Stream on Intel® Xeon Phi™ Coprocessors (updated October 22nd, 2013)

Overview

This document demonstrates the best methods to obtain peak memory bandwidth performance on Intel® Xeon Phi™ coprocessor using the *de facto* industry standard benchmark for the measurement of computer memory bandwidth - "STREAM"

Introduction

The STREAM benchmark is a simple, synthetic benchmark designed to measure sustainable memory bandwidth (in MB/s) and a corresponding computation rate for four simple vector kernels (Copy, Scale, Add and Triad). Its source code is freely available from http://www.cs.virginia.edu/stream/. STREAM is also a part of the HPCC Benchmark suite.

STREAM Rules

The general rule for STREAM is that each array must be at least 4x the size of the sum of all the last-level caches used in the run, or 1 million elements -- whichever is *larger*.

Standard vs. Tuned

There are two categories created by the STREAM author for citing memory bandwidth score. The kernels in the published link above "as is" are considered "Standard". The "Tuned" category has been added to allow users or vendors to submit results based on modified source code. This category explicitly allows assembly-language coded kernels. The code needs to be based on the sample harness provided by the author in the STREAM webpage. The Intel® Xeon Phi™ coprocessor results on the STREAM benchmark fall under "Standard" category.

Triad

Of all the vector kernels Triad is the most complex scenario and is highly relevant to HPC.

The STREAM Triad kernel is as follows:

```
#pragma parallel for

for (i =0; i<N; i++)

{

a[i] = b[i] + c[i] * SCALAR;

 }
```

Directions to Compile and Run STREAM on Intel® Xeon Phi™ Coprocessors

1) Without the use of 2MB pages

- Download the latest stream.c from http://www.cs.virginia.edu/stream/FTP/Code/

- Use the Intel(R) Parallel Studio XE 2013
- Compile with the following knobs: (Please check "Compiler Knobs" section below to know what each knob signifies)

-mmic -O3 -openmp -DSTREAM_ARRAY_SIZE=64000000 -opt-prefetch-distance=64,8 -opt-streaming-cache-evict=0 -opt-streaming-stores always

- Upload the binary & dependencies to the Intel® Xeon Phi™ coprocessor (You may have to change path depending on the compiler version)

- **scp stream mic0:/tmp/stream**
- **scp /opt/intel/composer_xe_2013.1.117/compiler/lib/mic/libiomp5.so mic0:/tmp/stream**

- Login to the Intel® Xeon Phi™ coprocessor and go to the path where your binary is located (cd /tmp) ; set two environment variables and run your binary as follows:

- **export KMP_AFFINITY=scatter**
  For Intel® Xeon Phi™ coprocessor 7110P (61 cores, 1.1GHz, 5.5GT/s)
- **export OMP_NUM_THREADS=60**
  - note: Use one less than number of physical cores
- **export LD_LIBRARY_PATH=/tmp:$LD_LIBRARY_PATH**
- **Run binary (./stream)**

2) Using 2MB pages

Note: You will need "root" access to allocate 2MB pages in this case

a) Method 1 via libhugetlbfs library (see Method 2 below, no root access required)

- Download the latest stream.c from http://www.cs.virginia.edu/stream/FTP/Code/

- Download the latest libhugetlbfs package (for 2MB pages) from http://sourceforge.net/projects/libhugetlbfs/files/
- Unzip/Untar the libhugetlbfs package (downloaded above)
  - You will get a libhugetlbfs* folder

- Go to the libhugetlbfs* directory and build the libhugetlbfs.so for Intel® Xeon Phi™ Coprocessor
  - Use the Intel(R) Parallel Studio XE 2013

- **make clean**
- **make ARCH=x86_64 CC64='icc –mmic' libs BUILDTYPE=NATIVEONLY**
  - Look for your library (libhugetlbfs.so) in obj64 directory

