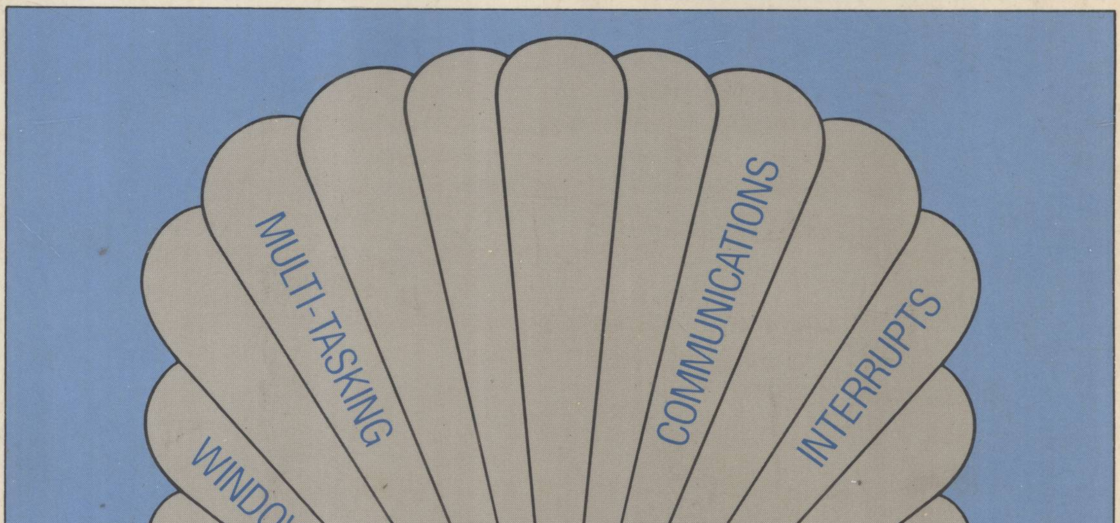
SYSTEMS SOFTWARE TOOLS



Ted J. Biggerstaff

PRENTICE-HALL SOFTWARE SERIES

TP31
B592

8761987

Systems Software Tools

Ted J. Biggerstaff

Microelectronics and Computer Technology Corporation
Austin, Texas

华南工学院
图书馆藏书

E8761987

Prentice-Hall, Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging-in-Publication Data

Biggerstaff, Ted J. (date)
Systems software tools.

Includes index.
1. Computer software. 2. IBM Personal Computer—Programming. 3. C (Computer program language)
I. Title.
QA76.754.B52 1986 005.4 85-25724
ISBN 0-13-881772-3
ISBN 0-13-881764-2 (pbk.)

Editorial/production supervision
and interior design: **Lisa Schulz**
Manufacturing buyer: **Gordon Osbourne**

Prentice-Hall Software Series, Brian W. Kernighan, Advisor

©1986 by Prentice-Hall
A Division of Simon & Schuster, Inc.
Englewood Cliffs, New Jersey 07632

If your diskette is defective or damaged in transit, return it directly to Prentice-Hall at the address below for a no-charge replacement within 90 days of the date of purchase. Mail the defective diskette together with your name and address.

Prentice-Hall, Inc.
Attention: Ryan Colby
College Operations
Englewood Cliffs, NJ 07632

The author and publisher of this book have used their best efforts in preparing this book and software. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

All rights reserved. No part of this book may be reproduced, in any form or by any means, without permission in writing from the publisher.

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-881772-3
ISBN 0-13-881764-2 {PBK.} 025

PRENTICE-HALL INTERNATIONAL (UK) LIMITED, *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sydney*
EDITORA PRENTICE-HALL DO BRASIL, LTDA., *Rio de Janeiro*
PRENTICE-HALL CANADA INC., *Toronto*
PRENTICE-HALL HISPANOAMERICANA, S.A., *Mexico*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
PRENTICE-HALL OF SOUTHEAST ASIA PTE. LTD., *Singapore*
WHITEHALL BOOKS LIMITED, *Wellington, New Zealand*

8761987

To my parents, *Harold and Dorothy,*
and my wife, *Christina*.

