

MANNING
涵盖 OpenCL v. 1.1

OpenCL 实战

OpenCL
IN ACTION

[美] Matthew Scarpino 著
陈睿 译



人民邮电出版社
POSTS & TELECOM PRESS

OpenCL 实战

OpenCL
IN ACTION

[美] Matthew Scarpino 著
陈睿 译

人民邮电出版社
北京

图书在版编目 (C I P) 数据

OpenCL实战 / (美) 斯卡皮诺 (Scarpino, M.) 著 ;
陈睿译. — 北京 : 人民邮电出版社, 2014. 7
ISBN 978-7-115-34734-3

I. ①O… II. ①斯… ②陈… III. ①图形软件—程序
设计 IV. ①TP391.41

中国版本图书馆CIP数据核字(2014)第076178号

版 权 声 明

Simplified Chinese-language edition copyright ©2014 by Posts & Telecom Press. All rights reserved.

Original English language edition, entitled OpenCL in Action, by Matthew Scarpino, published by Manning Publications Co., 209 Bruce Park Avenue, Greenwich, CT 06830. Copyright © 2012 by Manning Publications Co.

本书中文简体字版由 Manning Publications Co.授权人民邮电出版社独家出版。未经出版者书面许可, 不得以任何方式复制或抄袭本书内容。

版权所有, 侵权必究。

◆ 著 [美] Matthew Scarpino

译 陈 睿

责任编辑 陈冀康

责任印制 彭志环 焦志炜

◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路11号

邮编 100164 电子邮件 315@ptpress.com.cn

网址 <http://www.ptpress.com.cn>

北京艺辉印刷有限公司印刷

◆ 开本: 800×1000 1/16

印张: 26.5

字数: 571 千字

2014年7月第1版

印数: 1-2 500 册

2014年7月北京第1次印刷

著作权合同登记号 图字: 01-2012-4604 号

定价: 89.00 元

读者服务热线: (010) 81055410 印装质量热线: (010) 81055316

反盗版热线: (010) 81055315



内容提要

OpenCL (Open Computing Language) 是第一个面向异构系统通用目的并行编程的开放式、免费标准，也是一个统一的编程环境。OpenCL 当前已经广泛应用于各种并行环境和不同行业领域。

本书是 OpenCL 编程的实践指南，旨在介绍如何在实际的应用程序中使用 OpenGL 解决问题。全书共 16 章和 4 个附录。正文可以分为 3 个部分。第一部分是第 1~10 章，主要讨论的是 OpenCL 语言及其各项功能。第二部分是第 11~14 章，展示如何用 OpenCL 来处理高性能计算领域经常会碰到的大规模任务。最后一部分，包括 15 章和 16 章，展示如何用 OpenCL 来加速 OpenGL 应用程序的执行。附录 A 介绍了 SDK 并演示如何安装 AMD 和 Nvidia 所提供的 SDK。附录 B 讨论了 OpenGL 和着色器程序开发的基础知识。附录 C 介绍如何安装和使用 MinGW。附录 D 讨论了嵌入式 OpenCL 标准。

本书适合于需要在异构平台和并发环境下进行开发的专业人士阅读，要求读者有一定的 C 语言基础。本书也适合对 OpenCL 和高性能计算感兴趣的读者参考。

前言

1997年的夏天，我被震惊了。除了能在自己的专业（微电子工程）做实习生，我所能找到的最好的工作就是在一个专门研究高速信号处理的实验室做事，负责用 C 和 MPI（Message Passing Interface，消息传递接口）编程实现二维快速傅里叶变换（fast Fourier transform, FFT）。好消息是这个实验室有 16 台新的 SPARC 计算机，坏消息是我对 MPI 和 FFT 一无所知。

在当时还是新奇古怪的 Amazon.com 上，我购买了一些相关书籍，然后才费力弄清楚 MPI 的基本处理过程：应用程序将一系列指令部署到多台计算机上，每台计算机根据它的 ID 访问数据。在每台处理器完成相应的任务之后，将结果返回给 ID 为 0 的处理器。

