

Contents

Preface	xix
----------------	------------

About the Author	xxiii
-------------------------	--------------

Chapter 1 Introduction And Overview	3
--	----------

1.1	<i>Operating Systems</i>	3
1.2	<i>Approach Used In The Text</i>	5
1.3	<i>A Hierarchical Design</i>	5
1.4	<i>The Xinu Operating System</i>	7
1.5	<i>What An Operating System Is Not</i>	8
1.6	<i>An Operating System Viewed From The Outside</i>	9
1.7	<i>Remainder Of The Text</i>	10
1.8	<i>Perspective</i>	11
1.9	<i>Summary</i>	11

Chapter 2 Concurrent Execution And Operating System Services	15
---	-----------

2.1	<i>Introduction</i>	15
2.2	<i>Programming Models For Multiple Activities</i>	16
2.3	<i>Operating System Services</i>	17
2.4	<i>Concurrent Processing Concepts And Terminology</i>	17
2.5	<i>Distinction Between Sequential And Concurrent Programs</i>	19
2.6	<i>Multiple Processes Sharing A Single Piece Of Code</i>	21
2.7	<i>Process Exit And Process Termination</i>	23
2.8	<i>Shared Memory, Race Conditions, And Synchronization</i>	24
2.9	<i>Semaphores And Mutual Exclusion</i>	28
2.10	<i>Type Names Used In Xinu</i>	30
2.11	<i>Operating System Debugging With Kputc And Kprintf</i>	31
2.12	<i>Perspective</i>	32
2.13	<i>Summary</i>	32

Chapter 3 An Overview Of The Hardware And Runtime Environment 37

- 3.1 Introduction* 37
- 3.2 Physical And Logical Organizations Of A Platform* 38
- 3.3 Instruction Sets* 38
- 3.4 General-purpose Registers* 39
- 3.5 I/O Buses And The Fetch-Store Paradigm* 41
- 3.6 Direct Memory Access* 42
- 3.7 The Bus Address Space* 42
- 3.8 Bus Startup And Configuration* 43
- 3.9 Calling Conventions And The Runtime Stack* 44
- 3.10 Interrupts And Interrupt Processing* 47
- 3.11 Vectored Interrupts* 48
- 3.12 Exception Vectors And Exception Processing* 48
- 3.13 Clock Hardware* 49
- 3.14 Serial Communication* 49
- 3.15 Polled vs. Interrupt-driven I/O* 49
- 3.16 Storage Layout* 50
- 3.17 Memory Protection* 51
- 3.18 Hardware Details And A System On Chip Architecture* 51
- 3.19 Perspective* 52
- 3.20 Hardware References* 52

Chapter 4 List And Queue Manipulation 57

- 4.1 Introduction* 57
- 4.2 A Unified Structure For Linked Lists Of Processes* 58
- 4.3 A Compact List Data Structure* 59
- 4.4 Implementation Of The Queue Data Structure* 61
- 4.5 Inline Queue Manipulation Functions* 62
- 4.6 Basic Functions To Extract A Process From A List* 63
- 4.7 FIFO Queue Manipulation* 65
- 4.8 Manipulation Of Priority Queues* 68
- 4.9 List Initialization* 70
- 4.10 Perspective* 71
- 4.11 Summary* 72

Chapter 5 Scheduling And Context Switching 75

- 5.1 Introduction* 75
- 5.2 The Process Table* 76
- 5.3 Process States* 79
- 5.4 Ready And Current States* 80

5.5	<i>A Scheduling Policy</i>	80
5.6	<i>Implementation Of Scheduling</i>	81
5.7	<i>Deferred Rescheduling</i>	85
5.8	<i>Implementation Of Context Switching</i>	85
5.9	<i>State Saved In Memory</i>	86
5.10	<i>Context Switch Operation</i>	87
5.11	<i>An Address At Which To Restart A Process</i>	91
5.12	<i>Concurrent Execution And A Null Process</i>	92
5.13	<i>Making A Process Ready And The Scheduling Invariant</i>	93
5.14	<i>Other Process Scheduling Algorithms</i>	94
5.15	<i>Perspective</i>	95
5.16	<i>Summary</i>	95