Preface

The structure of this book is predicated upon several ideas. First, it is easier for people to learn concepts when simultaneously presented with a basic principle and one or more specific examples of that principle. Consider the English student trying to decide whether to use "which" or "that" to introduce a clause modifying a noun. Referring to his dictionary, he finds that "which" is used for nonrestrictive clauses and that "that" is largely confined to restrictive clauses. Now, exactly what does that mean? The dictionary provides examples that clarify the contrast. "A computer language that does not allow pointers is a clumsy language." "The FORTRAN language, which is considered by many to be the first high-level language, was developed at IBM." Ah, yes, now I see. The clause introduced by "that" is necessary to identify the specific computer languages referred to by the subject of the first example sentence. In contrast, the clause introduced by "which" in the second example sentence could be eliminated and the reader would still know exactly what the subject noun phrase is referring to.

Examples provide the student with a structure he can use to derive, test, and adjust his understanding of basic principles. Examples also provide a rich set of implications—albeit quite specific—that are either not derivable from the general principle or are derivable only with great labor. In this book, I will present both the basic principles, and working examples that illustrate the basic principles.

The second idea behind this book is that people learn best by doing. There is a popular aphorism that recognizes the value of doing:

I hear, I forget; I see, I remember; I do, I understand.

But one must perform a fair amount of preparatory work in order to start exper-

此为试读,需要完整PDF请访问: www.ertongbook.com

imenting (i.e., "doing") with systems software. If a person is studying simple, standalone algorithms it is easy enough to start from scratch and develop working variations on the algorithms. If, on the other hand, the person is studying systems, there are few small, standalone parts of systems that can be extracted for experimentation. To experiment with systems software, one usually has to implement a fairly large piece of the system before much of anything starts to work in an interesting way. Thus, it is useful to have a starting framework that relieves the experimenter of the rather large task of setting up a structure in which to perform an implementation experiment. Providing such a framework is the second objective of this book. The system software described herein is intended to serve as a starting framework for:

1. Student lab assignments in an operating systems course
2. Ambitious student projects, such as quarter, senior and masters projects
3. Hobby computing projects
4. Serious software engineering developments

It is not a complete system with all of the facilities that one would find in a production system. It is a minimal, working system intended as a base to be modified and extended.

The third idea behind this book is that most current operating systems texts do a good job of presenting the general principles but are less successful at presenting the student with examples that are rich enough to have interesting internal interactions among their parts; that actually run on some real machine and, therefore, can be experimented with; that, in addition to dealing with those systems issues for which simple, elegant theories exist, deal with those scruffy, homely systems issues for which there are no simple, elegant theories; and that, because of the richness of the examples, raise a plethora of design issues and problems that most students do not encounter until they get their first job assignment to develop a complex systems design. This book is targeted to fill this gap and serve as a teaching supplement or lab book for systems courses that require hands-on development of system software, or that would benefit from the study of reasonably rich working examples with documentation.

It also is intended as a case book for the working software practitioner. In that role, it can serve as a source for ideas and software algorithms that are the starting point for real developments, or it can serve as a self-study guide for the software practitioner delving into new systems areas.

The fourth idea behind this book is that it should be complete and stand alone in the knowledge that it presents. That is, it is intended to provide at least the rudiments of the various kinds of knowledge required to develop and understand the examples. This goal requires that it present a broad array of information areas, including the C programming language, operating systems concepts, windowing concepts, communications concepts, the 8086/88 and associated hardware, and the PC-DOS operating system. It presents these various information areas not with the intent of providing a thorough, in-depth treatment, but with the intent of providing sufficient information for the student to be able to understand, modify, and extend the examples presented.

I would like to thank several people who have helped to bring this book about. First, Brian Kernighan caught several bugs and many instances of ugly constructions. Not all ugly constructions managed to get removed, but most of those you don't see are due to Brian! In addition, a number of organizational improvements were due to Brian's suggestions. Second, Jim Fegen provided enthusiasm, encouragement, and suggestions that helped to shape the book. Third, Lisa Schulz and other anonymous copyeditors put some spit and polish on my prose. Finally, my wife tolerated my absence for the long time that these programs and this book required.

