SAMPLE

KernelLaunch

1 Overview

1.1 Location

`$(AMDAPPSDKSAMPLESROOT)\samples\opencl\cl\app`

1.2 How to Run

See the *Getting Started* guide for how to build samples. You first must compile the sample.

Use the command line to change to the directory where the executable is located. The pre-compiled sample executable is at `$(AMDAPPSDKSAMPLESROOT)\samples\opencl\bin\x86\` for 32-bit builds, and `$(AMDAPPSDKSAMPLESROOT)\samples\opencl\bin\x86_64\` for 64-bit builds.

Type the following command(s).

1. `KernelLaunch`
   This runs the program with the default options:
   `-d 0 -nl 1000 -nkb 1 -nr -s 2 -nk 1 -nb 1024 -nw 7 -if 0`
2. `KernelLaunch -h`
   This prints the help file.

1.3 Command Line Options

Table 1 lists, and briefly describes, the command line options.

Table 1 Command Line Options

Short Form	Description
`-h`	Shows all command options and their respective meaning.
`-d <n>`	Number of GPU devices.
`-nl <n>`	Number of timing loops.
`-nr <n>`	Repeat each timing <n> times.
`-nk <n>`	Number of loops in kernel.
`-nkb <n>`	Number of kernel launches per batch.
`-nb <n>`	Buffer size in bytes.
`-nw <n>`	Number of wavefronts per SIMD. Default = 7.
`-l`	Print complete timing log.
`-s <n>`	Skip first <n> timings for average. Default = 1.
`-if <n>`	Number of input flags.

2 Introduction

This sample measures the execution time of a minimal kernel launch.

In the default setup, the code measures a single synchronous call to `clEnqueueNDRangeKernel` 1000 times, then calculates the average over all runs.

The `-nkb` argument allows launching multiple calls to the `clEnqueueNDRange` kernel into a batch in order to compute the asynchronous launch time, which typically is far lower than executing the kernels synchronously.

3 Implementation Details

In order to verify a complete data round trip, a small buffer is copied to the device, and its content is verified by a device read kernel. The result is then read back to the host.

The read kernel is launched with a minimal number of threads (1 wavefront). To simulate longer running kernels, the `-nk` option controls the number of read loops inside the kernel.

4 Notes and Caveats

- All transfer steps are executed synchronously to ensure accurate bandwidth measurement. It is recommended that the application code not follow this model, but submit as many commands to a CL queue as possible before forcing the queue to drain.
- Do not run graphics applications while benchmarking compute or transfer operations.
- An argument of `-nk 0` skips the read step and gives the lowest possible time between kernel launch and completion; however, data verification is not done (and is reported as failed). Also, an optimized runtime or compiler might decide not to run the resulting no-op kernel at all, which might render the time measurement inaccurate.
- The `-l` option can be used to identify some of the one-time costs that exist for a given transfer chain. For instance, during the first one or two iterations, the GPU and CPU achieve maximum clock rates. Also, buffers are allocated and transported to their final location. These costs show up as increased execution times for the first few OpenCL calls.
- The read and write GPU kernels are written for clarity, and can achieve 85% of hardware peak bandwidth using the optimum number of work-items.
- The implemented data verification is basic.

Contact

Advanced Micro Devices, Inc.
One AMD Place
P.O. Box 3453
Sunnyvale, CA, 94088-3453
Phone: +1.408.749.4000

For AMD Accelerated Parallel Processing:

URL:	**developer.amd.com/appsdk**
Developing:	**developer.amd.com/**
Support:	**developer.amd.com/appsdksupport**
Forum:	**developer.amd.com/openclforum**

The contents of this document are provided in connection with Advanced Micro Devices, Inc. ("AMD") products. AMD makes no representations or warranties with respect to the accuracy or completeness of the contents of this publication and reserves the right to make changes to specifications and product descriptions at any time without notice. The information contained herein may be of a preliminary or advance nature and is subject to change without notice. No license, whether express, implied, arising by estoppel or otherwise, to any intellectual property rights is granted by this publication. Except as set forth in AMD's Standard Terms and Conditions of Sale, AMD assumes no liability whatsoever, and disclaims any express or implied warranty, relating to its products including, but not limited to, the implied warranty of merchantability, fitness for a particular purpose, or infringement of any intellectual property right.

AMD's products are not designed, intended, authorized or warranted for use as components in systems intended for surgical implant into the body, or in other applications intended to support or sustain life, or in any other application in which the failure of AMD's product could create a situation where personal injury, death, or severe property or environmental damage may occur. AMD reserves the right to discontinue or make changes to its products at any time without notice.

Copyright and Trademarks