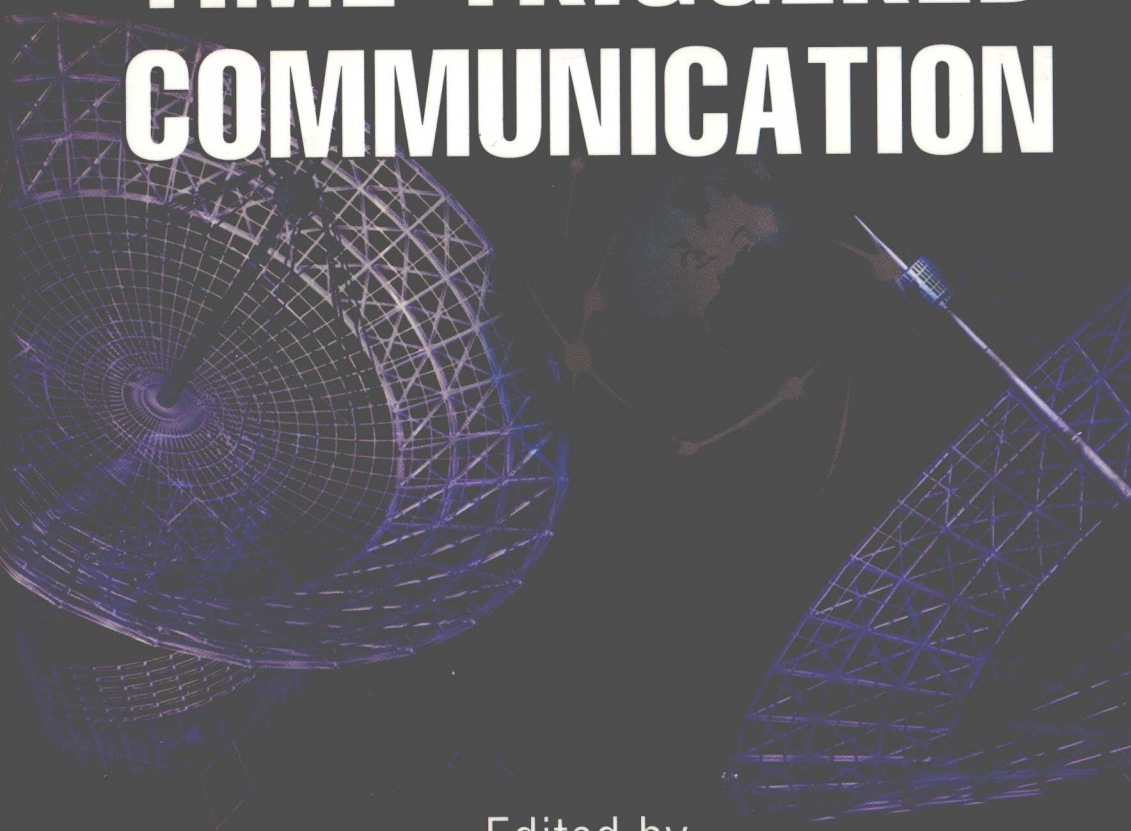
TIME-TRIGGERED COMMUNICATION



Edited by

ROMAN OBERMAISSER

CRC Press
Taylor & Francis Group

TIME-TRIGGERED COMMUNICATION

Edited by

ROMAN OBERMAISSER


CRC Press
Taylor & Francis Group
Boca Raton London New York

CRC Press is an imprint of the
Taylor & Francis Group, an **informa** business

CRC Press
Taylor & Francis Group
6000 Broken Sound Parkway NW, Suite 300
Boca Raton, FL 33487-2742

Printed in the United States of America on acid-free paper
Version Date: 20110629

International Standard Book Number: 978-1-4398-4661-2 (Hardback)

TIME-TRIGGERED COMMUNICATION

Embedded Systems

Series Editor

Richard Zurawski
SA Corporation, San Francisco, California, USA

Communication Architectures for Systems-on-Chip, *edited by José L. Ayala*

Real Time Embedded Systems Design and Analysis with Open-Source Operating Systems, *Ivan Cibrario Bertolotti and Gabriele Manduchi*

