Contents

About This Manual xi
Customer Support xv

Part I The CM Fortran Programming Environment

Chapter 1 The CM-5 System 3

1.1 The CM-5 Hardware 3
1.1.1 The Processing Nodes 5
1.1.2 The Partition Manager 5
1.1.3 The Vector Units 6
1.1.4 VU Hardware and Communication Speeds 6
1.1.5 Processing Elements (PEs) 7
1.1.6 Parallel Memory 7
1.2 Low-Level CM-5 Software 8
1.2.1 The CM Run-Time System 8
1.2.2 SPARC and DPEAC Assembly Code 8

Chapter 2 The CM Fortran Compiler 9

2.1 CM Fortran Execution Model 9
2.1.1 Scalar/Parallel Data and Instructions 9
2.1.2 Parallel Data in CM Fortran 10
2.1.3 CM Fortran Program Execution 11
2.2 The CM Fortran Compiler 12
2.2.1 PE Code Blocks 12
2.2.2 Parallel Computation and Parallel Communication 14
2.2.3 The Compilation Process 15
2.2.4 Useful Compiler Switches 17
2.2.5 Controlling the Compilation Process 18

Copyright © 1994 Thinking Machines Corporation

2.2 The CM Fortran Compiler *(continued)*
2.2.6 Generating a Listing File 19
2.2.7 Generating a FORALL Transformation Printout 20
2.2.8 Enabling the CM Fortran Code Optimizer 20
2.2.9 Compiling Code for Debugging and Performance Analysis . . 21
2.2.10 Array Padding and Default Axis Ordering 21

Chapter 3 CM Fortran Timing and Analysis Tools 23

3.1 CM Timers, Timesharing and You 24
3.2 The **`CM_timer`** Facility 25
3.2.1 Tips on Timing Parallel Computation and Communication . . 26
3.3 Using Prism to Analyze Performance 27
3.3.1 Prism Performance Data Display 28
3.3.2 Timing of PE Code Blocks 31
3.3.3 Timing of Parallel I/O Operations 31
3.3.4 Prism Restrictions 31
3.3.5 Using Prism and the CM Timer Utility Together 32
3.4 Examining Assembler Source Code 32
3.4.1 Counting Code Blocks 33
3.4.2 Estimating SPARC Cycles 33
3.4.3 Counting Floating-Point Operations 34
3.4.4 WARNING: Don't Misinterpret the Assembler Numbers . . . 34
3.4.5 Sample PE Assembler Output File 34
3.5 Timing Parallel I/O — Example 37

Version 2.1, January 1994
Copyright © 1994 Thinking Machines Corporation

Part II CM Fortran Performance Techniques

Chapter 4 CM Fortran Performance Aspects **41**

Chapter 5 Parallel Array Layout **43**

5.1 Array Dimensions and Array Layouts 43

5.2 Controlling Array Layouts 44

5.3 The Canonical Array Layout 45
5.3.1 The Array Subgrid 45
5.3.2 The Physical Grid 45
5.3.3 Viewing Array Geometries 46
5.3.4 Garbage Element Padding 48
5.3.5 VU Vector Length Padding 49
5.3.6 Seeing Array Padding 51
5.3.7 Physical and Subgrid Axis Ordering 52
5.3.8 The Complete Canonical Array Layout 55
5.3.9 Send Addresses 55
5.3.10 Canonical Layout on CMs without Vector Units 57

5.4 Controlling Array Layout: Specifying Axis Ordering and Weights 59
5.4.1 Serial Axes 59
5.4.2 Serial Axes and Array Layouts 61
5.4.3 Weighting Axes 62

5.5 Specifying Detailed Array Layouts 63
5.5.1 IMPORTANT: Detailed Layouts Can Restrict Flexibility 63
5.5.2 Using Detailed Layout Axis Descriptors 64
5.5.3 Axis Masks (Processor Masks) 65
5.5.4 Example 67
5.5.5 Vector Unit Performance Note 68
5.5.6 Serial Axes via Axis Masks 69
5.5.7 Axis Mask Restrictions 69
5.5.8 Using **`CMF_ALLOCATE_DETAILED_ARRAY`** Procedure 70
5.5.9 Example: Locating an Array Element 70

5.6 Performance Issues 72
5.6.1 Effect of Subgrid Length and Physical Grid 72
5.6.2 Effect of Serial Axes 73
5.6.3 Effect of Garbage Elements 74
5.6.4 Geometries of Array Arguments 74

5.7 Querying an Array's Layout 75

Copyright © 1994 Thinking Machines Corporation

Chapter 6 Array Alignment and Aliasing **77**

6.1 Using Array Alignment 77
6.1.1 WARNING — Use **ALIGN** With Caution! 77
6.1.2 Default Alignments 78
Conformable Arrays 78
Dynamic and Assumed-Shape Arrays 78
Nonconformable Arrays 79
Nonconformable Arrays with Serial Dimensions 79

6.2 The **ALIGN** Directive 80
6.2.1 Multiple Source Arrays 81
6.2.2 Multidimensional Source Arrays 82
6.2.3 Aligning Different-Sized Dimensions 82
6.2.4 Alignment Limitations 84

6.3 Benefits and Costs of Explicit Alignment 84
6.3.1 Memory Use 85
6.3.2 Subgrid Looping 85
6.3.3 Communication Costs 85

