GPU-STREAM: now in 2D!

Tom Deakin, James Price, Matt Martineau, Simon McIntosh-Smith
Department of Computer Science, University of Bristol, UK

Introduction

We present a major update to the GPU-STREAM benchmark implementation, first shown at SC'15. The original benchmark allowed comparison of achievable memory bandwidth performance through the STREAM kernels on OpenCL devices. GPU-STREAM v2.0 extends the benchmark to another dimension: the kernels are implemented in a wide range of popular state-of-the-art parallel programming models. This allows an intuitive comparison of performance across a diverse set of programming models and devices, investigating whether choice of model matters to performance and performance portability. In particular we investigate **7 parallel programming languages**

- OpenMP 4.x
- OpenACC
- Kokkos
- RAJA
- SYCL
- CUDA
- OpenCL

across **12 devices**

- 6 GPUs from NVIDIA and AMD
- Intel Xeon Phi (Knights Landing)
- 4 generations of Intel Xeon CPUs
- IBM Power 8

Devices

Name	Peak Memory BW (GB/s)
NVIDIA K20X GPU	250
NVIDIA K40 GPU	288
NVIDIA K80 GPU (1 GPU)	240
NVIDIA GTX 980 Ti GPU	336.5
AMD S9150 GPU	320
AMD Fury X GPU	512
Intel E5-2670 (Sandy Bridge) CPU	2×51.2=102.4
Intel E5-2697 v2 (Ivy Bridge) CPU	2×59.7=119.4
Intel E5-2698 v3 (Haswell) CPU	2×68=136
Intel E5-2699 v4 (Broadwell) CPU	2×76.8=153.6
Intel Xeon Phi (Knights Landing) 7210	∼5×102 = 510
IBM Power 8 CPU	2×192=384

Conclusions

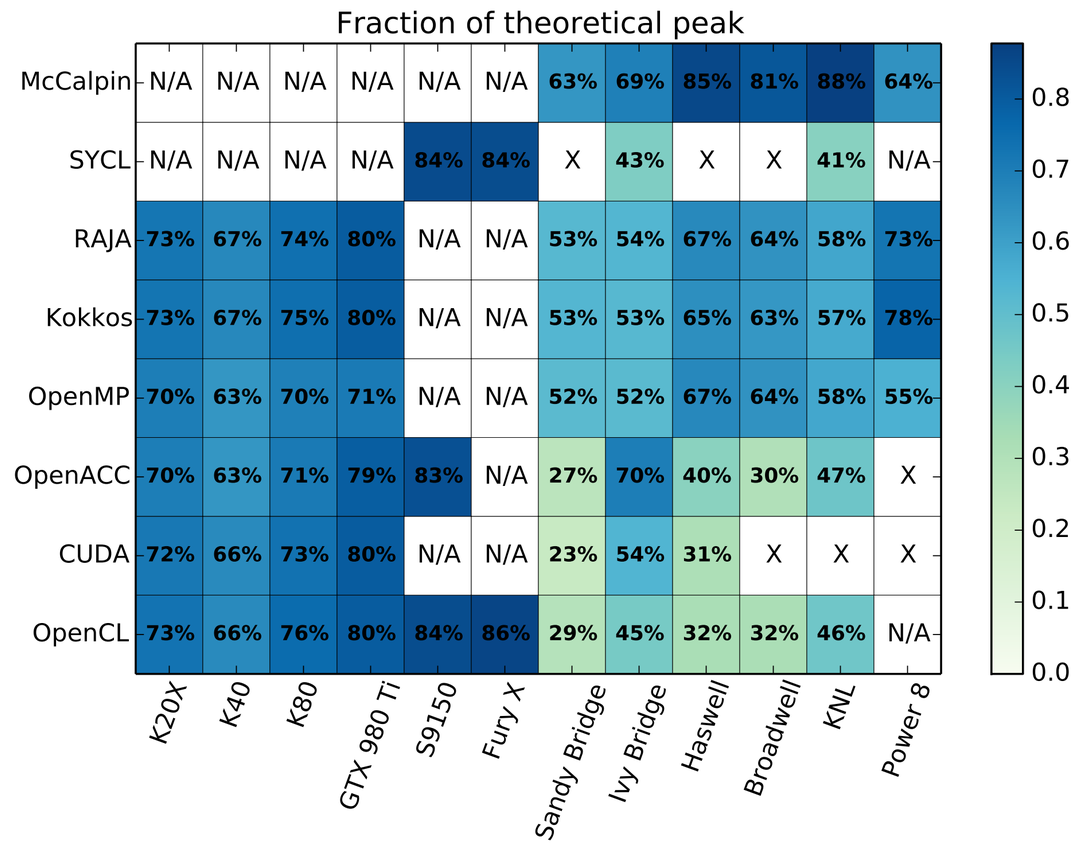
- Support of programming models on AMD GPUs lacking across the board
  - But they had the best memory bandwidth of the GPUs tested
- Have C++ code? Go for the RAJA or Kokkos C++-based approaches
- Have C or Fortran code? Go for the directive based approach of OpenMP
- OpenMP, RAJA and Kokkos all get near to 'close-to-the-metal' performance
- C++ OpenMP takes a performance hit over C OpenMP
- Intel Xeon Phi (Knights Landing) achieved the best memory bandwidth overall

Call to action!

