install an ioMemory VSL software package that is compiled for the new kernel.

Failure to follow this procedure may result in driver load issues.

Disabling the ioMemory VSL Software


The ioMemory VSL software automatically loads by default when the operating system starts. You can disable the ioMemory VSL software for diagnostic or troubleshooting purposes.



To disable auto-load, uninstall the ioMemory VSL software to keep it from loading, or move it out of the `/lib/modules/<kernel_version>` directory.

Disabling Auto-Attach

When the ioMemory VSL software is installed, it is configured to automatically attach any devices when the ioMemory VSL software is loaded. Sometimes you may want to disable the auto-attach feature (to assist in troubleshooting or diagnostics). To do so:

 You can also use the ioSphere software to enable or disable auto-attach. See the ioSphere software documentation for more information.

1. Edit the following file:

```
/etc/modprobe.d/iomemory-vsl.conf
```

2. Add the following line to that file:

```
options iomemory-vsl auto_attach=0
```

3. Save the file. To re-enable auto-attach, simply edit the file and either remove that line or change it to the following:

```
options iomemory-vsl auto_attach=1
```

Unmanaged Shutdown Issues

Unmanaged shutdowns due to power loss or other circumstances can force the ioMemory device to perform a consistency check during the restart. This may take several minutes or more to complete.

Although data written to the ioMemory device is not lost due to unmanaged shutdowns, important data structures may not have been properly committed to the device. This consistency check (also called a rescan) repairs these data structures.

Improving Rescan Times

The rescan of the device (also called a consistency check) the VSL performs after an unmanaged shutdown may take an extended period of time depending on the total capacity of the device(s) that the ioMemory VSL software needs to scan.

Default Fast Rescan

By default, all ioMemory devices formatted with the `fio-format` utility or ioSphere are formatted to have improved rescan times. You can disable this default fast rescan by reformatting the device and using the `-R` option. Disabling this feature will reclaim some reserve capacity that is normally set aside to help improve rescan times.

If you leave the default fast rescan feature in place you can also take further steps to improve rescan times by implementing one of the following module parameters.



Faster Rescans Using Module Parameters

These two module parameters require the default fast rescan formatting structure, and they also use system memory (RAM) to help improve rescan times. The extra memory enables the rescan process to complete faster, which reduces downtime after a hard shutdown. This memory allocation is only temporary and is freed up after the rescan process is complete.

If you decide to use one of these parameters, you will need to set the upper limit of RAM used by that parameter. To do this, you will need to determine how much RAM each parameter may use in your scenario, how much system RAM is available, and (therefore) which parameter is more suited for your use case.

For more information on setting module parameters, see [Using Module Parameters on page 61](#).

Here is a quick comparison of the two parameters:

- **RMAP Parameter**
 - **Fastest:** This improvement results in the fastest rescan times.
 - **Less Scalable:** (All or nothing.) This parameter requires enough RAM to function. If the RAM limit is set too low, then the ioMemory VSL software will not use RMAP at all, and it will revert back to the default fast rescan process.
 - **Target Scenario:** This parameter will improve any use case if there is enough RAM available for the parameter. It is more suited for smaller capacity ioMemory devices and/or systems with fewer ioMemory devices installed. We also recommend it for devices that have been used for many small random writes.
- **RSORT Parameter**
 - **Faster:** This improves rescan times over the default fast rescan process.
 - **Scalable:** With this parameter, the ioMemory VSL software works with the system RAM to improve rescan times until it reaches the RAM limit set in the parameter. At that point, the software reverts back to the default fast rescan process.
 - **Target Scenario:** This parameter will improve rescan times in any use scenario. It is especially useful in systems with multiple ioMemory devices and/or larger-capacity ioMemory devices. We also recommend it when ioMemory devices are used to store databases.

RMAP Parameter

The `rmap_memory_limit_MiB` parameter sets the upper memory (RAM) limit (in mebibytes) used by the ioMemory VSL software to perform the RMAP rescan process. You should only use this option if you have enough memory for all of your ioMemory devices in the system. If you do not have enough memory to use this option, use the `RSORT` parameter instead.

Because this parameter requires a set amount of memory, it often works best with fewer ioMemory devices and/or smaller-capacity ioMemory devices in a system, but the determining factor is how much memory is in the system and whether there is enough to set the appropriate memory limit.

This parameter requires 4,008 bytes of RAM per block of ioMemory device capacity.



1. First determine the number of blocks that are formatted for each device.
 - a. This information is visible when you format the device using the `fio-format` utility.
 - b. Or you can estimate the number of block using the device capacity and the formatted sector size.

This example shows a quick estimation of the number of blocks on a 400GB device with 512B size sectors (2 sectors per KB):

$$400\text{GB} * 1000\text{MB/GB} * 1000\text{KB/MB} * 2 \text{ Blocks/kB} = 800,000,000 \text{ Blocks}$$

2. Multiply the number of blocks by 4.008 bytes of RAM per block (and translate that into MiB) to determine the memory limit that is required for this parameter to function.
 - a. In the example above there were 800 million blocks:

$$800,000,000 \text{ Blocks} * 4.008\text{B/Block} * 1\text{KiB}/1024\text{B} * 1\text{MiB}/1024\text{KiB} = \\ \sim 3058\text{MiB of RAM}$$

- b. In this example, you would need about 3100 MiB of RAM available in your system for a 400GB ioMemory device formatted for 512B sectors, and you would need to set the RMAP parameter to 3100.



Default Value

The RMAP parameter is, by default, set to 3100. It is set to this low default value so the rescan process does not use all of the RAM in systems that have less available memory.

- If the RMAP value is too low for the number of ioMemory device blocks in the system, then the ioMemory VSL software will not use the RMAP process to improve rescan times, it will just use the default fast rescan process. (RMAP is an all-or-nothing setting.)
- If you don't have enough system memory to use the RMAP parameter, consider using the RSORT parameter. The RSORT parameter will use its RAM limit to improve the rescan process, and then the ioMemory VSL software revert to the default fast rescan process to finish the consistency check.

3. Set the module parameter to the value you have determined. See [Using Module Parameters on page 61](#) for more information on setting parameters.

