NTNU

Innovation and Creativity

# *Offload Compiler Runtime for the Intel® Xeon Phi™ Coprocessor*

**C.J. Newton, R. Deodhar, S. Dmitriev, et al.**

**TDT24 Paper presentation**

Christian Chavez
Institutt for Datateknikk og Informasjonsvitenskap

November 10, 2014

## Outline

NTNU
Innovation and Creativity

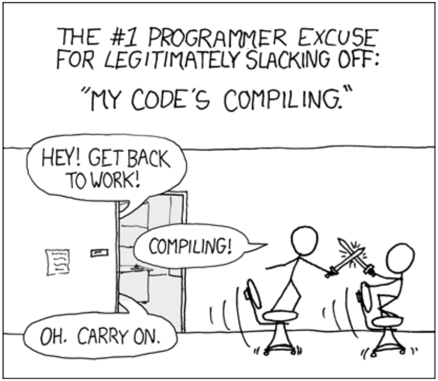
## Motivation

What happens when you have a codebase containing millions of lines of code?

NTNU
Innovation and Creativity

## Motivation

What happens when you have a codebase containing millions of lines of code?



NTNU
Innovation and Creativity

## GPGPUs

So, how can *GPGPUs* help?

1. Split up and run sub-parts of the compilation **concurrently** with host (CPU)

## GPGPUs

So, how can *GPGPUs* help?

1. Split up and run sub-parts of the compilation **concurrently** with host (CPU)
2. Run **computationally heavy** parts of the compiler algorithms, while the CPU works on the more serial ones.

## Intel coprocessors

Intel wants to enter this market with the help of the **Xeon Phi** processor family

## Intel coprocessors

Intel wants to enter this market with the help of the **Xeon Phi** processor family

— Intended as a **competitor** against Nvidia CUDA GPGPUs in the HPC market

## Xeon Phi drawbacks

Obvious *drawbacks* with Xeon Phi (compared with the CUDA GPGPUs)

1. Not as **mature** ecosystem/tools/environment

NTNU
Innovation and Creativity

## Xeon Phi drawbacks

Obvious *drawbacks* with Xeon Phi (compared with the CUDA GPGPUs)

1. Not as **mature** ecosystem/tools/environment
2. Still requires a high degree of **computation to communication** to be beneficial to run on the Phi instead of the CPU

NTNU
Innovation and Creativity

## Xeon Phi drawbacks

Obvious *drawbacks* with Xeon Phi (compared with the CUDA GPGPUs)

1. Not as **mature** ecosystem/tools/environment
2. Still requires a high degree of **computation to communication** to be beneficial to run on the Phi instead of the CPU
3. Still new and relatively untested/unused wrt. compilers

NTNU
Innovation and Creativity

## Xeon Phi drawbacks

Obvious *drawbacks* with Xeon Phi (compared with the CUDA GPGPUs)

1. Not as **mature** ecosystem/tools/environment
2. Still requires a high degree of **computation to communication** to be beneficial to run on the Phi instead of the CPU
3. Still new and relatively untested/unused wrt. compilers
4. Being Intel, everything is “canned” (not open source)

NTNU
Innovation and Creativity

## Xeon Phi benefits

Obvious *benefits* with Xeon Phi (compared with the CUDA GPGPUs)

1. The architecture supports regular Intel CPU instructions[1]

[1] Not confirmed exactly how much

NTNU
Innovation and Creativity

## Xeon Phi benefits

Obvious *benefits* with Xeon Phi (compared with the CUDA GPGPUs)

1. The architecture supports regular Intel CPU instructions[1]
   - The Xeon Phi does not require its own codebase for being run.

[1] Not confirmed exactly how much

NTNU
Innovation and Creativity

## Runtime offloading tools

The papers focus is on the following tools:

— Intel Manycore Platform Software Stack (MPSS)

## Runtime offloading tools

The papers focus is on the following tools:

- Intel Manycore Platform Software Stack (MPSS)
  - Intel Coprocessor Offload Infrastructure (COI)

NTNU
Innovation and Creativity

## Runtime offloading tools

The papers focus is on the following tools:

— Intel Manycore Platform Software Stack (MPSS)

  - Intel Coprocessor Offload Infrastructure (COI)

— COI utilizes:

NTNU
Innovation and Creativity

## Runtime offloading tools

The papers focus is on the following tools:

— Intel Manycore Platform Software Stack (MPSS)
- Intel Coprocessor Offload Infrastructure (COI)

— COI utilizes:
- Library calls

## Runtime offloading tools

The papers focus is on the following tools:

- Intel Manycore Platform Software Stack (MPSS)
  - Intel Coprocessor Offload Infrastructure (COI)
- COI utilizes:
  - Library calls
  - Language pragmas

NTNU
Innovation and Creativity

## Runtime offloading tools

The papers focus is on the following tools:

— Intel Manycore Platform Software Stack (MPSS)
  - Intel Coprocessor Offload Infrastructure (COI)

— COI utilizes:
  - Library calls
  - Language pragmas
  - Language keywords

## Runtime offloading tools

The papers focus is on the following tools:

- Intel Manycore Platform Software Stack (MPSS)
  - Intel Coprocessor Offload Infrastructure (COI)
- COI utilizes:
  - Library calls
  - Language pragmas
  - Language keywords
  - A FIFO command queue (dataobject: COIPipeline)

NTNU
Innovation and Creativity

## Runtime offloading tools

The papers focus is on the following tools:

— Intel Manycore Platform Software Stack (MPSS)
  - Intel Coprocessor Offload Infrastructure (COI)

— COI utilizes:
  - Library calls
  - Language pragmas
  - Language keywords
  - A FIFO command queue (dataobject: COIPipeline)

— OpenMP

NTNU
Innovation and Creativity

## Platform specifications

Test platform specifications:

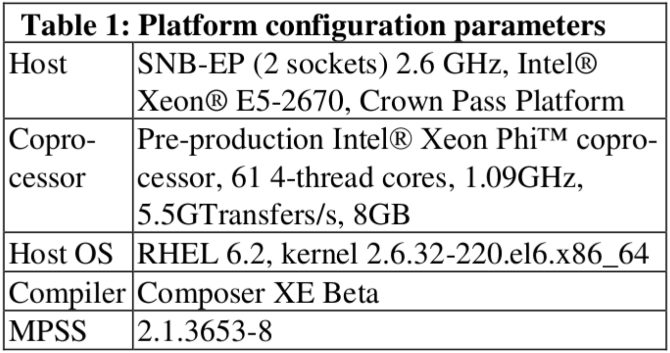


**Table 1: Platform configuration parameters**

| | |
|---|---|
| Host | SNB-EP (2 sockets) 2.6 GHz, Intel® Xeon® E5-2670, Crown Pass Platform |
| Copro-cessor | Pre-production Intel® Xeon Phi™ copro-cessor, 61 4-thread cores, 1.09GHz, 5.5GTransfers/s, 8GB |
| Host OS | RHEL 6.2, kernel 2.6.32-220.el6.x86_64 |
| Compiler | Composer XE Beta |
| MPSS | 2.1.3653-8 |

NTNU
Innovation and Creativity

Two sets of tests:

1. A test spanning a variety of application domains, coming from customers.

Two sets of tests:

1. A test spanning a variety of application domains, coming from customers.
    - Hence, some names are occluded in deference to customers.

Two sets of tests:

1. A test spanning a variety of application domains, coming from customers.
    - Hence, some names are occluded in deference to customers.
    - The tests are meaningfully representative to said customers.

Two sets of tests:

1. A test spanning a variety of application domains, coming from customers.
    - Hence, some names are occluded in deference to customers.
    - The tests are meaningfully representative to said customers.
2. An array of benchmarks from Scalable HeterOgeneous Computing (*SHOC*) benchmarks suite.

NTNU
Innovation and Creativity

Two sets of tests:

1. A test spanning a variety of application domains, coming from customers.
    - Hence, some names are occluded in deference to customers.
    - The tests are meaningfully representative to said customers.
2. An array of benchmarks from Scalable HeterOgeneous Computing (*SHOC*) benchmarks suite.
    - Only used tests that were not exclusive to OpenCL and CUDA

NTNU
Innovation and Creativity

Two sets of tests:

1. A test spanning a variety of application domains, coming from customers.
   - Hence, some names are occluded in deference to customers.
   - The tests are meaningfully representative to said customers.
2. An array of benchmarks from Scalable HeterOgeneous Computing (*SHOC*) benchmarks suite.
   - Only used tests that were not exclusive to OpenCL and CUDA
   - Only used tests that have both offload and native versions

NTNU
Innovation and Creativity

Two sets of tests:

1. A test spanning a variety of application domains, coming from customers.
    - Hence, some names are occluded in deference to customers.
    - The tests are meaningfully representative to said customers.
2. An array of benchmarks from Scalable HeterOgeneous Computing (*SHOC*) benchmarks suite.
    - Only used tests that were not exclusive to OpenCL and CUDA
    - Only used tests that have both offload and native versions
    - Results were measured by Intel in August 2012.

