Using OpenMP—The Next Step

Affinity, Accelerators, Tasking, and SIMD

Ruud van der Pas, Eric Stotzer, and Christian Terboven

The MIT Press
Cambridge, Massachusetts
London, England

Contents

	Series Foreword	xiii
	Foreword	xv
	Preface	xix
1	**A Recap of OpenMP 2.5**	1
1.1	OpenMP Directives and Syntax	1
1.2	Creating a Parallel Program with OpenMP	2
	1.2.1 The Parallel Region	3
	1.2.2 The OpenMP Execution Model	4
	1.2.3 The OpenMP Memory Model	6
1.3	The Worksharing Constructs	18
	1.3.1 The Loop Construct	18
	1.3.2 The Sections Construct	19
	1.3.3 The Single Construct	21
	1.3.4 The Fortran Workshare Construct	23
	1.3.5 The Combined Worksharing Constructs	23
1.4	The Master Construct	25
1.5	Nested Parallelism	25
1.6	Synchronization Constructs	28
	1.6.1 The Barrier Construct	28
	1.6.2 The Critical Construct	30
	1.6.3 The Atomic Construct	32
	1.6.4 The Ordered Construct	33
1.7	The OpenMP 2.5 Environment Variables	34
1.8	The OpenMP 2.5 Runtime Functions	35
1.9	Internal Control Variables in OpenMP	37
1.10	Concluding Remarks	38
2	**New Features in OpenMP**	41
2.1	Enhancements to Existing Constructs	41

	2.1.1	The Schedule Clause	41
	2.1.2	The If Clause	43
	2.1.3	The Collapse Clause	43
	2.1.4	The Linear Clause	46
	2.1.5	The Critical Construct	46
	2.1.6	The Atomic Construct	47
2.2	New Environment Variables		53
2.3	New Runtime Functions		60
	2.3.1	Runtime Functions for Thread Management, Thread Scheduling, and Nested Parallelism	61
	2.3.2	Runtime Functions for Tasking, Cancellation, and Thread Affinity	64
	2.3.3	Runtime Functions for Locking	67
	2.3.4	Runtime Functions for Heterogeneous Systems	70
	2.3.5	Usage Examples of the New Runtime Functions	76
2.4	New Functionality		86
	2.4.1	Changed Ownership of Locks	86
	2.4.2	Cancellation	87
	2.4.3	User-Defined Reduction	93
	2.4.4	The Doacross Loop	100
2.5	Concluding Remarks		102
3	**Tasking**		103
3.1	Hello Task		103
	3.1.1	Parallelizing a Palindrome	104
	3.1.2	Parallelizing a Sentence with a Palindrome	106
	3.1.3	Closing Comments on the Palindrome Example	109
3.2	Using Tasks to Parallelize a Linked List		109
	3.2.1	The Sequential Version of the Linked List Program	110
	3.2.2	The Parallel Version of the Linked List Program	113
	3.2.3	Closing Comments on the Linked List Example	118
3.3	Sorting Things Out with Tasks		118

3.3.1 The Sequential Quicksort Algorithm 119
3.3.2 The OpenMP Quicksort Algorithm 122
3.3.3 Fine-Tuning the OpenMP Quicksort Algorithm 124
3.3.4 Closing Comments on the OpenMP Quicksort Algorithm 127
3.4 Overlapping I/O and Computations Using Tasks 128
3.4.1 Using Tasks and Task Dependences 129
3.4.2 Using the Taskloop Construct 133
3.4.3 Closing Comments on the Pipeline Example 136
3.5 The Data Environment with Tasks 136
3.6 What is a Task? 138
3.7 Task Creation, Synchronization, and Scheduling 141
3.8 The Taskloop Construct 146
3.9 Concluding Remarks 149

