

lead to disaster. Thus, this third edition includes a new chapter on *cloud and fog computing* that argues for the *fog architectural style* to decouple the *world of cloud* from the *world of real-time embedded systems*. The second major change is the rewritten chapter on *real-time networks* that now also covers IEEE 802.1 Time-Sensitive Networking (TSN), the incorporation of the time-triggered paradigm in the mainstream IT networking set of standards. All other chapters have been revised and updated. This third edition includes approximately 80 new references in total.

Since the publication of the first edition, a visible paradigm shift from the event-triggered to the time-triggered design methodology for dependable distributed real-time systems has taken place in a number of applications.

It is assumed that the reader of this book has a background in basic computer science or computer engineering or has some practical experience in the design or implementation of embedded systems.

The glossary, provided at the end, is an integral part of this book, providing definitions for many of the technical terms used throughout. If the reader is not sure about the meaning of a term, they are advised to refer to the glossary.

Acknowledgments

It is impossible to name all students, colleagues from industry, and fellow scientists who have contributed to this third edition of the book by asking intelligent questions or making constructive comments over the last decade—thanks to all of you.

Vienna, Austria

Hermann Kopetz
Wilfried Steiner

May 2022

Contents

1	The Real-Time Environment	1
1.1	When Is a Computer System Real-Time?	2
1.2	Functional Requirements	3
1.2.1	Data Collection	3
1.2.2	Direct Digital Control	6
1.2.3	Man-Machine Interaction	6
1.3	Temporal Requirements	7
1.3.1	Where Do Temporal Requirements Come From?	7
1.3.2	Minimal Latency Jitter	9
1.3.3	Minimal Error-Detection Latency	10
1.4	Dependability Requirements	10
1.4.1	Reliability	10
1.4.2	Safety	11
1.4.3	Maintainability	12
1.4.4	Availability	12
1.4.5	Security	13
1.5	Classification of Real-Time Systems	13
1.5.1	Hard Real-Time System Versus Soft Real-Time System	14
1.5.2	Fail-Safe Versus Fail-Operational	16
1.5.3	Guaranteed Response Versus Best Effort	16
1.5.4	Resource-Adequate Versus Resource-Inadequate	17
1.5.5	Event-Triggered Versus Time-Triggered	17
1.6	The Real-Time System Market	18
1.6.1	Embedded Real-Time Systems	18
1.6.2	Plant Automation Systems	21
1.6.3	Multimedia Systems	22
1.7	Examples of Real-Time Systems	23
1.7.1	Controlling the Flow in a Pipe	23
1.7.2	Engine Control	24
1.7.3	Rolling Mill	25

Bibliographic Notes	27
Review Questions and Problems	28
2 Simplicity	31
2.1 Cognition	32
2.1.1 Problem-Solving	32
2.1.2 Definition of a Concept	34
2.1.3 Cognitive Complexity	35
2.1.4 Simplification Strategies	37
2.2 The Conceptual Landscape	38
2.2.1 Concept Formation	38
2.2.2 Scientific Concepts	40
2.2.3 The Concept of a Message	41
2.2.4 Semantic Content of a Variable	42
2.3 The Essence of Model Building	43
2.3.1 Purpose and Viewpoint	44
2.3.2 The Grand Challenge	46
2.4 Emergence	46
2.4.1 Irreducibility	47
2.4.2 Prior and Derived Properties	47
2.4.3 Complex Systems	48
2.5 How Can We Achieve Simplicity?	51
Bibliographic Notes	53
Points to Remember	53
Review Questions and Problems	55
3 Global Time	57
3.1 Time and Order	58
3.1.1 Different Orders	58
3.1.2 Clocks	59
3.1.3 Precision and Accuracy	61
3.1.4 Time Standards	63
3.2 Time Measurement	64
3.2.1 Global Time	64
3.2.2 Interval Measurement	66
3.2.3 π/Δ -Precedence	66
3.2.4 Fundamental Limits of Time Measurement	68
3.3 Dense Time Versus Sparse Time	68
3.3.1 Dense Time Base	69
3.3.2 Sparse Time Base	70
3.3.3 Space-Time Lattice	71
3.3.4 Cyclic Representation of Time	72
3.4 Internal Clock Synchronization	73
3.4.1 The Synchronization Condition	73
3.4.2 Central Master Synchronization	75
3.4.3 Fault-Tolerant Synchronization Algorithms	75