Time-Triggered Communication, *edited by Roman Obermaisser*

Editor

Roman Obermaisser is a full professor for embedded systems at the Department of Electrical Engineering and Computer Science of the University of Siegen in Germany. He studied computer sciences at Vienna University of Technology and received his master's degree in 2001. In February 2004, Professor Obermaisser finished his doctoral studies in computer science at Vienna University of Technology with Professor Hermann Kopetz as research advisor. In July 2009, he received the habilitation ("Venia docendi") certificate for Technical Computer Science. He is the author of numerous journal papers, books and conference publications.

Professor Obermaisser has participated in European research projects (e.g., universAAL, DECOS, NextTTA, INDEXYS) and was the technical coordinator of the FP7 research projects GENESYS (GENeric Embedded SYStem Platform) and ACROSS (Artemis Cross-Domain Architecture). He was also a member of the working groups "reference designs/architectures" and "middleware/seamless connectivity" in the European technology platform ARTEMIS, where a roadmap for European research in the area of embedded systems was defined. His leading role in the scientific community is shown through his chairing of and participation in many program committees (e.g., chair of the program committee of the IEEE Symposiums for Object and Component-Oriented Real-Time Distributed Computing, chair of the IEEE Workshop on Architectures and Applications for Mixed-Criticality Systems, chair of the IFIP Workshops on Software Technologies for Future Embedded and Ubiquitous Systems).

Professor Obermaisser's research focuses on system architectures, which provide the scientific and engineering foundation for the construction of embedded systems. The goals of his research are to discover design principles and to develop platform services that enable a component-based development of embedded systems in such a way that the ensuing systems can be built cost-effectively and exhibit key non-functional properties (e.g., dependability, timeliness, composability, maintainability). His investigations have resulted in contributions ranging from conceptual models of component-based system architectures to model-based development solutions and distributed algorithms for fault-tolerance and embedded operating system technologies for safety-relevant applications.

