KNL System Software

CUG 2017, Systems Support

Peter Hill, Clark Snyder, John Sygulla
Compute Products R&D
Cray Inc.
Bloomington, Minnesota, USA
{pfh,csnyder,jsygulla}@cray.com

***Abstract* — Intel® Xeon Phi™ "Knights Landing" (KNL) presents opportunities and challenges for system software. This paper starts with an overview of KNL architecture. We describe some of the key differences from traditional Xeon processors, such as processor (NUMA) and memory (MCDRAM) modes. We describe which KNL modes are most useful, and why. From there, we describe a day in the life of a KNL system, emphasizing unique features such as mode reconfiguration (selecting the processor and memory configuration for a job) and the zone sort feature (which optimizes performance of MCDRAM cache). As part of this coverage, we'll look at implementation, scaling and performance issues.**

Keywords: Cray XC40, Cray Linux Environment, CLE, Intel Xeon Phi, KNL, MCDRAM, reconfiguration, zone sort

I. Overview of KNL architecture

The second-generation Xeon Phi processor, known as Knights Landing (KNL), is self-hosted — it boots a standard operating system. It incorporates a novel memory-on-package feature, where each processor includes 16 GB of high-bandwidth MCDRAM memory, in addition to support for DDR4 memory.

The KNL processor has the following characteristics:

- It is binary-compatible with traditional Xeon
- It has support for Intel® AVX-512 instructions
- It has up to 36 tiles, where each tile has two cores, and 1 MB L2 cache; and each core has four threads, using Intel® Hyper-Threading technology
- It has 16 GB of MCDRAM memory
- It has support for up to 384 GB of DDR4-2133 or DDR4-2400 memory (six DIMMs)
- It has 36 lanes of PCI Express Revision 3.0

The KNL processor and MCDRAM can be configured in twenty different combinations of NUMA and MCDRAM modes. For example, the *quadrant* NUMA mode can be combined with the *cache* MCDRAM mode, and this combination is known as *quad/cache*.

The five NUMA modes are shown in Table 1.

Table 1. Processor NUMA modes

Mode	*Name*	*Description*
a2a	All-to-All	Addresses are uniformly hashed across distributed directories
quad	Quadrant	Addresses are hashed to a directory in the same quadrant as the memory
hemi	Hemisphere	Addresses are hashed to a directory in the same hemisphere as the memory
snc4	(4) Sub-NUMA Clusters	Tiles are divided into four sub-NUMA clusters; each cluster is one NUMA node
snc2	(2) Sub-NUMA Clusters	Tiles are divided into two sub-NUMA clusters; each cluster is one NUMA node

The four MCDRAM modes are shown in Table 2.

Table 2. MCDRAM modes

Mode	*Name*	*Description*
cache	Cache	MCDRAM is used as a cache between the processor and DDR4 memory
flat	Flat	MCDRAM is physically addressable, in a separate NUMA node
equal	Hybrid Equal	50% of MCDRAM is Flat, and 50% of MCDRAM is Cache
split	Hybrid Split	75% of MCDRAM is Flat, and 25% of MCDRAM is Cache

II. Using KNL modes

With twenty different combinations of KNL modes, a natural question is "which KNL modes are most useful, and why?" We'll start with a recommendation for two combinations, *quad/flat* and *quad/cache*.

For the "why" of the question, it depends on the application, of course; but we can offer a few guidelines.

NUMA mode: *Quad* is preferred. It delivers similar performance to *snc2* — but with *quad* mode, placement is straightforward, and you have more flexibility in choosing the number of MPI ranks versus OpenMP threads. If you consider *snc4* mode, note that it complicates placement on KNL processors with 68 cores, because the tiles cannot be evenly distributed across the four NUMA nodes.

MCDRAM mode: If your application uses less than 16 GB memory per node, you should probably use *flat*. If your application uses more than 16 GB memory per node, it can be difficult to use *flat* mode — and *cache* mode will typically perform as well as *flat*.

The Cray Performance Team provided these specific examples of KNL modes for applications:

- AMG, GTC, MiniGhost, MiniDFT, SNAP and UMT all benefit from *quad/cache*.
- MiniFE and MILC both benefit from *quad/flat*.

The Cray Performance Team provided these suggestions when using *quad/flat*: If your working set is less than 16 GB, consider using the *numactl* command to place all memory allocations on NUMA node 1, which is MCDRAM memory. If your working set is greater than 16 GB, but the key data fits in less than 16 GB, consider using *numactl* to place all memory allocations on NUMA node 0, which is DDR4 memory; and then use the *memkind* library for explicit allocation of key data on MCDRAM.

