Practical Usage of the Intel Math Kernel Library (MKL)

The Hands-On Tutorial (HOT) Series

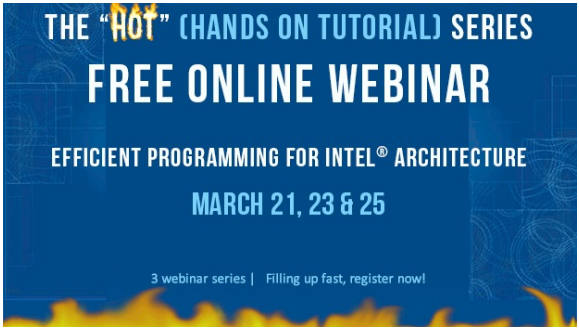
Andrey Vladimirov, PhD — Colfax International
colfaxresearch.com

Disclaimer

While best efforts have been used in preparing this training, Colfax International makes no representations or warranties of any kind and assumes no liabilities of any kind with respect to the accuracy or completeness of the contents and specifically disclaims any implied warranties of merchantability or fitness of use for a particular purpose. The publisher shall not be held liable or responsible to any person or entity with respect to any loss or incidental or consequential damages caused, or alleged to have been caused, directly or indirectly, by the information or programs contained herein. No warranty may be created or extended by sales representatives or written sales materials.

About the Series

Hands-On Tutorial (HOT) series: webinars on efficient programming for the Intel architecture. Select topics of parallel programming and performance optimization with detailed practical demonstrations.



colfaxresearch.com/hot-16-03

Learn More

Comprehensive Hands-On Workshop (HOW) series begins April 18, 2016. Free remote access to training servers (space is limited!):

colfaxresearch.com/how-16-04

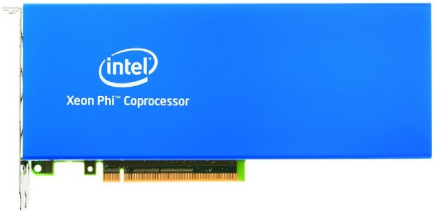
§2. Intel Architecture

Computing Platforms

Intel Xeon Processor

Current: Haswell
Upcoming: Broadwell

Intel Xeon Phi Corocessor, 1st generation



Current: Knights Corner (KNC)

Intel Xeon Phi Processor, 2nd generation*

* socket and coprocessor versions

Upcoming: Knights Landing (KNL)

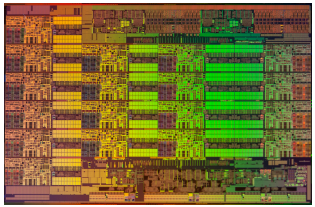
Multi-Core Architecture

Intel Many Integrated Core (MIC) Architecture

Intel Xeon CPU: Purpose and Specifications

General-purpose platform for demanding computing applications.

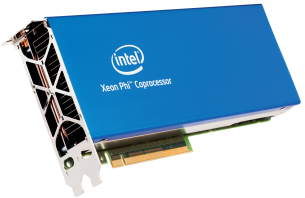
- Up to ~ 1 TFLOP/s in DP
- Up to ~ 2 TFLOP/s in SP
- Up to 768 GiB DDR4 RAM
- ~ 126 GB/s bandwidth
- Hardware-rich: forgiving of sub-optimal code

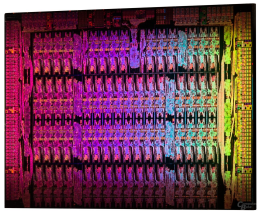


Intel Xeon Phi Processors (1st Gen)

Specialized platform for demanding computing applications.

- PCIe end-point device
- ~ 1.2 TFLOP/s in DP
- ~ 2.4 TFLOP/s in SP
- Up to 16 GiB GDDR5 RAM
- ~ 176 GB/s bandwidth
- Heterogeneous clustering
- Runs special Linux distribution

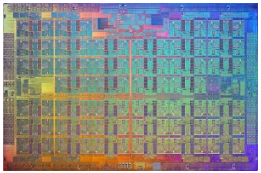




Intel Xeon Phi Processors (2nd Gen)

Specialized platform for demanding computing applications.