RSORT Parameter

The `rsort_memory_limit_MiB` parameter sets the memory (RAM) limit used by the ioMemory VSL software to perform the RSORT rescan process. The RSORT rescan process is faster than the default rescan process and we recommend using it to rescan devices that are used datastores for databases.

If this parameter is given any memory limit, the ioMemory VSL software will use the RSORT process until either the rescan is done or it consumes the memory limit. If the process runs out of memory, it will revert to the default fast rescan process. However, in order to optimize the use of this process, you can calculate the target RAM usage and set the limit based on that target. There is no penalty for setting a high limit, the RSORT process will only use the RAM it needs (up to the limit that is set).



This target is based on 32 bytes per write extent. For example, if your database writes 16kB at a time, there is one write extent per 16kB of ioMemory device capacity.

Blocks per Write Extent

One measure of the the benefits of the RSORT process is to see how many blocks are written per write extent. The RSORT process improves rescan times over the default fast rescan process on when a device has 8 or more blocks written per extent. For example, if your ioMemory device is formatted to 512B sector sizes (2 sectors per KB), and your database writes in 8KB chunks, then your database writes 16 blocks per write extent and RSORT would improve the rescan times.

1. First determine the number of blocks that are formatted for each device.
 - a. This information is visible when you format the device using the `fio-format` utility.
 - b. Or you can estimate the number of block using the total device capacities and their formatted sector sizes.

This example shows a quick estimation of the number of blocks on 1200GB of ioMemory device capacity with 512B size sectors (2 sectors per KB):

```
1200GB * 1000MB/GB * 1000KB/MB * 2 Blocks/kB = 2,400,000,000  
Blocks
```

2. Divide the number of blocks by the write extents per block to determine the total possible number of write extents on the device(s).
 - a. In the example above there were 2.4 billion blocks. We will assume 16KB write extents (32 blocks per write on 512B sectors):

```
2,400,000,000 Blocks * 1 Write Extent/32 Blocks = 150,000,000  
Writes
```

3. Multiply the number of writes by 32 bytes of RAM per write (and translate that into MiB) to determine the memory target for this parameter.
 - a. In the example above there were 150 million write extents:

```
150,000,000 Writes * 32B/Write * 1KiB/1024B * 1MiB/1024KiB =  
~4578MiB of RAM
```

- b. In this example, you would want to set the RSORT limit to about 4600 MiB of RAM available in your system for 1200GB of ioMemory device capacity formatted for 512B sectors.

Default Value


The RMAP parameter is, by default, set to 0m and it has a maximim of 100000 (100GB).

4. Set the module parameter to the value you have determined. See [Using Module Parameters on page 61](#) for more information on setting parameters.




Appendix A - Command-line Utilities Reference

The ioMemory VSL software installation packages include various command-line utilities, installed by default in `/usr/bin`. These provide a number of useful ways to access, test, and manipulate your device.

 There are some additional utilities installed in the `/usr/bin` directory that are not listed below. Those additional utilities are dependencies (used by the main ioMemory VSL utilities), and you should not use them directly unless Customer Support advises you to do so.


| Utility | Purpose |
|---------------------------------|---|
| <code>fio-attach</code> | Makes an ioMemory device available to the OS. |
| <code>fio-beacon</code> | Lights the ioMemory device's external LEDs. |
| <code>fio-bugreport</code> | Prepares a detailed report for use in troubleshooting problems. |
| <code>fio-detach</code> | Temporarily removes an ioMemory device from OS access. |
| <code>fio-format</code> | Used to perform a low-level format of an ioMemory device. |
| <code>fio-pci-check</code> | Checks for errors on the PCI bus tree, specifically for ioMemory devices. |
| <code>fio-status</code> | Displays information about the device. |
| <code>fio-sure-erase</code> | Clears or purges data from the device. |
| <code>fio-update-iodrive</code> | Updates the ioMemory device's firmware. |

 There are `-h` (Help) and `-v` (Version) options for all of the utilities. Also, `-h` and `-v` cause the utility to exit after displaying the information.


fio-attach

Description

Attaches the ioMemory device and makes it available to the operating system. This creates a block device in `/dev` named `fiox` (where `x` is `a`, `b`, `c`, etc.). You can then partition or format the ioMemory device, or set it up as part of a RAID array. The command displays a progress bar and percentage as it operates.

 In most cases, the ioMemory VSL software automatically attaches the device on load and does a scan. You only need to run `fio-attach` if you ran `fio-detach` or if you set the ioMemory VSL software's `auto_attach` parameter to 0.




 If the ioMemory device is in minimal mode, then auto-attach is disabled until the cause of the device being in minimal mode is fixed.

Syntax

```
fio-attach <device> [options]
```

where <device> is the name of the device node (/dev/fctx), where *x* indicates the device number: 0, 1, 2, etc. For example, /dev/fct0 indicates the first ioMemory device detected on the system.

You can specify multiple ioMemory devices. For example, /dev/fct1 /dev/fct2 indicates the second and third ioMemory devices installed on the system.

| Option | Description |
|--------|---|
| -r | Force a metadata rescan. This may take an extended period of time, and is not normally required. <div> Only use this option when directed by Customer Support.</div> |
| -c | Attach only if clean. |
| -q | Quiet: disables the display of the progress bar and percentage. |

fio-beacon

Description

Lights the ioMemory device's LED(s) to locate the device. You should first detach the ioMemory device and then run `fio-beacon`.

Syntax

```
fio-beacon <device> [options]
```

where <device> is the name of the device node (/dev/fctx), where *x* indicates the card number: 0, 1, 2, etc. For example, /dev/fct0 indicates the first ioMemory device detected on the system. The device numbers are visible using `fio-status`.

| Option | Description |
|--------|--|
| -0 | Off: (Zero) Turns off the LED beacon. |
| -1 | On: Lights the LED beacon. |
| -p | Prints the PCI bus ID of the device at <device> to standard output. Usage and error information may be written to standard output rather than to standard error. |



fio-bugreport

Description

Prepares a detailed report of the device for use in troubleshooting problems. The results are saved in the /tmp directory in the file that indicates the date and time the utility was run.