These are general guidelines, and your mileage may vary.

III. KNL ON THE CRAY XC40 SYSTEM

The Cray XC40 system, which was introduced with dual-socket Xeon compute nodes, can now be configured with single-socket KNL compute nodes. Xeon and KNL compute nodes can be combined into a single large-scale system.

At this time, supported KNL processors on a Cray XC40 system are:

Xeon Phi 7250	1.4 GHz	68 cores, 272 threads	DDR4-2400
Xeon Phi 7230	1.3 GHz	64 cores, 256 threads	DDR4-2400
Xeon Phi 7210	1.3 GHz	64 cores, 256 threads	DDR4-2133

Each KNL compute node can be configured with either 96 GB or 192 GB of DDR4 memory.

As an option, each KNL compute node can be configured with either 128 GB or 256 GB of M.2 solid-state disk (SSD), for local storage.

Support for KNL processors was introduced with the Cray Linux Environment (CLE) 6.0/8.0 release in 2016. The CLE system software required significant new features to support KNL processors, and we will discuss those features in the next section.

IV. A DAY IN THE LIFE OF A KNL SYSTEM

A typical workload on a Cray XC40 system consists of many batch jobs, under the control of a workload manager (WLM). Once it is running, a batch job can launch one or more applications, and each application is placed on one or more compute nodes.

With the introduction of the KNL processor, we have new requirements. At the beginning of each job, we may need to reconfigure one or more processors with the requested mode. We need to optimize performance on the MCDRAM cache, if that is in use. If the optional on-node SSD is configured with managed space, that space will need to be cleared at the end of each job.

A. *Mode Reconfiguration*

KNL mode reconfiguration is performed by the BIOS. In order to start a reconfiguration, we need to set the new NUMA and/or MCDRAM configuration for the node, and then *reinitialize* the node — which consists of the following steps: Shut down the operating system on the node; "bounce" the node (an initialization step that includes the BIOS); and transfer the operating system image to the node. Finally, we need to wait for the node to finish booting.

This is under the control of the workload manager (WLM). The WLM does this by making requests using the Cray *capmc* interface, as shown in Table 3.

Table 3. Cray *capmc* mode reconfiguration requests

Name	***Description***
get_numa_cfg_capabilities	Get supported NUMA configurations
get_numa_cfg	Get current NUMA configuration
set_numa_cfg	Set new NUMA configuration
get_mcdram_cfg_capabilities	Get supported MCDRAM configurations
get_mcdram_cfg	Get current MCDRAM configuration
set_mcdram_cfg	Set new MCDRAM configuration
node_reinit	Reinitialize node

A command-line interface, the *capmc* command, allows manual control by the system administrator or operator.

The following example demonstrates manual control of KNL mode reconfiguration, with the same sequence of operations that is used by the WLM. In this example, we reconfigure four nodes in *quad/cache*:

```
smw:~ # capmc set_numa_cfg \
        --mode quad -n 8,9,10,11 -p
smw:~ # capmc set_mcdram_cfg \
        --mode cache -n 8,9,10,11 -p
smw:~ # capmc node_reinit -n 8,9,10,11
```

One important aspect of reconfiguration is resiliency. We now allow each customer site to tune the number of retries that are allowed during the "bounce" step. On a large system, two retries will often allow all nodes to be reinitialized successfully. In addition, Intel, AMI and Cray have together made many improvements to the KNL BIOS, which is now faster and more reliable.

We continue our work to reduce the time taken to reinitialize a node, because each mode reconfiguration affects system utilization; a node isn't doing useful work while it is being reconfigured. Our customers will see these additional improvements both in UP03 patches, and in the upcoming UP04 release later in 2017.

Table 4 shows the time taken to reconfigure approximately 3,200 KNL compute nodes on an 18-cabinet Cray XC40 system.

We show the time taken for each of the steps performed during reconfiguration, followed by the time taken for the nodes to finish booting. We show these results for two different update releases of CLE 6.0/8.0: UP01 (June 2016) and UP03 (February 2017).

After the upgrade from UP01 to UP03, reconfiguration time was reduced from 2799 seconds (one example) to 1059 seconds (average of three tests) — a reduction of 62%.