Chapter 6 More Process Management

99

6.1	<i>Introduction</i>	99
6.2	<i>Process Suspension And Resumption</i>	99
6.3	<i>Self-suspension And Information Hiding</i>	100
6.4	<i>The Concept Of A System Call</i>	101
6.5	<i>Interrupt Control With Disable And Restore</i>	103
6.6	<i>A System Call Template</i>	104
6.7	<i>System Call Return Values SYSERR And OK</i>	105
6.8	<i>Implementation Of Suspend</i>	105
6.9	<i>Suspending The Current Process</i>	107
6.10	<i>The Value Returned By Suspend</i>	107
6.11	<i>Process Termination And Process Exit</i>	108
6.12	<i>Process Creation</i>	111
6.13	<i>Other Process Manager Functions</i>	115
6.14	<i>Summary</i>	117

Chapter 7 Coordination Of Concurrent Processes

123

7.1	<i>Introduction</i>	123
7.2	<i>The Need For Synchronization</i>	123
7.3	<i>A Conceptual View Of Counting Semaphores</i>	125
7.4	<i>Avoidance Of Busy Waiting</i>	125
7.5	<i>Semaphore Policy And Process Selection</i>	126
7.6	<i>The Waiting State</i>	127
7.7	<i>Semaphore Data Structures</i>	128
7.8	<i>The Wait System Call</i>	129
7.9	<i>The Signal System Call</i>	130
7.10	<i>Static And Dynamic Semaphore Allocation</i>	131
7.11	<i>Example Implementation Of Dynamic Semaphores</i>	132

7.12	<i>Semaphore Deletion</i>	133
7.13	<i>Semaphore Reset</i>	135
7.14	<i>Coordination Across Parallel Processors (Multicore)</i>	136
7.15	<i>Perspective</i>	137
7.16	<i>Summary</i>	137

Chapter 8 Message Passing

143

8.1	<i>Introduction</i>	143
8.2	<i>Two Types Of Message Passing Services</i>	143
8.3	<i>Limits On Resources Used By Messages</i>	144
8.4	<i>Message Passing Functions And State Transitions</i>	145
8.5	<i>Implementation Of Send</i>	146
8.6	<i>Implementation Of Receive</i>	148
8.7	<i>Implementation Of Non-Blocking Message Reception</i>	149
8.8	<i>Perspective</i>	149
8.9	<i>Summary</i>	150

Chapter 9 Basic Memory Management

153

9.1	<i>Introduction</i>	153
9.2	<i>Types Of Memory</i>	153
9.3	<i>Definition Of A Heavyweight Process</i>	154
9.4	<i>Memory Management In Our Example System</i>	155
9.5	<i>Program Segments And Regions Of Memory</i>	156
9.6	<i>Dynamic Memory Allocation</i>	157
9.7	<i>Design Of The Low-level Memory Manager</i>	158
9.8	<i>Allocation Strategy And Memory Persistence</i>	159
9.9	<i>Keeping Track Of Free Memory</i>	159
9.10	<i>Implementation Of Low-level Memory Management</i>	160
9.11	<i>Data Structure Definitions Used With Free Memory</i>	161
9.12	<i>Allocating Heap Storage</i>	162
9.13	<i>Allocating Stack Storage</i>	165
9.14	<i>Releasing Heap And Stack Storage</i>	167
9.15	<i>Perspective</i>	170
9.16	<i>Summary</i>	170

Chapter 10 High-level Memory Management and Virtual Memory

175

10.1	<i>Introduction</i>	175
10.2	<i>Partitioned Space Allocation</i>	176
10.3	<i>Buffer Pools</i>	176
10.4	<i>Allocating A Buffer</i>	178

10.5	<i>Returning Buffers To The Buffer Pool</i>	179
10.6	<i>Creating A Buffer Pool</i>	181
10.7	<i>Initializing The Buffer Pool Table</i>	183
10.8	<i>Virtual Memory And Memory Multiplexing</i>	184
10.9	<i>Real And Virtual Address Spaces</i>	185
10.10	<i>Hardware For Demand Paging</i>	186
10.11	<i>Address Translation With A Page Table</i>	187
10.12	<i>Metadata In A Page Table Entry</i>	188
10.13	<i>Demand Paging And Design Questions</i>	189
10.14	<i>Page Replacement And Global Clock</i>	190
10.15	<i>Perspective</i>	191
10.16	<i>Summary</i>	191

