

## <<大规模并行处理器程序设计>>

### 图书基本信息

书名：<<大规模并行处理器程序设计>>

13位ISBN编号：9787302229735

10位ISBN编号：7302229732

出版时间：2010-7

出版时间：清华大学出版社

作者：（美）柯克 等著

页数：258

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

## <<大规模并行处理器程序设计>>

### 前言

Mass-market computing systems that combine multicore CPUs and many-core GPUs have brought terascale computing to the laptop and petascale computing to clusters. Armed with such computing power, we are at the dawn of pervasive use of computational experiments for science, engineering, health, and business disciplines. Many will be able to achieve break-throughs in their disciplines using computational experiments that are of unprecedented level of scale, controllability, and observability. This book provides a critical ingredient for the vision: teaching parallel programming to millions of graduate and undergraduate students so that computational thinking and parallel programming skills will be as pervasive as calculus. We started with a course now known as ECE498AL. During the Christmas holiday of 2006, we were frantically working on the lecture slides and lab assignments. David was working the system trying to pull the early GeForce 8800 GTX GPU cards from customer shipments to Illinois, which would not succeed until a few weeks after the semester began. It also became clear that CUDA would not become public until a few weeks after the start of the semester. We had to work out the legal agreements so that we can offer the course to students under NDA for the first few weeks. We also needed to get the words out so that students would sign up since the course was not announced until after the pre-enrollment period.

## <<大规模并行处理器程序设计>>

### 内容概要

本书介绍了并行程序设计与GPU体系结构的基本概念，并详细探讨了用于构建并行程序的各种技术，用案例演示了并行程序设计的整个开发过程，即从并行计算的思想开始，直到最终实现实际且高效的并行程序。

本书特点 介绍了并行计算的思想，使得读者可以把这种问题的思考方式渗透到高性能并行计算中去。

介绍了CUDA的使用，CUDA是NVIDIA公司专门为大规模并行环境创建的一种软件开发工具。

介绍如何使用CUDA编程模式和OpenCL来获得高性能和高可靠性。

<<大规模并行处理器程序设计>>

作者简介

作者：（美国）柯克（David B.Kirk）（美国）Wen-mei W.Hwu

# <<大规模并行处理器程序设计>>

## 书籍目录

Preface Acknowledgments Dedication

CHAPTER 1 INTRODUCTION 1.1 GPUs as Parallel Computers 1.2 Architecture of a Modern GPU 1.3 Why More Speed or Parallelism? 1.4 Parallel Programming Languages and Models 1.5 Overarching Goals 1.6 Organization of the Book

CHAPTER 2 HISTORY OF GPU COMPUTING 2.1 Evolution of Graphics pipelines 2.1.1 The Era of Fixed-Function Graphics Pipelines 2.1.2 Evolution of Programmable Real-Time Graphics 2.1.3 Unified Graphics and Computing Processors 2.1.4 GPGPU: An Intermediate Step 2.2 GPU Computing 2.2.1 Scalable GPUs 2.2.2 Recent Developments 2.3 Future Trends

CHAPTER 3 INTRODUCTION TO CUDA 3.1 Data Parallelism 3.2 CUDA Program Structure 3.3 A Matrix-Matrix Multiplication Example 3.4 Device Memories and Data Transfer 3.5 Kernel Functions and Threading 3.6 Summary 3.6.1 Function declarations 3.6.2 Kernel launch 3.6.3 Predefined variables 3.6.4 Runtime AP

CHAPTER 4 CUDA THREADS 4.1 CUDA Thread Organization 4.2 Using blockIdx and threadIdx 4.3 Synchronization and Transparent Scalability 4.4 Thread Assignment 4.5 Thread Scheduling and Latency Tolerance 4.6 Summary 4.7 Exercises

CHAPTER 5 CUDATM MEMORIES 5.1 Importance of Memory Access Efficiency 5.2 CUDA Device Memory Types 5.3 A Strategy for Reducing Global Memory Traffic 5.4 Memory as a Limiting Factor to Parallelism 5.5 Summary 5.6 Exercises

CHAPTER 6 PERFORMANCE CONSIDERATIONS 6.1 More on Thread Execution 6.2 Global Memory Bandwidth 6.3 Dynamic Partitioning of SM Resources 6.4 Data Prefetching 6.5 Instruction Mix 6.6 Thread Granularity 6.7 Measured Performance and Summary 6.8 Exercises

CHAPTER 7 FLOATING POINT CONSIDERATIONS 7.1 Floating-Point Format 7.1.1 Normalized Representation of M 7.1.2 Excess Encoding of E 7.2 Representable Numbers 7.3 Special Bit Patterns and Precision 7.4 Arithmetic Accuracy and Rounding 7.5 Algorithm Considerations 7.6 Summary 7.7 Exercises

CHAPTER 8 APPLICATION CASE STUDY: ADVANCED MRI RECONSTRUCTION 8.1 Application Background 8.2 Iterative Reconstruction 8.3 Computing FHD Step 1. Determine the Kernel Parallelism Structure Step 2. Getting Around the Memory Bandwidth Limitation. Step 3. Using Hardware Trigonometry Functions Step 4. Experimental Performance Tuning 8.4 Final Evaluation 8.5 Exercises

CHAPTER 9 APPLICATION CASE STUDY: MOLECULAR VISUALIZATION AND ANALYSIS

CHAPTER 10 PARALLEL PROGRAMMING AND COMPUTATIONAL THINKING

CHAPTER 11 A BRIEF INTRODUCTION TO OPENCL

CHAPTER 12 CONCLUSION AND FUTURE OUTLOOK

APPENDIX A MATRIX MULTIPLICATION HOST-ONLY VERSION SOURCE CODE

APPENDIX B GPU COMPUTE CAPABILITIES

Index

## 章节摘录

插图：The raster operation (ROP) stage in Figure 2.2 performs the final raster operations on the pixels. It performs color raster operations that blend the color of overlapping/adjacent objects for transparency and antialiasing effects. It also determines the visible objects for a given viewpoint and discards the occluded pixels. A pixel becomes occluded when it is blocked by pixels from other objects according to the given view point. Figure 2.3 illustrates antialiasing, one of the ROP stage operations. Notice the three adjacent triangles with a black background. In the aliased output, each pixel assumes the color of one of the objects or the background. The limited resolution makes the edges look crooked and the shapes of the objects distorted. The problem is that many pixels are partly in one object and partly in another object or the background. Forcing these pixels to assume the color of one of the objects introduces distortion into the edges of the objects. The antialiasing operation gives each pixel a color that is blended, or linearly combined, from the colors of all the objects and background that partially overlap the pixel. The contribution of each object to the color of the pixel is the amount of the pixel that the object overlaps. Finally, the frame buffer interface (FBI) stage in Figure 2.1 manages memory reads from and writes to the display frame buffer memory.

<<大规模并行处理器程序设计>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>