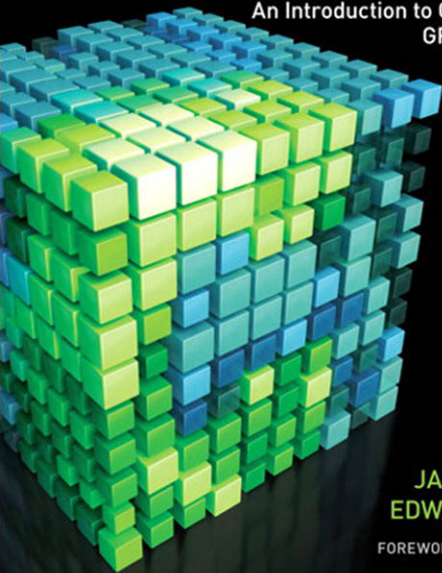




# CUDA

## BY EXAMPLE

An Introduction to General-Purpose  
GPU Programming



JASON SANDERS  
EDWARD KANDROT

FOREWORD BY JACK DONGARRA

# CUDA by Example

# **CUDA by Example: An Introduction to General-Purpose GPU Programming, Portable Documents**

## **Table of Contents**

Contents

Foreword

Preface

Acknowledgments

About the Authors

### **1 WHY CUDA? WHY NOW?**

1.1 Chapter Objectives

1.2 The Age of Parallel Processing

1.2.1 Central Processing Units

1.3 The Rise of GPU Computing

1.3.1 A Brief History of GPUs

1.3.2 Early GPU Computing

1.4 CUDA

1.4.1 What Is the CUDA Architecture?

1.4.2 Using the CUDA Architecture

1.5 Applications of CUDA

1.5.1 Medical Imaging

1.5.2 Computational Fluid Dynamics

1.5.3 Environmental Science

1.6 Chapter Review

### **2 GETTING STARTED**

# **Table of Contents**

## 2.1 Chapter Objectives

## 2.2 Development Environment

### 2.2.1 CUDA-Enabled Graphics Processors

### 2.2.2 NVIDIA Device Driver

### 2.2.3 CUDA Development Toolkit

### 2.2.4 Standard C Compiler

## 2.3 Chapter Review

# 3 INTRODUCTION TO CUDA C

## 3.1 Chapter Objectives

## 3.2 A First Program

### 3.2.1 Hello, World!

### 3.2.2 A Kernel Call

### 3.2.3 Passing Parameters

## 3.3 Querying Devices

## 3.4 Using Device Properties

## 3.5 Chapter Review

# 4 PARALLEL PROGRAMMING IN CUDA C

## 4.1 Chapter Objectives

## 4.2 CUDA Parallel Programming

### 4.2.1 Summing Vectors

### 4.2.2 A Fun Example

## 4.3 Chapter Review

# 5 THREAD COOPERATION

## 5.1 Chapter Objectives

## 5.2 Splitting Parallel Blocks

### 5.2.1 Vector Sums: Redux

### 5.2.2 GPU Ripple Using Threads

## 5.3 Shared Memory and Synchronization

# **Table of Contents**

5.3.1 Dot Product

5.3.1 Dot Product Optimized (Incorrectly)

5.3.2 Shared Memory Bitmap

5.4 Chapter Review

## **6 CONSTANT MEMORY AND EVENTS**

6.1 Chapter Objectives

6.2 Constant Memory

6.2.1 Ray Tracing Introduction

6.2.2 Ray Tracing on the GPU

6.2.3 Ray Tracing with Constant Memory

6.2.4 Performance with Constant Memory

6.3 Measuring Performance with Events

6.3.1 Measuring Ray Tracer Performance

6.4 Chapter Review

## **7 TEXTURE MEMORY**

7.1 Chapter Objectives

7.2 Texture Memory Overview

7.3 simulating Heat transfer

7.3.1 simple Heating Model

7.3.2 Computing temperature Updates

7.3.3 Animating the simulation

7.3.4 Using texture Memory

7.3.5 Using two-Dimensional texture Memory

7.4 Chapter Review

## **8 GRAPHICS INTEROPERABILITY**

8.1 Chapter objectives

8.2 Graphics Interoperation

8.3 GPU Ripple with Graphics Interoperability

# **Table of Contents**

8.3.1 the GPUAnimBitmap structure

8.3.2 GPU Ripple Redux

8.4 Heat transfer with Graphics Interop

8.5 DirectX Interoperability

8.6 Chapter Review

## **9 ATOMICS**

9.1 Chapter objectives

9.2 Compute Capability

9.2.1 the Compute Capability of NVIDIA GPUs

9.2.2 Compiling for a Minimum Compute Capability

9.3 Atomic operations overview

9.4 Computing Histograms

9.4.1 CPU Histogram Computation

9.4.2 GPU Histogram Computation

9.5 Chapter Review

## **10 STREAMS**

10.1 Chapter Objectives

10.2 Page-Locked Host Memory

10.3 CUDA Streams

10.4 Using a Single CUDA Stream

10.5 Using Multiple CUDA Streams

10.6 GPU Work Scheduling

10.7 Using Multiple CUDA Streams Effectively

10.8 Chapter Review

## **11 CUDA C ON MULTIPLE GPUS**

11.1 Chapter Objectives

11.2 Zero-Copy Host Memory

# **Table of Contents**

11.2.1 Zero-Copy Dot Product

11.2.2 Zero-Copy Performance

11.3 Using Multiple GPUs

11.4 Portable Pinned Memory

11.5 Chapter Review

## **12 THE FINAL COUNTDOWN**

12.1 Chapter Objectives

12.2 CUDA Tools

12.2.1 CUDA Toolkit

12.2.2 CUFFT

12.2.3 CUBLAS

12.2.4 NVIDIA GPU Computing SDK

12.2.5 NVIDIA Performance Primitives

12.2.6 Debugging CUDA C

12.2.7 CUDA Visual Profiler

12.3 Written Resources

12.3.1 Programming Massively Parallel Processors: A Hands-on Approach

12.3.2 CUDA U

12.3.3 NVIDIA Forums

12.4 Code Resources

12.4.1 CUDA Data Parallel Primitives Library

12.4.2 CULAtools

12.4.3 Language Wrappers

12.5 Chapter Review

## **A: ADVANCED ATOMICS**

A.1 Dot Product Revisited

A.1.1 Atomic Locks

A.1.2 Dot Product Redux: Atomic Locks

# **Table of Contents**

## **A.2 Implementing a Hash table**

A.2.1 Hash table overview

A.2.2 A CPU Hash table

A.2.3 Multithreaded Hash table

A.2.4 A GPU Hash table

A.2.5 Hash table Performance

## **A.3 Appendix Review**

**Index**