Contributors

Günther Bauer
TTTech Computertechnik AG
Vienna, Austria

Kenan Bilic
Vienna University of Technology
Vienna, Austria

Kevin Driscoll
Honeywell International Inc.
Maple Grove, MN

Christian El Salloum
Vienna University of Technology
Vienna, Austria

Petru Eles
Linkoping University
Linköping, Sweden

Wilfried Elmenreich
Alpen-Adria-Universität Klagenfurt
Klagenfurt, Austria

Alois Goller
TTTech Computertechnik AG
Vienna, Austria

Brendan Hall
Honeywell International Inc.
Eden Prairie, MN

Roland Kammerer
Vienna University of Technology
Vienna, Austria

Heinz Kantz
Thales Austria GmbH
Vienna, Austria

Hermann Kopetz
Vienna University of Technology
Vienna, Austria

Roman Obermaisser
University of Siegen
Siegen, Germany

Michael Paulitsch
EADS Innovation Works
Munich, Germany

Paul Pop
Technical University of Denmark
Kongens Lyngby, Denmark

Traian Pop
Ericsson AB
Linköping, Sweden

Christoph Scherrer
Thales Austria GmbH
Vienna, Austria

Eric Schmidt
TTTech Automotive GmbH
Vienna, Austria

Wilfried Steiner
TTTech Computertechnik AG
Vienna, Austria

Contents

List of Figures xvii

List of Tables xxiii

Editors xxv

Contributors xxvii

1 Introduction 1
R. Obermaisser
1.1 Scope of the Book . . . 2
1.2 Structure of the Book . . . 3

2 Basic Concepts and Principles of Time-Triggered Communication 5
R. Obermaisser and H. Kopetz
2.1 Introduction . . . 6
2.2 System Structure . . . 6
2.3 Concepts of Dependability . . . 9
2.3.1 Dependability Threats – Failure, Error, Fault . . . 10
2.3.2 Fault Containment . . . 10
2.3.3 Failure Modes . . . 10
2.3.4 Fault Hypothesis . . . 11
2.4 Global Time and State . . . 12
2.4.1 Time and Clocks . . . 13
2.4.2 Precision and Accuracy . . . 15
2.4.3 Global Time . . . 16
2.4.4 Sparse Time . . . 17
2.4.5 State of a System . . . 19
2.5 Autonomous Control of Communication Networks . . . 20
2.5.1 Types of Temporal Control Signals . . . 20
2.5.1.1 Event Triggers . . . 20
2.5.1.2 Time Triggers . . . 21
2.5.2 Information Semantics . . . 21
2.5.3 Temporal Firewall . . . 21
2.5.4 Transport Protocols . . . 22
2.5.5 Flow Control . . . 23

3 Properties of Time-Triggered Communication Systems **25**
R. Obermaisser and H. Kopetz
3.1 Introduction . . . 26
3.2 Composability . . . 27
3.2.1 Component-Based Design . . . 27
3.2.2 Component Interfaces . . . 28
3.2.2.1 Linking Interface . . . 28
3.2.2.2 Technology Independent Interface (TII) . . . 29
3.2.2.3 Technology Dependent Interface (TDI) . . . 29
3.2.2.4 Local Interface . . . 29
3.2.3 Linking Interface Specification . . . 30
3.2.4 Composition of Nodes . . . 31
3.2.4.1 Independent Development of Nodes . . . 31
3.2.4.2 Stability of Prior Services . . . 32
3.2.4.3 Non-Interfering Interactions . . . 32
3.2.4.4 Preservation of the Node Abstraction in the Case of Failures . . . 32
3.3 Determinism and Predictability . . . 33
3.3.1 The Concept of Determinism . . . 33
3.3.2 Replica Determinism . . . 34
3.3.2.1 Differing Inputs . . . 35
3.3.2.2 Deviations of Computational Progress Relative to Real Time . . . 35
3.3.2.3 Oscillator Drift . . . 35
3.3.2.4 Preemptive Scheduling . . . 36
3.3.2.5 Nondeterministic Language Features . . . 36
3.3.3 Building a Replica Determinate System . . . 36
3.3.3.1 Sparse Time-Base . . . 36
3.3.3.2 Agreement on Input . . . 36
3.3.3.3 Static Control Structure . . . 37
3.3.3.4 Deterministic Algorithms . . . 37
3.3.3.5 Deterministic Communication System . . . 37
3.4 Diagnosability . . . 37
3.4.1 Detection of Errors and Anomalies . . . 38
3.4.2 Decision Making – Analysis of Diagnostic Information . . . 39
3.4.3 Use of Diagnostic Information and Analysis Results . . . 40
3.5 Certifiability . . . 41
3.5.1 Safety Case . . . 41
3.5.2 Modular Certification . . . 43
3.5.3 Certification in Application Domains . . . 43
3.5.4 Time-Triggered Communication Protocols and Certification 44
3.6 Fault Containment and Error Containment . . . 45
3.6.1 Independent Fault Containment Regions . . . 46
3.6.2 Strict Control on Node Interactions . . . 46
3.6.3 Replica Determinism . . . 47

3.6.4 Recovery and Repair 47
3.7 Performance 48
3.7.1 Periodic, Sporadic and Aperiodic Messages 48
3.7.2 Performance Attributes 49