- Socket version or coprocessor
- 3+ TFLOP/s in DP
- 6+ TFLOP/s in SP
- Up to 16 GiB MCDRAM
- ~ 400 GB/s MCDRAM bandwidth
- Up to 384 GiB DDR4 RAM
- ~ 90 GB/s DDR4 bandwidth
- Supports common OS
- Public disclosures



§3. Intel Math Kernel Library

Performance Tuning Methodology

- **Scalar optimization** (compiler-friendly practices)
- **Vectorization** (must use 16- or 8-wide vectors)
- **Multi-threading** (must scale to 100+ threads)
- **Memory access** (streaming access or tiling)
- **Communication** (offload, MPI traffic control)

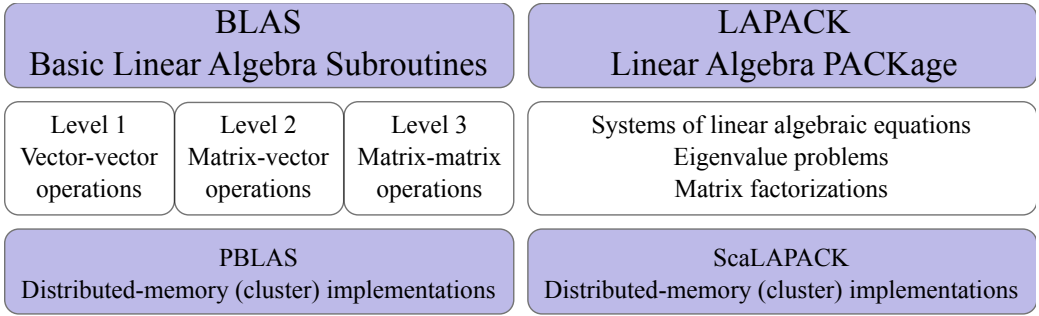
Details in the HOW Series and our book.

Scope of MKL

Intel Math Kernel Library (MKL) — standard mathematical functions optimized for Intel architecture.

Linear Algebra	Fast Fourier Transform	Vector Math	Vector Random Number Generators	Summary Statistics	Data Fitting
BLAS LAPACK Sparse solvers ScaLAPACK	Multidimentional (up to 7D) FFTW interfaces Cluster FFT	Trigonometric Hyperbolic Exponential Logarithmic Power/Root Rounding	Congruential Recursive Wichmann-Hill Mersenne Twister Sobol Neiderreiter Non-deterministic	Kurtosis Variation coefficitent Quantiles, order statistics Min/max Variance-covariance	Splines Interpolation Cell search