3.4.4 State Correction Versus Rate Correction	79
3.5 External Clock Synchronization	79
3.5.1 External Time Sources	79
3.5.2 Time Gateway	81
3.5.3 Time Formats	81
Points to Remember	82
Bibliographic Notes	83
Review Questions and Problems	83
4 Real-Time (RT) Model	87
4.1 Model Outline	88
4.1.1 Components and Messages	88
4.1.2 Cluster of Components	89
4.1.3 Temporal Control Versus Logical Control	90
4.1.4 Event-Triggered Control Versus Time-Triggered Control	92
4.2 Component State	92
4.2.1 Definition of State	93
4.2.2 The Pocket Calculator Example	93
4.2.3 Ground State	95
4.2.4 Database Components	96
4.3 The Message Concept	97
4.3.1 Message Structure	97
4.3.2 Event Information Versus State Information	98
4.3.3 Event-Triggered (ET) Message	99
4.3.4 Time-Triggered (TT) Message	99
4.4 Component Interfaces	100
4.4.1 Interface Characterization	101
4.4.2 Linking Interface (LIF)	102
4.4.3 Technology-Independent Interface (TII)	103
4.4.4 Technology-Dependent Interface (TDI)	103
4.4.5 Local Interfaces	103
4.5 Gateway Component	104
4.5.1 Property Mismatches	105
4.5.2 LIF Versus Local Interface of a Gateway Component	105
4.5.3 Standardized Message Interface	107
4.6 Linking Interface Specification	107
4.6.1 Transport Specification	108
4.6.2 Operational Specification	109
4.6.3 Meta-Level Specification	110
4.7 Component Integration	111
4.7.1 Principles of Composability	111
4.7.2 Integration Viewpoints	112
4.7.3 System of Systems	113
Bibliographic Notes	117
Review Questions and Problems	117

5	Temporal Relations	119
5.1	Real-Time Entities	120
5.1.1	Sphere of Control	120
5.1.2	Discrete and Continuous Real-Time Entities	121
5.2	Observations	121
5.2.1	Untimed Observation	121
5.2.2	Indirect Observation	122
5.2.3	State Observation	122
5.2.4	Event Observation	123
5.3	Real-Time Images and Real-Time Objects	124
5.3.1	Real-Time Images	124
5.3.2	Real-Time Objects	124
5.4	Temporal Accuracy	125
5.4.1	Definition	125
5.4.2	Classification of Real-Time Images	127
5.4.3	State Estimation	129
5.4.4	Composability Considerations	130
5.5	Permanence and Idempotency	130
5.5.1	Permanence	130
5.5.2	Duration of the Action Delay	132
5.5.3	Accuracy Interval Versus Action Delay	133
5.5.4	Idempotency	133
5.6	Determinism	133
5.6.1	Definition of Determinism	134
5.6.2	Consistent Initial States	136
5.6.3	Nondeterministic Design Constructs (NDDCs)	137
5.6.4	Recovery of Determinism	138
	Points to Remember	139
	Bibliographic Notes	140
	Review Questions and Problems	140
6	Dependability	143
6.1	Basic Concepts	144
6.1.1	Faults	145
6.1.2	Errors	146
6.1.3	Failures	147
6.2	Information Security	150
6.2.1	Secure Information Flow	151
6.2.2	Security Threats	152
6.2.3	Cryptographic Methods	154
6.2.4	Network Authentication	157
6.2.5	Protection of Real-Time Control Data	157
6.3	Anomaly Detection	158
6.3.1	What Is an Anomaly?	158
6.3.2	Failure Detection	160

6.3.3	Error Detection	161
6.4	Fault Tolerance	162
6.4.1	Fault Hypotheses	163
6.4.2	Fault-Tolerant Unit	164
6.4.3	The Membership Service	167
6.5	Robustness and Resilience	168
6.5.1	The Concept of Robustness	168
6.5.2	The Concept of Resilience	170
6.6	Component Reintegration	170
6.6.1	Finding a Reintegration Point	170
6.6.2	Minimizing the Ground State	171
6.6.3	Component Restart	172
	Points to Remember	173
	Bibliographic Notes	174
	Review Questions and Problems	174
7	Real-Time Communication	177
7.1	Requirements	178
7.1.1	Timeliness	178
7.1.2	Dependability and Security	179
7.1.3	Flexibility	180
7.1.4	Communication Bandwidth and Cost Efficiency	181
7.2	Design Principles and Pitfalls	181
7.2.1	Real-Time Network Model	182
7.2.2	Message Types	182
7.2.3	Flow Control	183
7.2.4	Design Limitations	184
7.2.5	Design Pitfalls	186
7.3	Event-Triggered Communication	187
7.3.1	CAN	188
7.3.2	Ethernet	189
7.4	Rate-Constrained Communication	190
7.4.1	Avionics Full-Duplex Switched Ethernet (AFDX): ARINC 664-p7	191
7.4.2	Audio/Video Bridging: IEEE 802.1 AVB	192
7.5	Time-Triggered Communication	192
7.5.1	TTP	194
7.5.2	TTEthernet	195
7.5.3	Time-Sensitive Networking: IEEE 802.1 TSN	197
	Points to Remember	198
	Bibliographic Notes	200
	Review Questions and Problems	200