- Comment or Remove the Following lines in **/path-to-libhugetlbfs-dir/ldscripts/elf_x86_64.xBDT** file (required for Intel® Xeon Phi™ Coprocessor)

OUTPUT_FORMAT("elf64-x86-64", "elf64-x86-64","elf64-x86-64")
OUTPUT_ARCH (i386:x86-64)
SEARCH_DIR ("/usr/x86_64-linux-gnu/lib64"); SEARCH_DIR("/usr/local/lib64");
SEARCH_DIR("/lib64"); SEARCH_DIR("/usr/lib64"); SEARCH_DIR("/usr/x86_64-linux-gnu/lib");
SEARCH_DIR ("/usr/local/lib"); SEARCH_DIR("/lib"); SEARCH_DIR("/usr/lib");

- Use the Intel(R) Parallel Studio XE 2013
- Compile your stream source with the following knobs:

-mmic -O3 -openmp -DSTREAM_ARRAY_SIZE=64000000 -opt-prefetch-distance=64,8 -opt-streaming-cache-evict=0 -opt-streaming-stores always -Wl,-T/path-to-libhugetlbfs-dir/ldscripts/elf_x86_64.xBDT -L/path-to-libhugetlbfs-dir/obj64

- Allocate required no. of hugepages on the Intel® Xeon Phi™ coprocessor: (From Host) – **as "root" (sudo su)**

  - **ssh mic0 'echo 623 > /proc/sys/vm/nr_hugepages'**

P.S: Above we have allocated "623" 2MB pages as an example; this can be changed depending on your application

- Mount huge pages on the Intel® Xeon Phi™ coprocessor: (From Host) – **as "root" (sudo su)**

  - **ssh mic0 'mkdir -p /mnt/hugetlbfs'**
  - **ssh mic0 'mount -t hugetlbfs none /mnt/hugetlbfs'**

- Upload the binary and dependencies to the Intel® Xeon Phi™ coprocessor:

  - **scp stream_2MB mic0:/tmp/stream_2MB**
  - **scp /opt/intel/composer_xe_2013.1.117/compiler/lib/mic/libsvml.so mic0:/tmp/**
  - **scp /opt/intel/composer_xe_2013.1.117/compiler/lib/mic/libintlc.so.5 mic0:/tmp/**
  - **scp /opt/intel/composer_xe_2013.1.117/compiler/lib/mic/libintlc.so mic0:/tmp/**
  - **scp /opt/intel/composer_xe_2013.1.117/compiler/lib/mic/libimf.so mic0:/tmp/**
  - **scp /opt/intel/composer_xe_2013.1.117/compiler/lib/mic/libirng.so mic0:/tmp/**
  - **scp /path-to-libhugetlbfs-dir/obj64/libhugetlbfs.so mic0:/tmp/**

- Login to the Intel® Xeon Phi™ coprocessor and go to the path where your binary is located (cd /tmp) and set two environment variables

  - **export KMP_AFFINITY=scatter**
    For Intel® Xeon Phi™ coprocessor 7110P (61 cores, 1.1GHz, 5.5GT/s):
  - **export OMP_NUM_THREADS=60**
    - note: Use one less than number of physical cores
  - **export LD_LIBRARY_PATH=/tmp:$LD_LIBRARY_PATH**
  - **Run binary (./stream_2MB)**

b) Method 2

- Upgrade to the Intel® Manycore Platform Software Stack (Intel® MPSS) gold update (**KNC_gold_update_1-2.1.4982-15) or later**

  http://software.intel.com/en-us/articles/intel-manycore-platform-software-stack-mpss

- This update has the "Transparent Huge pages" support which automatically promotes 4K pages to 2MB pages for stack and heap allocated data
- "Transparent huge pages" is a Linux kernel feature introduced in kernel version 2.6.38
- Using this Software Stack one does not have to use huge pages (libhugetlbfs library) method described in Method1 above to get the extra performance for "STREAM"
- We can achieve peak performance for STREAM without huge pages (thus not needing any "root" access)
- Follow the same steps as "Without the use of 2MB pages"

