Your Path to Knights Landing, the Next Generation of Intel® Xeon Phi™ technology

By James Reinders, Director of Parallel Programming Evangelism, Intel

In 2012, the world saw the first systems built on Intel's first many-core commercial product—the Intel® Xeon Phi™ coprocessor also known as “Knights Corner.” Today, Intel® Xeon® processors and these first-generation Intel Xeon Phi coprocessors contribute more FLOP/s (floating point operations per second) to the Top500 List than any other form of computation, including GPU accelerators. Tianhe-2, the world's fastest supercomputer on the Top500 List, uses Intel Xeon processors and Intel Xeon Phi coprocessors.

Made specifically for parallel processing, Knights Corner coprocessors have up to 61 x86 cores, each running up to 1.1 GHz. Each core has four hardware threads, which means that you can have as many as 244 threads of execution available per coprocessor for operation in parallel. That's a lot.

While 1.1 GHz per core is lower than an Intel Xeon processor, a conscious trade-off has been made for very high parallelism versus single-threaded performance. A program designed for high degrees of parallelism will make the Knights Corner coprocessor shine—delivering up to 2.2 times the performance of processors and four times the power efficiency. Think two TeraFLOP/s at single-precision operation or one TeraFLOP/s double precision.
In short, the overall performance of the device can be quite a bit higher than with a processor, but it does demand a parallel program to get that performance.

**Knights Landing: The Next Generation**

Intel has been working for the last few years to create the next generation of Intel Xeon Phi technology—codenamed "Knights Landing." The first commercial systems to use this technology have already been announced and are expected to reach the market later this year.

The first was the Cori Supercomputer at the National Energy Research Scientific Computing (NERSC) Center at the U.S. Department of Energy's Lawrence Berkeley National Laboratory. Another notable project is the Trinity Supercomputer at the National Nuclear Security Administration (NNSA), a $174 million deal awarded to Cray that will feature Intel® Haswell processors and Knights Landing components.

> "Many software developers have been inspired. They have been working hard on their applications, looking for that concurrency and examining how can they take advantage of new levels of computational power. The reward is a payoff in higher performance, whether the program runs on an Intel Xeon processor or an Intel Xeon Phi coprocessor."

**What’s New**

Knights Landing brings significant enhancements to the Intel Xeon Phi family of coprocessors. For example:

> Knights Landing will have at least as many cores as a Knights Corner device, but it will be manufactured with Intel’s 14-nanometer technology, which will add performance and power advantages. (The Knights Corner product line is made at a 22-nm process size.)

> You’ll see a huge jump in performance, from one TeraFLOP/s for peak performance in the first generation to three TeraFLOP/s double-precision peak performance per single socket node.

> Knights Landing can perform as either a stand-alone, bootable processor (running the host OS) or a coprocessor (PCIe* end-point device). With the processor version, you won’t be required to have a separate host processor as you do with Knights Corner. This versatility has a lot of implications. For one, all memory on a node or system can be dedicated to Knights Landing instead of being split between the many-core and multicore processing devices.
> With individual cores on the device supporting out-of-order execution, Knights Landing will improve performance and programmability—most notably, better capabilities for executing serial portions of programs.

> Large, high-bandwidth, on-package memory will boost applications that are very bandwidth-hungry and also have a substantial memory footprint. You can still use off-package memory; a tremendous boost for memory hungry applications will come with both on-device and off-device memory, just as with a regular processor.

> Versions will be available with integrated fabric, which will enable higher efficiency in cluster applications, such as those using the standard, portable MPI message-passing library.

> Knights Landing will offer five times the bandwidth (compared to DDR4) in one-third the space—with power efficiency more than 25 percent better than a discrete coprocessor and five times better than previous processors.

Knights Landing will standardize vector instructions around the Intel® AVX-512 specification. That’s exciting, because as Intel Xeon Phi products become available in a processor format, it is valuable to have the instruction set completely synchronized with processors. (This is probably the most significant instruction difference between Knights Corner and Knights Landing. For testing purposes, a software emulator is available with full support for AVX-512 as implemented in Knights Landing.)
Another advantage relates to work under way with other vendors, including the GNU Project*, producers of the GNU Compiler Collection* (GCC*). The current version of GCC for Knights Corner cannot make use of these wide 512-bit vector instructions. That is set to change by the time Knights Landing is released.

**What Doesn’t Change: A Key Merit of Knights Landing**

Intel’s vision is to span from multicore to many-core architecture with consistent programming models, languages, tools, and techniques. Our belief is this: You should be able to write one program (using familiar compilers, libraries, abstract parallel programming models, and standards) and get top performance on both a multicore CPU and on the Many Integrated Core (MIC) architecture—in other words, first- and second-generation Intel Xeon Phi products.

You may need to modify your source code to support highly parallel operation, but that investment can be done in a consistent fashion. You are not forced to write in a different method when using lots of cores versus using a few (Figure 1).