又过了些日子，我才对 MPI 有了更进一步的认识（阻塞和非阻塞数据传输，同步和异步通信）。在编程学习过程中，我发现自己爱上了分布式计算。我喜欢 16 台庞然大物锁步处理数据，它们就像是同场竞赛的运动员。我感觉自己就像是一名编舞，悉心策划一出精彩的舞蹈演出，亦或是一名作曲家，为整个乐团创作交响乐。实习结束时，我已经能用 MPI 编写出多个版本的二维快速傅里叶变换了，但最后因为网络延时的关系，实验室的研究人员否决了这个不太可行的方案。

那个暑假之后，我就对高性能计算倾注了极大的热情，全心投身于数字信号处理器、FPGA（Field-Programmable Gate Arrays，现场可编程门阵列）以及 Cell 处理器（它可是 Sony PS3 的“大脑”）的应用之中。但我想说的是，所有这些都不能够替代用 OpenCL 对 GPU（Graphics Processing Unit，图像处理单元）的编程体验。现在的超级计算机设计也已经说明，没有哪块 CPU 可以提供和 GPU 相并论的“计算处理能力/功耗”的性能。同时，也没有哪门语言能够像 OpenCL 一样，得到如此广泛的设备支持。

当 AMD 在 2009 年发布它的 OpenCL 开发工具时，我就已经爱上了 OpenCL。不仅是因为 OpenCL 能提供新的数据类型（向量数据类型）以及众多的数学函数，还因为它在很多方面都和 MPI 很像。两个工具集都免费、开源，也都可以用 C 或 C++ 来编写应用程序。同时，应用程序也都是将指令发送到多个设备上，其中处理单元根据 ID 来

确定访问相应的数据。MPI 和 OpenCL 也都允许使用相似的阻塞/非阻塞传输和同步/异步通信来进行数据传输。

OpenCL 在高性能计算领域还很新，也许很多程序员都还不知道它的存在。因此，为了帮助扩大这门语言的影响，我决定编写 *OpenCL in Action* 这本书。我很享受写书的过程，也希望它能帮助初学者了解、认识 OpenCL 和分布式计算。

在 2011 年的夏天，我开始写作此书的时候，我不禁有种周而复始的感觉。昨天晚上，我将本书收官在第 14 章的 FFT 应用程序上。它让我回想起了和 MPI 打交道的日子，但我还是惊讶于这些年的技术变迁。1997 年，我实验里的 16 台 SPARC 机器花了将近 1 分钟的时间才完成 32k FFT 的运算。而到了 2011 年，300 美元的显卡就能在几秒之内完成上百万个数据点的 FFT 变换。

技术在变化，但是编程的乐趣不变。也许现如今，分布式计算的学习曲线越来越陡，但回报的甘饴，而非补偿，也胜过以往。

关于本书

OpenCL 是一个复杂的主题。即使是编写最简单的应用程序，开发人员也需要知道主机编程、设备编程以及主机和设备之间的数据传输机制。本书的目的就是要展示如何解决这些问题，以及如何在实际的应用程序应用它们。

本书的格式是辅导书风格，即每个概念后面都会跟有示例代码，来演示如何在应用程序中实现刚才所学到的知识。刚开始的应用程序都很简单，有些只是为了获取设备和数据结构的相关信息。但越到后面，代码会变得越复杂，同时也会对主机和目标设备有更全面的使用。在后面的章节中，重点就会从学习 OpenCL 的基本概念和工作机制，转移到如何用 OpenCL 来应对海量数据的高速处理的难题。

读者

在写作本书时，我对读者的假定是，从未听说过 OpenCL，对分布式计算和高性能计算也是一无所知。我将尽我所能，讲清楚诸如任务并行化以及 SIMD（单指令多数据流）这样的概念。

但因为 OpenCL API 是基于 C 的，所以本书会假定读者对 C 的基本概念十分清楚。读者也需要对指针、数组以及像 malloc 和 free 等内存访问函数相当的熟悉，也最好清楚 math 库文件中的 C 函数，因为 OpenCL 内核函数中也有相似函数名和用法的函数。

OpenCL 应用程序可以在不同类型的设备上运行，但最主要的优势是，它可以被用来对 GPU 进行编程。所以，为了理解本书的内容，你最好是有一块带显卡或是混合 CPU-GPU 设备（如 AMD 的 Fusion）的计算机。

