AMD **Accelerated**
Parallel Processing
TECHNOLOGY

SAMPLE

OpenCL C++ Wrapper

1 Overview

1.1 Location

`$<APPSDKSamplesInstallPath>\samples\opencl\cpp_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 default executables are placed in `$<APPSDKSamplesInstallPath>\samples\opencl\bin\x86\` for 32-bit builds, and `$<APPSDKSamplesInstallPath>\samples\opencl\bin\x86_64\` for 64-bit builds.

2 Introduction

This is an introductory sample to show how to program using the OpenCL C++ Wrapper API.

3 Implementation Details

To use the OpenCL C++ Wrapper API, the source file must include the `cl.hpp` header file. The sample program starts by compiling the OpenCL source file. Note that there is no code to initialize the device and to create an OpenCL context in this sample. This is because a default device has been chosen and initialized behind the scene. Optionally, developers who want greater control over the device selection can achieve that using the C++ API.

One of the features introduced by the C++ API is the C++ exception mechanism. Instead of returning an error code, the API functions throw an exception in case of an error. To enable C++ exceptions for the C++ wrapper API, add a `#define __CL_ENABLE_EXCEPTIONS` before including the `cl.hpp` header file:

```
#define __CL_ENABLE_EXCEPTIONS
#include<CL/cl.hpp>
...
try
{
vectorAddProgram.build("");
}catch(cl::Errore){
    std::cout<<e.what()<<std::endl;
}
```

Another feature in the C++ wrapper API is that the kernel can be created as a functor, and launching a kernel looks similar to calling a normal C++ function:

```
typedefcl::make_kernel<cl::Buffer&, cl::Buffer&, cl::Buffer&>KernelType;
//create kernel as a functor
KernelTypevectorAddKernel(vectorAddProgram,"vectorAdd");
...
//execute the kernel by calling the kernel functor
e=vectorAddKernel(arg, outputBuffer, inputABuffer, inputBBuffer);
```

There is no clean-up code in the sample to release OpenCL-related resources; they are implicitly handled by the destructor of the OpenCL objects.

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/**
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