Chapter 11 High-level Message Passing

195

11.1	<i>Introduction</i>	195
11.2	<i>Inter-process Communication Ports</i>	195
11.3	<i>The Implementation Of Ports</i>	196
11.4	<i>Port Table Initialization</i>	197
11.5	<i>Port Creation</i>	199
11.6	<i>Sending A Message To A Port</i>	200
11.7	<i>Receiving A Message From A Port</i>	202
11.8	<i>Port Deletion And Reset</i>	204
11.9	<i>Perspective</i>	207
11.10	<i>Summary</i>	207

Chapter 12 Interrupt Processing

211

12.1	<i>Introduction</i>	211
12.2	<i>The Advantage Of Interrupts</i>	212
12.3	<i>Interrupt Processing</i>	212
12.4	<i>Vectored Interrupts</i>	213
12.5	<i>Integration Of Interrupts And Exceptions</i>	214
12.6	<i>ARM Exception Vectors Using Code</i>	215
12.7	<i>Assignment Of Device Interrupt Vector Numbers</i>	219
12.8	<i>Interrupt Dispatching</i>	220
12.9	<i>The Structure Of Interrupt Software</i>	221
12.10	<i>Disabling Interrupts</i>	223
12.11	<i>Constraints On Functions That Interrupt Code Invokes</i>	225
12.12	<i>The Need To Reschedule During An Interrupt</i>	225
12.13	<i>Rescheduling During An Interrupt</i>	226
12.14	<i>Perspective</i>	227
12.15	<i>Summary</i>	228

Chapter 13 Real-time Clock Management 233

- 13.1 Introduction* 233
- 13.2 Timed Events* 234
- 13.3 Real-time Clocks And Timer Hardware* 234
- 13.4 Handling Real-time Clock Interrupts* 235
- 13.5 Delay And Preemption* 236
- 13.6 Implementation Of Preemption* 237
- 13.7 Efficient Management Of Delay With A Delta List* 238
- 13.8 Delta List Implementation* 239
- 13.9 Putting A Process To Sleep* 241
- 13.10 Timed Message Reception* 244
- 13.11 Awakening Sleeping Processes* 248
- 13.12 Clock Interrupt Processing* 249
- 13.13 Clock Initialization* 251
- 13.14 Perspective* 254
- 13.15 Summary* 255

Chapter 14 Device-independent Input And Output 259

- 14.1 Introduction* 259
- 14.2 Conceptual Organization Of I/O And Device Drivers* 260
- 14.3 Interface And Driver Abstractions* 261
- 14.4 An Example I/O Interface* 262
- 14.5 The Open-Read-Write-Close Paradigm* 263
- 14.6 Bindings For I/O Operations And Device Names* 264
- 14.7 Device Names In Xinu* 265
- 14.8 The Concept Of A Device Switch Table* 265
- 14.9 Multiple Copies Of A Device And Shared Drivers* 266
- 14.10 The Implementation Of High-level I/O Operations* 269
- 14.11 Other High-level I/O Functions* 271
- 14.12 Open, Close, And Reference Counting* 275
- 14.13 Null And Error Entries In Devtab* 277
- 14.14 Initialization Of The I/O System* 278
- 14.15 Perspective* 283
- 14.16 Summary* 283

Chapter 15 An Example Device Driver 287

- 15.1 Introduction* 287
- 15.2 Serial Communication Using UART Hardware* 287
- 15.3 The Tty Abstraction* 288
- 15.4 Organization Of A Tty Device Driver* 289