In keeping with this vision, Intel Xeon Phi products support critical, top-of-mind standards such as OpenMP* and MPI, Fortran, C and C++, and Intel® Threaded Building Blocks ([Intel® TBB](https://software.intel.com/en-us/articles/threading-building-blocks)).
These standards have been around a long time, have support from many vendors, and work on many different platforms. Many applications have been written using these standards, which were created for general-purpose devices, specifically CPUs of many different varieties. Because Knights Landing is designed to be very versatile—capable of doing anything you would expect a CPU to do—all of these standards are supported very well.

Knights Corner made good on this promise. Knights Landing adds a number of new innovations and improves on it.

**Programming for High Levels of Parallelism**

Systems based on Knights Landing are coming in the second half of 2015, but you can do a lot now to get ready for them. The most important considerations come down to one very important concept: concurrency.

As with any highly parallel device, the performance of each thread individually is lower than a CPU, but with hundreds of threads of execution, Knights Landing can deliver much higher overall performance than a CPU. It puts a lot of parallelism at your disposal, but you have to use it.

As Figure 2 shows, if your program uses only four or eight threads of execution, it will actually perform better on an Intel Xeon processor. Not until you start to use dozens of threads will you see an Intel Xeon Phi coprocessor outperform an Intel Xeon processor. More likely, you would use hundreds of threads to see this performance.

© 2013, James Reinders & Jim Jeffers, diagram used with permission

---

© 2014, Intel Corporation. All rights reserved. Intel, the Intel logo, Intel Inside, Cilk, VTune, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries. Other names and brands may be claimed as the property of others.
Figure 3 shows an actual customer example of relative performance for a single-threaded application written in Fortran using MPI. The Intel Xeon processor outperforms the Intel Xeon Phi coprocessor—Knights Corner, in this case.

Illustrative example

Fortran code using MPI, single-threaded originally. Runs on Intel® Xeon Phi™ coprocessor natively (no offload).

Based on an actual customer example. Shown to illustrate a point about common techniques. Your results may vary!

However, when the program is tuned for parallelism, it’s a much different story. The Knights Corner coprocessor far outperforms the Intel Xeon processor. Note that the optimizations done for Intel Xeon Phi coprocessors also improved performance on Intel Xeon processors (Figure 4). It’s a win-win. This was a fairly straightforward piece of work with a customer. It didn’t take a great deal of effort to find where we could expose more concurrency in the application and take advantage of it.

These results beg the question, Why didn’t we do this work years ago? Why didn’t we optimize our applications to get this performance boost on processor-based systems? Why are so few supercomputer workloads tuned this way?
The short answer is that having 61 cores to work on is very motivating. Once you get excited, port your application, and take advantage of 61 cores, you'll find that virtually every parallel platform using processors will benefit.

Many software developers have been inspired. They have been working hard on their applications, looking for that concurrency and examining how can they take advantage of new levels of computational power. The reward is a payoff in higher performance, whether the program runs on an Intel Xeon processor or an Intel Xeon Phi coprocessor.

So perhaps the first step toward Knights Landing readiness is the inspiration—proving to yourself that you can get your application to scale on a highly parallel device, and that would be Knights Corner.

The good news is that attaining higher levels of parallelism is easy to do using current techniques. OpenMP or Intel TBB lend themselves very well for this. Or you could use a certain number of MPI ranks on the device, along with OpenMP or TBB, to reach a very high thread count on a device. The next thing you know, you’re seeing performance above anything you could expect out of a CPU. But you have to have that parallelism.
Is Your Application Ready for Knights Landing?

It could be, but it’s unlikely. If it hasn’t been running on a highly parallel device, you probably haven’t shaken out all the impediments to scaling. There are three scaling criteria you need to look for the most:

- Thread scaling, or being able to scale as you increase the number of cores
- Vectorization scaling to ensure profitable use of vectorization for data parallelism
- Fabric scaling for cluster applications

There may be some cache-related optimizations as well, but many methods to attain thread scaling and vectorization require or incorporate the concepts of good cache utilization. It is best to focus on these three elements.

The best way to know if your application is ready for this scaling is to do some plotting on a high-core-count system, such as a Knights Corner system or an Intel Xeon EX processor-based system. Establish a performance and configuration discipline and cluster baseline for your workload at scale:

- For thread scalability, plot performance against the number of ranks per node.
- For vector scalability, plot vectorization profitability. Turn vectorization on and off, or compare performance with minimal amounts of vectorization versus maximum vectorization flags.
- For fabric scalability, plot performance against the number of ranks per node. Does performance go up when you increase the number of ranks?

Performance plotting is the first basic but very effective step in understanding whether or not you have opportunities to tune for better performance. Then you can take it from there.

For example, using OpenMP or TBB, you can incrementally increase your thread scaling to use more and more of the whole device as a rank. What happens to performance as you add more worker threads? On Knights Corner, about three threads per core is likely to be optimal, but it depends on the application. All require at least two threads per core to reach peak performance.
For vectorization scaling, most developers use intrinsic functions, autovectorization or the SIMD (single instruction, multiple data) directives included in the OpenMP 4.0 standard. Intel® compilers also include additional innovative reporting capabilities that help with vectorization.