Ted J. Biggerstaff

Contents

ILLUSTRATIONS *ix*

TABLES *xiii*

PREFACE *xv*

CHAPTER 1 INTRODUCTION *1*

1.1. The Systems Software Tools *1*

1.2. The Elements of Personal Computer Software Tools *3*

Interrupt-Driven Software *3*

Asynchronous Communications *4*

Multitasking *4*

Window Management *4*

1.3. The Target Computer *4*

1.4. Overview of the Book *5*

CHAPTER 2 A REVIEW OF THE C PROGRAMMING LANGUAGE *6*

2.1. Introduction *6*

2.2. The C Programming Language *7*

Basic Data Types and Their Constant Forms *7*

Variable Declarations *9*

Operators 16
Statement and Blocks 21
Functions and Programs 31
Formatted Input/Output 32
Preprocessor Statements 37
2.3. The C Compiler 37
Exercises 39

CHAPTER 3 ACCESS TO SYSTEM SERVICES 40
3.1. BASIC Functions 40
BIOS Functions 40
DOS Functions 44
C Interface to BIOS Functions 45
C Interface to DOS Functions 51
3.2. Application of the BASIC Functions 53
Keyboard Management 53
Screen Management 57
3.3. Exercises 68

CHAPTER 4 INTERRUPT AND COMMUNICATIONS HARDWARE 69
4.1. Introduction 69
4.2. The Interrupt Hardware 70
Internal Structure of the 8259A 72
Features of the 8259A 72
Reading and Writing Ports 79
Programming the 8259A 79
4.3. Asynchronous Communications 86
Serial Asynchronous Communications 86
RS-232C Interface Standard 89
The ACE Hardware 93
Exercises 104

CHAPTER 5 A TERMINAL EMULATOR 105
5.1. Introduction 105
5.2. Anatomy of a Terminal Emulator 107
The Main Program 111
The Interrupt Service Routine 115
Initializing the Asynchronous Interrupts 122
Initializing the 8250 128
Buffer Management and Critical Sections 129
Sending Characters to the Comgear 132
Processing Incoming Characters 135
Processing Outgoing Characters 136
Controlling the Terminal 136
Exercises 144

CHAPTER 6 CONCEPTS OF MULTITASKING AND WINDOW INTERFACES *145*

6.1. Multitasking *145*

Processes (or Tasks) *146*

CPU Scheduling Strategies *148*

Interprocess Communication *150*

File System Management *152*

Sharing Code *152*

Deadlock and Its Prevention *153*

6.2. Window or Display Management *155*

Text or Graphics *156*

Overlayed or Tiled *157*

Transcription and Image Buffers *158*

Process per Window *158*

Exercises *159*

CHAPTER 7 A MULTITASKING WINDOW SYSTEM *160*

7.1. Requirements *160*

7.2. Architecture *162*

Scheduling *162*

The User Interface *162*

The System Interface *163*

7.3. System Overview *163*

7.4. BASIC Data Structure Conventions *165*

The DOS Interface *165*

Pointers *167*

Assemble Language/c Interface Conventions *168*

DOS Function Error Conventions *169*

Assembly Language Conventions *169*

CHAPTER 8 CONCURRENCY STRUCTURES AND THEIR MANAGEMENT *176*

8.1. Data Management *176*

Global Data Structures *176*

The Process Table *179*

Queues and Their Management *180*

Memory Management *181*

8.2. Process Management *185*

Context Switching *186*

The Scheduler *195*

Process State Change *199*

Process Creation *206*

Program Loading and Initiation *210*

Termination *227*

Exercises *230*

华东工学院 ★ 图书馆藏书

CHAPTER 9 THE USER INTERFACE 232
9.1. Window Management 232
A Simple Display Manager 232
Window Table Definition 233
Emulating DOS Display Functions 235
Emulating BIOS Display Functions 246
Window Management Functions 253
9.2. Keyboard Management 263
Requirements 263
Architecture 264
Data Structures and Their Management 265
Emulating DOS Keyboard Functions 266
Intercepting BIOS Keyboard Functions 276
9.3. The Command Interface 281
Requirements 281
Architecture 282
9.4. Summary 288
The Framework 288
Possible Extensions 289
The Final Word 290
Exercises 290