8	Power and Energy Awareness	201
8.1	Power and Energy	202
8.1.1	Basic Concepts	202
8.1.2	Energy Estimation	204
8.1.3	Thermal Effects and Reliability	208
8.2	Hardware Power Reduction Techniques	209
8.2.1	Device Scaling	209
8.2.2	Low-Power Hardware Design	211
8.2.3	Voltage and Frequency Scaling	211
8.2.4	Sub-threshold Logic	212
8.3	System Architecture	213
8.3.1	Technology-Agnostic Design	213
8.3.2	Pollack's Rule	214
8.3.3	Power Gating	216
8.3.4	Real Time Versus Execution Time	217
8.4	Software Techniques	218
8.4.1	System Software	218
8.4.2	Application Software	219
8.4.3	Software Tools	219
	Points to Remember	198
	Bibliographic Notes	200
	Review Questions and Problems	200
9	Real-Time Operating Systems	223
9.1	Inter-Component Communication	224
9.1.1	Technology-Independent Interface (TII)	224
9.1.2	Linking Interface (LIF)	225
9.1.3	Technology-Dependent Interface (TDI)	225
9.1.4	Generic Middleware (GM)	225
9.2	Task Management	226
9.2.1	Simple Tasks	226
9.2.2	Trigger Tasks	228
9.2.3	Complex Tasks	229
9.3	The Dual Role of Time	229
9.3.1	Time as Data	230
9.3.2	Time as Control	231
9.4	Inter-Task Interactions	232
9.4.1	Coordinated Static Schedules	232
9.4.2	The Non-blocking Write (NBW) Protocol	233
9.4.3	Semaphore Operations	234
9.5	Process Input/Output	234
9.5.1	Analog Input/Output	234
9.5.2	Digital Input/Output	235
9.5.3	Interrupts	236

9.5.4	Fault-Tolerant Actuators	238
9.5.5	Intelligent Instrumentation	239
9.5.6	Physical Installation	240
9.6	Agreement Protocols	241
9.6.1	Raw Data, Measured Data, and Agreed Data	241
9.6.2	Syntactic Agreement	242
9.6.3	Semantic Agreement	242
9.7	Error Detection	243
9.7.1	Monitoring Task Execution Times	243
9.7.2	Monitoring Interrupts	243
9.7.3	Double Execution of Tasks	243
9.7.4	Watchdogs	244
	Points to Remember	245
	Bibliographic Notes	245
	Review Questions and Problems	246
10	Real-Time Scheduling	247
10.1	The Scheduling Problem	248
10.1.1	Classification of Scheduling Algorithms	248
10.1.2	Schedulability Test	249
10.1.3	The Adversary Argument	250
10.2	Worst-Case Execution Time	251
10.2.1	WCET of Simple Tasks	252
10.2.2	WCET of Complex Tasks	254
10.2.3	Anytime Algorithms	255
10.2.4	State of Practice	255
10.3	Static Scheduling	256
10.3.1	Static Scheduling Viewed as a Search	257
10.3.2	Increasing the Flexibility in Static Schedules	258
10.4	Dynamic Scheduling	260
10.4.1	Scheduling Independent Tasks	260
10.4.2	Scheduling Dependent Tasks	262
10.5	Alternative Scheduling Strategies	263
10.5.1	Scheduling in Distributed Systems	263
10.5.2	Feedback Scheduling	264
	Points to Remember	265
	Bibliographic Notes	266
	Review Questions and Problems	267
11	System Design	269
11.1	System Design	270
11.1.1	The Design Process	270
11.1.2	The Role of Constraints	272
11.1.3	System Design Versus Software Design	272

11.2	Design Phases	274
11.2.1	Purpose Analysis	275
11.2.2	Requirements Capture	275
11.2.3	Architecture Design	276
11.2.4	Design of Components	277
11.3	Design Styles	277
11.3.1	Model-Based Design	277
11.3.2	Component-Based Design	279
11.3.3	Architecture Design Languages	280
11.3.4	Test of a Decomposition	281
11.4	Design of Safety-Critical Systems	283
11.4.1	What Is Safety?	284
11.4.2	Safety Analysis	285
11.4.3	Safety Case	288
11.4.4	Safety Standards	291
11.5	Design Diversity	293
11.5.1	Diverse Software Versions	294
11.5.2	An Example of a Fail-Safe System	295
11.5.3	Multilevel System	296
11.6	Design for Maintainability	296
11.6.1	Cost of Maintenance	297
11.6.2	Maintenance Strategy	298
11.6.3	Software Maintenance	299
11.7	The Time-Triggered Architecture	300
11.7.1	Principle of a Consistent Global Time	300
11.7.2	Principle of Component Orientation	301
11.7.3	Principle of Coherent Communication	302
11.7.4	Principle of Fault Tolerance	303
	Bibliographic Notes	303
	Points to Remember	303
	Review Questions and Problems	305
12	Validation	307
12.1	Validation Versus Verification	308
12.2	Testing Challenges	309
12.2.1	Design for Testability	310
12.2.2	Test Data Selection	310
12.2.3	Test Oracle	312
12.2.4	System Evolution and Technology Readiness Levels (TRLs)	313
12.3	Testing of Component-Based Systems	314
12.3.1	Component Provider	314
12.3.2	Component User	314
12.3.3	Communicating Components	315