Example:

```
/tmp/fio-bugreport-20100121.173256-sdv9ko.tar.bz2
```

Syntax

```
fio-bugreport
```

Notes

This utility captures the current state of the device. When a performance or stability problem occurs with the device, run the `fio-bugreport` utility and contact Customer Support at support@fusionio.com for assistance in troubleshooting.



Upload Report

For best results, do not email the bug report file. Instead please create a case (by emailing Customer Support), and then upload the report to the case using a web browser.

Sample Output

```
-bash-3.2# fio-bugreport
Collecting fio-status -a
Collecting fio-status
Collecting fio-pci-check
Collecting fio-pci-check -v
Collecting fio-read-lebmap /dev/fct0
Collecting fio-read-lebmap -x /dev/stdout/dev/fct0
Collecting fio-read-lebmap -t /dev/fct0
Collecting fio-get-erase-count/dev/fct0
Collecting fio-get-erase-count -b /dev/fct0
Collecting lspci
Collecting lspci -vvvvv
Collecting lspci -tv
Collecting messages file(s)
Collecting procfusion file(s)
Collecting lsmod
Collecting uname -a
Collecting hostname
Collecting sar -r
Collecting sar
Collecting sar -A
Collecting syslog file(s)
Collecting proc file(s)
Collecting procirq file(s)
```



```
Collecting dmidecode
Collecting rpm -qa iohome*
Collecting find /lib/modules
Please attach the bugreport tar file
/tmp/fio-bugreport-20090921.173256-sdv9ko.tar.bz2
to your support case, including steps to reproduce the problem.
If you do not have an open support case for this issue, please open a
support
case with a problem description and then attach this file to your new case.
```


For example, the filename for a bug report file named `fio-bugreport-20090921.173256-sdv9ko.tar.bz2` indicates the following:

- Date (20090921)
- Time (173256, or 17:32:56)
- Misc. information (sdv9ko.tar.bz2)

fio-detach

Description

Detaches the ioMemory device and removes the corresponding `fctxioMemory` device block device from the OS. The `fio-detach` utility waits until the device completes all read/write activity before executing the detach operation. By default, the command also displays a progress bar and percentage as it completes the detach.

 Before using this utility, ensure that the device you want to detach is **NOT** currently mounted and in use.

Syntax

```
fio-detach <device> [options]
```

where `<device>` is the name of the device node (`/dev/fctx`), where `x` indicates the card number: 0, 1, 2, etc. For example, `/dev/fct0` indicates the first ioMemory device detected on the system.

You can specify multiple ioMemory devices. For example, `/dev/fct1 /dev/fct2` indicates the second and third ioMemory devices installed on the system. You can also use a wildcard to indicate all ioMemory devices on the system. For example, `/dev/fct*`

| Option | Description |
|-----------------|---|
| <code>-q</code> | Quiet: Disables the display of the progress bar and percentage. |

Notes


Attempting to detach an ioMemory device may fail with an error indicating that the device is busy. This typically may occur if the ioMemory device is part of a software RAID (0,1,5) volume, is mounted, or some process has the device open.

The tools `fuser`, `mount`, and `lsof` can be helpful in determining what is holding the device open.





fio-format


Description


 ioMemory devices ship pre-formatted, so `fio-format` is generally not required except to change the logical size or block size of a device, or to erase user data on a device. To ensure the user data is truly erased, use `fio-sure-erase`, see [fio-sure-erase on page 54](#) for more information.

Performs a low-level format of the ioMemory device. By default, `fio-format` displays a progress-percentage indicator as it runs.

 Use this utility with care, as it deletes all user information on the device. You will be prompted as to whether you want to proceed with the format.

 Using a larger block (sector) size, such as 4096 bytes, can significantly reduce worst-case ioMemory VSL host memory consumption. However, some applications are not compatible with non-512-byte sector sizes.

 If you do not include the `-s` or `-o` options, the device size defaults to the advertised capacity. If used, the `-s` and `-o` options must include the size or percentage indicators.

 Do not interrupt the formatting! We recommend adding power backup to your system to prevent power failures during formatting. If formatting is interrupted, please contact Customer Support.


Syntax

```
fio-format [options] <device>
```

where `<device>` is the name of the device node (`/dev/fctx`), where `x` indicates the device number: 0, 1, 2, etc. For example, `/dev/fct0` indicates the first ioMemory device detected on the system. Use `fio-status` to view this number.

| Options | Description |
|--------------------------------------|---|
| <code>-b <size B K></code> | Set the block (sector) size, in bytes or kibibytes (base 2). The default is 512 bytes. For example: <code>-b 512B</code> or <code>-b 4K</code> (B in 512B is optional). |
| <code>-f</code> | Force the format size, bypassing normal checks and warnings. This option may be needed in rare situations when <code>fio-format</code> does not proceed properly. (The "Are you sure?" prompt still appears unless you use the <code>-y</code> option.) |
| <code>-q</code> | Quiet mode: Disable the display of the progress-percentage indicator. |
| <code>-s <size M G T %></code> | Set the device capacity as a specific size (in TB, GB, or MB) or as a percentage of the advertised capacity: |




| | |
|--|--|
| | <ul style="list-style-type: none">• T Number of terabytes (TB) to format• G Number of gigabytes (GB) to format• M Number of megabytes (MB) to format• % Percentage, such as 70% (the percent sign must be included) |
| <code>-o <size B K M G T %></code> | <p>Over-format the device size (to greater than the advertised capacity), where the maximum size equals the maximum physical capacity. If a percentage is used, it corresponds to the maximum physical capacity of the device. (Size is required for the <code>-o</code> option; see the <code>-s</code> option above for size indicator descriptions.)</p> <div> Before you use this option, please discuss your use case with Customer Support.</div> |
| <code>-R</code> | Disable fast rescan on unclean shutdown to reclaim some reserve capacity. |
| <code>-y</code> | Auto-answer "yes" to all queries from the application (bypass prompts). |

You must re-attach the device in order to use the ioMemory device. See [fio-attach on page 46](#) for details.

fio-pci-check

Description

Checks for errors on the PCI bus tree, specifically for ioMemory devices. This utility displays the current status of each ioMemory device. It also prints the standard PCI Express error information and resets the state.

 It is perfectly normal to see a few correctable errors when `fio-pci-check` is initially run. Subsequent runs should reveal only one or two errors during several hours of operation.