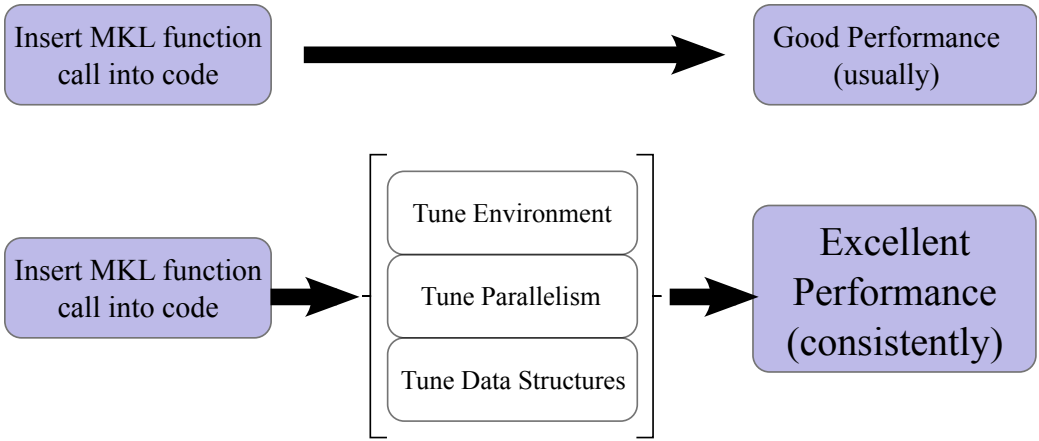
Linear Algebra Support



Using MKL with Intel Xeon Phi Coprocessors

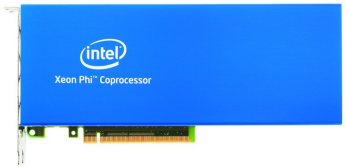
	Native	Compiler-Assisted Offload (CAO)	Automatic Offload (AO)
Programming:	Call MKL functions like from CPU code	Call MKL functions from #pragma offload	Call MKL functions from CPU code
Compilation:	With argument -mmic	Conventional for CPU	Conventional for CPUs
Running	On Xeon Phi	On host CPU	On host CPU; MKL_MIC_ENABLE=1
Data movement	Data already on coprocessor	Managed by programmer	Managed by MKL
Functions	All	All	Some; problem size > threshold
Multiple Copr.	Use cluster functions	DIY	Supported out-of-box

Performance Tuning with MKL



§4. Hands-On Labs

“Standard Candle” Testbench



vs.

One Intel Xeon Phi 7120P coprocessor (2012)
TDP: 300 W, RCP: $4129

Two Intel Xeon E5-2697 v3 CPUs (2014)
TDP: 290 W, RCP: $5404

See also “Intel Xeon Product Family: Performance Brief”

Batch Processing

- Feed multiple signals to MKL
- Let the library decide on the parallel strategy.

```
MKL_Complex16* data =
                (MKL_Complex16*) malloc(sizeof(MKL_Complex16)*fft_size*num_fft);

DFTI_DESCRIPTOR_HANDLE handle;
DftiCreateDescriptor(&handle, DFTI_DOUBLE, DFTI_COMPLEX, 1, (MKL_LONG)fft_size);
DftiSetValue(handle, DFTI_NUMBER_OF_TRANSFORMS, num_fft);
DftiSetValue(handle, DFTI_INPUT_DISTANCE, fft_size);
DftiSetValue(handle, DFTI_OUTPUT_DISTANCE, fft_size);
DftiSetValue(handle, DFTI_PLACEMENT, DFTI_INPLACE);
DftiCommitDescriptor(handle);
```

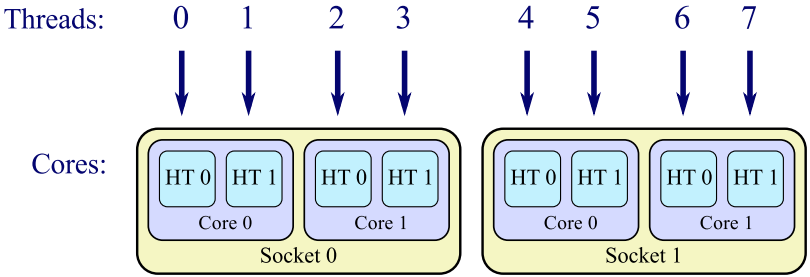
Thread Affinity Tuning

Xeon	**Xeon Phi**
MKL uses 1 thread/core	MKL uses 4 threads/core
`OMP_NUM_THREADS`=#phys. cores	`OMP_NUM_THREADS`=4×#cores
`KMP_AFFINITY`=scatter	`KMP_AFFINITY`=scatter
`KMP_AFFINITY`=compact (HT off)	`KMP_AFFINITY`=compact
`KMP_AFFINITY`=compact,1 (HT on)	

See also HOW series Session 8.