路线图

本书被分为三个部分。第一部分，包括第 1~10 章，主要讨论的是 OpenCL 语言和它的各个功能。第二部分，包括第 11~14 章，展示如何用 OpenCL 来处理高性能计算领域会经常碰到的大规模任务。最后一部分，包括 15 章和 16 章，展示如何用 OpenCL

来加速 OpenGL 应用程序的执行。

第一部分的设计是为了迎合从未编写过 OpenCL 程序的程序员的需要。第 1 章来引入 OpenCL 的话题，解释它是什么，从哪里来，以及基本的运行过程。第 2 章和第 3 章介绍如何编写在主机上运行的应用程序，第 4 章和第 5 章介绍如何编写在 OpenCL 兼容设备上运行的内核。第 6 章和第 7 章讨论主机编程和内核编程过程中会碰到的高级话题，确切地讲，第 6 章讨论的是图像处理，第 7 章讨论的是事件处理和同步化的问题。

第 8 章和第 9 章讨论的是从第 2 章到第 5 章所遇到的概念，但所用到的的是不同的编程语言。第 8 章讨论如何用 C++ 来编写主机/内核代码，第 9 章讨论如何用 Java 和 Python 来编写 OpenCL 应用程序。如果你不是必须要用 C 来编程，我强烈推荐你看看这几章中所介绍的工具集。

第 10 章是第 1 部分和第 2 部分的桥梁。它展示的是如何利用 OpenCL 的并行编程的优势，举的例子是实现简单的归并算法，将一百万个数据点加起来。同时，也提出了一些编写实际应用程序的指导原则。

第 11 章到第 14 章关注的是用 OpenCL 来处理海量数据，应用程序一般都是对百万个数据点进行的操作。第 11 章讨论的是 MapReduce 算法的实现以及两种排序算法：双调排序和基数排序。第 12 章讨论的是稠密矩阵的运算，第 13 章讨论的是稀疏矩阵的运算。第 14 章讨论的则是如何使用 OpenCL 来实现快速傅里叶变换。

第 15 章和第 16 章是我个人最喜欢的，OpenCL 的优势就在于可以用来加速三维渲染，这也是游戏开发和科学可视化最重要的话题。第 15 章讨论的是 OpenCL-OpenGL 互操作的问题，展示两个工具集如何共享和顶点属性相对应的数据。第 16 章将展开对 OpenCL-OpenGL 互操作话题的讨论，来了解如何用 OpenCL 加速 OpenGL 的纹理处理。这些章节都需要对 OpenGL 3.3 和着色器程序开发有所了解，而这两个话题的相关知识都被放在附录 B 中。

本书结尾的附录提供了很多关于 OpenCL 的有用信息，但这些资料并不是直接在一般的 OpenCL 开发中使用。附录 A 先是介绍 SDK，然后演示如何安装 AMD 和 Nvidia 所提供的 SDK。附录 B 讨论的是 OpenGL 和着色器程序开发的基础知识。附录 C 介绍的是如何安装和使用 MinGW (Minimalist GNU for Windows)，MinGW 是一个类 GNU 环境，用于在 Windows 系统下编译生成可执行程序。最后，附录 D 讨论的是嵌入式 OpenCL 标准。

获取和编译示例代码

最后，代码才是王道。本书包含了超过 60 个 OpenCL 应用程序的代码，你可以到出版社的网站去下载源代码：www.manning.com/OpenCLinAction 或 www.manning.com/scarpino2/。

下载站点提供了一个指向包含源代码的压缩包，可以用基于 GNU 的编译工具来编

译。这个压缩包中的文件夹和各章/附录相对应，每一个顶层文件夹中的子文件夹和示例项目对应。例如，如果查看 Ch5/shuffle_test 路径，你可以看到第 5 章 shuffle_test 项目的源代码。

至于依赖文件，每个项目都需要开发系统上已经安装了 OpenCL 库文件（Windows 下是 OpenCL.lib，*nix 系统下是 libOpenCL.so 文件）。附录 A 所讨论的就是如何获取、安装相应的 SDK。

此外，第 6 章和第 16 章讨论的是图像处理，这些章节的源代码用到了开源 PNG 库。第 6 章介绍了如何获取相应的库文件。附录 B 和第 15、16 章都涉及了对 OpenGL 的访问，附录 B 介绍了如何获取、安装这个工具集。

