

Contents

Preface

xvii

Chapter 1

Embedded Computing

1

1.1 The Landscape of High-Performance Embedded Computing

1

1.2 Example Applications

5

1.2.1 Radio and Networking

5

1.2.2 Multimedia

11

1.2.3 Vehicle Control and Operation

15

1.2.4 Sensor Networks

18

1.3 Design Goals

21

1.4 Design Methodologies

22

1.4.1 Basic Design Methodologies

24

1.4.2 Embedded Systems Design Flows

26

1.4.3 Standards-Based Design Methodologies

28

1.4.4 Design Verification and Validation

31

1.4.5 A Methodology of Methodologies

31

1.4.6 Joint Algorithm and Architecture Development

32

1.5 Models of Computation

33

1.5.1 Why Study Models of Computation?

33

1.5.2 Finite versus Infinite State

34

1.5.3 Control Flow and Data Flow Models

38

1.5.4	Parallelism and Communication	41
1.5.5	Sources and Uses of Parallelism	45
1.6	Reliability, Safety, and Security	46
1.6.1	Why Reliable Embedded Systems Are Needed?	47
1.6.2	Fundamentals of Reliable System Design	48
1.6.3	Novel Attacks and Countermeasures	52
1.7	Consumer Electronics Architectures	54
1.7.1	Bluetooth	54
1.7.2	WiFi	56
1.7.3	Networked Consumer Devices	56
1.7.4	High-Level Services	58
1.8	Summary and a Look Ahead	60
	What We Have Learned	61
	Further Reading	61
	Questions	61
	Lab Exercises	63
Chapter 2	CPUs	65
2.1	Introduction	65
2.2	Comparing Processors	66
2.2.1	Evaluating Processors	66
2.2.2	A Taxonomy of Processors	67
2.2.3	Embedded versus General-Purpose Processors	69
2.3	RISC Processors and Digital Signal Processors	69
2.3.1	RISC Processors	70
2.3.2	Digital Signal Processors	71
2.4	Parallel Execution Mechanisms	77
2.4.1	Very Long Instruction Word Processors	77
2.4.2	Superscalar Processors	80
2.4.3	SIMD and Vector Processors	80
2.4.4	Thread-Level Parallelism	82
2.4.5	Processor Resource Utilization	83

2.5	Variable-Performance CPU Architectures	86
2.5.1	Dynamic Voltage and Frequency Scaling	86
2.5.2	Better-Than-Worst-Case Design	88
2.6	Processor Memory Hierarchy	89
2.6.1	Memory Component Models	89
2.6.2	Register Files	95
2.6.3	Caches	95
2.6.4	Scratch Pad Memories	98
2.7	Additional CPU Mechanisms	99
2.7.1	Code Compression	100
2.7.2	Code and Data Compression	116
2.7.3	Low-Power Bus Encoding	117
2.7.4	Security	122
2.8	CPU Simulation	126
2.8.1	Trace-Based Analysis	129
2.8.2	Direct Execution	130
2.8.3	Microarchitecture-Modeling Simulators	131
2.9	Automated CPU Design	132
2.9.1	Configurable Processors	134
2.9.2	Instruction Set Synthesis	143
2.10	Summary	150
	What We Have Learned	150
	Further Reading	151
	Questions	151
	Lab Exercises	152
Chapter 3	Programs	155
3.1	Introduction	155
3.2	Code Generation and Back-End Compilation	156
3.2.1	Models for Instructions	157
3.2.2	Register Allocation	160
3.2.3	Instruction Selection and Scheduling	163

3.2.4	Code Placement	167
3.2.5	Programming Environments	169
3.3	Memory-Oriented Optimizations	170
3.3.1	Loop Transformations	171
3.3.2	Global Optimizations	174
3.3.3	Buffer, Data Transfer, and Storage Management	176
3.3.4	Cache- and Scratch Pad–Oriented Optimizations	178
3.3.5	Main Memory-Oriented Optimizations	182
3.4	Program Performance Analysis	185
3.4.1	Performance Models	187
3.4.2	Path Analysis	188
3.4.3	Path Timing	190
3.5	Models of Computation and Programming	197
3.5.1	Interrupt-Oriented Languages	199
3.5.2	Data Flow Languages	200
3.5.3	Control-Oriented Languages	207
3.5.4	Java	211
3.5.5	Heterogeneous Models of Computation	214
3.6	Summary	218
	What We Have Learned	219
	Further Reading	219
	Questions	219
	Lab Exercises	221
Chapter 4	Processes and Operating Systems	223
4.1	Introduction	223
4.2	Real-Time Process Scheduling	224
4.2.1	Preliminaries	224
4.2.2	Real-Time Scheduling Algorithms	227
4.2.3	Scheduling for Dynamic Voltage Scaling	234
4.2.4	Performance Estimation	239
4.3	Languages and Scheduling	241