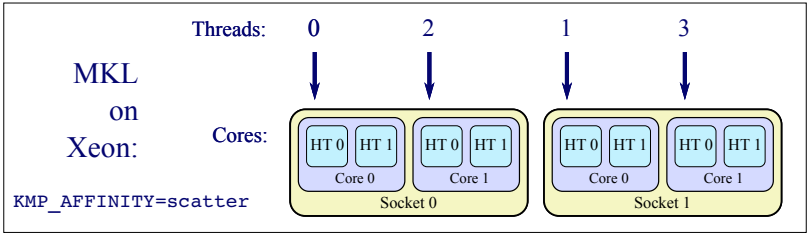
Thread Affinity: Compact Pattern

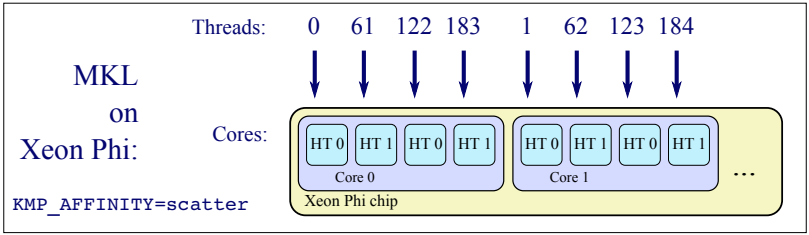
Generally beneficial for compute-bound applications.



Thread Affinity: Scatter Pattern

Generally beneficial for bandwidth-bound applications.





Data Container Tweaks

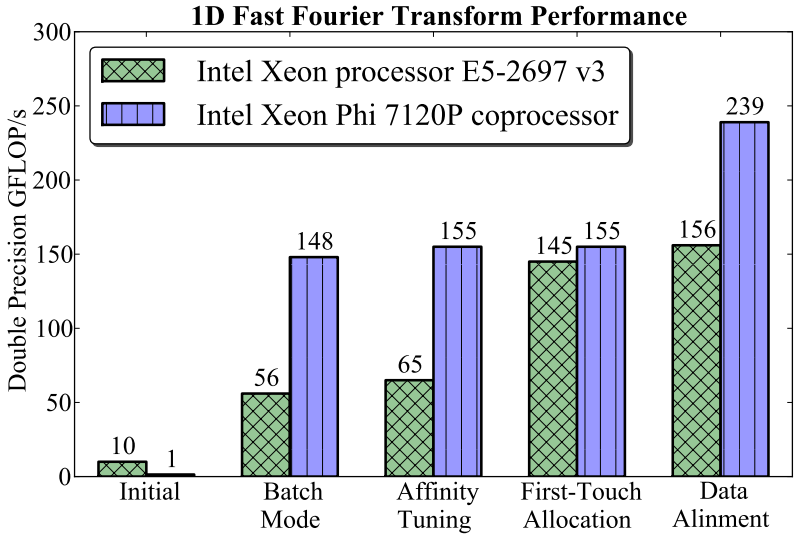
Parallel first-touch (see also HOW series Session 8):

```
#pragma omp parallel for
  for(int i = 0; i < num_fft; i++)
    for(int j = 0; j < fft_size; j++) {
      data[i*fft_size+j].real = cos(k*j);
      data[i*fft_size+j].imag = sin(k*j);
    }
```

Data alignment (see also HOW series Session 6)

```
MKL_Complex16* data =
        (MKL_Complex16*) mkl_malloc(sizeof(MKL_Complex16)*fft_size*num_fft, 64);
```