代码命名约定

尽管听起来很懒惰，但我还是情愿复制、粘贴源代码到我的应用程序之中，而不是选择从头开始。这不仅节约时间，而且减少了代码输入的错误。本书中的所有代码都是公开的，所以你可以免费下载，复制、粘贴任何你想要的代码。但在此之前，还是有必要了解我所使用的命名约定规则。

- 主机数据结构会在它的数据类型之后。cl_platform_id 结构被称为 platform（平台），cl_device_id 结构被称为 device（设备），cl_context 结构被称为 context（上下文）等。

- 在主机应用程序中，main 共调用了两个函数：调用 create_device 函数返回 cl_device，调用 build_program 函数创建和编译 cl_program。注意，create_device 函数会搜索和 GPU 相关联的第一个可用平台，如果没有找到任何可用的 GPU，它会转而搜索第一个兼容的 CPU。

- 主机应用程序在源文件的开头，通过宏定义来识别源文件和内核函数。具体而言，PROGRAM_FILE 宏来标识程序文件，KERNEL_FUNC 函数来标识内核函数。

- 所有的程序文件都是以.cl 后缀结尾。如果源文件只包含一个内核函数，函数名将和文件名相同。

- 对于 GNU 程序而言，所有的 makefile 都会假设库文件和头文件可以在环境变量指定的路径下找到。具体而言，makefile 会搜索 AMD 平台下 AMDAPPSDKROOT 指定的路径或者 Nvidia 平台下 CUDA 指定的路径。

作者在线

金无足赤，人无完人。如果本书在某个主题上说得不够清楚，或者出现了错误，请你通过 Manning 的作者在线给我提反馈意见。本书的作者在线论坛网址是 www.manning.com/OpenCLinAction，单击 Author Online 链接即可。

简单的问题和疑问，都会得到最及时的答复。但如果你是对双调排序实现的第 402 行代码不感冒，那我就得花些时间才能回复你了。我很愿意讨论所有和 OpenCL 相关的

一般性问题，但如果你是想寻求复杂、特定应用的帮助，例如对定制 FFT 程序进行调试，我还是建议你找一个专业顾问来帮忙，这样会更有效一些。

关于封面图

OpenCL in Action 的封面图的标题是“Kranjac”人，一群生活在 Slovenian Alps, Carniola 的原住民。插画来自 Balthasar Hacquet 的 *Images and Descriptions of Southwestern and Eastern Wenda, Illyrians, and Slavs*，该书由克罗地亚 Split 的 Ethnographic Museum 在 2008 年重印出版。Hacquet (1739~1815) 是一名奥地利医生、科学家，曾经在从意大利东北到斯洛文尼亚的 Julian Alps (以 Julius Caesar 的名字命名) 地区做过多年的植物学、地理学以及人类学的研究。这些手绘的插图都出现在 Hacquet 出版的科学论文和书籍中。

Hacquet 所出版的书刊中插画异常丰富、多种多样的特点也说明了 200 年以前的东阿尔卑斯山脉地区的与众不同和相互独立。在那个年代里，各个村庄之间的穿着都相互有别，村民的穿着也反映出他的社会地位。但到了后来，原住民的穿着就发生了很大的变化，曾经是如此的丰富，引以为豪的差异化都随时间如流水一般，悄然逝去。而就像在当今社会，很难将世界不同角落的民众细细分开一样，斯洛文尼亚山脉那些曾经曾经与众不同的村民，也已经和其他地区的斯洛文尼亚居民，或其他的欧洲大陆居民一样，别无二致。

我们 Manning 希望通过以封面插图的形式，将两个世纪前的服装以插图的形式重现，来慨叹计算机工业的发明、创新及其所带来的趣味。

致谢

我是在 2003 年为 Manning Publications 撰写第一本书的，尽管世事变迁，他们却还是一如既往地致力于出版高品质的书籍。我非常感谢全体 Manning 员工的辛勤工作以及奉献精神，但也想在此特别感谢以下几个人。

首先，我想感谢的是 Maria Townsley，她是我的开发编辑（developmental editor）。Maria 是我见过的最关注实现结果的编辑之一，除此之外，她在书籍的组织和书写理清清晰方面也提供了很多的建议。尽管我这人脾气不好，但到最后也不得不服气，承认她确实是对的。此外，虽然，我不断地反复修改表格内容，她都一直鼎力地给予支持和帮助。