Syntax

```
fio-pci-check [options]
```

| Option | Description |
|-------------------------------|---|
| <code>-d <value></code> | 1 = Disable the link; 0 = bring the link up (Not recommended). |
| <code>-e</code> | Enable PCI-e error reporting. |
| <code>-f</code> | Scan every device in the system. |
| <code>-n</code> | Do not perform any writes to config space. Will prevent errors from being cleared. |
| <code>-o</code> | Optimize the ioMemory device PCIe link settings by increasing the maximum read request size if it is too low. |
| <code>-r</code> | Force the link to retrain. |
| <code>-v</code> | Verbose: Print extra data about the hardware. |



fio-status

Description

Provides detailed information about the installed devices. This utility operates on either `fctx` or `fiox` devices. The utility depends on running as root and having the ioMemory VSL driver loaded. If no driver is loaded, a smaller set of status information is returned.


`fio-status` provides alerts for certain error modes, such as a minimal-mode, read-only mode, and write-reduced mode, describing what is causing the condition.

Syntax

```
fio-status [<device>] [<options>]
```

where `<device>` is the name of the device node (`/dev/fctx`), where `x` indicates the card number: 0, 1, 2, etc. For example, `/dev/fct0` indicates the first ioMemory device detected on the system.

If `<device>` is not specified, `fio-status` displays information for all cards in the system. If the ioMemory VSL driver is not loaded, this parameter is ignored.

| Option | Description |
|-----------|--|
| -a | Report all available information for each device. |
| -e | Show all errors and warnings for each device. This option is for diagnosing issues, and it hides other information such as format sizes. |
| -c | Count: Report only the number of ioMemory devices installed. |
| -d | Show basic information set plus the total amount of data read and written (lifetime data volumes). This option is not necessary when the <code>\-a</code> option is used. |
| -fj | Format JSON: creates the output in JSON format. |
| -fx | Format XML: creates the output in XML format. |
| -u | Show unavailable fields. Only valid with <code>-fj</code> or <code>-fx</code> . |
| -U | Show unavailable fields and details why. Only valid with <code>-fj</code> or <code>-fx</code> . <div> Some <code>fio-status</code> fields are unavailable depending on the operating system or device. For example, some legacy fields are unavailable on newer ioMemory devices.</div> |
| -F<field> | Print the value for a single field (see the next option for field names). Requires that a device be specified. Multiple <code>-F</code> options may be specified. |
| -l | List the fields that can be individually accessed with <code>-F</code> . |



Output Change

The standard formatting of `fio-status` output has changed compared to the output from ioMemory VSL software version 2.x. This will affect any custom management tools that used the output of this utility.

Basic Information: If no options are used, `fio-status` reports the following basic information:

- Number and type of devices installed in the system
- ioMemory VSL software version

Adapter information:

- Adapter type
- Product number
- External power status
- PCIe power limit threshold (if available)
- Connected ioMemory devices

Block device information:

- Attach status
- Product name
- Product number
- Serial number
- PCIe address and slot
- Firmware version
- Size of the device, out of total capacity
- Internal temperature (average and maximum, since ioMemory VSL software load) in degrees Centigrade
- Health status: healthy, nearing wearout, write-reduced or read-only
- Reserve capacity (percentage)
- Warning capacity threshold (percentage)

Data Volume Information: If the `-d` option is used, the following data volume information is reported *in addition* to the basic information:

- Physical bytes written
- Physical bytes read

All Information: If the `-a` option is used, all information is printed, which includes the following information *in addition* to basic and data volume information:

Adapter information:

- Manufacturer number
- Part number
- Date of manufacture
- Power loss protection status



- PCIe bus voltage (avg, min, max)
- PCIe bus current (avg, max)
- PCIe bus power (avg, max)
- PCIe power limit threshold (watts)
- PCIe slot available power (watts)
- PCIe negotiated link information (lanes and throughput)


Block device information:


- Manufacturer's code
- Manufacturing date
- Vendor and sub-vendor information
- Format status and sector information (if device is attached)
- FPGA ID and Low-level format GUID
- PCIe slot available power
- PCIe negotiated link information
- Card temperature, in degrees Centigrade
- Internal voltage (avg and max)
- Auxiliary voltage (avg and max)
- Percentage of good blocks, data and metadata
- Lifetime data volume statistics
- RAM usage


Error Mode Information: If the ioMemory VSL software is in minimal mode, read-only mode, or write-reduced mode when `fio-status` is run, the following differences occur in the output:

- Attach status is "Status unknown: Driver is in MINIMAL MODE:"
- The reason for the minimal mode state is displayed (such as "Firmware is out of date. Update firmware.")
- "Geometry and capacity information not available." is displayed.
- No media health information is displayed.


fio-sure-erase

 As a best practice, do not use this utility if there are any ioMemory devices installed in the system that you do not want to clear or purge. First remove any devices that you do not want to accidentally erase. Once the data is removed with this utility it is gone forever. **It is not recoverable.**

 Before you use this utility, be sure to back up any data that you wish to preserve.

 After using `fio-sure-erase`, format the device using `fio-format` before using the device again, see [fio-format on page 50](#).



 If the device is in Read-only mode, perform a format using `fio-format` before running `fio-sure-erase`. If the device is in Minimal mode, then `fio-sure-erase` cannot erase the device. Updating the firmware may take the device out of Minimal Mode. If the device remains in Minimal mode, contact Customer Support at support@fusionio.com for further assistance.

In order to run `fio-sure-erase`, the block device **must be detached**. See [fio-detach on page 49](#) section for more information.

Description

The `fio-sure-erase` is a command-line utility that securely removes data from ioMemory devices. It complies with the "Clear" and "Purge" level of destruction from the following standards:


1. DOD 5220.22-M - Comply with instructions for Flash EPROM
2. NIST SP800-88- Comply with instructions for Flash EPROM


For information regarding certifications please see <http://www.fusionio.com/sureerase/>. See below for more information on Clear and Purge support.


Syntax

```
fio-sure-erase [options] <device>
```

Where `<device>` is the name of the device node (`/dev/fctx`), where `x` indicates the card number: 0, 1, 2, etc. For example, `/dev/fct0` indicates the first ioMemory device detected on the system. Use `fio-status` to view this device node, see [fio-status on page 52](#).