GPU-STREAM is Open Source and available on GitHub at the link below, or via the QR code. The webpage maintains a repository of all our results.

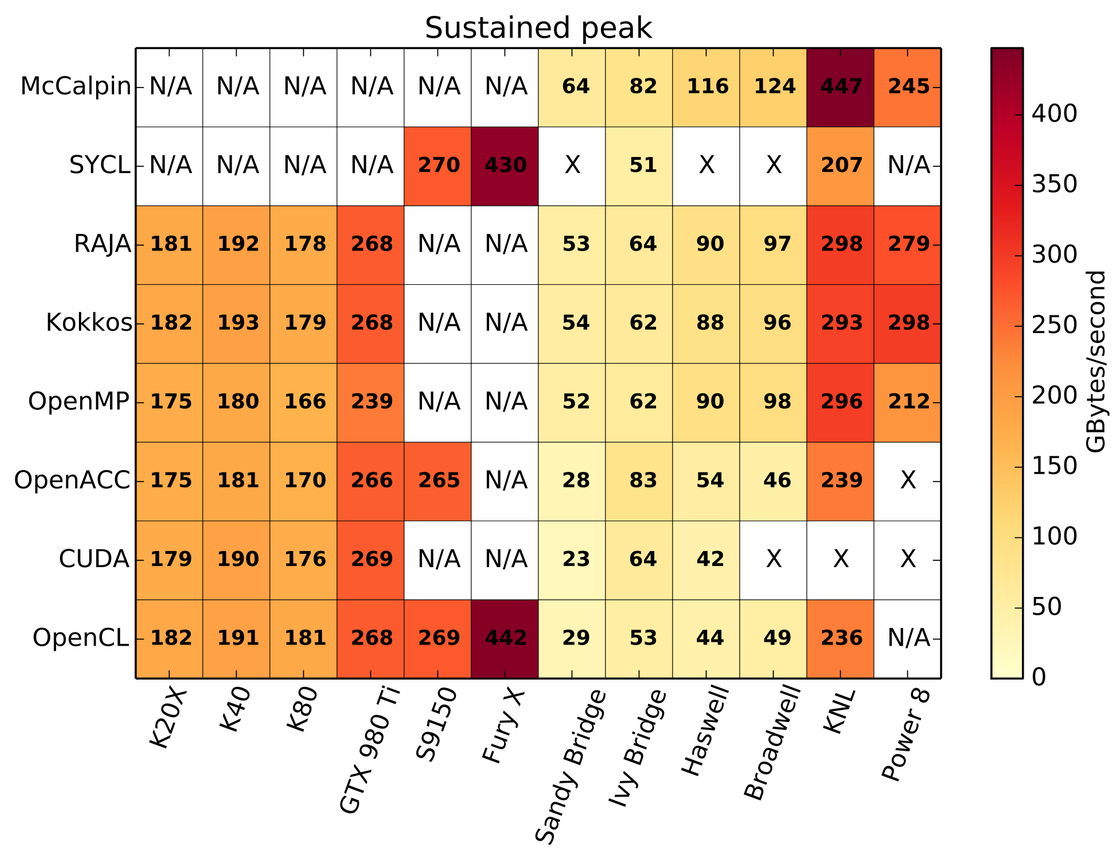
- Submit your results using GPU-STREAM to the website
- Contribute implementations in your favourite/new programming model
- Compilers for programming models

Percentage of peak performance



- C++ OpenMP takes a performace hit (up to 30% less)
  - Some of this is due to not knowing the array size at compile time
- RAJA and Kokkos both show performance similar to the close-to-the-metal CUDA and OpenCL
- OpenMP and OpenACC both show good NVIDIA GPU performance
- OpenACC CPU performance lacking compared to OpenMP

Raw performance



- KNL with MCDRAM demonstrates highest achievable memory bandwidth
  - AMD Fury X with HBM a close second
- GPUs still offer improvements over CPU memory bandwidth
  - IBM Power 8 CPU offers more bandwidth than any of NVIDIA's GPUs that we tested

References

[1] Tom Deakin and Simon McIntosh-Smith.
GPU-STREAM: Benchmarking the achievable memory bandwidth of Graphics Processing Units (poster).
In *Supercomputing*, Austin, Texas, 2015.

[2] Tom Deakin, James Price, Matt Martineau, and Simon McIntosh-Smith.
GPU-STREAM v2.0: Benchmarking the achievable memory bandwidth of many-core processors across diverse parallel programming models.
In *P^3MA workshop at International Super Computing*, 2016.

[3] John D McCalpin.
Memory Bandwidth and Machine Balance in Current High Performance Computers.
IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter, pages 19–25, dec 1995.