同时，我还想深深感谢整个 Manning 的发行团队。尤其是要感谢 Andy Carroll，感谢他对编辑本书的巨大付出。他的意见和观点不仅对本书的润色起到了巨大的作用，同时他的专业知识也让本书更好地做到了深入浅出。同样地，我也想在这里感谢 Maureen Spencer 和 Katie Tennant，感谢他们对最终稿的字斟句酌，感谢 Gordan Salinovic，感谢他对本书图片和结构的不懈努力。同时，我还想感谢 Mary Piergies，感谢她策划了全书，保证了本书能够维系 Manning 的高品质。

简单而言，Jörn Dinkla 是我合作过的最好的技术编辑。我已经将本书中的示例代码在 Linux 和 Mac OS 下分别进行了测试，而他更进一步，将代码用 AMD 和 Nvidia 的 SDK 在 Linux 下进行了测试。他不仅找出了我未曾发现的几个错误，而且很多次地花时间研究出错的个中原因。我不敢想象，如果缺少他的帮助，事情会演变成什么样子。在此，我深深感谢他的辛勤付出，对提升本书代码质量所作出的努力。

我还要感谢 Candace Gilhooley，感谢他对本书的宣传。由于 OpenCL 还很年轻，它的读者群并不像 Manning 的众多 Java 书一样，族群丰富。但不得不说，无论是在网络专栏、图书展览还是会议研讨中，Candace 已然对 OpenCL in Action 做足了宣传。

Manning 最强大的地方就在于它一直信赖反馈意见。在本书的编写和编辑期间，Karen Tegtmeier 和 Ozren Harlovic 邀请了众多的审稿人，并组织了多轮的审阅。正是由于

这些审稿人的反馈意见，这本书得以融入了很多我不大可能会考虑的重要主题，他们是：Olivier Chafik, Martin Beckett, Benjamin Ducke, Alan Commike, Nathan Levesque, David Strong, Seth Price, John J. Ryan III 和 John Griffin。

最后，我还要感谢 Jandex Indexing 的 Jan Bednarczuk，感谢她对索引本书内容的细致工作。她不仅在很短的时间创建了一个完整、专业的检索表，还发现了不少的书写错误。再次感谢。

目录

第一部分 OpenCL 编程基础

第 1 章 OpenCL 简介 3

- 1.1 OpenCL 的来临 4
- 1.2 为什么是 OpenCL 5
 - 1.2.1 可移植性 5
 - 1.2.2 标准化的向量处理 6
 - 1.2.3 并行编程 7
- 1.3 类比: OpenCL 处理和纸牌游戏 8
- 1.4 OpenCL 应用程序的第一印象 10
- 1.5 OpenCL 标准和扩展 13
- 1.6 框架和 SDK 13
- 1.7 小结 14

第 2 章 主机编程: 基本的数据结构 15

- 2.1 基本数据类型 16
- 2.2 获取平台信息 17
 - 2.2.1 创建平台结构 17
 - 2.2.2 获取平台信息 18
 - 2.2.3 示例代码: 测试平台的扩展 19
- 2.3 访问安装设备 20
 - 2.3.1 创建设备结构 21
 - 2.3.2 获取设备信息 22
 - 2.3.3 示例代码: 测试设备扩展 22
- 2.4 通过上下文管理设备 24

- 2.4.1 创建上下文 25
- 2.4.2 获取上下文信息 26
- 2.4.3 上下文和引用计数 27
- 2.4.4 示例代码: 检查上下文的引用计数 27

2.5 将设备代码保存在程序中 28

- 2.5.1 创建程序 29
- 2.5.2 编译程序 30
- 2.5.3 获取程序信息 31
- 2.5.4 示例代码: 构建来自多个源文件的程序 33

2.6 将函数打包为内核 34

- 2.6.1 创建内核 35
- 2.6.2 获取内核信息 35
- 2.6.3 示例代码: 获取内核信息 36

2.7 用命令队列保存内核 37

- 2.7.1 创建命令队列 38
- 2.7.2 入列内核执行命令 38

2.8 小结 39

第 3 章 主机编程: 数据传输和数据划分 41

- 3.1 设置内核参数 42
- 3.2 缓存对象 42
 - 3.2.1 分配缓存对象 43
 - 3.2.2 创建子缓存对象 45
- 3.3 图像对象 46
 - 3.3.1 创建图像对象 46
 - 3.3.2 获取图像对象的相关

