



# **Intel® Xeon Phi™ Processor Software**

**User's Guide for Linux\***

---

Revision 1.0  
January 31, 2018

## Disclaimer and legal information

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps. The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting: <http://www.intel.com/design/literature.htm>.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at <http://www.intel.com/> or from the OEM or retailer.

No computer system can be absolutely secure.

Intel, Xeon, Xeon Phi and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

Intel does not warrant or guarantee the performance or compatibility of third party commercial products. Reference in this site to any specific commercial product, process, or service, is for the information and convenience of the public, and does not constitute endorsement, or recommendation by Intel.

\*Other names and brands may be claimed as the property of others. Copyright© 2018, Intel Corporation. All rights reserved.

# Contents

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduction</b>                                      | <b>5</b>  |
| 1.1      | Notational Conventions . . . . .                         | 5         |
| 1.2      | Terminology . . . . .                                    | 5         |
| 1.3      | Additional documentation . . . . .                       | 6         |
| <b>2</b> | <b>Software overview</b>                                 | <b>7</b>  |
| 2.1      | The hwloc package . . . . .                              | 7         |
| 2.2      | The cpuid package . . . . .                              | 7         |
| 2.3      | The memkind library . . . . .                            | 7         |
| 2.4      | The micperf package . . . . .                            | 8         |
| 2.5      | The sysdiag package . . . . .                            | 8         |
| 2.6      | The zonesort package . . . . .                           | 8         |
| <b>3</b> | <b>Installation, upgrade and uninstallation</b>          | <b>9</b>  |
| 3.1      | Prerequisites and supported operating systems . . . . .  | 9         |
| 3.1.1    | Root access . . . . .                                    | 9         |
| 3.2      | Installation procedure . . . . .                         | 10        |
| 3.2.1    | Software distribution . . . . .                          | 10        |
| 3.2.2    | RHEL*/CentOS* . . . . .                                  | 10        |
| 3.2.3    | SLES* . . . . .  | 11        |
| 3.2.4    | Ubuntu* . . . . .  | 11        |
| 3.2.5    | Uninstall . . . . .                                      | 12        |
| 3.2.6    | Additional remarks . . . . .                             | 12        |
| 3.3      | Rebuilding source packages . . . . .                     | 12        |
| <b>4</b> | <b>Virtualization</b>                                    | <b>13</b> |
| 4.1      | QEMU installation remarks . . . . .                      | 13        |
| 4.2      | Configuration . . . . .                                  | 13        |
| 4.3      | QEMU raw command . . . . .                               | 14        |
| 4.3.1    | Libvirt xml configuration file . . . . .                 | 14        |
| 4.3.2    | Adding an Intel® OPA card to a virtual machine . . . . . | 14        |
| <b>5</b> | <b>Zonesort</b>  | <b>16</b> |
| 5.1      | Installing <i>zonesort</i> . . . . .                     | 16        |
| 5.2      | Using <i>zonesort</i> . . . . .                          | 17        |
| 5.3      | Administration . . . . .                                 | 17        |
| 5.4      | Debugging . . . . .                                      | 18        |
| <b>6</b> | <b>Huge pages</b>  | <b>19</b> |
| 6.1      | Huge pages on RHEL*/CentOS* . . . . .                    | 19        |
| 6.2      | Huge pages on SLES* . . . . .                            | 19        |
| 6.3      | Allocating all MCDRAM to 1G Pages . . . . .              | 20        |
| <b>7</b> | <b>References</b>  | <b>21</b> |

## Revision History

| Revision | Date           | Description   |
|----------|----------------|---|
| 1.0      | January 2018   | Document update for the release of Intel® Xeon Phi™ processor software version 2.3. |
| 0.7      | January 2018   | Added additional information about OS support.                                      |
| 0.6      | January 2018   | Added information about Intel® Xeon Phi™ processor x200 product family.             |
| 0.5      | October 2017   | Initial release of the L <sup>A</sup> T <sub>E</sub> X version of the user's guide. |
| 0.4      | September 2017 | Updated brand names throughout the document.  |
| 0.3      | August 2017    | Document update for the release of Intel® Xeon Phi™ processor software version 2.0. |
| 0.2      | June 2016      | Removed redundant and/or unnecessary sections.                                      |
| 0.1      | June 2016      | Initial draft of the user's guide.  |

# Chapter 1

## Introduction

Intel® Xeon Phi™ processor software is a set of software and utilities that enable functionalities of the Intel® Xeon Phi™ product family. This document will allow its readers to understand and utilize those features. This paper is meant to serve as a guide; usage and options are subjective to the customer's needs.