Table 4. Time to reconfigure KNL nodes with *capmc* for CLE 6.0/8.0 updates UP01 (3,233 nodes) and UP03 (3,240 nodes)

UP01, Seconds	*UP03, Seconds*	*Reinitialization Step*	*Description*
146	26	xtcli shutdown	Shut down OS on nodes
170	177	xtbounce	Bounce nodes
97	109	xtcli boot	Fan out OS image to nodes
2386	747	*wait for boot*	Boot OS on nodes
2799	**1059**	**Total time (seconds)**	

B. *Zone Sort*

When the MCDRAM is configured as cache, it provides a physically addressed, direct-mapped cache. Thus, each DDR address has only one possible destination in the MCDRAM cache. With 16 GB of cache, each physical address in DDR will conflict with every address that is a multiple of 16 GB away.

Over time, as jobs run on a compute node allocating and freeing memory, the actual physical memory that is free changes, as does the order in which this memory is kept on the kernel's free lists. As a result, a program will almost certainly be given different physical memory each time it runs. This results in inconsistent cache performance from run to run due to the underlying physical memory allocations causing different conflicts in the cache.

Zone sort reorders the pages of memory on the kernel's free lists to try to minimize cache conflicts among later memory allocations. This sorting is a discrete operation and not continuous.

On a Cray XC40 system, zone sort is used on compute nodes between applications along with other cleanup operations, such as memory compaction and clearing the file cache, in order to improve memory availability and reduce run-to-run variability. Our testing has shown the time to perform a zone sort is typically less than 50 milliseconds.

The Cray Performance Team validation of zone sort v1.5.0 included the following test on one quad/cache KNL node. The STREAM and DGEMM MPI benchmarks were run in alternation. With zone sort, STREAM triad rate was consistently 350-360 GB/sec. Without zone sort, STREAM performance would degrade to 100-150 GB/sec after the DGEMM runs, and remain in that state until the node was rebooted.

An additional feature, which will allow a user to request that zone sort run periodically while an application is running, is under development.

C. *Local storage*

If an SSD is present on a KNL compute node, the local storage may be configured in one or more of the following categories:

- Swap device for the local node
- Managed space — cleared at the end of each job
- Unmanaged space — the customer site may allocate, format, mount, and use this space as desired

The system administrator can keep track of SSD life remaining with the *xtcheckssd* command.

D. *KNL debug information and tools*

Mode reconfiguration: System administrators can use log files to get a better understanding of reconfiguration [1]. These files are on the System Management Workstation (SMW) in the /var/opt/cray/log/ directory:

- xtremoted-*yyyymmdd* – detailed logging of *capmc* configuration and reinitialization requests, with timestamps; also any errors that occur during the xtbounce step (note that errors are logged after the bounce has completed)
- commands/log.*yyyymmdd* – logging of commands, including those issued for reinitialization, with timestamps
- p0-current/console-*yyyymmdd* – console messages from each node, with timestamps; KNL mode is recorded for each boot; search for the string NODE_INFO_OS_BOOT_SUCCEEDED to determine when a node has finished booting

Zone sort: System administrators with root access can log into individual compute nodes, and enable dynamic debug messages to observe zone sort while a workload is active. These messages are too verbose for the console.

Query current KNL mode: Anyone with an account on the XC40 system can use the following techniques [2]:

- ALPS command on login/mom node: *apstat -M*
- SLURM command on login/mom node: *sjstat*
- Command on compute node [3]: *hwloc*

When launching an application with ALPS, the *aprun* and *cnselect* commands can be used together to target nodes with specific characteristics, including KNL mode.

System administrators can query current KNL mode when logged into the SMW [1]:

- Command on SMW: *xthwinv ...*
- Command on SMW: *capmc get ...*

Please be aware that the *capmc* command returns the most recent requested configuration, which may not be the current configuration.

System administrators can also determine historical KNL mode settings, by inspecting the *xtremoted* and *console* log files described above.

V. Conclusion

We have installed Cray XC40 systems at customer sites with up to 10,000 KNL compute nodes in a single system. Several of these systems also have traditional Xeon compute nodes, for workloads that benefit from both kinds of processors.

The features in CLE 6.0/8.0 support these diverse workloads, and we continue to make significant improvements.

Acknowledgment

The authors would like to thank Nathan Wichmann and Stephen Behling for their contributions to this paper.

References

[1] XC Series Software Initial Installation and Configuration Guide (S-2559), Cray Inc

[2] XC Series Programming Environment User Guide (S-2529), Cray Inc

[3] Hardware Locality (hwloc) documentation, open-mpi.org