- 信息 49
- 3.4 获取缓存对象的相关信息 49
- 3.5 内存对象的传输命令 51
 - 3.5.1 读/写数据传输 52
 - 3.5.2 映射内存对象 55
 - 3.5.3 内存对象间的数据复制 57
- 3.6 数据划分 59
 - 3.6.1 循环和工作项 60
 - 3.6.2 工作项的大小和偏移量 61
 - 3.6.3 一个简单的一维例子 62
 - 3.6.4 工作组和计算单元 63
- 3.7 小结 64
- 第4章 内核编程：数据类型和设备内存 66**
 - 4.1 内核编程简介 67
 - 4.2 标量数据类型 68
 - 4.2.1 访问双精度数据类型 69
 - 4.2.2 字节顺序 70
 - 4.3 浮点计算 71
 - 4.3.1 浮点数据类型 71
 - 4.3.2 双精度数据类型 72
 - 4.3.3 half数据类型 73
 - 4.3.4 检查IEEE-754的兼容性 73
 - 4.4 向量数据类型 75
 - 4.4.1 首选向量宽度 76
 - 4.4.2 初始化向量 78
 - 4.4.3 读取和修改向量分量 78
 - 4.4.4 字节顺序和内存访问 81
 - 4.5 OpenCL设备模型 83
 - 4.5.1 内核模型类比的第一部分：学校中学数学的学生 83
 - 4.5.2 设备模型类比的第二部分：设备上的工作项 84
 - 4.5.3 程序中的地址空间 86
 - 4.5.4 内存对齐 88
 - 4.6 局部和私有内核参数 88
 - 4.6.1 局部参数 89
 - 4.6.2 私有参数 89
 - 4.7 小结 90
- 第5章 内核编程：运算符和函数 92**
 - 5.1 运算符 93
 - 5.2 工作组和工作项函数 95
 - 5.2.1 维度和工作项 96
 - 5.2.2 工作组 97
 - 5.2.3 示例应用 97
 - 5.3 数据传输操作 98
 - 5.3.1 加载和保存同类型的数 98
 - 5.3.2 将标量数组加载保存到向量中 99
 - 5.3.3 将向量保存到标量数组中 100
 - 5.4 浮点型函数 100
 - 5.4.1 算术运算函数和取舍函数 100
 - 5.4.2 比较函数 102
 - 5.4.3 指数函数和对数函数 103
 - 5.4.4 三角函数 103
 - 5.4.5 其他类型的浮点函数 105
 - 5.5 整数函数 106
 - 5.5.1 加法函数和减法函数 106
 - 5.5.2 乘法运算 108
 - 5.5.3 其他类型的整数函数 109
 - 5.6 混洗和选择函数 111
 - 5.6.1 混洗函数 111
 - 5.6.2 选择函数 113
 - 5.7 向量测试函数 115
 - 5.8 几何函数 116
 - 5.9 小结 118
- 第6章 图像处理 120**
 - 6.1 图像对象和采样器 121
 - 6.1.1 主机上的图像对象：cl mem 121
 - 6.1.2 主机上的采样器：cl_sampler 122
 - 6.1.3 设备上的图像对象：image2d_t和image3d_t 125
 - 6.1.4 设备上的采样器：sampler_t 126
 - 6.2 图像处理函数 126
 - 6.2.1 图像读取函数 127
 - 6.2.2 写图像函数 128
 - 6.2.3 图像信息函数 129
 - 6.2.4 一个简单的例子 130
 - 6.3 图像放缩和插值 131
 - 6.3.1 最邻近插值 131
 - 6.3.2 双线性插值 132
- 第5章 内核编程：运算符和函数 92**
 - 5.1 运算符 93