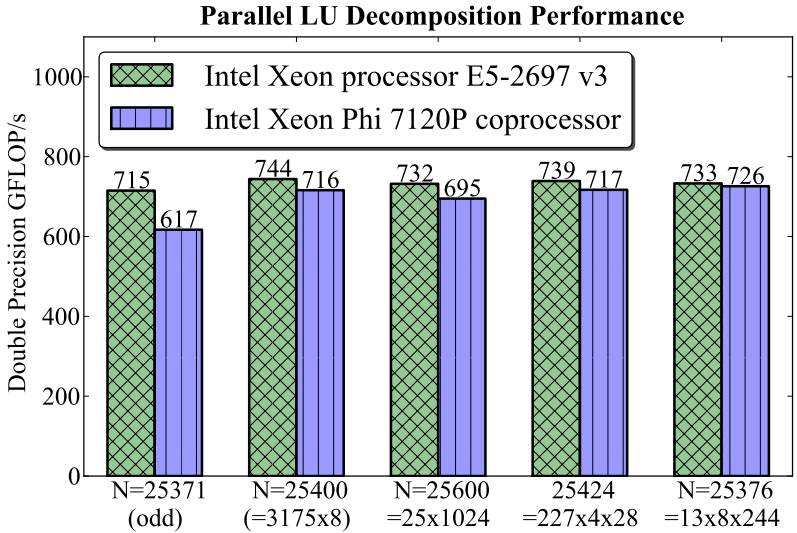
Complex-to-Complex 1D FFT Performance



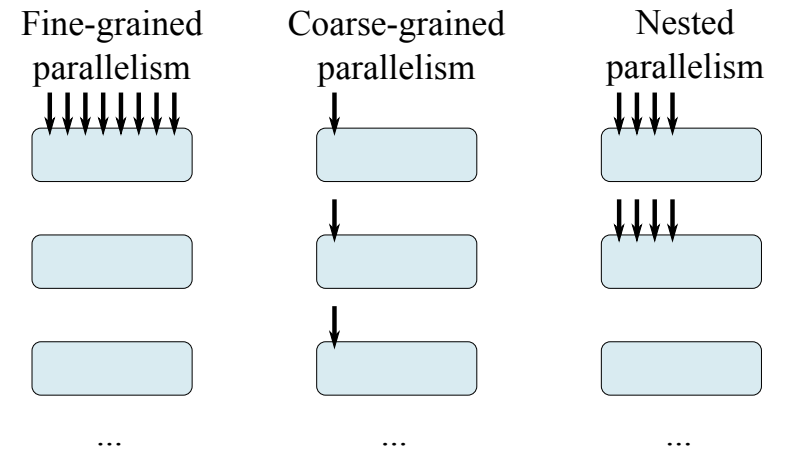
Problem Size Tuning

- Some functions require “good” problem sizes
- Good practice #1: array length a multiple of vector length
- Good practice #2: array length a multiple of number of threads
- Good practice #3: array length a multiple of a power of 2
- BLAS and LAPACK have “leading array dimension” for padding rows

LU Decomposition Size Tuning



Nested Parallelism



OpenMP Hot Teams

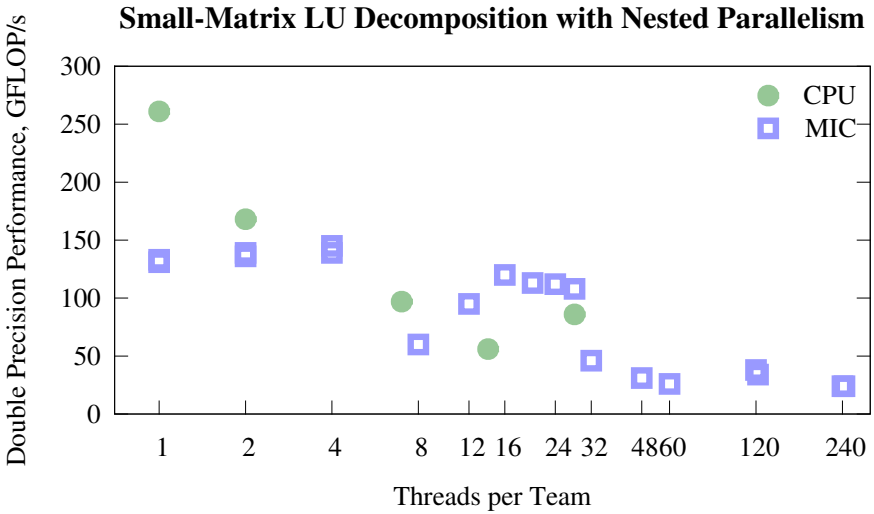
Xeon	Xeon Phi
`OMP_NUM_THREADS=2,14`	`OMP_NUM_THREADS=60,4`