4 Core Algorithms **53**
M. Paulitsch, W. Steiner, R. Obermaisser and C. El Salloum
4.1 Introduction 54
4.2 Clock Synchronization 55
4.2.1 Principle of Operation of Clock Synchronization 56
4.2.1.1 Resynchronization Initiation 57
4.2.1.2 Remote Clock Time Readings 57
4.2.1.3 Convergence Functions 58
4.2.2 Classifications of Clock Synchronization Algorithms 59
4.2.3 Limits in and Performance of Clock Synchronization Algorithms 61
4.2.4 Related Work on Clock Synchronization Algorithms 61
4.2.5 Time Standards and Sources 65
4.2.5.1 Time Standards 65
4.2.5.2 Time Sources 66
4.2.6 Time Aspects from an Application-Specific View 67
4.3 Startup and Restart 68
4.3.1 Introduction and Overview 68
4.3.2 Startup 70
4.3.2.1 Integration 71
4.3.2.2 Coldstart 74
4.3.3 Restart 77
4.3.3.1 Clique Detection Algorithms 78
4.4 Integration of Event-Triggered and Time-Triggered Communication 80
4.4.1 Integration of Event-Triggered and Time-Triggered Communication at MAC Layer 81
4.4.1.1 Event-Triggered and Time-Triggered Communication — Contention Avoidance 81
4.4.1.2 Event-Triggered and Time-Triggered Communication — Contention Detection with Preemption 82
4.4.1.3 Event-Triggered and Time-Triggered Communication — Contention Tolerance 83
4.4.2 Event-Triggered Overlay Networks 83
4.4.3 Generic Event Service 84
4.4.3.1 Higher Protocols: CORBA Internet Inter-ORB Protocol 85
4.4.3.2 Higher Protocols: Controller Area Network (CAN) 85
4.5 Diagnostic Services 88

4.5.1 Error Detection 88
4.5.1.1 Error Detection by Syntactic Checks 89
4.5.1.2 Error Detection by Semantic Checks 89
4.5.1.3 Error Detection by Active Redundancy 90
4.5.2 Membership Agreement 90

5 Time-Triggered Protocol (TTP/C) 93
R. Obermaisser
5.1 Protocol Overview 94
5.2 Protocol Services 95
5.2.1 Communication Services 96
5.2.1.1 Temporal Structuring of Communication 96
5.2.1.2 Timing of a TDMA Slot 97
5.2.1.3 Frame Types and States 98
5.2.2 Clock Synchronization 99
5.2.3 Restart, Re-Integration, Integration 100
5.2.4 Diagnostic Services 101
5.2.4.1 Life-Sign 101
5.2.4.2 Membership Service 102
5.2.4.3 Clique Detection 104
5.2.4.4 Communication System Blackout Detection . . . 104
5.2.5 Fault Isolation 104
5.2.6 Configuration Services 106
5.2.6.1 Mode Changes 106
5.2.6.2 Boot Loader 107
5.3 Protocol Parameterization 108
5.3.1 Message Descriptor List 108
5.4 Communication Interface 110
5.4.1 Status Area 110
5.4.2 Control Area 113
5.4.2.1 Message Area 114
5.5 Protocol States 114
5.6 Validation and Verification Efforts 116
5.6.1 Formal Analysis of Clock Synchronization Algorithm . . . 116
5.6.2 Formal Analysis of Fault Isolation and Consistency 117
5.6.3 Formal Analysis of Membership Service and Clique Avoidance 117
5.6.4 Fault Injection Experiments 118
5.7 Example Configurations and Implementations 119

6 FlexRay 121
C. El Salloum and K. Bilic
6.1 Protocol Overview 122
6.2 Protocol Services 122
6.2.1 Communication Services 122

6.2.1.1 Temporal Structuring of Communication 123
6.2.1.2 Frame Format 126
6.2.1.3 Coding and Decoding 129
6.2.2 Protocol Operation Control 130
6.2.3 Clock Synchronization 132
6.2.3.1 Global and Local Time 132
6.2.3.2 Synchronization Process 132
6.2.4 Wakeup and Startup 134
6.2.4.1 Wakeup 134
6.2.4.2 Startup 135
6.3 Diagnostic Services and Fault Isolation 137
6.3.1 Redundant Communication Channels 137
6.3.2 Bus Guardians 137
6.3.2.1 Local Bus Guardian 138
6.3.2.2 Central Bus Guardian 139
6.3.3 Checks on the Reception of a Frame 139
6.4 Protocol Parameterization 140
6.4.1 Cluster Parameters 140
6.4.2 Node Parameters 141
6.5 Controller Host Interface 142
6.5.1 Overview of the E-Ray IP Module 142
6.5.2 Programmers Model 144
6.5.2.1 Assignment of Message Buffers 144
6.5.2.2 Structure of the Message RAM 145
6.5.2.3 Message Handling 146
6.6 Example Configurations and Implementations 148
6.6.1 Topology and Layout of a FlexRay Network 148
6.6.1.1 Passive Bus Topology 148
6.6.1.2 Active Star Topology 149
6.6.1.3 Hybrid Network 149