Compiler Knobs

1. **–mmic :** build an application that runs natively on Intel(R) Xeon Phi coprocessor
2. **–O3 :** optimize for maximum speed and enable more aggressive optimizations that may not improve performance on some programs
3. **–openmp** : enable the compiler to generate multi-threaded code based on the OpenMP* directives (same as -fopenmp)
4. **-opt-prefetch-distance=64,8 :** Software Prefetch 64 cachelines ahead for L2 cache; Software Prefetch 8 cachelines ahead for L1 cache
5. **-opt-streaming-cache-evict=0 :** Turn off all cache line evicts
6. **-opt-streaming-stores always :** enables generation of streaming stores under the assumption that the application is memory bound
7. **-DSTREAM_ARRAY_SIZE=64000000 :** Increasing the size of the array size to be compliant with the STREAM Rules

Results

The results below are on a pre-production Intel® Xeon Phi™ coprocessor (specifications in the table below), µOS version 2.6.34.11-g65c0cd9 with Flash version 2.1.01.0375 and Intel MPSS version 2.1.4346-16 (Gold Stack). The OS running on the host is Red Hat Enterprise Linux Server release 6.1
The libhugetlbfs-2.12 version was used for 2MB pages.

Workload	**ECC**	**2MB pages**	**Intel® Xeon Phi™ 5110P 60c / 1.053GHz / 5.0GTS**	**Intel® Xeon Phi™ 7110P 61c / 1.1GHz / 5.5GTS**
Stream Triad	On	Yes	159GB/s	174 GB/s
Stream Triad	Off	Yes	171GB/s	181 GB/s
Stream Triad	On	No	150GB/s	164 GB/s
Stream Triad	Off	No	168GB/s	178 GB/s

The results below are on a pre-production Intel® Xeon Phi™ coprocessor (specifications in the table below), µOS version 2.6.38.8-g32944d0 with Flash version 2.1.05.0375 and Intel MPSS version 2.1.4982-15 (Gold Stack update). The OS running on the host is Red Hat Enterprise Linux Server release 6.1. Due to "Transparent Huge page" support no libhugetlbfs library required.

Workload	**ECC**	**Intel® Xeon Phi™ 7110P 61c / 1.1GHz / 5.5GTS**
Stream Triad	On	174 GB/s
Stream Triad	Off	181 GB/s

The results below are on a pre-production Intel® Xeon Phi™ coprocessor (specifications in the table below), µOS version 2.6.38.8-g2593b11with Flash version 2.1.03.0386 and Intel MPSS version 2.1.6720-15 (Gold Stack update). The OS running on the host is Red Hat Enterprise Linux Server release 6.1. Due to "Transparent Huge page" support no libhugetlbfs library required.

Workload	**ECC**	**Intel® Xeon Phi™ 7120P 61c / 1.238GHz / 5.5GTS**
Stream Triad	On	177 GB/s
Stream Triad	Off	192 GB/s

Additional Resources

Intel® C++ Compiler XE 13.0 User and Reference Guides:

- http://software.intel.com/en-us/articles/programming-and-compiling-for-intel-many-integrated-core-architecture
- http://software.intel.com/sites/products/documentation/doclib/stdxe/2013/composerxe/compiler/cpp-lin/index.htm

Stream Benchmark Open source:

- http://www.cs.virginia.edu/stream/

Acknowledgements

The author would like to thank the Intel Compiler Team, Paul Besl - Software Engineering Manager and John McCalpin - author of the STREAM Benchmark

About the Author

Karthik Raman is a Software Architect in the Intel Software and Services Group (SSG).

Notices

Optimization Notice

http://software.intel.com/en-us/articles/optimization-notice/

Performance Notice

For more complete information about performance and benchmark results, visit www.intel.com/benchmarks