 **Products with Multiple Devices**
`fio-sure-erase` works on individual ioMemory devices. For example, if you are planning to purge an ioDrive Duo device, you will need to perform this operation on each of the product's two ioMemory devices.

| Option | Description |
|-----------------|--|
| <code>-p</code> | Purge instead of Clear: performs a write followed by an erase. For more information on Purge, see below. <div>  Purging the device may take hours to accomplish, depending on the size of the device that needs to be purged. </div> |
| <code>-y</code> | No confirmation: do not require a yes/no response to execute the utility. |
| <code>-t</code> | Do not preserve current format parameters, including device and sector size (reset to default). |
| <code>-q</code> | Quiet: do not display the status bar. |

 If you run `fio-sure-erase` with no options, a Clear is performed. For more information, see below.

When the utility completes, each block of memory consists of uniform 1 bits or 0 bits.



Clear Support

A "Clear" is the default state of running `fio-sure-erase` (with no options), and refers to the act of performing a full low-level erase (every cell pushed to "1") of the entire NAND media, including retired erase blocks.

Metadata that is required for operation will not be destroyed (media event log, erase counts, physical bytes read/written, performance and thermal history), but any user-specific metadata will be destroyed.

The following describes the steps taken in the Clear operation:

1. Creates a unity map of every addressable block (this allows `fio-sure-erase` to address every block, including previously unmapped bad blocks).
2. For each block, performs an erase cycle (every cell is pushed to "1").
3. Restores the bad block map.
4. Formats the device (the purpose of this is to make the device usable again, the utility erases all of the headers during the clear).

Purge Support

A "Purge" is implemented by using the `-p` option with `fio-sure-erase`. Purge refers to the act of first overwriting the entire NAND media (including retired erase blocks) with a single character (every cell written to logical "0"), and then performing a full chip erase (every cell pushed to "1") across all media (including retired erase blocks).

Metadata that is required for operation will **not** be destroyed (media event log, erase counts, physical bytes read/written, performance and thermal history), but any user-specific metadata will be destroyed.

The following describes the steps taken in the Purge operation:

1. Creates a unity map of every addressable block (this allows `fio-sure-erase` to address every block, including previously unmapped bad blocks).
2. For each block, performs a write cycle (every cell written to "0").
3. For each block, performs an erase cycle (every cell pushed to "1").
4. Restores the bad block map.
5. Formats the drive (the purpose of this is to make the drive usable again, the utility erases all of the headers during the clear).

fio-update-iodrive



You should back up the data on the ioMemory device prior to any upgrade as a precaution.


Description


Updates the ioMemory device's firmware. This utility scans the PCIe bus for all ioMemory devices and updates them. A progress bar and percentage are shown for each device as the update completes.





It is extremely important that the power not be turned off during a firmware upgrade, as this could cause device failure. If a UPS is not already in place, consider adding one to the system prior to performing a firmware upgrade.





 Note that when running multiple firmware upgrades in sequence, it is critical to load the ioMemory VSL driver after each firmware upgrade step. Otherwise the on-device format will not be changed, and there will be data loss.

 Do not use this utility to downgrade the ioMemory device to an earlier version of the firmware. Doing so may result in data loss and void your warranty. Contact Customer Support at <http://support.fusionio.com> if you have issues with your upgrade.

 The default action (without using the `-d` or `-s` option) is to upgrade all ioMemory devices with the firmware contained in the `fusion_<version>.<date>.fff` firmware archive file. Confirm that all devices need the upgrade prior to running the update. If in doubt, use the `-p` (Pretend) option to view the possible results of the update.

 You must detach all ioMemory devices before updating the firmware.

 **Upgrade Path**
There is a specific upgrade path that you must take when upgrading ioMemory device. Consult the *ioMemory VSL Release Notes* for this ioMemory VSL software release before upgrading ioMemory devices.

 If you receive an error message when updating the firmware that instructs you to update the midprom information, contact Customer Support.


To update one or more specific devices:

- If the ioMemory VSL driver is loaded, use the `-d` option with the device number.


Syntax

```
fio-update-iodrive [options] <firmware-path>
```

where `<firmware-path>` is the full path to the firmware archive file `fusion_<version>.<date>.fff` available at <http://support.fusionio.com>. If you downloaded the `.fff` firmware archive file, then the firmware is most likely with the other downloaded packages. If you installed the firmware from the firmware package, the default path is `/usr/share/fio/firmware/`. This parameter is required.

| Option | Description |
|-----------------|---|
| <code>-d</code> | <p>Updates the specified devices (by <code>fctx</code>, where <code>{i}x{i}</code> is the number of the device shown in <code>fio-status</code>). If this option is not specified, all devices are updated.</p> <div> Use the <code>-d</code> or <code>-s</code> option with care, as updating the wrong ioMemory device could damage your device.</div> |



| Option | Description |
|---------|---|
| -f | Force upgrade (used when directed by Customer Support). If the ioMemory VSL driver is not loaded, this option also requires the -s option. <div> Use the -f option with care, as it could damage your card.</div> |
| -l | List the firmware available in the archive. |
| -p | Pretend: Shows what updates would be done. However, the actual firmware is not modified. |
| -c | Clears locks placed on a device. |
| -q | Runs the update process without displaying the progress bar or percentage. |
| -y | Confirm all warning messages. |
| -s | Updates the devices in the specified slots using '*' as a wildcard for devices. The slots are identified in the following PCIe format (as shown in lspci): <div>[[[<domain>]:<bus>]:<slot>].[<func>]]</div> |
| --split | Split the ioMemory device into virtual devices. |
| --merge | Merge the virtual devices of an ioMemory device. |

If you arrived at this section from [Upgrading the Firmware on page 18](#), you should return to that section.



Appendix B - Monitoring the Health of Devices

This section describes how the health of ioMemory devices can be measured and monitored in order to safeguard data and prolong device lifetime.

Health Metrics

The ioMemory VSL software manages block retirement using pre-determined retirement thresholds. The ioSphere software and the `fio-status` utilities show a health indicator that starts at 100 and counts down to 0. As certain thresholds are crossed, various actions are taken.