NTNU
Innovation and Creativity

Two sets of tests:

1. A test spanning a variety of application domains, coming from customers.
   - Hence, some names are occluded in deference to customers.
   - The tests are meaningfully representative to said customers.
2. An array of benchmarks from Scalable HeterOgeneous Computing (*SHOC*) benchmarks suite.
   - Only used tests that were not exclusive to OpenCL and CUDA
   - Only used tests that have both offload and native versions
   - Results were measured by Intel in August 2012.
   - The benchmark application *Triad* was also added, even though it has no native version

NTNU
Innovation and Creativity

## Customer tests results

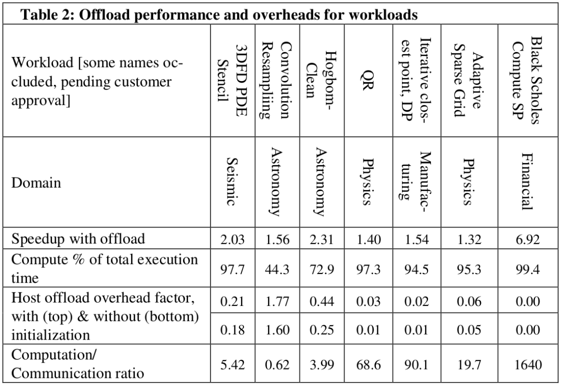


**Table 2: Offload performance and overheads for workloads**

| Workload [some names occluded, pending customer approval] | 3DFD PDE Stencil | Convolution Resampling | Hogbom-Clean | QR | Iterative closest point, DP | Adaptive Sparse Grid | Black Scholes Compute SP |
|---|---|---|---|---|---|---|---|
| Domain | Seismic | Astronomy | Astronomy | Physics | Manufacturing | Physics | Financial |
| Speedup with offload | 2.03 | 1.56 | 2.31 | 1.40 | 1.54 | 1.32 | 6.92 |
| Compute % of total execution time | 97.7 | 44.3 | 72.9 | 97.3 | 94.5 | 95.3 | 99.4 |
| Host offload overhead factor, with (top) & without (bottom) initialization | 0.21 | 1.77 | 0.44 | 0.03 | 0.02 | 0.06 | 0.00 |
| | 0.18 | 1.60 | 0.25 | 0.01 | 0.01 | 0.05 | 0.00 |
| Computation/ Communication ratio | 5.42 | 0.62 | 3.99 | 68.6 | 90.1 | 19.7 | 1640 |

NTNU
Innovation and Creativity

# SHOC tests results

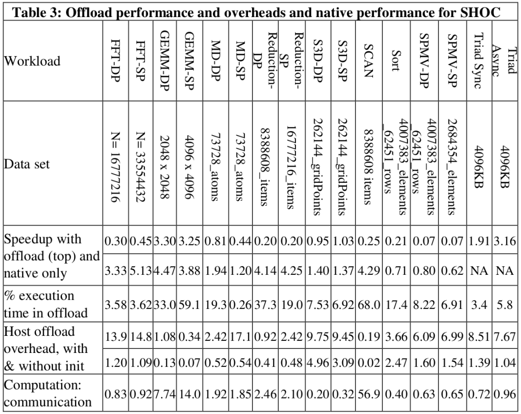


Table 3: Offload performance and overheads and native performance for SHOC

| Workload | FFT-DP | FFT-SP | GEMM-DP | GEMM-SP | MD-DP | MD-SP | Reduction-DP | Reduction-SP | S3D-DP | S3D-SP | SCAN | Sort | SPMV-DP | SPMV-SP | Triad Sync | Triad Async |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Data set | N= 16777216 | N= 33554432 | 2048 x 2048 | 4096 x 4096 | 73728_atoms | 73728_atoms | 8388608_items | 16777216_items | 262144_gridPoints | 262144_gridPoints | 8388608 items | 4007383_elements _62451_rows | 4007383_elements _62451_rows | 2684354_elements | 4096KB | 4096KB |
| Speedup with offload (top) and native only | 0.30 | 0.45 | 3.30 | 3.25 | 0.81 | 0.44 | 0.20 | 0.20 | 0.95 | 1.03 | 0.25 | 0.21 | 0.07 | 0.07 | 1.91 | 3.16 |
| | 3.33 | 5.13 | 4.47 | 3.88 | 1.94 | 1.20 | 4.14 | 4.25 | 1.40 | 1.37 | 4.29 | 0.71 | 0.80 | 0.62 | NA | NA |
| % execution time in offload | 3.58 | 3.62 | 33.0 | 59.1 | 19.3 | 0.26 | 37.3 | 19.0 | 7.53 | 6.92 | 68.0 | 17.4 | 8.22 | 6.91 | 3.4 | 5.8 |
| Host offload overhead, with & without init | 13.9 | 14.8 | 1.08 | 0.34 | 2.42 | 17.1 | 0.92 | 2.42 | 9.75 | 9.45 | 0.19 | 3.66 | 6.09 | 6.99 | 8.51 | 7.67 |
| | 1.20 | 1.09 | 0.13 | 0.07 | 0.52 | 0.54 | 0.41 | 0.48 | 4.96 | 3.09 | 0.02 | 2.47 | 1.60 | 1.54 | 1.39 | 1.04 |
| Computation: communication | 0.83 | 0.92 | 7.74 | 14.0 | 1.92 | 1.85 | 2.46 | 2.10 | 0.20 | 0.32 | 56.9 | 0.40 | 0.63 | 0.65 | 0.72 | 0.96 |