- 6.3.3 用 OpenCL 编程放大图像 134
- 6.4 小结 135
- 7 第 7 章 事件、性能分析及同步化 136
 - 7.1 主机提醒事件 137
 - 7.1.1 将事件和命令关联 137
 - 7.1.2 将事件和回调函数关联 138
 - 7.1.3 主机提醒的例子 139
 - 7.2 命令同步事件 140
 - 7.2.1 等待列表和命令事件 141
 - 7.2.2 等待列表和用户事件 142
 - 7.2.3 额外的命令同步函数 144
 - 7.2.4 获取和事件关联的数据 146
 - 7.3 性能分析事件 149
 - 7.3.1 配置性能分析命令 149
 - 7.3.2 对数据传输进行性能分析 151
 - 7.3.3 对数据划分进行分析 152
 - 7.4 工作项同步化 154
 - 7.4.1 障碍和栅栏 155
 - 7.4.2 原子操作 156
 - 7.4.3 原子命令和互斥 158
 - 7.4.4 异步数据传输 160
 - 7.5 小结 161
- 8 第 8 章 用 C++ 开发 163
 - 8.1 初步了解 164
 - 8.1.1 向量和字符串 164
 - 8.1.2 异常 165
 - 8.2 创建内核 166
 - 8.2.1 平台、设备以及上下文 166
 - 8.2.2 程序和内核 169
 - 8.3 内核参数和内存对象 172
 - 8.3.1 内存对象 173
 - 8.3.2 通用数据参数 177
 - 8.3.3 局部内存参数 178
 - 8.4 命令队列 179
 - 8.4.1 创建 CommandQueue 对象 179
 - 8.4.2 入列内核执行命令 179
 - 8.4.3 读写命令 181
 - 8.4.4 内存映射和复制命令 183
- 8.5 事件处理 185
 - 8.5.1 主机提醒 185
 - 8.5.2 命令同步化 187
 - 8.5.3 性能分析事件 188
 - 8.5.4 另外的事件函数 189
- 8.6 小结 190
- 9 第 9 章 用 Java 和 Python 来开发 192
 - 9.1 Aparapi 193
 - 9.1.1 Aparapi 安装 193
 - 9.1.2 Kernel 类 194
 - 9.1.3 工作项和工作组 195
 - 9.2 JavaCL 197
 - 9.2.1 JavaCL 安装 198
 - 9.2.2 JavaCL 开发概述 198
 - 9.2.3 用 JavaCL 来创建内核 199
 - 9.2.4 设定内核参数以及入列命令 202
 - 9.3 PyOpenCL 206
 - 9.3.1 PyOpenCL 安装和许可 206
 - 9.3.2 PyOpenCL 开发概述 207
 - 9.3.3 用 PyOpenCL 创建内核 207
 - 9.3.4 设置参数和执行内核 211
 - 9.4 小结 215
- 10 第 10 章 通用编程原则 217
 - 10.1 全局大小和局部大小 218
 - 10.1.1 找出工作组大小的上限值 218
 - 10.1.2 测试内核和设备 220
 - 10.2 数值归并 221
 - 10.2.1 OpenCL 的归并算法 221
 - 10.2.2 使用向量提升归并运算的速度 224
 - 10.3 工作组间的同步化 225
 - 10.4 设计高性能内核的 10 条技巧 227
 - 10.5 小结 229