APPENDIX A LISTING OF .h FILES FOR ASSEMBLY PROGRAMS 292

APPENDIX B LISTING OF .h FILES FOR MULTITASKING FUNCTIONS 294

INDEX 305

此为试读，需要完整PDF请访问： www.ertongbook.com

Illustrations

Figure 2-1 Graphical Representation of Declaration *15*
Figure 2-2 Example of a Simple Statement *22*
Figure 2-3 Block or Compound Statement Form *22*
Figure 2-4 Example of a Block *22*
Figure 2-5 **if** Statement Forms *23*
Figure 2-6 Example of **if** Statements *24*
Figure 2-7 Form of a **switch** Statement *25*
Figure 2-8 Example of a **switch** Statement *25*
Figure 2-9 Form of a **while** Loop *26*
Figure 2-10 Example of a **while** Loop *26*
Figure 2-11 Form of a **do-while** Loop *27*
Figure 2-12 Example of a **do-while** Loop *27*
Figure 2-13 Form of a **for** Loop *27*
Figure 2-14 **for** Loop Equivalent Construct *27*
Figure 2-15 Example of a **for** Loop *28*
Figure 2-16 Example of **continue** and **break** Statements *29*
Figure 2-17 Example of the **goto** Statement *30*
Figure 2-18 Form of a C Function *31*
Figure 2-19 Example of a C Function *31*
Figure 2-20 Example of a C **main** Program *33*
Figure 2-21 Formatted Output Functions *34*
Figure 2-22 Formatted Input Functions *34*
Figure 2-23 Field Specifier *35*
Figure 2-24 Format Conversion Characters *36*

Figure 2-25 Examples of Formatted Output *37*
Figure 2-26 Examples of Formatted Input *37*
Figure 2-27 Preprocessor Statements *38*

Figure 3-1 PC Memory Map *41*
Figure 3-2 Assembly Code for **sysint** *46*
Figure 3-3 Setting Up a Simulated Interrupt *49*
Figure 3-4 **bdos** Implementation *52*
Figure 3-5 **getc_noecho** Implementation *53*
Figure 3-6 Simpler **getc_noecho** Implementation *54*
Figure 3-7 **check_keyboard** Implementation *56*
Figure 3-8 **cls** Implementation *59*
Figure 3-9 **scroll_window** Implementation *62*
Figure 3-10 **move_cursor** Implementation *63*
Figure 3-11 **writln** Implementation *64*
Figure 3-12 **read_cursor** Implementation *66*
Figure 3-13 **fetch_dchar** Implementation *67*

Figure 4-1 IBM PC Interrupt Hardware *71*
Figure 4-2 Structure of the 8259A *73*
Figure 4-3 Fully Nested Mode Behavior *73*
Figure 4-4 Rotate on Non-specific Command *76*
Figure 4-5 Initialization Command Words (ICWs) *81*
Figure 4-6 BIOS Initialization of 8259A *82*
Figure 4-7 Operational Command Words (OCWs) *83*
Figure 4-8 OCW2 Command Format *85*
Figure 4-9 Serial Data Transmission over Telephone Lines *88*
Figure 4-10 Serial Data Format *88*
Figure 4-11 RS-232-C Connector and Signals *90*
Figure 4-12 RS-232-C Full Duplex Handshaking *93*
Figure 4-13 Logical Data Flow for the ACE *96*

Figure 5-1 Polled Design for a Terminal Emulator *106*
Figure 5-2 Terminal Emulator Data Flow *108*
Figure 5-3 Terminal Emulator Main Program *111*
Figure 5-4 Data Management of Global Variables *113*
Figure 5-5 General Form of Interrupt Service Routine *115*
Figure 5-6 Asynchronous Interrupt Service Routine *120*
Figure 5-7 C86 Interrupt Code Structure *123*
Figure 5-8 Enabling Conditions *125*
Figure 5-9 Initializing and Enabling Interrupts *126*
Figure 5-10 Initializing the 8250 *128*

Figure 5-11 Circular Buffer Implementation *131*
Figure 5-12 Circular Buffer with Synchronized Access *133*
Figure 5-13 Sending a Character to the 8250 *134*
Figure 5-14 Processing Incoming Characters *135*
Figure 5-15 Processing Outgoing Characters *136*
Figure 5-16 Processing Escape Sequences *139*
Figure 5-17 Global Data Used by **do_escape_seq** *142*
Figure 5-18 Processing Escape Sequences *143*

Figure 6-1 Physical Layout of a Window *156*

Figure 7-1 Organization of the Multitasking System *164*
Figure 7-2 An 8086/88 Pointer *167*
Figure 7-3a Physical Address Computation *167*
Figure 7-3b Example of Physical Address Computation *167*
Figure 7-4 **pfission** and **pfusion** *168*
Figure 7-5 Restoring Registers *169*
Figure 7-6 C Calling Convention Example *170*
Figure 7-7 Assembly Macros *170*