The key point to remember is that once you understand these three important characteristics of your application, there's a lot you can do to improve its performance on all machines—such as those using Intel Xeon processors and Intel Xeon Phi coprocessors.

What Are the Best Systems to Prepare on?

Without a doubt, Knights Corner is the best system to use today to prepare for Knights Landing. If your application is a good fit for Knights Corner, you can tune the scaling dimensions to get the desired performance attributes, reap the near-term benefits, then prepare to be impressed with application performance on Knights Landing.

For example, if you're presently using optimizations that have to shuttle data back and forth to the CPU because Knights Corner doesn't have enough memory, you'll see a significant performance boost with the added memory and reduced data movement when running on Knights Landing.

“Knights Landing offers high-level source compatibility with the intrinsic functions of Knights Corner, while adding a number of enhancements for better performance on vector instructions.”

What If the Application Is Not a Fit for Knights Corner?

That's possible. The chief reason would be memory. While 16GB is a lot of memory for a card, it's not a lot of memory for a node. So there's a chance you may not get as much of your application running on Knights Corner as you'd like, due to limitations that apply to anyone designing to a card. Another limiting factor for applications can be high input/output (I/O) or communication requirements.
Furthermore, the relatively low serial performance of Knights Corner may call for material portions of your code to run on the host processor, which tends to be about 10 times faster. An unwanted side effect of such an approach is moving more data back and forth across the PCI bus.

If Knights Corner is not the right choice, then you need to understand the three scaling dimensions of your application on another system with a very large number of cores. Eight or 10 cores is nowhere near enough, because it doesn't require really tackling a very high degree of scaling—the effective use of threads and vectors.

Theoretically, you could have eight or 16 cores and use performance data from those few cores to predict performance for hundreds of cores. In reality, we've never seen this successfully done because it seems to fall short of inspiring enough attention to scaling.

You really have to put the application to the real-world test. If not using a 61-core Knights Corner system, it could be a very high-core count Intel Xeon system—an Intel Xeon processor EX system—if you have the budget for it. But if there's any way to use Knights Corner to prepare for Knights Landing, that's the way to go. Efforts to make your code ready on Knights Corner will be handsomely rewarded, not only on today's Intel Xeon processor-based machines but on Knights Landing in the future.

Looking to the Future

The enhancements in Knights Landing help with the considerations and limitations discussed earlier. For example:

- Additional versatility and memory (the ability to be a processor and to have processor-sized memory) enable many more applications and use models.
- The integrated fabric and onboard, high-bandwidth memory redefine the kinds of workloads Knights Landing can handle.
- Significantly higher serial code performance reduces the effects of moving back and forth between parallel and serial operations—and the work required to juggle those movements on large applications.

Knights Landing offers high-level source compatibility with the intrinsic functions of Knights Corner, while adding a number of enhancements for better performance on vector instructions. Enhancements to the Intel AVX-512 specification, such as better support for unaligned data, make it easier to program the vectorization capability of Knights Landing compared to Knights Corner. (If you've coded specifically for Knights Corner, there may be a little bit of work to do here.)

In short, if you have a Knights Corner application today, you're very well lined up for Knights Landing. And where your applications might be limited today by Knights Corner, Knights Landing will bring relief.
Inspired by 61 Cores

This theme recurs again and again. Having 61 cores to work with is very motivating. Find a Knights Corner machine or build out an Intel Xeon machine with a very high number of cores, be inspired by that, and you'll be well on the road to Knights Landing. For more inspiration, take a look at our new book, *High Performance Parallelism Pearls*, which has outstanding real-world examples, with actual code, contributed by 69 experts from around the world.

In summary:

> **Expose the concurrency.** To show workloads’ real value, Intel Xeon Phi requires them to have great concurrency across multiple dimensions. Getting ready for Knights Landing therefore means investing generically in exposing that concurrency in a way that allows our tools and hardware to take advantage of it, regardless of platform.

> **Find opportunities for improvement.** Using a high number of cores, plot the different dimensions of concurrency—thread scaling, vector scaling, and fabric scaling—to understand where you can tune your application to capitalize on high degrees of parallelism.

> **Use familiar and proven methods.** Intel is delivering on the promise of highly parallel computing using consistent programming models, languages, tools, and techniques. There is no need to adopt new methods or reinvent the wheel as you advance from a few cores to hundreds of cores.

The payoff for these investments is higher performance, whether the program runs on an Intel Xeon processor, Knights Corner, or the new Knights Landing, with lasting value for decades to come.
Resources

Knights Corner: Your Path to Knights Landing webinar by James Reinders

Intel® Xeon Phi™ Coprocessor High-Performance Programming, Jim Jeffers, James Reinders, 2013

High Performance Parallelism Pearls: Multicore and Many-Core Approaches, Jim Jeffers, James Reinders, 2014


Structured Parallel Programming, Michael McCool, Arch Robinson, and James Reinders, 2012

Online developer resources: software.intel.com/mic-developer

Try Intel® Compilers

Available in these software tools:
Intel® Parallel Studio XE Composer, Professional, and Cluster Editions >