

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. `SimpleSPIR`
This command runs the program with the default options.
2. `SimpleSPIR -h`
This command 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	Long Form	Description
-h	--help	Shows all command options and their respective meanings.
	--device [cpu gpu]	Devices on which the OpenCL kernel is to be run. Acceptable values are <code>cpu</code> or <code>gpu</code> .
-q	--quiet	Quiet mode. Suppresses all text output.
-e	--verify	Verify results against reference implementation.
-t	--timing	Print timing-related statistics.
-v	--version	AMD APP SDK version string.
	--load [filename]	Load binary SPIR image and execute on device. This option is mandatory for this sample .
	--flags [filename]	Specify the filename containing the compiler flags for building the kernel.
-i	--iterations	Number of iterations.
-p	--platformId	Select the platformId to be used[0 to N-1 where N is number platform s available].
-d	--deviceId	Select deviceId to be used[0 to N-1 where N is number devices available].

2 Introduction

This sample demonstrates the consumption of SPIR Binary by using OpenCL APIs.

SPIR (Standard Portable Intermediate Representation) is a portable, non-source representation for device's programs. Application developers can use SPIR to avoid shipping kernel source and to manage the proliferation of devices and drivers from multiple vendors.

In this sample, a Matrix transpose implementation is used to demonstrate SPIR code consumption. The sample consumes the SPIR binary generated from the Matrix transpose OpenCL kernel.

This document does not dwell much into the Matrix Transpose implementation itself as the purpose is to demonstrate SPIR code consumption. For details about the Matrix transpose implementation, see the Matrix Transpose sample document.

3 Implementation

The implementation for the consumption of SPIR binary is as follows.

The SPIR binary, `MatrixTranspose_Kernels.fe.bc`, is loaded by using the `clCreateProgramWithBinary` OpenCL API to generate OpenCL Program.

The API is declared as follows:

```
cl_program clCreateProgramWithBinary(cl_context context, cl_uint
num_devices, const cl_device_id *device_list, const size_t *lengths,
const unsigned char **binaries, cl_int *binary_status, cl_int
*errcode_ret);
```

The array of SPIR Binaries for various platforms are given under the "binaries" argument of the `clCreateProgramWithBinary` API.

The generated OpenCL Program is built using the `clBuildProgram` OpenCL API with the following options provided:

```
cl_int clBuildProgram (cl_program program, cl_uint num_devices, const
cl_device_id *device_list, const char *options, (CL_CALLBACK
*PFN_notify)(cl_program program, void *user_data), void *user_data)
```

```
options = " -x spir -spir-std=1.2"
```

```
-x spir
```

The last line indicates that the binary used is in SPIR format.

```
-spir-std=1.2
```

This option tells the API that the SPIR binary is generated using the SPIR 1.2 specification. The number corresponding to the version of SPIR specification used must be provided.

In this sample, the SPIR related options are passed through the `SimpleSpir_oclflags.txt` file.

4 References

1. <https://www.khronos.org/spir>
2. <https://www.khronos.org/registry/cl/sdk/1.0/docs/man/xhtml/clCreateProgramWithBinary.html>
3. <https://www.khronos.org/registry/cl/sdk/1.0/docs/man/xhtml/clBuildProgram.html>
4. <http://en.wikipedia.org/wiki/Transpose>

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

© 2014 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.