### 1.1 Notational Conventions

This document uses the following notational conventions.

|                                  |  |
|----------------------------------|--|
| <i>zypper rm &lt;package&gt;</i> | Commands and their arguments in prose sections are italicized.   |
| <i>packages/x86_64/core</i>      | Files and directories in prose sections are italicized.  |
| command                          | Code and commands entered by the user. A backslash symbol: \ indicates that command is continued in the next line. |
| output                           | Terminal output by the computer.   |
| [host]\$                         | Commands that do not require root privileges.  |
| [host]#                          | Commands that require root privileges.   |

### 1.2 Terminology

|                 |   |
|-----------------|---|
| DTS             | Developer Tool Set  |
| EDAC            | Error Detection and Correction infrastructure in Linux* kernel which detects hardware problems.   |
| gcc             | The GNU C Compiler collection   |
| gdb             | The GNU Debugger  |
| MCDRAM          | High Bandwidth memory found in the processor package.   |
| MCE             | Machine Check Exception   |
| Intel® MKL      | Intel® Math Kernel Library  |
| PMU             | Performance Monitoring Unit - a set of counters used to understand events happening inside a CPU. |
| RHEL*           | Red Hat* Enterprise Linux*  |
| SLES*           | SUSE* Linux* Enterprise Server  |
| Upstream kernel | The Linux* kernel source code available at <a href="http://www.kernel.org">www.kernel.org</a> .   |
| Memkind         | A helper library that allows direct memory allocation in the MCDRAM.                              |

## 1.3 Additional documentation

| Document   | Location  |
|--|---|
| Micperf User's Guide, <i>README.txt</i> and <i>INSTALL.txt</i> | <i>/usr/share/doc/packages/micperf</i>  |
| <i>xpps-readme.txt</i>   | <i>xpps-&lt;version&gt;/&lt;os_version&gt;/doc</i>  |
| Offload Over Fabric User's Guide                               | <a href="https://software.intel.com/en-us/articles/xeon-phi-software">https://software.intel.com/en-us/articles/xeon-phi-software</a> |

# Chapter 2

## Software overview

The packages included in Intel® Xeon Phi™ processor software are meant to enable core functionalities of the Intel® Xeon Phi™ product family. If older versions of packages listed in this section are already installed on your host OS, they will be replaced.

### 2.1 The hwloc package

The *Portable Hardware Locality (hwloc)* provides hardware information, including NUMA memory nodes, shared caches, processor sockets, processor cores and processing units. It is mainly used by other applications so that they can utilize available hardware efficiently. For more information visit <https://www.open-mpi.org/projects/hwloc/>.

The *hwloc* package is distributed for RHEL\* and CentOS\*.

### 2.2 The cpuid package

*Cpuid* is a user space tool that provides an interface for querying information about x86 CPUs.

The *cpuid* package is only distributed for SLES\*.

### 2.3 The memkind library

The *memkind* library is a user-extensible heap manager that provides an efficient allocation mechanism for multithreaded applications and supports high bandwidth memory (MCDRAM).

The standard *memkind* API provides a set of standard heap management functions, each one prefixed by *memkind\_\**. Additional parameters specify the heap management “*kind*”. The standard API also includes functionality for managing *kinds*, error handling and debugging. To find out more about the *memkind* API please refer to its man page or to the *usr/share/doc/memkind/README* file.

Further reference is available in the *Intel® Xeon Phi™ Processor Programming and Leveraging High Bandwidth Memory whitepaper rev. 0.5*. This document can be downloaded from Intel® IPS or CDI Doc #570827. The source code repositories can be found at <http://memkind.github.io/memkind/>.

## 2.4 The micperf package

*Micperf* incorporates a variety of benchmarks into a simple and unified user experience. The user interface consists of five executables: one for execution of benchmarks (*micprun*), and four that interpret the output of the first one. The results can be displayed as professional quality plots, human readable text or comma separated values that can be easily imported into a variety of other applications.

