



UNIVERSIDAD NACIONAL DE ENTRE RÍOS  
FACULTAD DE INGENIERIA  
CENTRO DE MEDIOS  
BIBLIOTECA

# Contents

2356

<hr/>	
<b>INTRODUCTION</b>	
Why This Book?	xiii
Who Will Benefit/How to Use This Book	xiv
Prerequisites	xiv
Programming Languages	xv
Organization	xv
Trademarks	xviii
Acknowledgments	xviii
About the Author	xix
<hr/>	
<b>1 BASIC REAL-TIME CONCEPTS</b>	
1.1 Basic Computer Architecture	1
1.1.1 Bus transfer mechanisms, 2	
1.1.2 Input and output, 3	
1.1.3 Memory, 3	
1.1.4 CPU operation, 4	
1.2 Some Terminology	8
1.2.1 Software concepts, 8	
1.2.2 System concepts, 8	
1.2.3 Real-time definitions, 10	
1.2.4 Events and determinism, 11	
1.2.5 Synchronous and asynchronous events, 12	
1.2.6 Determinism, 13	
1.2.7 Time-loading, 14	
1.3 Real-Time Design Issues	14
1.4 Example Real-Time Systems	15
1.5 Brief History	17
1.5.1 Software, 17	
1.5.2 Hardware, 18	
1.6 Exercises	18
	vii

---

**2 COMPUTER  
HARDWARE**

2.1	CPU	21
2.1.1	Addressing modes, 22	
2.1.2	0-Address architecture, 25	
2.1.3	1-Address architecture, 32	
2.1.4	2-Address architecture, 34	
2.1.5	3-Address architecture, 36	
2.1.6	Compare, jump, and subroutine instructions, 38	
2.1.7	Interrupt enable and disable, 40	
2.1.8	Floating point instructions, 40	
2.1.9	Pipelining, 41	
2.1.10	Coprocessors, 42	
2.1.11	RISC machines, 43	
2.2	Memories	45
2.2.1	Core, 46	
2.2.2	Semiconductor memories, 46	
2.2.3	Fusible link, 47	
2.2.4	UVROM, 47	
2.2.5	EEPROM, 48	
2.3	Input and Output	48
2.3.1	Programmed I/O, 49	
2.3.2	Memory-mapped I/O, 51	
2.3.3	DMA, 52	
2.4	Other Devices	54
2.4.1	MUX transceivers, 54	
2.4.2	A/D circuitry, 55	
2.4.3	D/A circuitry, 56	
2.4.4	Interrupt controllers, 56	
2.4.5	Watchdog timer, 59	
2.5	Exercises	59

---

**3 LANGUAGE ISSUES**

3.1	Language Features	64
3.1.1	Parameter passing, 64	
3.1.2	Recursion, 68	
3.1.3	Dynamic allocation, 69	
3.1.4	Typing, 70	
3.1.5	Exception handling, 73	
3.1.6	Abstract data typing, 73	
3.1.7	Modularity, 76	
3.2	Survey of Commonly Used Programming Languages	79
3.2.1	BASIC, 79	
3.2.2	FORTRAN, 79	
3.2.3	C, 80	
3.2.4	Pascal, 83	
3.2.5	Modula-2, 84	
3.2.6	Ada, 85	
3.2.7	Assembly languages, 86	