NTNU
Innovation and Creativity

## Speedups 1/2

— Computation to communication ratio varied from 0.62 to 1640

NTNU
Innovation and Creativity

## Speedups 1/2

— Computation to communication ratio varied from 0.62 to 1640

1. But it was generally over 4, except for the convolution case.

NTNU
Innovation and Creativity

## Speedups 1/2

— Computation to communication ratio varied from 0.62 to 1640

1. But it was generally over 4, except for the convolution case.
2. The convolution case scaling across threads and SIMD elements on the coprocessor allowed the computation speedup to outweigh the overheads

NTNU
Innovation and Creativity

## Speedups 1/2

— Computation to communication ratio varied from 0.62 to 1640
   1. But it was generally over 4, except for the convolution case.
   2. The convolution case scaling across threads and SIMD elements on the coprocessor allowed the computation speedup to outweigh the overheads

— Data movement overhead varied from neglegible to 1.77x the computation time

NTNU
Innovation and Creativity

## Speedups 1/2

- Computation to communication ratio varied from 0.62 to 1640
  1. But it was generally over 4, except for the convolution case.
  2. The convolution case scaling across threads and SIMD elements on the coprocessor allowed the computation speedup to outweigh the overheads
- Data movement overhead varied from neglegible to 1.77x the computation time
- Black Scholes was the big performance winner, with a 6.92x speedup of the dual-socket SandyBridge

NTNU
Innovation and Creativity

## Speedups 1/2

— Computation to communication ratio varied from 0.62 to 1640
  1. But it was generally over 4, except for the convolution case.
  2. The convolution case scaling across threads and SIMD elements on the coprocessor allowed the computation speedup to outweigh the overheads

— Data movement overhead varied from neglegible to 1.77x the computation time

— Black Scholes was the big performance winner, with a 6.92x speedup of the dual-socket SandyBridge

— SHOC shows no correlation between speedup and computation to communication.

NTNU
Innovation and Creativity

## Speedups 1/2

- Computation to communication ratio varied from 0.62 to 1640
  1. But it was generally over 4, except for the convolution case.
  2. The convolution case scaling across threads and SIMD elements on the coprocessor allowed the computation speedup to outweigh the overheads
- Data movement overhead varied from neglegible to 1.77x the computation time
- Black Scholes was the big performance winner, with a 6.92x speedup of the dual-socket SandyBridge
- SHOC shows no correlation between speedup and computation to communication.
  - Paper lists this as potential future work. "Broader investigation is needed on that"

NTNU
Innovation and Creativity

## Speedups 2/2

"No single factor determines offload performance"

— Data movement overhead varied from neglegible to 1.77x the computation time

Offload speedups are most correlated with the following:

NTNU
Innovation and Creativity

## Speedups 2/2

"No single factor determines offload performance"

— Data movement overhead varied from neglegible to 1.77x the computation time

Offload speedups are most correlated with the following:

1. Inverse of offload overheads

NTNU
Innovation and Creativity

## Speedups 2/2

“No single factor determines offload performance”

— Data movement overhead varied from neglegible to 1.77x the computation time

Offload speedups are most correlated with the following:

1. Inverse of offload overheads
2. % of execution time in offload (which includes coprocessor invocation and data movement)

NTNU
Innovation and Creativity

## Speedups 2/2

"No single factor determines offload performance"

— Data movement overhead varied from neglegible to 1.77x the computation time

Offload speedups are most correlated with the following:

1. Inverse of offload overheads
2. % of execution time in offload (which includes coprocessor invocation and data movement)
3. Native performance (when there is no execution on the host at all)

NTNU
Innovation and Creativity

## Conclusion

- The Xeon Phi in conjunction with the platform used in the paper showed benchmark application speedups from 1.3x to 6.9 on “customer-relevant” examples spanning different application domains
- Offload is not always profitable, as SHOC showed
- Inhibitions or enhancements of speedup this paper has explored:
  1. When response time is of concern, there must be a speedup from native execution on only the coprocessor, relative to the execution on the host
  2. Ratio of computation to communication must be generally high
  3. Offload runtime overheads must be small relative to computation

NTNU
Innovation and Creativity

## Sources

— *Offload Compiler Runtime for the Intel® Xeon Phi™ Coprocessor*, Newton, R. Deodhar, S. Dmitriev, et al.[2]

— Randall Munroe, xkcd[3]

[2] `https://software.intel.com/sites/default/files/article/366893/offload-runtime-for-the-intelr-xeon-phitm-coprocessor.pdf`

[3] `http://xkcd.com/303/`

NTNU
Innovation and Creativity