

AMD OpenCL Programming Guide August 2013

Table of Contents

Chapter 1 OpenCL Architecture and AMD Accelerated Parallel Processing

- 1.1 Software Overview
 - 1.1.1 Synchronization
- 1.2 Hardware Overview for Southern Islands Devices
- 1.3 Hardware Overview for Evergreen and Northern Islands Devices
- 1.4 The AMD Accelerated Parallel Processing Implementation of OpenCL
 - 1.4.1 Work-Item Processing
 - 1.4.2 Flow Control
 - 1.4.3 Work-Item Creation
- 1.5 Memory Architecture and Access
 - 1.5.1 Memory Access
 - 1.5.2 Global Buffer
 - 1.5.3 Image Read/Write
 - 1.5.4 Memory Load/Store
- 1.6 Communication Between Host and
 - 1.6.1 PCI Express Bus
 - 1.6.2 Processing API Calls: The Command Processor
 - 1.6.3 DMA Transfers
 - 1.6.4 Masking Visible Devices
- 1.7 GPU Compute Device Scheduling
- 1.8 Terminology
 - 1.8.1 Compute Kernel
 - 1.8.2 Wavefronts and Work-groups
 - 1.8.3 Local Data Store (LDS)
- 1.9 Programming Model
- 1.10 Example Programs
 - 1.10.1 First Example: Simple Buffer Write
 - 1.10.2 Example: Parallel Min() Function .1-23

Chapter 2 Building and Running OpenCL Programs

- 2.1 Compiling the Program
 - 2.1.1 Compiling on Windows
 - 2.1.2 Compiling on Linux
 - 2.1.3 Supported Standard OpenCL Compiler Options
 - 2.1.4 AMD-Developed Supplemental Compiler Options
- 2.2 Running the Program

- 2.2.1 Running Code on Windows
- 2.2.2 Running Code on Linux
- 2.3 Calling Conventions

Chapter 3 Debugging OpenCL

- 3.1 AMD CodeXL GPU Debugger .
- 3.2 Debugging CPU Kernels with GDB
 - 3.2.1 Setting the Environment .3-
 - 3.2.2 Setting the Breakpoint in an OpenCL Kernel.3-2
 - 3.2.3 Sample GDB Session
 - 3.2.4 Notes

Chapter 4 OpenCL Performance and Optimization

- 4.1 CodeXL GPU Profiler
 - 4.1.1 Collecting OpenCL Application Traces
 - 4.1.2 Timeline View.
 - 4.1.3 Summary Pages View
 - 4.1.4 API Trace View
 - 4.1.5 Collecting OpenCL GPU Kernel Performance Counters
- 4.2 AMD APP KernelAnalyzer2
 - 4.2.1 Start KernelAnalyzer2
 - 4.2.2 Open Kernel Source
 - 4.2.3 Build Options – Choosing Target ASICS
 - 4.2.4 Build Options – Defining Kernel Compilation Options
 - 4.2.5 Analysis Input Tab
 - 4.2.6 Build the Kernel
 - 4.2.7 Build Statistics Tab
 - 4.2.8 The Analysis Tab
- 4.3 Analyzing Processor Kernels
 - 4.3.1 Intermediate Language and GPU Disassembly
 - 4.3.2 Generating IL and ISA Code.
- 4.4 Estimating Performance
 - 4.4.1 Measuring Execution Time
 - 4.4.2 Using the OpenCL timer with Other System Timers
 - 4.4.3 Estimating Memory Bandwidth
- 4.5 OpenCL Memory Objects
 - 4.5.1 Types of Memory Used by the Runtime
 - 4.5.2 Placement
 - 4.5.3 Memory Allocation
 - 4.5.4 Mapping
 - 4.5.5 Reading, Writing, and Copying
 - 4.5.6 Command Queue
- 4.6 OpenCL Data Transfer Optimization
 - 4.6.1 Definitions
 - 4.6.2 Buffers
- 4.7 Using Multiple OpenCL Devices
 - 4.7.1 CPU and GPU Devices

- 4.7.2 When to Use Multiple Devices
- 4.7.3 Partitioning Work for Multiple Devices
- 4.7.4 Synchronization Caveats
- 4.7.5 GPU and CPU Kernels
- 4.7.6 Contexts and Devices

Chapter 5 OpenCL Performance and Optimization for GCN Devices

- 5.1 Global Memory Optimization
 - 5.1.1 Channel Conflicts.
 - 5.1.2 Coalesced Writes
 - 5.1.3 Hardware Variations.
- 5.2 Local Memory (LDS) Optimization
- 5.3 Constant Memory Optimization
- 5.4 OpenCL Memory Resources: Capacity and Performance
- 5.5 Using LDS or L1 Cache
- 5.6 NDRange and Execution Range Optimization
 - 5.6.1 Hiding ALU and Memory Latency
 - 5.6.2 Resource Limits on Active Wavefronts
 - 5.6.3 Partitioning the Work
 - 5.6.4 Summary of NDRange Optimizations
- 5.7 Instruction Selection Optimizations
 - 5.7.1 Instruction Bandwidths
 - 5.7.2 AMD Media Instructions
 - 5.7.3 Math Libraries
 - 5.7.4 Compiler Optimizations
- 5.8 Additional Performance Guidance
 - 5.8.1 Loop Unroll pragma
 - 5.8.2 Memory Tiling
 - 5.8.3 General Tips
 - 5.8.4 Guidance for CUDA Programmers Using OpenCL
 - 5.8.5 Guidance for CPU Programmers Using OpenCL to Program GPUs
 - 5.8.6 Optimizing Kernel Code
 - 5.8.7 Optimizing Kernels for Southern Island GPUs
- 5.9 Specific Guidelines for Southern Islands GPUs

