Contents

Preface *page* xi

1 Introduction 1
- 1.1 Concurrency 2
- 1.2 Real-time systems 3
- 1.3 Ada's time and clock facilities 6
- 1.4 Summary 13
- 1.5 Further reading 13

2 The nature and uses of concurrent programming 15
- 2.1 Uses of concurrent programming 17
- 2.2 Program entities 18
- 2.3 Process representation 20
- 2.4 A simple embedded system 21
- 2.5 Summary 30
- 2.6 Further reading 30

3 Inter-process communication 31
- 3.1 Data communication 32
- 3.2 Synchronisation 33
- 3.3 Deadlocks and indefinite postponements 34
- 3.4 System performance, correctness and reliability 36
- 3.5 Dining philosophers problem 38
- 3.6 Shared variables and protected variables 39
- 3.7 Semaphores 41
- 3.8 Monitors 44
- 3.9 Message-based communication 48
- 3.10 Summary 53
- 3.11 Further reading 54

4 Task types and objects 55
4.1 Task creation 57
4.2 Task activation, execution, finalisation and termination 65
4.3 Task hierarchies 70
4.4 Task identification 75
4.5 Task creation, communication and synchronisation within task finalisation 77
4.6 Summary 77

5 The rendezvous 79
5.1 The basic model 79
5.2 The entry statement 81
5.3 The accept statement 83
5.4 The `Count` attribute 88
5.5 Entry families 88
5.6 Three-way synchronisation 90
5.7 Private entries 92
5.8 Exceptions and the rendezvous 93
5.9 Task states 94
5.10 Summary 94

6 The select statement and the rendezvous 97
6.1 Selective accept 97
6.2 Guarded alternatives 101
6.3 Delay alternative 103
6.4 The else part 107
6.5 The correct use of guards 109
6.6 The terminate alternative 111
6.7 The exception `Program_Error` 116
6.8 Summary of the selective accept statement 118
6.9 Conditional and timed entry calls 118
6.10 Mutual exclusion and deadlocks 121
6.11 The dining philosophers 124
6.12 Task states 127
6.13 Summary 127

7 Protected objects and data-oriented communication 129
7.1 Protected objects 129
7.2 Mutual exclusion 131
7.3 Condition synchronisation 133
7.4 Entry calls and barriers 135
7.5 Private entries and entry families 139

7.6 Restrictions on protected objects 142
7.7 Access variables and protected types 144
7.8 Elaboration, finalisation and exceptions 146
7.9 Shared data 147
7.10 The readers and writers problem 148
7.11 The specification of synchronisation agents 151
7.12 Shared variables 152
7.13 Volatile and atomic data 156
7.14 Task states 160
7.15 Summary 161

8 Avoidance synchronisation and the requeue facility 163
8.1 The need for requeue 163
8.2 Semantics of requeue 175
8.3 Requeuing to other entities 179
8.4 Real-time solutions to the resource control problem 183
8.5 Entry families and server tasks 186
8.6 Extended example 190
8.7 Task states 193
8.8 Summary 194

9 Exceptions, abort and asynchronous transfer of control 195
9.1 Exceptions 195
9.2 The abort statement 198
9.3 Asynchronous transfer of control 200
9.4 Understanding the asynchronous select statement 212
9.5 A robust readers and writers algorithm 217
9.6 Task states 221
9.7 Summary 221

10 Object-oriented programming and tasking 223
10.1 The Ada 2005 OOP model 224
10.2 Tasks and interfaces 231
10.3 Protected types and interfaces 239
10.4 Synchronized interfaces 244
10.5 Summary 246
10.6 Further reading 246

11 Concurrency utilities 247
11.1 Communication and synchronisation abstractions 248
11.2 Semaphores 248
11.3 Locks 257
11.4 Signals 263

11.5 Event variables 264
11.6 Buffers 266
11.7 Blackboards 268
11.8 Broadcasts 269
11.9 Barriers 276
11.10 Concurrent execution abstractions 277
11.11 Callables and futures 278
11.12 Executors 280
11.13 Completion services 284
11.14 Image processing example revisited 288
11.15 Summary 291

12 Tasking and systems programming 293
12.1 Device driving and interrupt handling 296
12.2 Model of interrupts 300
12.3 Task identifiers 311
12.4 Task attributes 313
12.5 Summary 316
12.6 Further reading 316

13 Scheduling real-time systems – fixed priority dispatching 317
13.1 Scheduling 317
13.2 Fixed priority dispatching 319
13.3 Priority ceiling locking 322
13.4 Entry queue policies 327
13.5 Active priorities and dispatching policies 327
13.6 Summary 329
13.7 Further reading 329

14 Scheduling real-time systems – other dispatching facilities 331
14.1 Non-preemptive dispatching 331
14.2 Round-robin dispatching 332
14.3 Earliest deadline first dispatching 335
14.4 Mixed scheduling 347
14.5 Dynamic priorities 348
14.6 Synchronous and asynchronous task control 354
14.7 Summary 359
14.8 Further reading 359

15 Timing events and execution-time control 361
15.1 Events and event handling 361
15.2 Timing events 362
15.3 Dual priority scheduling 366

15.4 Execution-time clocks 369
15.5 Execution-time timers 371
15.6 Group budgets 374
15.7 Task termination events 387
15.8 Summary 389
15.9 Further reading 389

16 Real-time utilities 391
16.1 Real-time task state 393
16.2 Real-time task release mechanisms 395
16.3 Periodic release mechanisms 397
16.4 Sporadic release mechanisms 405
16.5 Aperiodic release mechanisms and execution-time servers 407
16.6 Real-time tasks 415
16.7 The cruise control system example 419
16.8 Summary 432

17 Restrictions, metrics and the Ravenscar profile 433
17.1 Restricted tasking and other language features 433
17.2 The Ravenscar profile 436
17.3 Partition elaboration control 439
17.4 Examples of the use of the Ravenscar profile 440
17.5 Metrics and optimisations 448
17.6 Summary 449
17.7 Further reading 450

18 Conclusion 451
18.1 Support for concurrency 452
18.2 Support for real-time 452
18.3 New to Ada 2005 453
18.4 Outstanding issues and the future 453

References 455
Index 457