15.5	<i>Request Queues And Buffers</i>	290
15.6	<i>Synchronization Of Upper Half And Lower Half</i>	291
15.7	<i>UART Hardware FIFOs And Driver Design</i>	292
15.8	<i>The Concept Of A Control Block</i>	293
15.9	<i>Tty Control Block And Data Declarations</i>	293
15.10	<i>Minor Device Numbers</i>	296
15.11	<i>Upper-half Tty Character Input (ttygetc)</i>	297
15.12	<i>Upper-half Tty Read Function (ttyread)</i>	298
15.13	<i>Upper-half Tty Character Output (ttyputc)</i>	300
15.14	<i>Starting Output (ttykickout)</i>	301
15.15	<i>Upper-half Tty Multiple Character Output (ttywrite)</i>	302
15.16	<i>Lower-half Tty Driver Function (ttyhandler)</i>	303
15.17	<i>Output Interrupt Processing (ttyhandle_out)</i>	306
15.18	<i>Tty Input Processing (ttyhandle_in)</i>	308
15.19	<i>Tty Control Block Initialization (ttyinit)</i>	315
15.20	<i>Device Driver Control (ttycontrol)</i>	317
15.21	<i>Perspective</i>	319
15.22	<i>Summary</i>	320

Chapter 16 DMA Devices And Drivers (Ethernet)

325

16.1	<i>Introduction</i>	325
16.2	<i>Direct Memory Access And Buffers</i>	325
16.3	<i>Multiple Buffers And Rings</i>	326
16.4	<i>An Example Ethernet Driver Using DMA</i>	327
16.5	<i>Device Hardware Definitions And Constants</i>	328
16.6	<i>Rings And Buffers In Memory</i>	331
16.7	<i>Definitions Of An Ethernet Control Block</i>	333
16.8	<i>Device And Driver Initialization</i>	336
16.9	<i>Reading From An Ethernet Device</i>	343
16.10	<i>Writing To An Ethernet Device</i>	347
16.11	<i>Handling Interrupts From An Ethernet Device</i>	349
16.12	<i>Ethernet Control Functions</i>	352
16.13	<i>Perspective</i>	353
16.14	<i>Summary</i>	354

Chapter 17 A Minimal Internet Protocol Stack

357

17.1	<i>Introduction</i>	357
17.2	<i>Required Functionality</i>	358
17.3	<i>Simultaneous Conversations, Timeouts, And Processes</i>	359
17.4	<i>A Consequence Of The Design</i>	359
17.5	<i>ARP Functions</i>	360

17.6	<i>Definition Of A Network Packet</i>	371
17.7	<i>The Network Input Process</i>	373
17.8	<i>Definitions For IP</i>	377
17.9	<i>IP Functions</i>	377
17.10	<i>Definition Of The UDP Table</i>	388
17.11	<i>UDP Functions</i>	389
17.12	<i>Internet Control Message Protocol</i>	403
17.13	<i>Dynamic Host Configuration Protocol</i>	404
17.14	<i>Perspective</i>	412
17.15	<i>Summary</i>	413

Chapter 18 A Remote Disk Driver

417

18.1	<i>Introduction</i>	417
18.2	<i>The Disk Abstraction</i>	417
18.3	<i>Operations A Disk Driver Supports</i>	418
18.4	<i>Block Transfer And High-level I/O Functions</i>	418
18.5	<i>A Remote Disk Paradigm</i>	419
18.6	<i>The Important Concept Of Caching</i>	420
18.7	<i>Semantics Of Disk Operations</i>	421
18.8	<i>Definition Of Driver Data Structures</i>	421
18.9	<i>Driver Initialization (rdsinit)</i>	427
18.10	<i>The Upper-half Open Function (rdsopen)</i>	430
18.11	<i>The Remote Communication Function (rdscomm)</i>	432
18.12	<i>The Upper-half Write Function (rdswrite)</i>	435
18.13	<i>The Upper-half Read Function (rdsread)</i>	438
18.14	<i>Flushing Pending Requests</i>	442
18.15	<i>The Upper-half Control Function (rdscontrol)</i>	442
18.16	<i>Allocating A Disk Buffer (rdsbufalloc)</i>	445
18.17	<i>The Upper-half Close Function (rdsfclose)</i>	447
18.18	<i>The Lower-half Communication Process (rdsprocess)</i>	448
18.19	<i>Perspective</i>	453
18.20	<i>Summary</i>	454