At the 10% healthy threshold, a one-time warning is issued. See [Health Monitoring Techniques on page 59](#) for methods for capturing this alarm event.


At 0%, the device is considered unhealthy. It enters *write-reduced* mode, which somewhat prolongs its lifespan so data can be safely migrated off. In this state the ioMemory device behaves normally, except for the reduced write performance.


After the 0% threshold, the device will soon enter *read-only* mode -- any attempt to write to the ioMemory device causes an error. Some filesystems may require special mount options in order to mount a read-only block device in addition to specifying that the mount should be read-only.

For example, under Linux, `ext3` requires that "`-o ro,noload`" is used. The "`noload`" option tells the filesystem to not try and replay the journal.

Read-only mode should be considered a final opportunity to migrate data off the device, as device failure is more likely with continued use.

The ioMemory device may enter failure mode. In this case, the device is offline and inaccessible. This can be caused by an internal catastrophic failure, improper firmware upgrade procedures, or device wearout.

 For service or warranty-related questions, contact the company from which you purchased the device.

 For products with multiple ioMemory devices, these modes are maintained independently for each device.

Health Monitoring Techniques

`fio-status -a`: Output from the `fio-status` utility (using the `-a` option) shows the health percentage and device state. These items are referenced as "Media status" in the sample output below.

```
Found 3 ioMemory devices in this system
Fusion-io driver version: 3.x.x build xxxx

Adapter: Single Adapter
          Fusion-io ioDrive 1.30TB, Product Number:F00-001-1T30-CS-0001,
          SN:1133D0248, FIO SN:1134D9565
```



```
...
Media status: Healthy; Reserves: 100.00%, warn at 10.00%; Data: 99.12%
Lifetime data volumes:
  Physical bytes written: 6,423,563,326,064
  Physical bytes read   : 5,509,006,756,312
```

The following Health Status messages are produced by the `fio-status` utility:

- Healthy
- Read-only
- Reduced-write
- Unknown

ioSphere software: In the Device Report tab, look for the Reserve Space percentage in the right column. The higher the percentage, the healthier the drive is likely to be.

Software RAID and Health Monitoring

Software RAID stacks are typically designed to detect and mitigate the failure modes of traditional storage media. The ioMemory device attempts to fail as gracefully as possible, and these new failure mechanisms are compatible with existing software RAID stacks. An ioMemory device in a RAID group will fail to receive data at a sufficient rate if: a) the device is in a write-reduced state, and b) it is participating in a write-heavy workload. In this case, the device will be evicted from the RAID group. A device in read-only mode will be evicted when write I/Os are returned from the device as failed. Catastrophic failures are detected and handled just as though they are on traditional storage devices.




Appendix C - Using Module Parameters

The following table describes the module parameters you can set by editing the `/etc/modprobe.d/iomemory-vsl.conf` file and changing their values.


Each module parameter in the configuration file must be preceded by options `iomemory-vsl`. The `/etc/modprobe.d/iomemory-vsl.conf` file has some example parameters that are commented out. You may use these examples as templates and/or uncomment them in order to use them.

 These changes must be completed before the ioMemory VSL software is loaded in order to take effect.

| Module Parameter | Default (min/max) | Description |
|--|---------------------|--|
| <code>auto_attach</code> | 1 (0, 1) | 1 (default) = Always attach the device on driver load. 0 = Don't attach the device on driver load. |
| <code>external_power_override</code> | No devices selected | Allows selected devices to draw full power from the PCIe slot. Where the <value> for this parameter is a comma-separated list of adapter serial numbers.  Use with care, see Enabling PCIe Power Override on page 20 for more information. |
| <code>fio_dev_wait_timeout_secs</code> | 30 | Number of seconds to wait for <code>/dev/fio*</code> files to show up during driver load. For systems not using <code>udev</code> , this should be set to 0 to disable the timeout and avoid an unneeded pause during driver load. |
| <code>force_minimal_mode</code> | 0 | 1 = Force minimal mode on the device. 0 = Do not force minimal mode on the device. |
| <code>numa_node_forced_local</code> | 0 | 1 = Enable, 0 = Disable. See NUMA Configuration on page 63 for more information. |
| <code>numa_node_override</code> | Nothing Selected | A list of <affinity specification> couplets that specify the affinity settings of all devices in the system. Each item in the couplet is separated by a colon, and each couplet set is separated by a comma. Where each <affinity specification> couplet has the following syntax: |



| Module Parameter | Default (min/max) | Description |
|--------------------|---------------------|--|
| | | <div><device-id>=<node-number></div> <p>See NUMA Configuration on page 63 for more information on using this parameter.</p> |
| parallel_attach | 1 | 1 = Enable parallel attach of multiple devices. 0 = Disable parallel attach of multiple devices. |
| preallocate_memory | No devices selected | For the selected devices, pre-allocate all memory necessary to have the drive usable as swap space. Where the <value> for this parameter is a comma-separated list of device serial numbers. |
| tintr_hw_wait | 0 (0, 255) | Interval (microseconds) to wait between hardware interrupts. Also known as interrupt coalescing. 0 is off. |
| use_workqueue | 0 (0 or 3) | Linux only: 3 = use standard OS I/O elevators; 0 = bypass. |

 Other than `external_power_override` and `preallocate_memory`, module parameters are global — they apply to all ioMemory devices in the computer.



Appendix D - NUMA Configuration

About NUMA Architecture

Servers with a NUMA (Non-Uniform Memory Access) architecture may require special installation instructions in order to maximize ioMemory device performance. This includes most multi-socket servers.

On some servers with NUMA architecture, during system boot the BIOS will not associate PCIe slots with the correct NUMA node. Incorrect mappings result in inefficient I/O handling that can significantly degrade performance. Here are two methods for correcting the NUMA affinity:

- **numa_node_forced_local Parameter:** Forces I/O completions to happen on the same CPU that is running VSL processes for a particular device. This parameter is simple to implement (enabled or disabled), and is persistent.
- **numa_node_override Parameter:** Used to manually assign devices to specific NUMA nodes. This parameter is more complex. For optimal implementation, you should understand the NUMA architecture of the system in order to assign devices to the nodes that are closely linked to the device's PCIe slot. This parameter is persistent until any new I/O devices are installed in the system (thus changing the PCIe bus numbers in the system).