4 Thread Affinity 151
4.1 The Characteristics of a cc-NUMA Architecture 151
4.2 First Touch Data Placement 153
4.2.1 The Pros and Cons of First Touch Data Placement 153
4.2.2 How to Exploit the First Touch Policy 154
4.3 The Need for Thread Affinity Support 156
4.4 The OpenMP Thread Affinity Philosophy 157
4.5 The OpenMP Places Concept 160
4.5.1 Defining OpenMP Places Using Sets with Numbers 161
4.5.2 The OpenMP Place List 165
4.5.3 Defining OpenMP Places Using Abstract Names 166
4.5.4 How to Define the OpenMP Place List 167
4.6 Mapping Threads onto OpenMP Places 168
4.6.1 The Master Affinity Policy 172
4.6.2 The Close Affinity Policy 173
4.6.3 The Spread Affinity Policy 178

4.6.4 What's in a Name? 186
4.7 Making it Easier to Use the Thread Affinity Policies 188
4.7.1 The Sockets Abstract Place List 189
4.7.2 The Cores Abstract Place List 190
4.7.3 The Threads Abstract Place List 194
4.8 Where Are My Threads Running? 197
4.8.1 Affinity Examples for a Single-Level Parallel Region 198
4.8.2 Affinity Examples for a Nested Parallel Region 205
4.8.3 Making Things Easier Again 210
4.8.4 Moving Threads Around 215
4.9 Concluding Remarks 218

5 SIMD – Single Instruction Multiple Data 221
5.1 An Introduction to SIMD Parallelism 221
5.2 SIMD loops 224
5.2.1 The Simd Construct 224
5.2.2 The Simdlen Clause 228
5.2.3 The Safelen Clause 229
5.2.4 The Linear Clause 230
5.2.5 The Aligned Clause 233
5.2.6 The Composite Loop SIMD Construct 234
5.2.7 Use of the Simd Construct with the Ordered Construct 238
5.3 SIMD Functions 240
5.3.1 The Declare Simd Directive 241
5.3.2 SIMD Function Parameter Attributes 243
5.3.3 Conditional Calls to SIMD Functions 247
5.3.4 Multiple Versions of a SIMD Function 250
5.4 Concluding Remarks 250

6 Heterogeneous Architectures 253
6.1 Devices and Accelerators 253

6.2	Heterogeneous Program Execution	255
	6.2.1 A New Initial Thread	256
	6.2.2 Contention Groups	258
	6.2.3 A League of Teams	258
	6.2.4 The Target Task	259
6.3	Heterogeneous Memory Model	262
	6.3.1 Mapped Variables	263
	6.3.2 Device Data Environments	264
	6.3.3 Device Pointers	265
	6.3.4 Array Sections	267
6.4	The Target Construct	270
6.5	The Target Teams Construct	275
	6.5.1 The Distribute Construct	280
	6.5.2 Combined and Composite Accelerated Worksharing Constructs	283
6.6	Data Mapping Clauses	287
	6.6.1 The Map Clause	289
	6.6.2 Mapping Structure Members	295
	6.6.3 The Defaultmap Clause and Data-mapping Attributes	296
	6.6.4 Pointers and Zero-length Array Sections	300
6.7	The Declare Target Directive	301
6.8	The Data-mapping Constructs	305
	6.8.1 The Target Data Construct	306
	6.8.2 The Target Update Construct	308
	6.8.3 The Target Enter and Exit Data Constructs	310
6.9	The Nowait Clause on Device Constructs	314
6.10	Selecting a Device	317
	6.10.1 The Default Device and the Device Clause	317
	6.10.2 The If Clause on Device Constructs	319
6.11	The Device Pointer Clauses	320
	6.11.1 The Is_device_ptr Clause	321

6.11.2 The Use_device_ptr Clause 322
6.12 Device Memory Functions 323
6.13 Concluding Remarks 328

7 What is Next? 331
7.1 Memory Management 331
7.2 Heterogeneous Architectures 333
Glossary 335
References 355
Subject Index 359