The *micprun* executable, the primary application in the *micperf* package, executes the following benchmarks: Intel® MKL<sup>1</sup> *SMP Linpack*<sup>2</sup>, Intel® MKL *HPLinpack*, Intel® MKL *HPCG*, Intel® MKL *SGEMM*, Intel® MKL *DGEMM*, *STREAM*<sup>3</sup>, *Deepbench convolutions*<sup>4</sup> (*libxsmm\_layer*, *std\_conv\_bench*) and *fio*<sup>5</sup>.

These benchmarks were carefully chosen to demonstrate performance in all of the major bottlenecks in the system.

For more information please refer to the *micperf\_users\_guide.pdf* document included in the Intel® Xeon Phi™ processor software release package.

## 2.5 The sysdiag package

The *sysdiag* package contains the *SysDiag* processor diagnostics tool. *SysDiag* can monitor DDR, MCDRAM, PCI-E information, CPU temperature and CPU performance status data.

## 2.6 The zonesort package

The *zonesort* package contains the *zonesort* kernel module which prevents cache performance degradation resulting from increased number of cache collisions caused by memory fragmentation. Refer to section 5 for more information.

§

---

<sup>1</sup>Intel® Math Kernel Library (Intel® MKL).

<sup>2</sup>Jack Dongarra, Piotr Luszczek, and Antoine Petit. The linpack benchmark: past, present and future. *Concurrency and Computation: Practice and Experience*, 15(9):803–820, 2003

<sup>3</sup>John D. McCalpin. *Stream: Sustainable memory bandwidth in high performance computers*. Technical report, University of Virginia, Charlottesville, Virginia, 1991–2007. A continually updated technical report.

<sup>4</sup><https://github.com/baidu-research/DeepBench>

<sup>5</sup><https://github.com/axboe/fio>



## Chapter 3

# Installation, upgrade and uninstallation

This chapter describes Intel® Xeon Phi™ processor software installation and configuration.

**Note:** Before proceeding with installation, please read through this chapter to ensure all required components and facilities are available. Following these installation steps in the presented order is strongly recommended.

### 3.1 Prerequisites and supported operating systems

The target system must contain at least one Intel® Xeon Phi™ processor x200 or Intel® Xeon Phi™ processor x205. The table below lists major Linux\* distributions Intel® Xeon Phi™ processor software was validated against.

| Supported OS version                  | Kernel version            |
|---------------------------------------|---------------------------|
| CentOS* Linux* 7 (1708)               | 3.10.0-693.5.2.el7.x86_64 |
| Red Hat* Enterprise Linux* 7.4        | 3.10.0-693.5.2.el7.x86_64 |
| SUSE* Linux* Enterprise Server 12 SP3 | 4.4.73-5-default          |
| Ubuntu* 17.10                         | 4.13.0-11-generic         |

**Note:** Installing Ubuntu\* on Intel® Xeon Phi™ processor x200-based hosts is not officially supported.

If your host runs Red Hat\* Enterprise Linux\* Server 7.4 or CentOS\* Linux\* 7 (1708), ensure your OS kernel was updated to version 3.10.0-693.2.1.el7.x86\_64 or later which contains critical errata: <https://access.redhat.com/errata/RHBA-2017:2581>. To obtain the host's running kernel version, execute:

```
[host]$ uname -r
```

Some packages that will be installed require access to the standard distribution software repositories. If you disabled any of the standard repositories, please consider re-enabling them to prevent failed dependency issues. For more information, refer to documentation provided by your operating system vendor.

#### 3.1.1 Root access

Many of the tasks described in this document require root access privileges. Verify you have such privileges on the machines you will configure.

The use of *sudo* to acquire root privileges should be done carefully because its use may cause subtle and undesirable side effects. *Sudo* may not retain the non-root environment of the caller. This could, for example, result in use of an unexpected *PATH* variable leading to execution of wrong code.

When *su* is used to become root, the non-root environment is mostly retained. To retain *HOME*, *SHELL*, *USER* and *LOGNAME* use *su* with the *-m* switch. See the *su* man page for details.

## 3.2 Installation procedure

### 3.2.1 Software distribution

The latest Intel® Xeon Phi™ processor software distribution can be obtained from the [Drivers & Software website](#). Software releases are available in separate tar archives for each supported OS (RHEL\* and CentOS\* share a release package). Download the appropriate package for your operating system and extract it:

```
[host]$ tar xvf xpps-<version>-<os>.tar
[host]$ cd xpps-<version>/<os_version>/packages/x86_64/core
```

Follow the instructions below to install or update components of the Intel® Xeon Phi™ processor software.

