Embedded Software Development

The Open-Source Approach

Ivan Cibrario Bertolotti
National Research Council of Italy
Turin, Italy

Tingting Hu
National Research Council of Italy
Politecnico di Torino
Turin, Italy

CRC Press
Taylor & Francis Group
Boca Raton London New York

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

Contents

Foreword xiii

Preface xvii

The Authors xix

List of Figures xxi

List of Tables xxv

Chapter 1 Introduction 1

PART I Basics of Embedded Software Development

Chapter 2 Embedded Applications and Their Requirements 9

2.1 Role and Purpose of Embedded Systems 9
2.2 Microcontrollers and Their Internal Structure 13
2.2.1 Flash Memory 15
2.2.2 Static Random Access Memory (SRAM) 17
2.2.3 External Memory 19
2.2.4 On-Chip Interconnection Architecture 23
2.3 General-Purpose Processors versus Microcontrollers 29
2.4 Embedded Software Development Process 32
2.5 Summary 37

Chapter 3 GCC-Based Software Development Tools 39

3.1 Overview 40
3.2 Compiler Driver Workflow 42
3.3 C Preprocessor Workflow 44
3.4 The Linker 48
3.4.1 Symbol Resolution and Relocation 49
3.4.2 Input and Output Sequences 51
3.4.3 Memory Layout 55
3.4.4 Linker Script Symbols 58
3.4.5 Section and Memory Mapping 59

3.5 The C Runtime Library 62
3.6 Configuring and Building Open-Source Software 65
3.7 Build Process Management: GNU Make 69
3.7.1 Explicit Rules 70
3.7.2 Variables 72
3.7.3 Pattern Rules 74
3.7.4 Directives and Functions 77
3.8 Summary 78

Chapter 4 Execution Models for Embedded Systems 81

4.1 The Cyclic Executive 81
4.2 Major and Minor Cycles 83
4.3 Task Splitting and Secondary Schedules 87
4.4 Task-Based Scheduling 91
4.5 Task State Diagram 95
4.6 Race Conditions in Task-Based Scheduling 98
4.7 Summary 102

Chapter 5 Concurrent Programming Techniques 105

5.1 Task Management 105
5.2 Time and Delays 111
5.3 Semaphores 114
5.4 Message Passing 125
5.5 Summary 136

Chapter 6 Scheduling Algorithms and Analysis 137

6.1 Scheduling Algorithms for Real-Time Execution 137
6.2 Scheduling Analysis 144
6.3 Summary 162

Chapter 7 Configuration and Usage of Open-Source Protocol Stacks 163

7.1 Introduction to the LWIP Protocol Stack 163
7.2 Operating System Adaptation Layer 167
7.3 Configuration Options 169
7.4 Netconn Interface 175
7.5 Network Buffer Management 181
7.6 POSIX Networking Interface 187
7.7 Summary 197

Chapter 8 Device Driver Development 199

8.1 General Structure of a Device Driver 199
8.2 Interrupt Handling 201
8.2.1 Interrupt Handling at the Device Level 203

8.2.2 Interrupt Handling at the Interrupt Controller Level . 204
8.2.3 Interrupt Handling at the CPU Level 206
8.2.4 Some Remarks 211
8.3 Device Driver Interfaces 211
8.4 Synchronization Issues 214
8.5 Example: Ethernet Device Driver 218
8.5.1 Ethernet Block Architecture 218
8.5.2 Initialization 221
8.5.3 Interrupt Handler 230
8.5.4 Receive Process 232
8.5.5 Transmit Process 237
8.6 Summary 238

Chapter 9 Portable Software 241

9.1 Portability in Embedded Software Development 241
9.2 Portability Issues in C-Language Development 247
9.3 Application Programming Interfaces 255
9.4 GCC Extensions to the C Language 258
9.4.1 Object Attributes 259
9.4.2 Assembly Language Inserts 268
9.5 Summary 274

Chapter 10 The FREERTOS Porting Layer 275

10.1 General Information 275
10.2 Basic Data Types 280
10.3 Time Representation and Architectural Details 281
10.4 Context Switch 282
10.5 Interrupt Handling and Critical Regions 288
10.6 Task Stack Initialization 290
10.7 Tick Timer 294
10.8 Architecture-Dependent Scheduler Startup 296
10.9 Summary 297

Chapter 11 Performance and Footprint at the Toolchain Level 299

11.1 Overview of the GCC Workflow and Optimizations 300
11.1.1 Language-Dependent Workflow 301
11.1.2 Language-Independent Workflow 303
11.1.3 Target-Dependent Workflow 314
11.2 Optimization-Related Compiler Options 316
11.3 Architecture-Dependent Compiler Options 320
11.4 Source-Level Optimization: A Case Study 323
11.4.1 Summary of the Algorithm 324
11.4.2 Base Source Code 327

11.4.3 Optimizations 331
11.5 Summary 342

Chapter 12 Example: A MODBUS TCP Device 343

12.1 Toolchain and Operating System 343
12.2 General Firmware Structure 345
12.3 MODBUS Slave Protocol Stack 348
12.3.1 TCP/IP Protocol Stack Interface 349
12.3.2 Operating System Adaptation Layer 352
12.3.3 MODBUS TCP Layer 356
12.3.4 MODBUS Application Layer and Event Loop 358
12.4 USB-Based Filesystem 360
12.4.1 FAT Filesystem 360
12.4.2 USB Protocol Stack and Driver 364
12.5 Application Code 367
12.6 Performance and Footprint 368
12.6.1 Memory Footprint 369
12.6.2 System Performance 371
12.7 Summary 373

PART II Advanced Topics

Chapter 13 Model Checking of Distributed and Concurrent Systems 377

13.1 Introduction 377
13.2 The SPIN Model Checker 379
13.3 The PROMELA Modeling Language 380
13.3.1 Processes 381
13.3.2 Data Types 383
13.3.3 Expressions 386
13.3.4 Control Statements 388
13.3.5 Atomicity 393
13.3.6 Channels 396
13.4 Property Specification 400
13.5 Performance Hints 401
13.6 Summary 404

Chapter 14 Model Checking: An Example 407

14.1 Introduction 407
14.2 Distributed Master Election Protocol 408
14.3 Formal PROMELA Protocol Model 412
14.3.1 Simplified Time Model 412
14.3.2 Broadcast Channel Model 415
14.3.3 Masters Model 417

14.4 Formal Verification Results 419
14.5 Summary 425

Chapter 15 Memory Protection Techniques 427

15.1 Memory Management Units (MMUs) 427
15.2 Memory Protection Units (MPUs) 432
15.3 MPUs versus MMUs 438
15.4 Memory Checksumming 439
15.5 CRC Calculation 444
15.6 Data Structure Marking 455
15.7 Stack Management and Overflow Detection 460
15.8 Summary 465

Chapter 16 Security and Dependability Aspects 467

16.1 Introduction to SPLINT 468
16.2 Basic Checks 470
16.3 Memory Management 478
16.4 Buffer Overflows 492
16.5 Function Interface Annotations 499
16.6 Summary 506

References 507