	3.3	Code Generation	87
	3.3.1	Know your compiler, 88	
	3.4	Exercises	88
<hr/>			
<b>4 THE SOFTWARE LIFE CYCLE</b>	4.1	Phases of the Software Life Cycle	91
	4.1.1	Concept phase, 93	
	4.1.2	Requirements phase, 93	
	4.1.3	Design phase, 95	
	4.1.4	Programming phase, 96	
	4.1.5	Testing phase, 98	
	4.1.6	Maintenance phase, 99	
	4.2	Nontemporal Transitions in the Software Life Cycle	99
	4.3	Exercises	101
<hr/>			
<b>5 REAL-TIME SPECIFICATION AND DESIGN TECHNIQUES</b>	5.1	Human Languages	104
	5.2	Mathematical Specification	105
	5.3	Flowcharts	106
	5.4	Structure Charts	108
	5.5	Pseudocode and Programming Design Languages	109
	5.6	Finite State Automata	111
	5.7	Data Flow Diagrams	115
	5.7.1	DeMarco's rules, 116	
	5.7.2	Hatley and Pribhai's extensions, 119	
	5.8	Petri Nets	120
	5.9	Warnier-Orr Notation	126
	5.10	Statecharts	129
	5.10.1	Depth, 129	
	5.10.2	Orthogonality, 130	
	5.10.3	Broadcast communication, 130	
	5.11	Exercises	133
<hr/>			
<b>6 REAL-TIME KERNELS</b>	6.1	Polled Loop Systems	138
	6.1.1	Polled loop with interrupts, 139	
	6.2	Phase/State-Driven Code	141
	6.3	Coroutines	143
	6.4	Interrupt-Driven Systems	145
	6.4.1	Context switching, 146	
	6.4.2	Round-robin systems, 148	
	6.4.3	Preemptive priority systems, 149	
	6.4.4	Hybrid systems, 151	
	6.5	Foreground/Background Systems	152
	6.5.1	Background processing, 152	
	6.5.2	Initialization, 153	
	6.5.3	Real-time operation, 154	
	6.6	Full-Featured Real-Time Operating Systems	156
	6.6.1	Task control block model, 157	
	6.7	Exercises	160

<b>7</b>	<b>INTERTASK COMMUNICATION AND SYNCHRONIZATION</b>	<ul style="list-style-type: none"> <li>7.1 Buffering Data 163                             <ul style="list-style-type: none"> <li>7.1.1 Time-relative buffering, 164</li> <li>7.1.2 Ring buffers, 165</li> </ul> </li> <li>7.2 Mailboxes 167                             <ul style="list-style-type: none"> <li>7.2.1 Mailbox implementation, 167</li> <li>7.2.2 Other operations on mailboxes, 168</li> <li>7.2.3 Queues, 168</li> </ul> </li> <li>7.3 Critical Regions 169</li> <li>7.4 Semaphores 170                             <ul style="list-style-type: none"> <li>7.4.1 Mailboxes and semaphores, 171</li> <li>7.4.2 Counting semaphores, 172</li> <li>7.4.3 Problems with semaphores, 174</li> <li>7.4.4 The test-and-set instruction, 175</li> </ul> </li> <li>7.5 Event Flags and Signals 176</li> <li>7.6 Exercises 178</li> </ul>
<b>8</b>	<b>REAL-TIME MEMORY MANAGEMENT</b>	<ul style="list-style-type: none"> <li>8.1 Process Stack Management 181                             <ul style="list-style-type: none"> <li>8.1.1 Task control block model, 182</li> <li>8.1.2 Managing the stack, 182</li> <li>8.1.3 Run-time ring buffer, 184</li> <li>8.1.4 Maximum stack size, 185</li> <li>8.1.5 Multiple stack arrangements, 185</li> </ul> </li> <li>8.2 Dynamic Allocation 188                             <ul style="list-style-type: none"> <li>8.2.1 Swapping, 189</li> <li>8.2.2 Overlays, 189</li> <li>8.2.3 MFT, 189</li> <li>8.2.4 MVT, 191</li> <li>8.2.5 Demand paging, 192</li> <li>8.2.6 Real-time garbage collection, 195</li> <li>8.2.7 Contiguous file systems, 195</li> </ul> </li> <li>8.3 Static Schemes 196</li> <li>8.4 Exercises 196</li> </ul>
<b>9</b>	<b>SYSTEM PERFORMANCE ANALYSIS AND OPTIMIZATION</b>	<ul style="list-style-type: none"> <li>9.1 Response Time Calculation 199                             <ul style="list-style-type: none"> <li>9.1.1 Polled loops, 199</li> <li>9.1.2 Coroutines/phase-driven code, 200</li> <li>9.1.3 Interrupt systems, 200</li> </ul> </li> <li>9.2 Interrupt Latency 202                             <ul style="list-style-type: none"> <li>9.2.1 Propagation delay, 202</li> <li>9.2.2 Macroinstruction execution times, 203</li> <li>9.2.3 Interrupts disabled, 203</li> <li>9.2.4 Preemption, 204</li> <li>9.2.5 Low priority interrupts high, 204</li> </ul> </li> <li>9.3 Time-Loading and Its Measurement 204                             <ul style="list-style-type: none"> <li>9.3.1 Using a logic analyzer, 205</li> <li>9.3.2 Instruction counting, 205</li> <li>9.3.3 Pictorial representation, 208</li> <li>9.3.4 Instruction execution time simulators, 209</li> </ul> </li> </ul>

