

David B. Kirk
Wen-mei W. Hwu

Programming Massively Parallel Processors

A Hands-on Approach



NVIDIA

MK
MORGAN KAUFMANN

Contents

Preface	xi
Acknowledgments	xvii
Dedication	xix
CHAPTER 1 INTRODUCTION	1
1.1 GPUs as Parallel Computers	2
1.2 Architecture of a Modern GPU	8
1.3 Why More Speed or Parallelism?	10
1.4 Parallel Programming Languages and Models	13
1.5 Overarching Goals	15
1.6 Organization of the Book	16
CHAPTER 2 HISTORY OF GPU COMPUTING	21
2.1 Evolution of Graphics Pipelines	21
2.1.1 The Era of Fixed-Function Graphics Pipelines	22
2.1.2 Evolution of Programmable Real-Time Graphics	26
2.1.3 Unified Graphics and Computing Processors	29
2.1.4 GPGPU: An Intermediate Step	31
2.2 GPU Computing	32
2.2.1 Scalable GPUs	33
2.2.2 Recent Developments	34
2.3 Future Trends	34
CHAPTER 3 INTRODUCTION TO CUDA	39
3.1 Data Parallelism	39
3.2 CUDA Program Structure	41
3.3 A Matrix–Matrix Multiplication Example	42
3.4 Device Memories and Data Transfer	46
3.5 Kernel Functions and Threading	51
3.6 Summary	56
3.6.1 Function declarations	56
3.6.2 Kernel launch	56
3.6.3 Predefined variables	56
3.6.4 Runtime API	57
CHAPTER 4 CUDA THREADS	59
4.1 CUDA Thread Organization	59
4.2 Using <code>blockIdx</code> and <code>threadIdx</code>	64
4.3 Synchronization and Transparent Scalability	68

4.4	Thread Assignment	70
4.5	Thread Scheduling and Latency Tolerance	71
4.6	Summary	74
4.7	Exercises	74
CHAPTER 5	CUDA™ MEMORIES	77
5.1	Importance of Memory Access Efficiency	78
5.2	CUDA Device Memory Types	79
5.3	A Strategy for Reducing Global Memory Traffic	83
5.4	Memory as a Limiting Factor to Parallelism	90
5.5	Summary	92
5.6	Exercises	93
CHAPTER 6	PERFORMANCE CONSIDERATIONS	95
6.1	More on Thread Execution	96
6.2	Global Memory Bandwidth	103
6.3	Dynamic Partitioning of SM Resources	111
6.4	Data Prefetching	113
6.5	Instruction Mix	115
6.6	Thread Granularity	116
6.7	Measured Performance and Summary	118
6.8	Exercises	120
CHAPTER 7	FLOATING POINT CONSIDERATIONS	125
7.1	Floating-Point Format	126
7.1.1	Normalized Representation of M	126
7.1.2	Excess Encoding of E	127
7.2	Representable Numbers	129
7.3	Special Bit Patterns and Precision	134
7.4	Arithmetic Accuracy and Rounding	135
7.5	Algorithm Considerations	136
7.6	Summary	138
7.7	Exercises	138
CHAPTER 8	APPLICATION CASE STUDY: ADVANCED MRI RECONSTRUCTION	141
8.1	Application Background	142
8.2	Iterative Reconstruction	144
8.3	Computing F^{Hd}	148
	Step 1. Determine the Kernel Parallelism Structure	149
	Step 2. Getting Around the Memory Bandwidth Limitation	156

Step 3. Using Hardware Trigonometry Functions	163
Step 4. Experimental Performance Tuning	166
8.4 Final Evaluation.....	167
8.5 Exercises	170
CHAPTER 9 APPLICATION CASE STUDY: MOLECULAR VISUALIZATION AND ANALYSIS	173
9.1 Application Background.....	174
9.2 A Simple Kernel Implementation	176
9.3 Instruction Execution Efficiency.....	180
9.4 Memory Coalescing.....	182
9.5 Additional Performance Comparisons	185
9.6 Using Multiple GPUs	187
9.7 Exercises	188
CHAPTER 10 PARALLEL PROGRAMMING AND COMPUTATIONAL THINKING	191
10.1 Goals of Parallel Programming	192
10.2 Problem Decomposition	193
10.3 Algorithm Selection	196
10.4 Computational Thinking.....	202
10.5 Exercises	204
CHAPTER 11 A BRIEF INTRODUCTION TO OPENCL™	205
11.1 Background.....	205
11.2 Data Parallelism Model.....	207
11.3 Device Architecture.....	209
11.4 Kernel Functions	211
11.5 Device Management and Kernel Launch	212
11.6 Electrostatic Potential Map in OpenCL	214
11.7 Summary	219
11.8 Exercises	220
CHAPTER 12 CONCLUSION AND FUTURE OUTLOOK	221
12.1 Goals Revisited.....	221
12.2 Memory Architecture Evolution	223
12.2.1 Large Virtual and Physical Address Spaces	223
12.2.2 Unified Device Memory Space	224
12.2.3 Configurable Caching and Scratch Pad.....	225
12.2.4 Enhanced Atomic Operations	226
12.2.5 Enhanced Global Memory Access	226

12.3	Kernel Execution Control Evolution	227
12.3.1	Function Calls within Kernel Functions	227
12.3.2	Exception Handling in Kernel Functions	227
12.3.3	Simultaneous Execution of Multiple Kernels	228
12.3.4	Interruptible Kernels	228
12.4	Core Performance	229
12.4.1	Double-Precision Speed	229
12.4.2	Better Control Flow Efficiency	229
12.5	Programming Environment	230
12.6	A Bright Outlook.....	230

APPENDIX A MATRIX MULTIPLICATION HOST-ONLY VERSION

	SOURCE CODE	233
A.1	matrixmul.cu.....	233
A.2	matrixmul_gold.cpp.....	237
A.3	matrixmul.h.....	238
A.4	assist.h	239
A.5	Expected Output	243

APPENDIX B GPU COMPUTE CAPABILITIES **245**

B.1	GPU Compute Capability Tables.....	245
B.2	Memory Coalescing Variations.....	246

Index	251
--------------------	------------