### 3.2.2 RHEL\*/CentOS\*

#### Installing *memkind*:

```
[host]# yum install \
memkind-<version>-<release>.x86_64.rpm
memkind-devel-<version>-<release>.x86_64.rpm
```

#### Installing *sysdiag*:

```
[host]# yum install \
sysdiag-<version>-<release>.x86_64.rpm
```

#### Installing *micperf*:

```
[host]# yum install \
micperf-<version>-<release>.x86_64.rpm
```

#### Installing *hwloc*:

```
[host]# yum install \
hwloc-<version>-<release>.x86_64.rpm \
hwloc-devel-<version>-<release>.x86_64.rpm \
hwloc-gui-<version>-<release>.x86_64.rpm \
hwloc-libs-<version>-<release>.x86_64.rpm \
hwloc-sbin-<version>-<release>.x86_64.rpm
```

**Note:** Enable the *rhel-7-server-optional-rpms* repository to install the *hwloc-devel* package. For more information visit [https://access.redhat.com/documentation/en-us/red\\_hat\\_subscription\\_management/1/html/rhsm/supplementary-repos](https://access.redhat.com/documentation/en-us/red_hat_subscription_management/1/html/rhsm/supplementary-repos).

#### Installing *zonesort*:

```
[host]# yum install \
kmod-zonesort-<version>-<release>.x86_64.rpm
```

### 3.2.3 SLES\*

#### Installing *memkind*:

```
[host]# zypper install \  
memkind-<version>-<release>.x86_64.rpm \  
memkind-devel-<version>-<release>.x86_64.rpm
```

#### Installing *sysdiag*:

```
[host]# zypper install \  
sysdiag-<version>-<release>.x86_64.rpm
```

#### Installing *micperf*:

```
[host]# zypper install \  
micperf-<version>-<release>.x86_64.rpm
```

#### Installing *cpuid*:

```
[host]# zypper install \  
cpuid-<version>-<release>.x86_64.rpm
```

#### Installing *zonesort*:

```
[host]# zypper install \  
zonesort-kmp-default-<version><kernel_version>-<release>.x86_64.rpm
```

**Note:** The *zonesort* module was pre-compiled for installation on SLES\* 12 SP3. If you use a different version of this OS, recompile the module. The source code can be found in the `xpps-<xpps-version>/<os_version>/packages/source/core` directory.

### 3.2.4 Ubuntu\*

#### Installing *memkind*:

```
[host]# apt install memkind_<version>-<release>_amd64.deb
```

#### Installing *sysdiag*:

```
[host]# apt install sysdiag_<version>-<release>_amd64.deb
```

#### Installing *micperf*:

```
[host]# apt install micperf_<version>-<release>_amd64.deb
```

#### Installing *zonesort*:

```
[host]# apt install \  
zonesort_<version>-<release>_amd64.deb
```

### 3.2.5 Uninstall

To check for a previous install of Intel® Xeon Phi™ processor software processor software execute:

- **RHEL\*/CentOS\*/SLES\*:**  
`[host]$ rpm -qa | grep +xpps`
- **Ubuntu\*:**  
`[host]$ apt list --installed | grep +xpps`

The command above lists packages correlating to Intel® Xeon Phi™ processor software which need to be uninstalled.

- **RHEL\*/CentOS\*:**  
`[host# yum remove <package-name>`
- **SLES\*:**  
`[host]# zypper rm <package-name>`
- **Ubuntu\*:**  
`[host]# apt remove <package-name>`

### 3.2.6 Additional remarks

The *hwloc* module requires the *hwloc-dump-hwdata* files to be present in the */var/run/hwloc* directory. On SLES\* the *hwloc-dump-hwdata* application has to be run manually. On other systems Intel® Xeon Phi™ processor software provides a service which runs *hwloc-dump-hwdata* during system boot.

## 3.3 Rebuilding source packages

The Intel® Xeon Phi™ processor software source code is available in the *xpps-<xpps-version>/<os\_version>/packages/source/core* directory. Refer to your OS documentation for detailed instructions on rebuilding software from source.