Chapter 19 File Systems

459

19.1	<i>What Is A File System?</i>	459
19.2	<i>An Example Set Of File Operations</i>	460
19.3	<i>Design Of A Local File System</i>	461
19.4	<i>Data Structures For The Xinu File System</i>	461
19.5	<i>Implementation Of The Index Manager</i>	462
19.6	<i>Clearing An Index Block (lfibclear)</i>	467
19.7	<i>Retrieving An Index Block (lfibget)</i>	468

19.8	Storing An Index Block (<i>lfi</i> bput)	469
19.9	Allocating An Index Block From The Free List (<i>lfi</i> ballo	471
19.10	Allocating A Data Block From The Free List (<i>lfd</i> ballo	472
19.11	Using The Device-Independent I/O Functions For Files	474
19.12	File System Device Configuration And Function Names	474
19.13	The Local File System Open Function (<i>lfs</i> open)	475
19.14	Closing A File Pseudo-Device (<i>lfl</i> close)	483
19.15	Flushing Data To Disk (<i>lfl</i> flush)	483
19.16	Bulk Transfer Functions For A File (<i>lfl</i> write, <i>lfl</i> read)	486
19.17	Seeking To A New Position In the File (<i>lfl</i> seek)	488
19.18	Extracting One Byte From A File (<i>lfl</i> getc)	489
19.19	Changing One Byte In A File (<i>lfl</i> putc)	490
19.20	Loading An Index Block And A Data Block (<i>lf</i> setup)	492
19.21	Master File System Device Initialization (<i>lfs</i> init)	496
19.22	Pseudo-Device Initialization (<i>lfl</i> init)	497
19.23	File Truncation (<i>lfr</i> truncate)	499
19.24	Initial File System Creation (<i>lfs</i> create)	501
19.25	Perspective	503
19.26	Summary	504

Chapter 20 A Remote File Mechanism

509

20.1	Introduction	509
20.2	Remote File Access	509
20.3	Remote File Semantics	510
20.4	Remote File Design And Messages	510
20.5	Remote File Server Communication (<i>rfs</i> comm)	518
20.6	Sending A Basic Message (<i>rfs</i> ndmsg)	520
20.7	Network Byte Order	522
20.8	A Remote File System Using A Device Paradigm	522
20.9	Opening A Remote File (<i>rfs</i> open)	524
20.10	Checking The File Mode (<i>rfs</i> getmode)	527
20.11	Closing A Remote File (<i>rfl</i> close)	528
20.12	Reading From A Remote File (<i>rfl</i> read)	529
20.13	Writing To A Remote File (<i>rfl</i> write)	532
20.14	Seeking On A Remote File (<i>rfl</i> seek)	535
20.15	Character I/O On A Remote File (<i>rfl</i> getc, <i>rfl</i> putc)	536
20.16	Remote File System Control Functions (<i>rfs</i> control)	537
20.17	Initializing The Remote File System (<i>rfs</i> init, <i>rfl</i> init)	541
20.18	Perspective	543
20.19	Summary	543

Chapter 21 A Syntactic Namespace**547**

- 21.1 *Introduction* 547
- 21.2 *Transparency And A Namespace Abstraction* 547
- 21.3 *Myriad Naming Schemes* 548
- 21.4 *Naming System Design Alternatives* 550
- 21.5 *Thinking About Names Syntactically* 550
- 21.6 *Patterns And Replacements* 551
- 21.7 *Prefix Patterns* 551
- 21.8 *Implementation Of A Namespace* 552
- 21.9 *Namespace Data Structures And Constants* 552
- 21.10 *Adding Mappings To The Namespace Prefix Table* 553
- 21.11 *Mapping Names With The Prefix Table* 555
- 21.12 *Opening A Named File* 559
- 21.13 *Namespace Initialization* 560
- 21.14 *Ordering Entries In The Prefix Table* 562
- 21.15 *Choosing A Logical Namespace* 563
- 21.16 *A Default Hierarchy And The Null Prefix* 564
- 21.17 *Additional Object Manipulation Functions* 564
- 21.18 *Advantages And Limits Of The Namespace Approach* 566
- 21.19 *Generalized Patterns* 566
- 21.20 *Perspective* 567
- 21.21 *Summary* 568