`OMP_NESTED=1`
`OMP_PROC_BIND=spread,close`
`OMP_PLACES=cores`
	`OMP_NESTED=1`
	`OMP_PROC_BIND=spread,close`
	`OMP_PLACES=threads`

`KMP_HOT_TEAMS_MODE=1`
`KMP_HOT_TEAMS_MAX_LEVEL=2`
`OMP_MAX_ACTIVE_LEVELS=2`
	`KMP_HOT_TEAMS_MODE=1`
	`KMP_HOT_TEAMS_MAX_LEVEL=2`
	`OMP_MAX_ACTIVE_LEVELS=2`

`MKL_DYNAMIC=false`	`MKL_DYNAMIC=false`

Small Matrix LU Decomposition with Nested Parallelism



Automatic Offload

```
MKL_MIC_ENABLE=1

OMP_NUM_THREADS=... # CPU thread setting
KMP_AFFINITY=... # CPU affinity setting

MIC_KMP_PLACE_THREADS=4t # MIC thread setting
MIC_KMP_AFFINITY=... # MIC affinity setting
```

Compilation of R with MKL

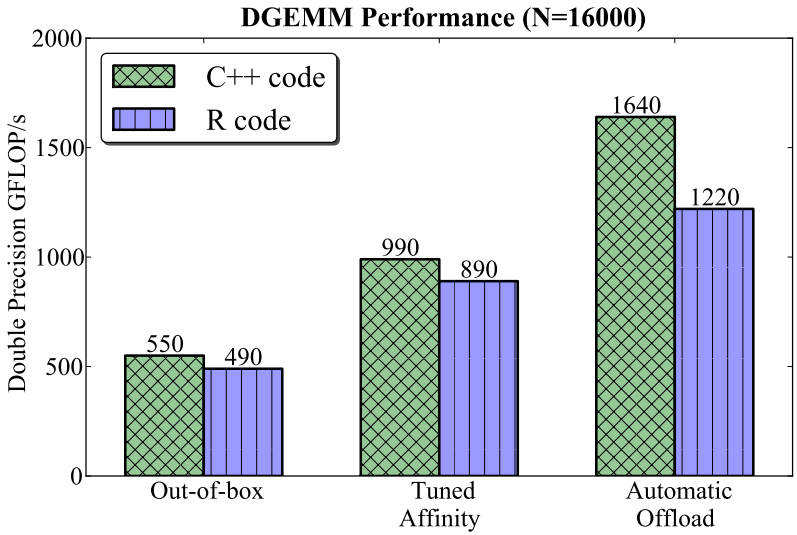
```
./configure \
   --with-blas="-L/opt/intel/mkl/lib/intel64 \
                -lmkl_intel_lp64 -lmkl_core -lmkl_intel_thread -lpthread -lm" \
   --with-lapack \
   CC=icc CFLAGS="-O2 -qopenmp -I/opt/intel/mkl/include" \
   CXX=icpc CXXFLAGS="-O2 -qopenmp -I/opt/intel/mkl/include" \
   F77=ifort FFLAGS="-O2 -qopenmp -I/opt/intel/mkl/include" \
   FC=ifort FCFLAGS="-O2 -qopenmp -I/opt/intel/mkl/include" \
   --prefix=/opt/R

user@host% make

user@host% su

root@host% make install
```