Using the numa_node_forced_local Parameter

The `numa_node_forced_local` parameter is either enabled or disabled, and therefore does not offer as much user control as other options. It is persistent and it is enabled by modifying the `/etc/modprobe.d/iomemory-vsl.conf` file and editing or adding the following line:

```
numa_node_forced_local=1
```

This parameter forces an I/O completion for a particular device to happen on a CPU within the local `numa_node` that the other ioMemory VSL processes are running on (rather than trying to complete an I/O on the CPU that the host issued it on). Because the I/O completions are grouped with other ioMemory VSL processes, they are less likely to compete with processes from other device drivers.

This parameter may or may not improve performance depending on your configuration and workloads. You should test this parameter with your use case to determine if it improves performance.

Using the numa_node_override Parameter

Use this parameter to map devices with specific NUMA nodes.



The example below shows the final implementation of custom affinity settings. This implementation required an analysis of the specific system, including the system architecture, type and number of ioMemory devices installed, and the particular PCIe slots that were used. Your particular circumstances will require a custom analysis of your set-up. This analysis requires understanding of your system's NUMA architecture compared to your particular installation.



Your actual settings may be different than the example below, depending on your server configuration. In order to create the correct settings for your specific system, use `fio-status` to list all of the devices and determine the `<device-id>` (see below). Then use the example below of setting the `numa_node_override` parameter as a template and modify it for your particular system.

Determining the Device ID

You should present each `<device-id>` in the following format:

```
<domain>:<bus>:<device>.<function>
```

Typically the domain is 0000, consult your server documentation to determine your device ID. The remainder of the device ID string is visible in `fio-status` output. For example:

```
# fio-status
Found 2 ioMemory devices in this system
...
    PCI:04:00.0
...
    PCI:15:00.0
```

In the example above the device IDs would be 0000:04:00.0 and 0000:15:00.0 on a system that had a domain of 0000.



Note that the PCI device ID, including the bus number, may change if you change any of the PCI devices in the system. For example, if you add a network card or another ioMemory device. If the device ID changes, you will have to update the configuration.

numa_node_override Parameter

Configuring your ioMemory devices for servers with NUMA architecture requires the use of the `numa_node_override` parameter by modifying the `iomemory-vsl.conf` file.

The `numa_node_override` parameter is a list of `<affinity specification>` couplets that specify the affinity settings of all devices in the system. Each item in the couplet is separated by an equal sign (=), and each couplet set is separated by a comma.

Syntax:

```
numa_node_override=<affinity specification>[,<affinity specification>...]
```

Where each `<affinity specification>` has the following syntax:

```
<device-id>=<node-number>
```

Simple Example:

```
numa_node_override=0000:04:00.0=1,0000:1d:00.0=0,0000:05:00.0=2,
0000:1e:00.0=3
```



Has the effect of creating :

| <device-id> | Node/Group | Processor Affinity |
|--------------|------------|--------------------------|
| 0000:04:00.0 | node 1 | all processors in node 1 |
| 0000:1d:00.0 | node 0 | all processors in node 0 |
| 0000:05:00.0 | node 2 | all processors in node 2 |
| 0000:1e:00.0 | node 3 | all processors in node 3 |

Advanced Configuration

If your server has multiple NUMA nodes and multiple ioMemory devices installed, you will need to make sure that the ioMemory devices are spread out among the various nodes.

While it may be optimal to pair devices to nodes that are electronically closer each device's PCIe slot (which would require an advanced understanding of your server's NUMA architecture and an analysis of the device installation), just simply spreading out all of the devices' node affinity among the available nodes should result in improved performance.

In the example above the device IDs would be 0000:04:00.0 and 0000:15:00.0 on a system that had a domain of 0000.



Note that the PCI device ID may change if you change any of the PCI devices in the system. For example, if you add a network card or another ioMemory device. If the device ID changes, you will have to update the configuration.


| <device-id> | Node/Group | Processor Affinity |
|--------------|------------|--------------------------|
| 0000:04:00.0 | node 1 | all processors in node 1 |
| 0000:1d:00.0 | node 0 | all processors in node 0 |
| 0000:05:00.0 | node 2 | all processors in node 2 |
| 0000:1e:00.0 | node 3 | all processors in node 3 |





Appendix E - Upgrading Devices from VSL 2.x to 3.x

This version of the ioMemory VSL software supports new features, including the latest generation of ioMemory architecture and improved Flashback protection. These features require the latest version of the ioMemory device firmware. Every ioMemory device in a system running 3.1.x or later must be upgraded to the latest version of the firmware.

For example, if you have a system running 2.x ioMemory VSL software with ioDrive devices previously installed, and you want to install new ioDrive2 devices (that require the latest version of the firmware), then you will need to upgrade all of the existing devices to the latest firmware version.

 You cannot revert a device's firmware to an earlier version once you have upgraded the device (without voiding your warranty). If you experience problems with your upgrade, please contact Customer Support at support@fusionio.com.


 Upgrading devices (previously configured for VSL 2.x.x) to work with VSL 3.x.x will require a low-level media format of the device. No user data will be maintained during the process. Be sure to backup all data as instructed.

 **Upgrade Path**
Depending on the current firmware version of your devices, you may need to upgrade your device's firmware multiple times in order to preserve internal structures. Consult the ioMemory VSL software for the upgrade path. Visit <http://support.fusionio.com> for all of the required software and firmware versions.

For more information on upgrading from one version to the next, see the *ioMemory VSL Release Notes* (available at <http://support.fusionio.com>) for the version you will upgrade the device to. Then follow the upgrade instructions in that version's user guide for your operating system (including the firmware update instructions).

Upgrade Procedure


Be sure to follow the upgrade path in the *ioMemory VSL Release Notes*. Make sure that all previously installed ioDrive devices are updated with the appropriate firmware.

 If you plan to use ioDrive devices and ioDrive2 devices in the same host, perform this upgrade on all existing ioDrive devices **before** installing the new ioDrive2 devices.



1. Prepare each existing ioDrive device for upgrade.

- a. Backup user data on each device.

 The upgrade process will require a low-level media format of the device. No user data will be maintained during the process; be sure to make a complete backup.

