

1 Overview

1.1 Location `$<APPSDKSamplesInstallPath>\samples\opencl\cl\`

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 `$<APPSDKSamplesInstallPath>\samples\opencl\bin\x86\` for 32-bit builds, and `$<APPSDKSamplesInstallPath>\samples\opencl\bin\x86_64\` for 64-bit builds.

Type the following command(s).

1. `ImageBandwidth`
This runs the program with the default options: `-t 0 -d 0 -nl 10 -nr 1 -nk 100 -nkl 1 -x 1024 -y 1024 -ox 0 -oy 0 -rx 1024 -ry 1024 -s 2 -if 0 -of 1 -cf 2 -p 0`
2. `ImageBandwidth -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
<code>-t <n></code>	Test type: 0 clEnqueue[Map,Unmap]Image 1 clEnqueue[Read,Write]Image 2 clEnqueueCopyImage 3 clEnqueueCopyBufferToImage and clEnqueueCopyImageToBuffer 4 clEnqueue[Read,Write]Image, prepinned
<code>-d <n></code>	Number of GPU devices.
<code>-x <n></code>	Image width in pixels.
<code>-y <n></code>	Image height in pixels.
<code>-ox <n></code>	Subimage origin x in pixels.
<code>-oy <n></code>	Subimage origin y in pixels.
<code>-rx <n></code>	Subimage region x in pixels.
<code>-ry <n></code>	Subimage region y in pixels.
<code>-nl <n></code>	Number of timing loops.
<code>-nr <n></code>	Time each OpenCL command <n> times.
<code>-nk <n></code>	Number of loops in the kernel.
<code>-nkl <n></code>	Number of kernel launches.

Short Form	Description
-l	Print complete timing log.
-s <n>	Skip first <n> timings for average (default = 2).
-[if,of,cf] <n>	Input, output, copy mem object flags (ok to use multiple): <ul style="list-style-type: none"> 0 CL_MEM_READ_ONLY 1 CL_MEM_WRITE_ONLY 2 CL_MEM_READ_WRITE 3 CL_MEM_USE_HOST_PTR 4 CL_MEM_COPY_HOST_PTR 5 CL_MEM_ALLOC_HOST_PTR 6 CL_MEM_USE_PERSISTENT_MEM_AMD
-p <n>	Pixel format. <ul style="list-style-type: none"> 0 CL_RGBA/CL_UNSIGNED_INT32 1 CL_RGBA/CL_UNSIGNED_INT8
-m	Always map as MAP_READ MAP_WRITE.
-db	Disable host memory bandwidth baseline.
-v (or --version)	AMD APP SDK version string.
-h	Print all options and their meanings.

2 Introduction

This sample can run the following tests:

- Create a simple baseline for host memory read and write performance. This can be used to ensure sanity of device image access performance numbers created by the other tests.
- Benchmark a round-trip chain of synchronous, serialized transfer steps between the host and the device.
- The sample can create a log over many iterations to locate one-time effects or variations over time.

The following transfer paths can be tested via command line option:

```
clEnqueueMapImage/UnmapImage
clEnqueueReadImage/WriteImage
clEnqueueCopyImage
clEnqueueCopyBufferToImage
clEnqueueCopyImageToBuffer
clEnqueueReadImage/WriteImage, prepinned
```

This sample allows selection of any of the various CL image creation attributes for the source and destination images of the transfer chain.

3 Implementation Details

Details on the various buffer types and recommended transfer paths are provided in Chapter 4 of the *AMD APP OpenCL Programming Guide*.

Inside the read and write kernels, the code iterates over a given image multiple times (-nk option) to amortize kernel launch costs. A particular access pattern is used to minimize cache hits and provide a closer measure of actual-off chip memory bandwidth. For small images, this access

pattern still produces read cache or write combine hits. A simple way to further minimize cache hits is to run with `-nk 1 -nkl 10` (for example), so that the kernel is launched multiple times, while reading the image only once inside the kernel.

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 1 or 2 iterations, the GPU and CPU achieve maximum clock rates. Also, memory objects are allocated and transported to their final location. These costs show up as increased execution times for the first few OpenCL calls.

All transfer steps are executed synchronously to ensure accurate bandwidth measurement. The application code should not follow this model, but submit as many commands to a CL queue as possible before forcing the queue to drain.

Currently, runtime bug 6432 breaks mapping of `CL_MEM_USE_PERSISTENT_MEM` images using offsets and/or regions. The sample produces data verification failures for this path.

4 Notes and Caveats

- Do not run graphics applications while benchmarking compute or transfer operations.
- The read and write GPU kernels are written for clarity, and should achieve around 85% of HW peak with the right number of threads.
- The data verification used is basic.
- User-specified image sizes, offsets, and region boundaries can be adjusted by the sample code to nearby sizes that work well across all measurement stages.

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

© 2013 Advanced Micro Devices, Inc. All rights reserved. AMD, the AMD Arrow logo, ATI, the ATI logo, Radeon, FireStream, and combinations thereof are trademarks of Advanced Micro Devices, Inc. OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos. Other names are for informational purposes only and may be trademarks of their respective owners.