# Chapter 4

## Virtualization

**Note:** Virtualization is not officially supported on hosts with Intel® Xeon Phi™ processors x200.

QEMU version 2.9 or newer is required to fully support virtualization on the Intel® Xeon Phi™ processors. Older versions may work, however, they will only support up to 255 virtual CPUs. Currently, virtualization is supported on hosts running RHEL\*, CentOS\* and SLES\*.

### 4.1 QEMU installation remarks

Please note that some operating systems may provide old naming conventions for *QEMU* binary files in order to provide backward compatibility. Therefore, if *QEMU* needs to be used manually the file `/usr/libexec/qemu-kvm` should be used instead of `/usr/bin/qemu-system-x86_64`. All functionalities should remain unchanged.

The convention mentioned above can be disregarded if *QEMU* is used through the libvirt API. If your installation of *QEMU* was upgraded to version 2.9 or later, you must restart the libvirt daemon with the following command:

```
[host]# systemctl restart libvirtd
```

#### **RHEL\*/CentOS\*:**

In RHEL\* install *QEMU* version 2.9 provided by Red Hat\* Virtualization packages. Installing *QEMU* from standard repositories may install an older version that lack pass-through support for all available virtual CPUs. In CentOS\* download and install software provided in the `xpps-virt-<version>-<os>.tar` package, which is provided as a part of the Intel® Xeon Phi™ processor software. You can also install those packages on RHEL\*.

#### **SLES\*:**

SLES\* provides all components required for virtualization. Install *QEMU* using the command below.

```
[host]# zypper install qemu qemu-kvm libvirt
```

### 4.2 Configuration

To fully support more than 255 virtual CPUs configure *QEMU* according to instructions below.

- The `-machine` option should be set to `q35 kernel_irqchip=split`.

- Intel® iommu must set with the `-device intel-iommu,intremap=on,eim=on` option.
- Intel® iommu driver must be present on the host OS.

### 4.3 QEMU raw command

The code snippet below presents an example *QEMU* command that creates a virtual machine with 288 virtual CPUs.

```
LC_ALL=C
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin
QEMU_AUDIO_DRV=none
/usr/libexec/qemu-kvm
-name guest=rh74_288vcpus,debug-threads=on
-machine q35,accel=kvm,usb=off,dump-guest-core=off
-cpu host -m 4096
-realtime mlock=off
-smp 288,sockets=1,cores=72,threads=4
-drive file=/tmp/rh74_288vcpus/rhel74.qcow2, \
  format=qcow2,if=none,id=drive-sata0-0-0
-device intel-iommu,intremap=on,eim=on
-machine kernel_irqchip=split
-msg timestamp=on
-device ide-hd,bus=ide.0,drive=drive-sata0-0-0,id=sata0-0-0,bootindex=1
-nographic
```

#### 4.3.1 Libvirt xml configuration file

Review the *example\_vm\_288vcpus.xml* file included in the Intel® Xeon Phi™ processor software release package.

#### 4.3.2 Adding an Intel® OPA card to a virtual machine

To add a PCI card to a VM, you can either use the *hostdev* option to the virtual machine configuration xml file or add the `-device` parameter to *QEMU* command line. In both cases the device's parameters must correspond to the host's physical PCI bus. Alternatively, you can add a device using the *virt-manager* GUI application. Using a PCI device in parallel by the host OS and a virtual machine is not supported.

**Adding a device in the xml file:**

```
<hostdev mode='subsystem' type='pci' managed='yes'>
  <source>
    <address domain='0x0000' bus='0x01' slot='0x00' />
  </source>
</hostdev>
```

**Adding a device in the QEMU command:**

```
-device pcie-root-port,port=0xa,chassis=3, \
    id=pci.3,bus=pcie.0,addr=0x1.0x2
-device vfio-pci,host=01:00.0,id=hostdev0, \
    bus=pci.2,addr=0x0
```

## Chapter 5

# Zonesort

The cache mode design places MCDRAM as a direct mapped cache. On Linux\* systems this design causes cache performance degradation over time due to increased number of cache collisions caused by memory fragmentation. To mitigate the performance degradation use the *zonesort* page sorting module provided in the Intel® Xeon Phi™ processor software.

### 5.1 Installing *zonesort*

If the Intel® Xeon Phi™ processor software is installed and running on your system, the correct module can be used; proceed to the next section. Otherwise, if your machine is running one of the supported operating systems, install the correct kernel module package by following the steps below.