9.4	Reducing Response Times and Time-Loading	210
9.4.1	Compute at slowest cycle, 211	
9.4.2	Scaled arithmetic, 211	
9.4.3	Binary angular measurement, 212	
9.4.4	Look-up tables, 213	
9.4.5	Basic optimization theory, 215	
9.5	Analysis of Memory Requirements	224
9.5.1	Memory-mapped I/O and DMA memory, 225	
9.5.2	Program area, 225	
9.5.3	RAM area, 226	
9.5.4	Stack area, 226	
9.5.5	Memory management schemes, 227	
9.6	Reducing Memory-Loading	227
9.6.1	Variable selection, 227	
9.6.2	Reuse variables, 228	
9.6.3	Memory fragmentation, 228	
9.6.4	Self-modifying code, 228	
9.7	Exercises	228
<hr/>		
<b>10</b>	<b>QUEUEING MODELS</b>	
10.1	Probability Functions	231
10.1.1	Continuous, 232	
10.2	Discrete	233
10.3	Basic Buffer Size Calculation	235
10.3.1	Handling bursts of data, 236	
10.3.2	Variable buffer size calculation, 237	
10.4	Some Results from Queuing Theory	239
10.4.1	The M/M/1 queue, 240	
10.4.2	Service and production rates, 241	
10.4.3	More buffer calculations, 242	
10.4.4	Response time modeling, 243	
10.4.5	Other queuing models, 244	
10.5	Exercises	244
<hr/>		
<b>11</b>	<b>RELIABILITY, TESTING, AND FAULT TOLERANCE</b>	
11.1	Reliability	245
11.1.1	Formal definition, 246	
11.1.2	Calculating system reliability, 247	
11.2	Testing	253
11.2.1	Unit level testing, 253	
11.2.2	System level testing, 256	
11.2.3	Statistically based testing, 257	
11.2.4	Cleanroom testing, 257	
11.3	Fault Tolerance	258
11.3.1	General problem handling, 258	
11.3.2	N-version programming, 260	
11.3.3	Built-in-test software, 260	
11.3.4	CPU testing, 260	
11.3.5	Memory testing, 261	

	11.3.6 Spurious and missed interrupts, 264	
	11.3.7 Deadlock, 265	
	11.4 Exercises	268
<hr/>		
<b>12 MULTIPROCESSING SYSTEMS</b>	12.1 Distributed Systems	271
	12.1.1 Embedded, 272	
	12.1.2 Organic, 273	
	12.1.3 System specification, 273	
	12.1.4 Reliability in distributed systems, 275	
	12.1.5 Calculation of reliability in distributed systems, 276	
	12.1.6 Increasing reliability in distributed systems, 279	
	12.2 Non-von Neumann Architectures	281
	12.2.1 Data flow architectures, 282	
	12.2.2 Systolic processors, 283	
	12.2.3 Wavefront processors, 286	
	12.3 Exercises	289
<hr/>		
<b>13 HARDWARE/ SOFTWARE INTEGRATION</b>	13.1 Goals of Real-Time System Integration	291
	13.1.1 System unification, 292	
	13.1.2 System validation, 292	
	13.2 Tools	293
	13.2.1 Multimeters, 293	
	13.2.2 Oscilloscope, 293	
	13.2.3 Logic analyzer, 294	
	13.2.4 In-circuit emulator, 296	
	13.2.5 Software simulators, 296	
	13.2.6 Hardware prototypes/simulators, 297	
	13.2.7 Debuggers, 297	
	13.3 Methodology	297
	13.3.1 Establish baseline, 297	
	13.3.2 Backoff method, 297	
	13.3.3 Patching, 298	
	13.4 The Software Heisenberg Uncertainty Principle	299
	13.4.1 Real world analogies, 299	
	13.4.2 The software Heisenberg uncertainty principle, 300	
	13.4.3 Testing of software, 300	
	13.4.4 Time- and memory-loading, 300	
	13.4.5 Other implications, 300	
	13.5 Exercises	301
<hr/>		
<b>A GLOSSARY</b>		303
<hr/>		
<b>B BIBLIOGRAPHY</b>		321
<hr/>		
<b>INDEX</b>		331