Contents

1 The "Nulticore" Dilemma 1

2 Virtualizable Architecture for embedded MPSoC 5
- 2.1 The Term Virtualization 5
- 2.2 Virtualization for Embedded Multi-Processor Architectures 7
  - 2.2.1 Characteristics of Embedded Processors 7
  - 2.2.2 Purpose of Virtualization in Embedded Multi-Processor Arrays . 9
  - 2.2.3 Requirements for Embedded Virtualization 12
  - 2.2.4 Prerequisites to enable Virtualization Features 13
  - 2.2.5 Virtualization Procedure 17
  - 2.2.6 Analysis and Characteristics of the Virtualization Procedure . . . 23
  - 2.2.7 Intermediate Conclusions 28
- 2.3 Dynamically Reconfigurable Task-to-Processor Interconnection Network 30
  - 2.3.1 Interconnection Requirements for a virtualizable MPSoC 30
  - 2.3.2 Architectural Alternatives 32
  - 2.3.3 Permutation Networks 34
  - 2.3.4 A Sorting Permutation Network as Task-Processor Interconnect 39
  - 2.3.5 Optimization 47
  - 2.3.6 Intermediate Conclusions 57
- 2.4 Task Scheduling via Interconnection Network 59
  - 2.4.1 Binding Definitions of Task Groups and Task Budgets 59
  - 2.4.2 Scheduling an n-to-1 Relation 61
  - 2.4.3 Scheduling several n-to-1 Relations 63
  - 2.4.4 Self-Scheduling of Tasks 64
  - 2.4.5 Intermediate Conclusions 75
- 2.5 Virtualized Task-to-Task Communication 76
  - 2.5.1 Communication Alternatives 76
  - 2.5.2 Conventional Point-to-Point Transfer in non-Virtualized Systems 79
  - 2.5.3 Architectural Enhancements and Code Changes 81
  - 2.5.4 Basic Communication Scheme 83
  - 2.5.5 Enhanced Communication Scheme 89
  - 2.5.6 Properties and Advanced Features 93
- 2.6 Reliability Features of the Virtualization Layer 95
- 2.7 Features for Energy Management 101

Bibliografische Informationen
http://d-nb.info/105961698X

digitalisiert durch DEUTSCHE NATIONAL BIBLIOTHEK

2.8 Intermediate Conclusions 104

3 The Virtualizable MPSoC: Requirements, Concepts, and Design Flows 107
3.1 Requirements for Designing on Top of the virtualizable Architecture . . 109
3.1.1 Weaknesses of Existing Design Tools 109
3.1.2 The FripGa Design Framework 110
3.2 Design Concepts for Reliable, Self-Healing Systems 116
3.2.1 Dynamic Binding Strategies 117
3.2.2 Self-Healing Strategies in the virtualizable MPSoC 119
3.3 Design Flow for Parallelized Execution 122
3.3.1 Problem Motivation 122
3.3.2 Parallelization Topologies 123
3.3.3 Data Dependencies 125
3.3.4 Design Flow 126
3.3.5 Discussion 129
3.4 Design Flow for Agile Processing 135
3.4.1 Motivation and Challenge 135
3.4.2 Prerequisites 136
3.4.3 Design Flow Modification 136
3.4.4 Scheduling Schemes for Agile Processing Designs 141
3.4.5 Discussion 146
3.5 Conclusions 147

4 Application Scenarios 149
4.1 Reshaping in an Automotive Assistance System 149
4.1.1 System Architecture 150
4.1.2 Driver Assistance Tasks 150
4.1.3 Road Types 151
4.1.4 Functional Profiles 152
4.1.5 Timing Constraints 152
4.1.6 Binding Derivation 154
4.1.7 Task Scheduling 157
4.1.8 Setup of the MPSoC and Binding Storage 158
4.1.9 Functional Reshaping 160
4.1.10 Reliability Reshaping 162
4.1.11 Synthesis Evaluation 168
4.1.12 Outcome and Outlook 170
4.2 Reshaping in an Energy-Constrained Quadrocopter System 172
4.2.1 Envisaged Scenario 173
4.2.2 Sensors and Control Systems of the Quadrocopter 174
4.2.3 Example Mission Sequence 176
4.2.4 Role of the virtualizable MPSoC 178

4.2.5 Analysis of Runtime Behavior 179
4.2.6 Synthesis Evaluation 181
4.2.7 Conclusion 182

5 Conclusion and Outlook **185**