第二部分 用 OpenCL 来编程实现实际的算法

第 11 章 归并与排序 233

11.1 MapReduce 234

11.1.1 MapReduce 简介 234

11.1.2 MapReduce 和 OpenCL 236

11.1.3 MapReduce 例子: 字符串查找 238

11.2 双调排序 240

11.2.1 理解双调排序算法 240

11.2.2 用 OpenCL 来实现双调排序 243

11.3 基数排序 249

11.3.1 理解基数排序 250

11.3.2 用向量实现基数排序 250

11.4 小结 252

第 12 章 矩阵和 QR 分解 253

12.1 矩阵转置 254

12.1.1 矩阵简介 254

12.1.2 矩阵转置的理论和实现 254

12.2 矩阵乘法 257

12.2.1 矩阵乘法理论 257

12.2.2 用 OpenCL 编程实现矩阵乘法 258

12.3 Householder 变换 259

12.3.1 向量投影 260

12.3.2 向量反射 261

12.3.3 外积和 Householder 矩阵 262

12.3.4 用 OpenCL 编程实现向量反射 263

12.4 QR 分解 264

12.4.1 计算 Householder 向量和 R 矩阵 265

12.4.2 计算 Householder 矩阵和矩阵 Q 266

12.4.3 用 OpenCL 编程实现 QR 分解 267

12.5 小结 270

第 13 章 稀疏矩阵 272

13.1 差分方程和稀疏矩阵 273

13.2 稀疏矩阵的存储以及 Harwell-Boeing 数据集 274

13.2.1 Harwell-Boeing 数据集简介 274

13.2.2 访问 Matrix Market 文件中的数据 275

13.3 最速下降法 278

13.3.1 正定矩阵 279

13.3.2 最速下降法理论 279

13.3.3 用 OpenCL 编程实现 SD 算法 281

13.4 共轭梯度法 283

13.4.1 正交化和共轭 283

13.4.2 正交化和 GRAM-SCHMIDT 法 283

13.4.3 共轭梯度法 285

13.5 小结 287

第 14 章 信号处理和快速傅里叶变换 289

14.1 频率分析简介 289

14.2 离散傅里叶变换 291

14.2.1 DFT 背后的理论 292

14.2.2 OpenCL 和 DFT 298

14.3 快速傅里叶变换 299

14.3.1 DFT 的三条性质 299

14.3.2 构建快速傅里叶变换 302

14.3.3 用 OpenCL 来实现 FFT 306

14.4 小结 311

第三部分 用 OpenCL 来加速 OpenGL

15 第 15 章 将 OpenCL 和 OpenGL 结合 315

- 15.1 在 OpenGL 和 OpenCL 之间共享数据 316
 - 15.1.1 创建 OpenCL 上下文 317
 - 15.1.2 在 OpenGL 和 OpenCL 之间共享数据 319
 - 15.1.3 同步化对共享数据的访问 322
- 15.2 获取信息 323
 - 15.2.1 获取 OpenGL 对象和纹理信息 323
 - 15.2.2 获取 OpenGL 上下文的 相关信息 324
- 15.3 基本的互操作例子 325
 - 15.3.1 初始化 OpenGL 操作 325
 - 15.3.2 初始化 OpenCL 操作 326
 - 15.3.3 创建数据对象 327
 - 15.3.4 执行内核 327
 - 15.3.5 渲染图形 328
- 15.4 互操作和动画 329
 - 15.4.1 确定顶点数据 329
 - 15.4.2 动画和显示 330
 - 15.4.3 执行内核 331
- 15.5 小结 332

16 第 16 章 纹理和渲染缓存 334

- 16.1 图像滤波 335
 - 16.1.1 高斯模糊 337
 - 16.1.2 图像锐化 337
 - 16.1.3 图像浮雕化 338
- 16.2 用 OpenCL 来对纹理滤波 339
 - 16.2.1 init_gl 函数 339
 - 16.2.2 init_cl 函数 339
 - 16.2.3 configure_shared_data 函数 340
 - 16.2.4 execute_kernel 函数 341
 - 16.2.5 display 函数 342
- 16.3 小结 343

A 附录 A 安装和使用软件 开发包 344

- A.1 了解 OpenCL SDK 344
 - A.1.1 检查设备的兼容性 344
 - A.1.2 OpenCL 头文件和库文件 345
- A.2 Windows 上的 OpenCL 347
 - A.2.1 在 Windows 上安装 AMD 显卡驱动 347
 - A.2.2 用 AMD 显卡来编译 Windows 应用程序 349
 - A.2.3 在 Windows 上安装 Nvidia 的显卡驱动 349
 - A.2.4 用 Nvidia 显卡来编译 Windows 应用程序 351
- A.3 Linux 上的 OpenCL 351
 - A.3.1 在 Linux 上安装 AMD 的显卡驱动 351
 - A.3.2 在 Linux 上安装 Nvidia 显卡驱动 352
 - A.3.3 在 Linux 上编译 OpenCL 应用程序 354
- A.4 在 Mac OS 上安装 OpenCL 355
- A.5 小结 356

B 附录 B 用 OpenGL 作实时渲染 357

- B.1 安装 OpenGL 358
 - B.1.1 在 Windows 上安装 OpenGL 359
 - B.1.2 在 Linux 上安装 OpenGL 359
 - B.1.3 在 Mac OS 上安装 OpenGL 360
- B.2 在主机上开发 OpenGL 应用程序 360
 - B.2.1 将数据放到顶点缓存对象 (VBO) 之中 361
 - B.2.2 配置顶点属性 363