Use a backup method of your choice. For best results, use software and backup devices that have proven effective in the past. Do not backup the data onto another ioMemory device on the same system. The back up must be to a local disk or to an externally attached volume.

- b. Run the `fio-bugreport` utility and save the output. This will capture the device information for each device in the system. This device information will be useful in troubleshooting any upgrade issues. Sample command:

```
fio-bugreport
```

- c. Detach ioDrive devices, for example:

```
fio-detach /dev/fct*
```

For more information, see [fio-detach on page 49](#).

2. Unload the current ioMemory VSL driver, for example:

```
$ modprobe -r iomemory-vsl
```

3. Uninstall the 2.x ioMemory VSL software.

- a. Uninstall the utilities:

- Sample RPM command:

```
$ rpm -e fio-util fio-snmp-agentx fio-common fio-firmware  
iomanager-gui iomanager-jre libfio libfio-doc libfusionjni  
fio-sysvinit fio-smis fio-snmp-mib libfio-dev
```

- Sample DEB command:

```
$ dpkg -r fio-util fio-snmp-agentx fio-common fio-firmware  
iomanager-gui iomanager-jre libfio libfio-doc libfusionjni  
fio-sysvinit fio-smis fio-snmp-mib libfio-dev
```

- b. To uninstall the software, you must specify the kernel version of the package you are uninstalling. Run the appropriate command to find the installed packages:

- RPM command:

```
$ rpm -qa | grep -i iomemory
```



- DEB command:

```
$ dpkg -l | grep -i iomemory
```

- Sample output:

```
iomemory-vsl-2.6.18-194.el5-2.2.2.82-1.0
```

- c. Uninstall the ioMemory VSL software by running a command similar to this example (specify the kernel version of the package you wish to uninstall):

- Sample RPM command:


```
$ rpm -e iomemory-vsl-2.6.18-194.el5-2.2.0.82-1.0
```

- Sample DEB command:

```
$ dpkg -r iomemory-vsl-2.6.32-24-server
```

4. Install the new ioMemory VSL software and related packages.

- a. Download the ioMemory VSL software binary package for your kernel and all supporting packages at <http://support.fusionio.com>

 If you don't see a binary for your kernel, follow the instructions in the **Building the ioMemory VSL from Source** section of [Installing RPM Packages on page 9](#) or [Installing DEB Packages on page 12](#). To see your current kernel version, run: `uname -r`

- b. Install the ioMemory VSL software and utilities using the appropriate commands:

- RPM commands:

```
rpm -Uvh iomemory-vsl-<kernel-version>-<VSL-version>.x86_64.rpm  
rpm -Uvh lib*.rpm  
rpm -Uvh fio*.rpm
```

- DEB commands:

```
dpkg -i iomemory-vsl-<kernel-version>-<VSL-version>_amd64.deb  
dpkg -i lib*.deb  
dpkg -i fio*.deb
```

- See [Installing RPM Packages on page 9](#) or [Installing DEB Packages on page 12](#) for full instructions on installing those packages.

- c. Reboot the system.

5. Update the firmware on each device to the latest version using `fio-update-iodrive`.



Prevent Power Loss

Take measures to prevent power loss during the update, such as a UPS. Power loss during an update may result in device failure. For all warnings, alerts, and options pertaining to this utility, see [fio-update-iodrive on page 56](#).

Sample syntax:

```
fio-update-iodrive <firmware-path>
```

Where <firmware-path> is the full path to the firmware archive file (fusion_<version>.<date>.fff) available at <http://support.fusionio.com>. This command will update all of the devices to the selected firmware. If you wish to update specific devices, consult [fio-update-iodrive on page 56](#) for more options.

6. Reboot the system
7. Load the ioMemory VSL software, for example:

```
$ modprobe iomemory-vsl
```

For more information, see [Loading the ioMemory VSL Driver on page 15](#).



If run, `fio-status` will warn that the upgraded devices are missing a lebmap. This is expected, and will be fixed in the next step.

8. Destructive Step



Running `fio-format` in the next step will erase the entire device, including user data. Once this format is started, the device cannot be downgraded to the 2.x driver without voiding your warranty. If you experience problems with your upgrade, please contact Customer Support at support@fusionio.com.

9. Format each device using `fio-format`, for example:

```
fio-format <device>
```

You will be prompted to confirm you wish to erase all data on the device.



The format may take an extended period of time, depending on the wear on the device.

10. Attach all ioDrive devices, for example:

```
fio-attach /dev/fct*
```

11. Check the status of all devices using `fio-status`, for example:

```
fio-status -a
```



Your ioDrive devices have now been successfully upgraded for this version of the ioMemory VSL software. You may now install any ioDrive2 devices.



Fusion Powered Support

We offer Fusion-io Customer Services and Support by phone, e-mail and on the Web. For the most up-to-date contact information visit at <http://support.fusionio.com> .

E-Mail

Our support e-mail address is: support@fusionio.com

E-mail is the fastest way to get simple questions answered. Please give a detailed description of your problem with your complete contact information (name, phone number, email address, location address).

Warranty Support

Warranty Support is available via support@fusionio.com and at <http://support.fusionio.com> .

Telephone Support

ioFX Support

North America: (855) 322-5767

Enterprise Support

North America: (877) 816-5740

Country Numbers

For product support outside of North America, please use the number for the country/region closest to you. If that is not possible, please contact North America at (801) 424 5474.

| | | | | | |
|--|--------------------------------|-------------------------------|------------------------------------|-------------------------------------|--|
| Australia 1800 353 941 (02) 8278 1489 | Belgium 02 700 74 86 | China 40-08866109 | Denmark 4331 4999 | Finland 097 251 9979 | France 01 57 32 48 90 |
| Germany (069) 17 07 76 790 | Hong Kong 3071 3587 | Italy 02 23331509 | Japan (03) 6743-9765 | Luxembourg (224) 87 19 84 | Mexico 01 882 816 5740 |
| Netherlands 070 7703993 | Norway 23 02 49 99 | Singapore 6818 5692 | South Korea 02 3483 6689 | Sweden 08 593 663 99 | United Kingdom (020) 3564 9935 |

Web

Go online to find tips, FAQs, and troubleshooting help, or download the latest user guides, software, and support packages at: <http://support.fusionio.com> .