AMD **Accelerated** Parallel Processing TECHNOLOGY

SAMPLE

Matrix Multiplication Using Image

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 default executables are placed in `$(AMDAPPSDKSAMPLESROOT)\samples\opencl\bin\x86` for 32-bit builds and `$(AMDAPPSDKSAMPLESROOT)\samples\opencl\bin\x86_64\` for 64-bit builds.

Type the following command(s).

1. `MatrixMulImage`
   Implements MatrixMultiplication using images.
2. `MatrixMulImage -h`
   This prints the help message.

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 meaning.
	`--device`	Devices on which the program is to be run. Acceptable values are `cpu` or `gpu`.
`-q`	`--quiet`	Quiet mode. Suppresses all text output.
`-e`	`--verify`	Verify results against reference implementation.
`-t`	`--timing`	Print timing.
	`--dump`	Dump binary image for all devices.
	`--load`	Load binary image and execute on device.
	`--flags`	Specify compiler flags to build the kernel.
`-p`	`--platformId`	Select platformId to be used (0 to N-1, where N is the number of available platforms).
`-d`	`--deviceId`	Select deviceId to be used (0 to N-1, where N is the number of available devices).
`-v`	`--version`	AMD APP SDK version string.
`-x`	`--height0`	Height of matrix A.
`-y`	`--width0`	Width of matrix A and height of matrix B.
`-z`	`--width1`	Width of matrix B.
`-i`	`--iterations`	Number of iterations for kernel execution.

2 Implementation Details

This sample computes the following relation among matrices:

C = AB

Dimensions of matrix A = (y, x) {width0, height0}.

Dimensions of matrix B = (z, y) {width1, width0}.

This results in a matrix C, with dimension = {z, x}.

Matrices A, B, and C are allocated using image buffers.

For (width A / 4) iterations of the loop: Each thread reads four float4s from matrix A, then the corresponding four float4s from matrix B. Each thread then calculates the partial matrix multiplication and updates the partial sum. Thus, each thread computes four float4s (16 floating values of matrix C). See Figure 1.

The number of global threads = {widthC / 4, heightC / 4}.

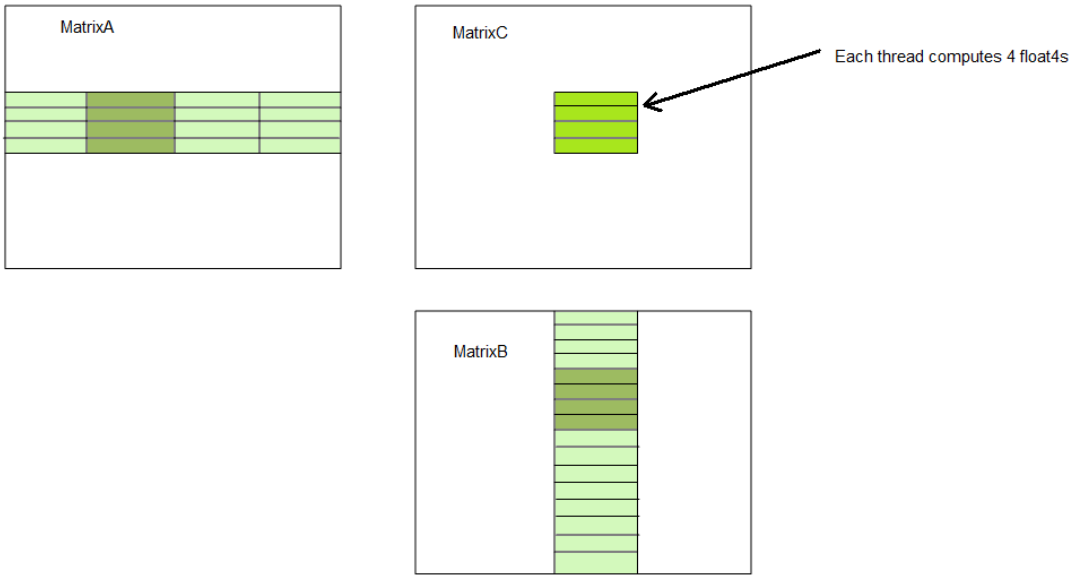

Figure 1 Matrix Multiplication

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