7 SAFEbus **153**
M. Paulitsch and K. Driscoll
7.1 SAFEbus 154
7.1.1 Background 154
7.2 Protocol Overview 155
7.3 Protocol Services 157
7.3.1 Communication Services 157
7.3.1.1 Determinism and Partitioning 159
7.3.1.2 Data-Message Structure 160
7.3.1.3 Bus Encoding 161
7.3.1.4 Out-of-Band Signaling Pulses 162
7.3.2 Clock Synchronization 163
7.3.3 Restart, Re-Integration, Integration 164
7.3.4 Diagnostic Services 169

7.3.4.1 Debugging Mechanisms . . . 169
7.3.5 Fault Isolation . . . 170
7.3.5.1 Babble Protection . . . 170
7.3.5.2 Byzantine Protection . . . 171
7.3.5.3 Availability vs. Integrity Trade-Off . . . 171
7.3.5.4 Zombie Module Protection . . . 172
7.3.6 Configuration Services . . . 172
7.3.6.1 Frame Changes . . . 172
7.3.7 Protocol Parameterization . . . 173
7.3.7.1 Table Memory . . . 173
7.3.7.2 Frame Description Language . . . 174
7.3.7.3 Table Versioning . . . 174
7.4 Communication Interface . . . 176
7.5 Validation and Verification Efforts . . . 178
7.6 Example Configurations and Implementations . . . 178

8 Time-Triggered Ethernet **181**
W. Steiner, G. Bauer, B. Hall and M. Paulitsch
8.1 Protocol Overview . . . 182
8.2 Protocol Services . . . 184
8.2.1 Communication Services . . . 185
8.2.1.1 Communication Modes . . . 185
8.2.1.2 Frame Formats . . . 187
8.2.1.3 Coding and Decoding . . . 190
8.2.1.4 Media Access Control . . . 190
8.2.1.5 Permanence Function . . . 195
8.2.2 Clock Synchronization . . . 196
8.2.2.1 Clock Synchronization Overview . . . 196
8.2.2.2 First Step Convergence: Compression Master . . . 197
8.2.2.3 Second Step Convergence: Synchronization Master . . . 200
8.2.3 Startup and Restart . . . 201
8.2.3.1 Integration . . . 203
8.2.3.2 Coldstart . . . 204
8.2.3.3 Restart . . . 205
8.2.3.4 Clique Detection . . . 205
8.2.4 Diagnostic Services . . . 206
8.2.5 Fault Isolation . . . 207
8.2.5.1 Central Guardian . . . 207
8.2.5.2 High-Integrity Design . . . 209
8.2.6 Configuration Services . . . 210
8.3 Protocol Parameterization . . . 210
8.3.1 Physical Topology . . . 210
8.3.2 Protocol-Control Flow Parameterization . . . 211
8.3.3 Dataflow Parameterization . . . 211

8.3.3.1 Time-Triggered Parameters . . . 212
8.3.3.2 Rate-Constrained Parameters . . . 212
8.3.3.3 Best-Effort Parameters . . . 213
8.4 Communication Interface . . . 213
8.5 Validation and Verification Efforts . . . 214
8.5.1 Formal Verification and Analysis . . . 214
8.5.2 Certified Development Process . . . 215
8.5.3 Model-Based Testing . . . 215
8.6 Example Configurations and Implementations . . . 216
8.6.1 Configurations . . . 216
8.6.1.1 Master-Based Configuration . . . 216
8.6.1.2 Dual-Fault Tolerant Configuration . . . 217
8.6.1.3 System-of-Systems Configuration . . . 217
8.6.2 Implementations . . . 219