Chapter 6 OpenCL Performance and Optimization for Evergreen and Northern Islands Devices

- 6.1 Global Memory Optimization
 - 6.1.1 Two Memory Paths
 - 6.1.2 Channel Conflicts.
 - 6.1.3 Float4 Or Float1
 - 6.1.4 Coalesced Writes
 - 6.1.5 Alignment
 - 6.1.6 Summary of Copy Performance

- 6.1.7 Hardware Variations
- 6.2 Local Memory (LDS) Optimization
- 6.3 Constant Memory Optimization
- 6.4 OpenCL Memory Resources: Capacity and Performance
- 6.5 Using LDS or L1 Cache
- 6.6 NDRange and Execution Range Optimization
 - 6.6.1 Hiding ALU and Memory Latency
 - 6.6.2 Resource Limits on Active Wavefronts
 - 6.6.3 Partitioning the Work
 - 6.6.4 Optimizing for Cedar
 - 6.6.5 Summary of NDRange Optimizations
- 6.7 Using Multiple OpenCL Devices
 - 6.7.1 CPU and GPU Devices
 - 6.7.2 When to Use Multiple Devices
 - 6.7.3 Partitioning Work for Multiple Devices
 - 6.7.4 Synchronization Caveats
 - 6.7.5 GPU and CPU Kernels
 - 6.7.6 Contexts and Devices
- 6.8 Instruction Selection Optimizations
 - 6.8.1 Instruction Bandwidths
 - 6.8.2 AMD Media Instructions
 - 6.8.3 Math Libraries
 - 6.8.4 VLIW and SSE Packing
 - 6.8.5 Compiler Optimizations
- 6.9 Clause Boundaries
- 6.10 Additional Performance Guidance
 - 6.10.1 Loop Unroll pragma
 - 6.10.2 Memory Tiling
 - 6.10.3 General Tips.
 - 6.10.4 Guidance for CUDA Programmers Using OpenCL
 - 6.10.5 Guidance for CPU Programmers Using OpenCL to Program GPUs
 - 6.10.6 Optimizing Kernel Code
 - 6.10.7 Optimizing Kernels for Evergreen and 69XX-Series GPUs

Chapter 7 OpenCL Static C++ Programming Language

- 7.1 Overview
 - 7.1.1 Supported Features
 - 7.1.2 Unsupported Features
 - 7.1.3 Relations with ISO/IEC C++
- 7.2 Additions and Changes to Section 5 – The OpenCL C Runtime
 - 7.2.1 Additions and Changes to Section 5.
- 7.1 – Creating Kernel Objects
 - 7.2.2 Passing Classes between Host and Device
- 7.3 Additions and Changes to Section The OpenCL C Programming Language
 - 7.3.1 Building C++ Kernels
 - 7.3.2 Classes and Derived Classes

- 7.3.3 Namespaces
- 7.3.4 Overloading
- 7.3.5 Templates
- 7.3.6 Exceptions
- 7.3.7 Libraries
- 7.3.8 Dynamic Operation
- 7.4 Examples
 - 7.4.1 Passing a Class from the Host to the Device and Back
 - 7.4.2 Kernel Overloading
 - 7.4.3 Kernel Template

Appendix A OpenCL Optional Extensions

- A.1 Extension Name Convention
- A.2 Querying Extensions for a Platform
- A.3 Querying Extensions for a Device
- A.4 Using Extensions in Kernel Programs
- A.5 Getting Extension Function Pointers
- A.6 List of Supported Extensions that are Khronos-Approved
- A.7 cl_ext Extensions
- A.8 AMD Vendor-Specific Extensions
 - A.8.1 cl_amd_fp64
 - A.8.2 cl_amd_vec3
 - A.8.3 cl_amd_device_persistent_memory
 - A.8.4 cl_amd_device_attribute_query
 - A.8.5 cl_amd_compile_options
 - A.8.6 cl_amd_offline_devices
 - A.8.7 cl_amd_event_callback
 - A.8.8 cl_amd_popcnt
 - A.8.9 cl_amd_media_ops
 - A.8.10 cl_amd_media_ops2
 - A.8.11 cl_amd_printf
 - A.8.12 cl_amd_predefined_macros
 - A.8.13 cl_amd_bus_addressable_memory
- A.9 Supported Functions for cl_amd_fp64/cl_khr_fp64
- A.10 Extension Support by Device

Appendix B The OpenCL Installable Client Driver (ICD)

- B.1 Overview
- B.2 Using ICD

Appendix C Compute Kernel

- C.1 Differences from a Pixel Shader
- C.2 Indexing

C.3 Performance Comparison

C.4 Pixel Shader

C.5 Compute Kernel

C.6 LDS Matrix Transpose

C.7 Results Comparison

Appendix D Device Parameters

Appendix E OpenCL Binary Image Format (BIF) v2.0

E.1 Overview

E.1.1 Executable and Linkable Format (ELF) Header

E.1.2 Bitness

E.2 BIF Options

Appendix F Open Decode API Tutorial

F.1 Overview

F.2 Initializing

F.3 Creating the Context

F.4 Creating the Session

F.5 Decoding

F.6 Destroying Session and Context

Appendix G OpenCL-OpenGL Interoperability

G.1 Under Windows

G.1.1 Single GPU Environment

G.1.2 Multi-GPU Environment

G.1.3 Limitations

G.2 Linux Operating System

G.2.1 Single GPU Environment

G.2.2 Multi-GPU Configuration

G.3 Additional GL Formats Supported