Chapter 22 System Initialization**573**

- 22.1 *Introduction* 573
- 22.2 *Bootstrap: Starting From Scratch* 573
- 22.3 *An Example Of Booting Over A Network* 574
- 22.4 *Operating System Initialization* 575
- 22.5 *Xinu Initialization* 576
- 22.6 *Xinu System Startup* 579
- 22.7 *Transforming A Program Into A Process* 583
- 22.8 *Perspective* 584
- 22.9 *Summary* 584

Chapter 23 Subsystem Initialization And Memory Marking**589**

- 23.1 *Introduction* 589
- 23.2 *Self-initializing Modules* 590
- 23.3 *Self-initializing Modules In A Concurrent System* 591
- 23.4 *Self-initialization In The Presence Of Reboot* 593
- 23.5 *Initialization Using Accession Numbers* 593

23.6	<i>A Generalized Memory Marking Scheme</i>	595
23.7	<i>Data Declarations For The Memory Marking System</i>	596
23.8	<i>Implementation Of Marking</i>	598
23.9	<i>Perspective</i>	599
23.10	<i>Summary</i>	599

Chapter 24 Exception Handling

603

24.1	<i>Introduction</i>	603
24.2	<i>Terminology: Faults, Checks, Traps, And Exceptions</i>	603
24.3	<i>Vectored Exceptions And Maskable Interrupts</i>	604
24.4	<i>Types Of Exceptions</i>	604
24.5	<i>Handling Exceptions</i>	605
24.6	<i>Exception Vector Initialization</i>	606
24.7	<i>Panic In The Face Of Catastrophic Problems</i>	606
24.8	<i>Implementation Of Panic</i>	607
24.9	<i>Perspective</i>	607
24.10	<i>Summary</i>	608

Chapter 25 System Configuration

611

25.1	<i>Introduction</i>	611
25.2	<i>The Need For Multiple Configurations</i>	611
25.3	<i>Configuration In Xinu</i>	613
25.4	<i>Contents Of The Xinu Configuration File</i>	613
25.5	<i>Computation Of Minor Device Numbers</i>	616
25.6	<i>Steps In Configuring A Xinu System</i>	616
25.7	<i>Perspective</i>	617
25.8	<i>Summary</i>	617

Chapter 26 An Example User Interface: The Xinu Shell

621

26.1	<i>Introduction</i>	621
26.2	<i>What Is A User Interface?</i>	622
26.3	<i>Commands And Design Principles</i>	622
26.4	<i>Design Decisions For A Simplified Shell</i>	623
26.5	<i>Shell Organization And Operation</i>	623
26.6	<i>The Definition Of Lexical Tokens</i>	624
26.7	<i>The Definition Of Command-Line Syntax</i>	625
26.8	<i>Implementation Of The Xinu Shell</i>	625
26.9	<i>Storage Of Tokens</i>	628
26.10	<i>Code For The Lexical Analyzer</i>	629

26.11	<i>The Heart Of The Command Interpreter</i>	633
26.12	<i>Command Name Lookup And Builtin Processing</i>	641
26.13	<i>Arguments Passed To Commands</i>	641
26.14	<i>Passing Arguments To A Non-builtin Command</i>	643
26.15	<i>I/O Redirection</i>	646
26.16	<i>An Example Command Function (sleep)</i>	647
26.17	<i>Perspective</i>	649
26.18	<i>Summary</i>	650

Appendix 1 Porting An Operating System **653**

A1.1	<i>Introduction</i>	653
A1.2	<i>Motivation: Evolving Hardware</i>	654
A1.3	<i>Steps Taken When Porting An Operating System</i>	654
A1.4	<i>Programming To Accommodate Change</i>	660
A1.5	<i>Summary</i>	662

Appendix 2 Xinu Design Notes **663**

A2.1	<i>Introduction</i>	663
A2.2	<i>Overview</i>	663
A2.3	<i>Xinu Characteristics</i>	664
A2.4	<i>Xinu Implementation</i>	665
A2.5	<i>Major Concepts And Implementation</i>	667

Index **669**