9 TTCAN **221**
R. Kammerer
9.1 Protocol Overview . . . 221
9.2 Protocol Services . . . 222
9.2.1 Communication Services . . . 222
9.2.2 Clock Synchronization . . . 224
9.2.3 Sending and Receiving Messages in TTCAN . . . 229
9.2.4 Restart, Re-Integration, Integration . . . 230
9.2.5 Diagnostic Services . . . 232
9.2.6 Error Detection and Fault Isolation . . . 234
9.2.7 Configuration Services . . . 238
9.3 Protocol Parameterization . . . 239
9.4 Communication Interface . . . 241
9.5 Validation and Verification Efforts . . . 242
9.6 Example Configurations and Implementations . . . 243

10 LIN **245**
W. Elmenreich
10.1 Protocol Overview . . . 245
10.2 Protocol Services . . . 246
10.2.1 Communication Services . . . 246
10.3 LIN 2.x . . . 247
10.3.1 Clock Synchronization . . . 248
10.3.2 Restart, Re-Integration, Integration . . . 248
10.3.3 Diagnostic Services . . . 248
10.3.4 Error Detection and Fault Isolation . . . 249
10.3.5 Configuration Services and Protocol Parameterization . . . 250
10.4 Communication Interface . . . 252
10.5 Validation and Verification Efforts . . . 253
10.6 Example Configurations and Implementations . . . 253

11 TTP/A **255**
W. Elmenreich
11.1 Protocol Overview 255
11.2 OMG Smart Transducer Standard 256
11.3 Interface File System (IFS) 256
11.4 Protocol Services 259
11.4.1 Communication Services 259
11.4.2 Clock Synchronization 261
11.4.3 Restart, Re-Integration, Integration 262
11.4.4 Diagnostic Services 262
11.4.5 Fault Isolation 263
11.4.6 Configuration Services and Protocol Parameterization . . . 263
11.5 Communication Interface 264
11.6 Validation and Verification Efforts 265
11.7 Example Configurations and Implementations 265
11.7.1 TTP/A Slave Nodes 265
11.7.2 TTP/A Master 266

12 BRAIN **269**
M. Paulitsch, B. Hall and K.R. Driscoll
12.1 Protocol Overview 270
12.1.1 Development History and Design Goals 270
12.1.2 Minimal Overhead Replication and Input Agreement 273
12.2 Protocol Mechanisms and Services 274
12.2.1 High-Integrity Data Propagation 274
12.2.1.1 Self-Checking Data Relay 274
12.2.1.2 Independent Path Data Integrity Reconstitution . . 276
12.2.1.3 Self-Checking Processor Pair Broadcast 277
12.2.2 Clock Synchronization, Startup and Clique Resolution . . . 279
12.2.2.1 Self-Checking Master Coordination 281
12.2.2.2 Connectivity Building and Clique Aggregation . . 282
12.2.2.3 Synchronous Mode Clique Aggregation Breakthrough 285
12.3 Fault Isolation 286
12.3.1 Time-Triggered Sequenced Guardian Roles 286
12.3.1.1 Directional Integrity Exchange 287
12.3.1.2 Skip Guardian Link Forwarding 288
12.3.1.3 Self-Checking Pair Neighbor Guardian 288
12.3.2 Asynchronous Guardian Roles 289
12.3.2.1 Startup Enforcement 289
12.3.2.2 Source Authentication 290
12.3.2.3 Additional Guardian Fault Containment Behavior 291
12.4 Diagnostic and Agreement Services 291
12.4.1 Host Task Set Agreement 291
12.5 Validation and Verification Efforts 292

12.6 Example Configurations, Implementations and Deployment Considerations . . . 292

13 ASCB – Avionics Standard Communications Bus **295**
M. Paulitsch
13.1 Protocol Overview . . . 295
13.2 Protocol Services . . . 296
13.2.1 Communication Services . . . 296
13.2.2 Clock Synchronization, Restart, Re-Integration and Integration . . . 296
13.2.3 Diagnostic Services . . . 299
13.2.4 Fault Isolation . . . 299
13.2.5 Configuration Services . . . 300
13.3 Protocol Parameterization . . . 300
13.4 Communication Interface . . . 300
13.5 Validation and Verification Efforts . . . 301
13.6 Example Configurations and Implementations . . . 301