12.4	Formal Methods	316
12.4.1	Formal Methods in the Real World	316
12.4.2	Classification of Formal Methods	317
12.4.3	Benefits of Formal Methods	317
12.4.4	Model Checking	319
12.5	Fault Injection	319
12.5.1	Software-Implemented Fault Injection	320
12.5.2	Physical Fault Injection	320
12.5.3	Sensor and Actuator Failures	321
	Points to Remember	322
	Bibliographic Notes	323
	Review Questions and Problems	323
13	Internet of Things	325
13.1	The Vision of the Internet of Things (IoT)	326
13.2	Drivers for an IoT	327
13.2.1	Uniformity of Access	327
13.2.2	Logistics	327
13.2.3	Energy Savings	328
13.2.4	Physical Security and Safety	328
13.2.5	Industrial	329
13.2.6	Medical	329
13.2.7	Lifestyle	329
13.3	Technical Issues of the IoT	329
13.3.1	Internet Integration	329
13.3.2	Naming and Identification	330
13.3.3	Near-Field Communication	331
13.3.4	IoT Device Capabilities Versus Cloud Computing	332
13.3.5	Autonomic Components	332
13.4	RFID Technology	333
13.4.1	Overview	334
13.4.2	The Electronic Product Code (EPC)	334
13.4.3	RFID Tags	335
13.4.4	RFID Readers	336
13.4.5	RFID Security	336
13.5	Wireless Sensor Networks (WSN)	338
	Points to Remember	339
	Bibliographic Notes	340
	Review Questions and Problems	341
14	Cloud and Fog Computing	343
14.1	Introduction	344
14.2	Characteristics of the Cloud	345

14.3	The Advent of Fog Computing	347
14.3.1	Fog Computing for Distributed Embedded Systems	348
14.3.2	Fog Computing Benefits and Risks	349
14.3.3	General Fog Computing and Comparison to Edge Computing	351
14.4	Selected Cloud and Fog Technologies	351
14.4.1	Resource Pooling	352
14.4.2	Connectivity	356
14.4.3	Configuration	357
14.4.4	System Design Automation	358
14.5	Example Use Cases	358
14.5.1	Cloud Computing-Enabled Use Cases	359
14.5.2	Fog Computing-Enabled Use Cases	360
14.5.3	Nerve	362
	Points to Remember	363
	Bibliographic Notes	364
	Review Questions and Problems	365
Annexes	367
References	383
Index	395

Chapter 1

The Real-Time Environment

Overview

The purpose of this introductory chapter is to describe the environment of real-time computer systems from a number of different perspectives. A solid understanding of the technical and economic factors that characterize a real-time application helps to interpret the demands that the system designer must cope with. The chapter starts with the definition of a real-time system and with a discussion of its functional and nonfunctional requirements. Particular emphasis is placed on the temporal requirements that are derived from the well-understood properties of control applications. The objective of a control algorithm is to drive a process such that a performance criterion is satisfied. Random disturbances occurring in the environment degrade system performance and must be taken into account by the control algorithm. Any additional uncertainty that is introduced into the control loop by the control system itself, e.g., a non-predictable jitter of the control loop, results in a degradation of the quality of control.

In Sects. 1.2, 1.3, 1.4 and 1.5, real-time applications are classified from a number of viewpoints. Special emphasis is placed on the fundamental differences between *hard* and *soft* real-time systems. Because soft real-time systems do not have severe failure modes, a less rigorous approach to their design is often followed. Sometimes resource-inadequate solutions that will not handle the rarely occurring peak-load scenarios are accepted on economic arguments. In a hard real-time application, such an approach is unacceptable because the safety of a design in all specified situations, even if they occur only very rarely, must be demonstrated vis-a-vis a certification agency. In Sect. 1.6, a brief analysis of the real-time system market is carried out with emphasis on the field of embedded real-time systems. An embedded real-time system is a part of a self-contained product, e.g., a television set or an automobile. Embedded real-time systems, also called *cyber-physical systems* (CPS), form the most important market segment for real-time technology and the computer industry in general.