4.4	Operating System Design	247
4.4.1	Memory Management in Embedded Operating Systems	248
4.4.2	Structure of a Real-Time Operating System	250
4.4.3	Operating System Overhead	251
4.4.4	Support for Scheduling	253
4.4.5	Interprocess Communication Mechanisms	254
4.4.6	Power Management	255
4.4.7	File Systems in Embedded Devices	255
4.5	Verification	259
4.6	Summary	264
	What We Have Learned	264
	Further Reading	264
	Questions	264
	Lab Exercises	265
Chapter 5	Multiprocessor Architectures	267
5.1	Introduction	267
5.2	Why Embedded Multiprocessors?	269
5.2.1	Requirements on Embedded Systems	270
5.2.2	Performance and Energy	272
5.2.3	Specialization and Multiprocessors	274
5.2.4	Flexibility and Efficiency	275
5.3	Multiprocessor Design Techniques	275
5.3.1	Multiprocessor Design Methodologies	276
5.3.2	Multiprocessor Modeling and Simulation	277
5.4	Multiprocessor Architectures	279
5.5	Processing Elements	288
5.6	Interconnection Networks	289
5.6.1	Models	290
5.6.2	Network Topologies	292
5.6.3	Routing and Flow Control	296
5.6.4	Networks-on-Chips	296

5.7	Memory Systems	304
5.7.1	Traditional Parallel Memory Systems	304
5.7.2	Models for Memory	306
5.7.3	Heterogeneous Memory Systems	307
5.7.4	Consistent Parallel Memory Systems	309
5.8	Physically Distributed Systems and Networks	312
5.8.1	Time-Triggered Architecture	313
5.8.2	FlexRay	316
5.8.3	Aircraft Networks	324
5.9	Multiprocessor Design Methodologies and Algorithms	326
5.10	Summary	332
	What We Have Learned	332
	Further Reading	333
	Questions	333
	Lab Exercises	334
Chapter 6	Multiprocessor Software	337
6.1	Introduction	337
6.2	What Is Different about Embedded Multiprocessor Software?	337
6.3	Real-Time Multiprocessor Operating Systems	339
6.3.1	Role of the Operating System	339
6.3.2	Multiprocessor Scheduling	342
6.3.3	Scheduling with Dynamic Tasks	359
6.4	Services and Middleware for Embedded Multiprocessors	361
6.4.1	Standards-Based Services	363
6.4.2	System-on-Chip Services	366
6.4.3	Quality-of-Service	370
6.5	Design Verification	376
6.6	Summary	378
	What We Have Learned	378
	Further Reading	379

Questions	379
Lab Exercises	381

Chapter 7	Hardware and Software Co-design	383
7.1	Introduction	383
7.2	Design Platforms	384
7.3	Performance Analysis	387
	7.3.1 High-Level Synthesis	387
	7.3.2 Accelerator Estimation	393
7.4	Hardware/Software Co-synthesis Algorithms	396
	7.4.1 Program Representations	397
	7.4.2 Platform Representations	398
	7.4.3 Template-Driven Synthesis Algorithms	400
	7.4.4 Co-synthesis of General Multiprocessors	407
	7.4.5 Multi-objective Optimization	416
	7.4.6 Control and I/O Synthesis	421
	7.4.7 Memory Systems	422
	7.4.8 Co-synthesis for Reconfigurable Systems	425
7.5	Hardware/Software Co-simulation	428
7.6	Summary	430
	What We Have Learned	430
	Further Reading	430
	Questions	431
	Lab Exercises	432
	Glossary	433
	References	467
	Index	501