14 Industrial Applications **303**
M. Paulitsch, E. Schmidt, C. Scherrer and H. Kantz
14.1 Introduction . . . 304
14.2 Time-Triggered Communication in Aerospace . . . 304
14.2.1 Requirements . . . 305
14.2.2 A General Discussion of Time-Triggered Communication to Meet Requirements . . . 311
14.2.3 Use of Time-Triggered Communication Networks in Aerospace and Space . . . 315
14.2.3.1 SAFEbus in Boeing 777 . . . 316
14.2.3.2 ASCB in Primus Epic . . . 321
14.2.3.3 Honeywell's Modular Aerospace Controller . . . 326
14.2.3.4 TTEthernet in Orion . . . 328
14.3 Time-Triggered Communication in Automotive Applications . . . 333
14.3.1 Typical Design of Automotive Applications . . . 337
14.3.2 Migration from CAN to FlexRay . . . 339
14.3.2.1 Event-Triggered Approach – FlexRay as CAN Replacement . . . 340
14.3.2.2 Time-Triggered Approach — FlexRay-Synchronous Task Execution . . . 342
14.3.2.3 Discussion . . . 344
14.3.3 Practical Experience with the Time-Triggered Approach in Automotive Subsystems . . . 345
14.4 Time-Triggered Communication Services in Railway Applications . 346
14.4.1 Railway Applications . . . 346
14.4.2 Requirements on Railway Applications . . . 348
14.4.3 Requirements on Communication Systems . . . 349

14.4.4 Generic System Architecture 350
14.4.4.1 TAS Control Platform Redundancy Architecture . 351
14.4.4.2 TAS Control Platform Communication System . . 351
14.4.4.3 TAS Control Platform Fault Tolerance Layer . . . 353
14.4.4.4 Connectivity 354
14.4.5 Application of Time-Triggered Protocols in the Railway Domain 355
14.4.5.1 Interlocking: Architecture (Components, Services, Interactions) 355
14.4.5.2 Field Element Controller 356
14.4.5.3 Availability Concept 357
14.4.6 Safety Concept 357
14.4.6.1 Timing Requirements 357
14.4.6.2 TTP-Configuration and Schedule 358
14.4.7 Conclusion and Outlook 359

15 Development Tools **361**
P. Pop, A. Goller, T. Pop and P. Eles
15.1 Introduction 363
15.2 Design Tasks 365
15.3 Schedule Generation 368
15.3.1 Requirements and Application Model 371
15.3.1.1 Application Model 374
15.3.2 Scheduling Complexity and Scheduling Strategies 374
15.3.2.1 Incremental Scheduling 376
15.3.2.2 Host Multiplexing 378
15.3.2.3 Dynamic Messaging 380
15.3.2.4 Scheduling Strategies in ^{TTP}Plan 381
15.3.3 Schedule Visualization 383
15.3.3.1 The Schedule Browser 384
15.3.3.2 The Schedule Editor 384
15.3.3.3 The Round-Slot Viewer 387
15.3.3.4 Visualization of Message Paths 387
15.4 Holistic Scheduling and Optimization 391
15.4.1 System Model 392
15.4.2 The FlexRay Communication Protocol 393
15.4.3 Timing Analysis 396
15.4.3.1 Schedulability Analysis of DYN Messages 397
15.4.3.2 Holistic Schedulability Analysis of FPS Tasks and DYN Messages 401
15.4.4 Bus Access Optimization 402
15.4.4.1 The Basic Bus Configuration 404
15.4.4.2 Greedy Heuristic 406
15.4.4.3 Simulated Annealing-Based Approach 407
15.4.4.4 Evaluation of Bus Optimization Heuristics 407