1. Navigate to the directory containing binary packages for Intel® Xeon Phi™ processor software.

```
[host]# cd xpps-<version>/<os-version>/packages/x86_64/core
```

2. Install the kernel module package:

**RHEL\*/CentOS\*:**

```
[host]# yum install kmod-zonesort-*.x86_64.rpm
```

**SLES\*:**

**Note:** If you do not use SLES\* 12 SP3, before installation, recompiling the kernel module from the source package is required.

```
[host]# zypper install zonesort-kmp-default-*.x86_64.rpm
```

**Ubuntu\*:**

```
[host]# apt install *zonesort*.x86_64.deb
```



## 5.2 Using zonesort

The *zonesort* module sorts kernel free memory pages lists to minimize cache misses when those pages are acquired by user processes. Since the module operates on free pages, enabling sorting before running user applications is recommended.

Due to high memory fragmentation, sorting pages alone may not be sufficient to restore initial performance. To decrease fragmentation and increase the amount of physically-contiguous pages, use memory compaction before sorting (see the example below).

Sorting can be called on-demand similar to the example below:

1. Load the module:  
[host]# modprobe zonesort\_module
2. Trigger memory compaction:  
[host]# echo 1 > /proc/sys/vm/compact\_memory
3. Trigger sorting (the call returns once sorting completes):  
[host]# echo <numa\_node\*> > \  
/sys/kernel/zone\_sort\_free\_pages/nodeid

Alternatively, you can configure sorting to trigger automatically with an interval:

1. Load the module:  
[host]# modprobe zonesort\_module
2. Set the interval of periodic sorting:  
[host]# echo <interval\_in\_sec> > \  
/sys/kernel/zone\_sort\_free\_pages/sort\_interval

Note that in case of periodic sorting the action will always be taken on all online nodes. Unlike using the *zone\_sort\_free\_pages/nodeid* interface, the node to be sorted cannot be chosen. Writing value 0 (zero) disables periodic sorting and cancels all pending activities. Actively running sorts will finish regardless. Memory compaction must be handled by the system administrator. The *zonesort* module does not call it. On-demand sorting is disabled. Writing to *zone\_sort\_free\_pages/nodeid* while *zone\_sort\_free\_pages/sort\_interval* is set to non-zero value will return *EBUSY*.

## 5.3 Administration

By default, due to security reasons, all interfaces exposed by the *zonesort* module can be written to only by root. To modify permissions we recommend using the *udev* manager, as presented in the example below.

1. Create the file */etc/udev/rules.d/99-zonesort.rules* with the following content:  
ACTION=="add", DEVPATH=="/module/zonesort\_module",  
SUBSYSTEM=="module",  
RUN+="/bin/chmod 0666 \  
/sys/kernel/zone\_sort\_free\_pages/sort\_interval \  
/sys/kernel/zone\_sort\_free\_pages/nodeid"
2. Reload the *udev* rules to apply changes:  
[host]# udevadm control --reload-rules

The inserted rule will change access permissions to the interfaces every time the module is being loaded.

## 5.4 Debugging

The *zonesort* module exposes additional interfaces, which may be useful for identifying the system's state:

- *buddy\_lists*:  
Provides details of the current state of the kernel buddy allocator. In order to use it, dump its contents to a file:

```
[host]# cat /sys/kernel/debug/buddy_lists > output_file
```

- *directmappedcache\_state*:  
Provides information similar to */proc/pagetypeinfo* but extended for the purpose of direct mapped cache debugging. The data can be obtained by printing the entry to standard output:

```
[host]# cat /sys/kernel/debug/directmappedcache_state
```

For further details on how to interpret the results please refer to the source code of the module, which is delivered with the Intel® Xeon Phi™ processor software.

### REMARKS:

- The module does not support explicitly allocated huge pages.
- The module has been validated for stock kernels of supported OS distributions. There is no guarantee the module will be functional when used with other kernels.
- The module can only be loaded if MCDRAM is configured as cache.

# Chapter 6

## Huge pages

Linux\* systems support 2 MB and 1 GB huge pages, which can be allocated at boot or at runtime. Huge pages can significantly increase performance, particularly for large memory and memory-intensive workloads.

When huge pages are allocated during boot time, they are distributed equally between nodes. Runtime allocation allows the system administrator to choose which NUMA node to allocate those pages from. However, due to memory fragmentation, runtime page allocation is more failure prone than boot time allocation.

