

Real-Time Systems and Programming Languages

Ada, Real-Time Java and C/Real-Time POSIX

Fourth Edition

Alan Burns and Andy Wellings

University of York



An imprint of Pearson Education

Harlow, England • London • New York • Boston • San Francisco • Toronto • Sydney • Singapore • Hong Kong
Tokyo • Seoul • Taipei • New Delhi • Cape Town • Madrid • Mexico City • Amsterdam • Munich • Paris • Milan

Contents

Preface	xiii
1 Introduction to real-time systems	1
1.1 Definition of a real-time system	2
1.2 Examples of real-time systems	4
1.3 Characteristics of real-time systems	9
1.4 Development cycle for real-time systems	15
1.5 Languages for programming real-time systems	20
Summary	23
Further reading	25
Exercises	26
2 Reliability and fault tolerance	27
2.1 Reliability, failure and faults	28
2.2 Failure modes	31
2.3 Fault prevention and fault tolerance	33
2.4 <i>N</i> -version programming	36
2.5 Software dynamic redundancy	41
2.6 The recovery block approach to software fault tolerance	46
2.7 A comparison between <i>N</i> -version programming and recovery blocks ..	49
2.8 Dynamic redundancy and exceptions	50
2.9 Measuring and predicting the reliability of software	52
2.10 Safety, reliability and dependability	53
Summary	55
Further reading	56
Exercises	57
3 Exceptions and exception handling	59
3.1 Exception handling in older real-time languages	60
3.2 Modern exception handling	62
3.3 Exception handling in Ada, Java and C	69
3.4 Recovery blocks and exceptions	85

Summary	88
Further reading	89
Exercises	89
4 Concurrent programming	95
4.1 Processes and tasks/threads	96
4.2 Concurrent execution	99
4.3 Task representation	103
4.4 Concurrent execution in Ada	105
4.5 Concurrent execution in Java	111
4.6 Concurrent execution in C/Real-Time POSIX	116
4.7 Multiprocessor and distributed systems	121
4.8 A simple embedded system	125
4.9 Language-supported versus operating-system-supported concurrency	131
Summary	132
Further reading	133
Exercises	133
5 Shared variable-based synchronization and communication	137
5.1 Mutual exclusion and condition synchronization	138
5.2 Busy waiting	139
5.3 Suspend and resume	142
5.4 Semaphores	145
5.5 Conditional critical regions	156
5.6 Monitors	157
5.7 Mutexes and condition variables in C/Real-Time POSIX	160
5.8 Protected objects in Ada	163
5.9 Synchronized methods in Java	171
5.10 Shared memory multiprocessors	179
5.11 Simple embedded system revisited	183
Summary	185
Further reading	186
Exercises	187
6 Message-based synchronization and communication	193
6.1 Process synchronization	193
6.2 Task naming and message structure	195
6.3 Message passing in Ada	196
6.4 Selective waiting	201
6.5 The Ada <code>select</code> statement	202
6.6 Non-determinism, selective waiting and synchronization primitives	205
6.7 C/Real-Time POSIX message queues	206
6.8 Distributed systems	210
6.9 Simple embedded system revisited	219
Summary	220
Further reading	221
Exercises	222

7	Atomic actions, concurrent tasks and reliability	227
7.1	Atomic actions	228
7.2	Atomic actions in C/Real-Time POSIX, Ada and Real-Time Java ...	232
7.3	Recoverable atomic actions	240
7.4	Asynchronous notification	245
7.5	Asynchronous notification in C/Real-Time POSIX	247
7.6	Asynchronous notification in Ada	255
7.7	Asynchronous notification in Real-Time Java	266
	Summary	278
	Further reading	280
	Exercises	280
8	Resource control	285
8.1	Resource control and atomic actions	286
8.2	Resource management	286
8.3	Expressive power and ease of use	287
8.4	The requeue facility	296
8.5	Asymmetric naming and security	302
8.6	Resource usage	303
8.7	Deadlock	304
	Summary	304
	Further reading	305
	Exercises	305
9	Real-time facilities	307
9.1	The notion of time	307
9.2	Access to a clock	309
9.3	Delaying a task	317
9.4	Programming timeouts	320
9.5	Specifying timing requirements	326
9.6	Temporal scopes	328
	Summary	332
	Further reading	333
	Exercises	333
10	Programming real-time abstractions	335
10.1	Real-time tasks	336
10.2	Programming periodic activities	338
10.3	Programming aperiodic and sporadic activities	341
10.4	The role of real-time events and their handlers	344
10.5	Controlling input and output jitter	349
10.6	Other approaches for supporting temporal scopes	356
	Summary	363
	Further reading	364
	Exercises	364

11 Scheduling real-time systems	365
11.1 The cyclic executive approach	366
11.2 Task-based scheduling	367
11.3 Fixed-priority scheduling (FPS)	370
11.4 Utilization-based schedulability tests for FPS	371
11.5 Response time analysis (RTA) for FPS	374
11.6 Sporadic and aperiodic tasks	378
11.7 Task systems with $D < T$	380
11.8 Task interactions and blocking	382
11.9 Priority ceiling protocols	386
11.10 An extendible task model for FPS	390
11.11 Earliest deadline first (EDF) scheduling	400
11.12 Dynamic systems and online analysis	405
11.13 Worst-case execution time	407
11.14 Multiprocessor scheduling	408
11.15 Scheduling for power-aware systems	413
11.16 Incorporating system overheads	414
Summary	419
Further reading	420
Exercises	421
12 Programming schedulable systems	425
12.1 Programming cyclic executives	425
12.2 Programming preemptive priority-based systems	426
12.3 Ada and fixed-priority scheduling	427
12.4 The Ada Ravenscar profile	430
12.5 Dynamic priorities and other Ada facilities	434
12.6 C/Real-Time POSIX and fixed-priority scheduling	436
12.7 Real-Time Java and fixed-priority scheduling	438
12.8 Programming EDF systems	443
12.9 Mixed scheduling	453
Summary	454
Further reading	454
Exercises	455
13 Tolerating timing faults	457
13.1 Dynamic redundancy and timing faults	457
13.2 Deadline miss detection	459
13.3 Overrun of worst-case execution time	467
13.4 Overrun of sporadic events	471
13.5 Overrun of resource usage	474
13.6 Damage confinement	475
13.7 Error recovery	485
Summary	492
Further reading	493
Exercises	493

14 Low-level programming	495
14.1 Hardware input/output mechanisms	495
14.2 Language requirements	502
14.3 Ada	504
14.4 Real-Time Java	514
14.5 C and older real-time languages	517
14.6 Scheduling device drivers	519
14.7 Memory management	521
Summary	527
Further reading	528
Exercises	528
15 Mine control case study	533
15.1 Mine drainage	533
15.2 The HRT-HOOD design method	538
15.3 The logical architecture design	539
15.4 The physical architecture design	545
15.5 Translation to Ada	546
15.6 Translation to Real-Time Java	564
15.7 Fault tolerance and distribution	570
Summary	572
Further reading	572
Exercises	573
16 Conclusions	575
References	579
Index	587

Supporting resources

Visit www.pearsoned.co.uk/burns to find valuable online resources

For instructors

- Solutions to exercises
- Example examination questions
- Code fragments
- PowerPoint slides

For more information please contact your local Pearson Education sales representative or visit www.pearsoned.co.uk/burns