6.4 Dynamic Array Alignment 86
6.4.1 The Compiler's View of Dynamic Arrays 86
6.4.2 **ALIGN** Directives and Dynamic Arrays 88
Aligning Dynamic Arrays with Non-Dynamic Arrays 88
Aligning Dynamic Arrays with Other Dynamic Arrays . . . 89
6.4.3 Using **ALIGN** Effectively with Dynamic Arrays 89
Align Multiple Arrays 89
Avoid Using Allocatables Unnecessarily 90
Be Careful! 90
6.4.4 Compile-Time Shape Checking 91
Run-Time Geometry Checking 93

6.5 Array Aliasing 93
6.5.1 Array Aliasing: A Quick Overview 94
6.5.2 Restrictions and Caveats 96

Chapter 7 Parallel Computation **99**

7.1 PE Code Blocks 99
7.1.1 How PE Code Blocks Are Invoked 100
7.1.2 Understanding Costs in PE Code Blocks 101
7.1.3 Example 102
7.1.4 Timing PE Code Blocks 103
Counting Flops 104
Assessing Computation Performance 105

Version 2.1, January 1994
Copyright © 1994 Thinking Machines Corporation

7.2 Constructing PE Code Blocks Properly 106
7.2.1 Compiler Optimization of Code Blocks 107
7.2.2 Things That Break Code Blocks 107
Flow Control 107
Function Calls 107
Contextualization 107
Parallel Communication Operations 108
Differing Array Geometries in Expressions 108
Parallel Reductions and Accessing Elements of Arrays . . . 109
Any Parallel I/O Operations 109
Compiling with the **`-g`** Debugging Option 109
7.3 CM Fortran Compiler Serial Optimizations 109
7.3.1 Recognizing Serial Dimensions 109
7.3.2 No-Motion Detection 110
7.3.3 Better Handling of Context 112
7.3.4 Nested and Flattened Loops 112
7.3.5 Simple Dependence Analysis 113
7.3.6 Contiguous Serial Sections Passed in Place 114
7.3.7 Gathers/Scatters with **`FORALL`** 117
7.3.8 Serial Spreads with **`FORALL`** or **`SPREAD`** 117
7.4 Other Optimizations 118
7.4.1 Single-Precision Memory Accesses in Vector Unit Code 118

Chapter 8 Parallel Communication 119

8.1 Compilation of Communication 120
8.1.1 Location Analysis 120
8.1.2 Location Equivalence 121
8.1.3 Combining of Locations 122
8.1.4 Location Comparison Restrictions 123
8.1.5 Communication Operator Selection 124
NEWS 124
Cross Geometry Move (VP Move) 124
Block Move 125
Sends and Gets 125
8.1.6 Comparing Communication Functions 126
8.2 Intrinsics 127
8.2.1 Spreads 127
8.2.2 Reductions 128
8.2.3 Shifts 128
8.3 Intrinsics That Generate Communication 129

Copyright © 1994 Thinking Machines Corporation

8.4 Array Argument Passing 130
8.4.1 Noncanonical Arrays as Arguments 130
8.4.2 Array Sections as Arguments 130
8.5 **FORALL** Optimizations 131
8.5.1 Operations Now Optimized 132
Gathers/Scatters on Serial Dimensions 132
Spread Operations on Serial Dimensions 133
Scan Operations 133
Comparison with Utility Library **CMF_SCAN_** 134
Send-with-Combiner Operations 137
8.5.2 Expressions Now Parallelized 138
8.5.3 Temporary Restrictions 138
8.5.4 Permanent Restrictions 139
8.5.5 Using Alternative Forms of **FORALL** 140
8.6 Simplifying Communication — Summary 141

Chapter 9 Memory Management 143

9.1 CM Memory Management 143
9.1.1 Scalar and Parallel Memory 143
9.1.2 The Parallel Stack and Heap 144
9.1.3 CM Memory Mapping 144
9.1.4 CM Fortran Array Types 146
9.2 Compiler Temporary Arrays 147
9.2.1 Communication-Computation Temporary Arrays 147
9.2.2 Common Subexpression (CSE) Temporary Arrays 150
9.3 Running Out of Memory 151
9.4 Dealing with Memory Errors 155
9.4.1 Find Out Where You Ran Out 155
9.4.2 Check the Size of Your Arrays 155
9.4.3 Try to Reduce Explicit Use of Memory 156
9.4.4 Reduce Use of Temporary Arrays 156
9.5 CMOST and Memory Allocation 158

Version 2.1, January 1994
Copyright © 1994 Thinking Machines Corporation

Chapter 10 Miscellaneous Performance Issues 161

10.1 The CMSSL Library 161
10.1.1 CMSSL Communication Routines 161
Polyshift Operation 162
All-to-All Operations 163
Matrix Transpose 163
Sparse Gather/Scatter 163
Vector Move (Extract and Deposit) 164
Array Permutations 164
Communication Compiler 165
10.1.2 Additional CMSSL Optimizations 165

10.2 Alternate Programming Models 165
10.2.1 Nodal Programming 166
10.2.2 Global/Local Programming 166
10.2.3 Performance Effects 167

Appendixes

Appendix A CMRTS and CM Memory Allocation 171

A.1 The CM Run-Time System (CMRTS) 171
A.1.1 Arrays in the CMRTS 172
A.1.2 An Example of a CMRTS Array 174
A.1.3 CMRTS Data Structures 178
`CMRT_desc_t` 178
`CMRT_array_geometry_t` 179
`CMRT_machine_geometry_t` 182
`CMCOM_axis_descriptor` 184

A.2 CMRTS Parallel Memory Allocation 187
A.2.1 Standard CMRTS Memory Allocation Functions 187
A.2.2 Node-Level Stack Operations 189

Index

Index 191

Copyright © 1994 Thinking Machines Corporation