Performance with C++ Code and in R

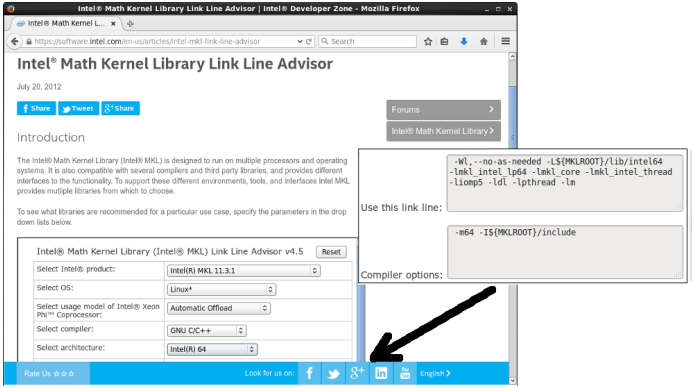


§5. Additional Information

Acquiring MKL

	Community License	Commercial License
Cost	Free	Per developer
Support	Community forum	Intel Premier Support
Use in products	Yes	Yes
Royalty-free	Yes	Yes

Compilation with MKL



https://software.intel.com/en-us/articles/intel-mkl-link-line-advisor

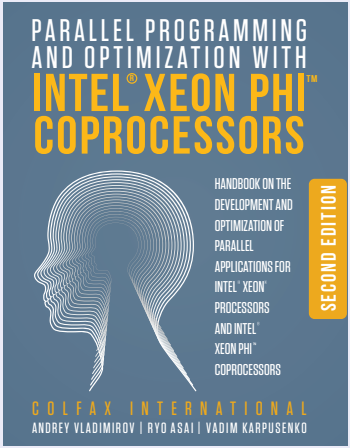
§6. Resources

Supplementary Materials: Textbook

ISBN: 978-0-9885234-0-1 (2nd edition, 508 pages, Electronic or Print)

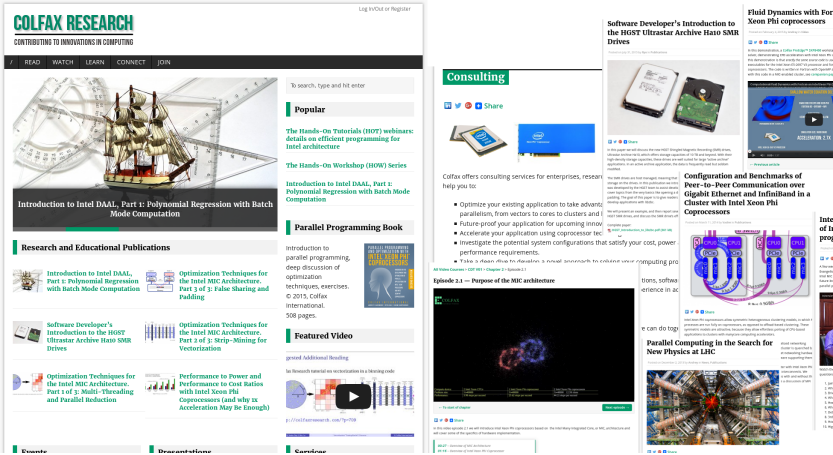
Parallel Programming
and Optimization with
Intel® Xeon Phi™
Coprocessors

Handbook on the Development and
Optimization of Parallel Applications
for Intel® Xeon® Processors
and Intel® Xeon Phi™ Coprocessors



http://xeonphi.com/book

Colfax Research



http://colfaxresearch.com/

Learn More

Comprehensive Hands-On Workshop (HOW) series begins April 18, 2016. Free remote access to training servers (space is limited!):

colfaxresearch.com/how-16-04

Slides, Code, Video

You can download slides, code and watch the video recording of this webinar here (requires registration for a free Colfax Research account):

colfaxresearch.com/hot-16-03

Past webinars (March 21 & 23, 2016: “From Scalar to Serial...” and “Finding the Low-Hanging Fruit for Performance Optimization”) — recordings on same page.