### 6.1 Huge pages on RHEL\*/CentOS\*

#### Boot time mode:

1G huge pages are enabled at boot by default in Red Hat\* Enterprise Linux\* and CentOS\*. To allocate different sizes of huge pages at boot, use the command below. This example allocates four 1GB huge pages and 1024 2MB huge pages:

```
'default_hugepagesz=1G hugepagesz=1G hugepages=4 hugepagesz=2M hugepages=1024'
```

#### Runtime mode:

Huge pages can be also allocated at runtime with the command below.

```
[host]# echo <number_of_pages> > sys/devices/system/node/node[0-9]* \
/hugepages/hugepages-<size_in_bytes>/nr_hugepages
```

### 6.2 Huge pages on SLES\*

#### Boot time mode:

The default Huge Page size in SLES\* is 2 MB. Additional configuration is required to enable huge Pages over the default size. Boot time mode distributes huge pages equally between the nodes.

To allocate different sizes of huge pages at boot time, use the following kernel boot parameters. This example allocates four 1GB huge pages and 1024 2MB huge pages:

```
'hugepagesz=1G hugepagesz=1G hugepages=4 hugepagesz=2M hugepages=1024'
```

**Runtime mode:**

The default SLES\* kernel does not support huge pages in real-time mode. Enabling support requires installing additional kernel patches, which are listed in the table below, and rebuilding the kernel with the following lines in the kernel config:

```
CONFIG_CMA=y
CONFIG_CMA_DEBUG=n
```

Patches to apply:

| Kernel commit SHA | Patch name   |
|-------------------|--|
| bae7f4a           | hugetlb: add hstate_is_gigantic()                                |
| a7407a2           | hugetlb: update_and_free_page(): don't clear the PG_reserved bit |
| 1cac6f2           | hugetlb: move helpers up in the file                             |
| 944d9fe           | hugetlb: add support for gigantic page allocation at runtime     |

To allocate huge pages use the following command:

```
[host]# echo <number_of_pages> > sys/devices/system/node/node \
[0-9]*/hugepages/hugepages-<size_in_bytes>/nr_hugepages
```

## 6.3 Allocating all MCDRAM to 1G Pages

To allocate all MCDRAM to 1G pages follow the steps below.

- Enter your platform's BIOS and set the *Treat MCDRAM as Hotplug* option to *enabled*.
- Add the *"movable\_node"* kernel command line. This allows a node to have only movable memory, and enables the following two things: when the system is booting, a node full of *hotpluggable* memory can be arranged to have only movable memory so the whole node can be hot-removed. Once the system is up, the option allows users to online all the memory of a node as movable memory so that the whole node can be hot-removed. Users who do not use the memory *hotplug* feature can leave this option on since they do not specify the *movable\_node* boot option, or they do not online memory as movable.

## Chapter 7

## References

1. Burger, T. W. (2011, March 5). Intel® Virtualization Technology for Directed I/O. Retrieved from Intel® Developer Zone: <https://software.intel.com/en-us/articles/intel-virtualization-technology-for-directed-io-vt-d-enhancing-intel-platforms-for-efficient-virtualization-of-io-devices>
2. Craig, L., & Bas, V. (2009). *Scaling Lean & Agile Development*. Boston: Addison-Wesley.
3. Danalis, A., Marin, G., McCurdy, C., Meredith, J. S., Roth, P. C., Spafford, K., ... Vetter, J. S. (2010). The scalable heterogeneous computing (shoc) benchmark suite. In *Proceedings of the 3rd Workshop on General-Purpose Computation on Graphics Processing Units. GPGPU*, 63-74.
4. <https://github.com/axboe/fio>. (2017).
5. <https://github.com/baidu-research/DeepBench>. (2017).
6. Intel Corporation. (2017). Intel® Math Kernel Library (Intel® MKL). Retrieved from <http://software.intel.com/en-us/intel-mkl>
7. Jack, D., Luszczek, P., & Petitet, A. (2003). The linpack benchmark: past, present and future. *Concurrency and Computation: Practice and Experience*.
8. McCalpin, J. D. (1991-2007). *Stream: Sustainable memory bandwidth in high performance computers. A continually updated technical report*. Charlottesville: University of Virginia.
9. McCalpin, J. D. (1995). Memory bandwidth and machine balance in current high performance computers. *IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter*, 19-25.