Figure 8-1 Multitasking System Global Variables *177*
Figure 8-2 The Process Table Entry Definition *179*
Figure 8-3 Definition of **queue** and **qentry** *181*
Figure 8-4 An Empty Queue *181*
Figure 8-5 A Queue Containing One Element *182*
Figure 8-6 Specification of Queue Management Functions *182*
Figure 8-7 **get_mem, ret_mem** and **memsize** *184*
Figure 8-8 **errno.h** *186*
Figure 8-9 Canonical Form of a Stopped Process *187*
Figure 8-10 **pswitch** *189*
Figure 8-11 **preempt** *192*
Figure 8-12 The Main Program *197*
Figure 8-13 Process State Diagram for Example *200*
Figure 8-14 **restart** *201*
Figure 8-15 **makready** and **outrange** *202*
Figure 8-16 **resume** *203*
Figure 8-17 **suspend** *204*
Figure 8-18 **kill** *205*
Figure 8-19 **createproc** *206*
Figure 8-20 **load.h** Header File *208*
Figure 8-21 Load Image as Defined by **.exe** File Header Fields *211*
Figure 8-22 Definitions from **proc.h** Header File *212*

Figure 8-23 **loadprog** 215
Figure 8-24 **startstate** 220
Figure 8-25 **lfsearch** 222
Figure 8-26 Example Environment String 224
Figure 8-27 **read_exe_hdr** 225
Figure 8-28 **create_com_hdr** 226
Figure 8-29 **wrapup** 227
Figure 8-30 **harikari** 228
Figure 8-31 **terminate** 229

Figure 9-1 Physical Layout of a Window 232
Figure 9-2 The Window Table 233
Figure 9-3 Relationship of a Window to Real and Virtual Displays 234
Figure 9-4 **capt21** 235
Figure 9-5 **charout** 242
Figure 9-6 **displayc** 243
Figure 9-7 **writeac** 248
Figure 9-8 **setpos** 249
Figure 9-9 **readpos** 252
Figure 9-10 Cursor Movement Utility Functions 254
Figure 9-11 **initwindow** 257
Figure 9-12 **findwindow** 259
Figure 9-13 **clear_window** 260
Figure 9-14 **change_window** and **next_window** 260
Figure 9-15 **delete_window** 261
Figure 9-16 **errormsg, windowmsg** and **disp_str** 262
Figure 9-17 Keyboard Management Architecture 265
Figure 9-18 **clearkeybuf** 266
Figure 9-19 **emptykeybuf** 267
Figure 9-20 **nextkeybuf** 267
Figure 9-21 **putkeybuf** 268
Figure 9-22 **pushkeybuf** 268
Figure 9-23a **keyin** 269
Figure 9-23b **dos3f** 271
Figure 9-23c **dos01** 273
Figure 9-23d **dos06** 274
Figure 9-23e **dos0a** 274
Figure 9-24 **bios16** 277
Figure 9-25a **command** 282
Figure 9-25b **command** Support Routines 286

Tables

Table 1-1 Comparison of Software Tool Characteristics *2*

Table 2-1 Typical C Data Type Sizes for the PC *8*
Table 2-2 Examples of C Data Constants *9*
Table 2-3 Descriptive Meta-Variables *10*
Table 2-4 Basic Declarations of C *11*
Table 2-5 Transformation Rules for Interpreting C Declarations *13*
Table 2-6 Arithmetic Operators *16*
Table 2-7 Relational and Logical Operators *17*
Table 2-8 Assignment Operators *17*
Table 2-9 Storage Related Operators *19*
Table 2-10 Bitwise Logical and Shift Operators *19*
Table 2-11 Miscellaneous Operators *20*
Table 2-12 Operators Ordered by Precedence Level *21*

Table 3-1 BIOS Interrupt Vectors *43*
Table 3-2 Functions of DOS Interrupt 0x21 *45*
Table 3-3 **video_io** Interrupt Functions *59*
Table 3-4 C **#defines** for Display Attributes *61*
Table 3-5 Atomic Display Attributes *61*

Table 4-1 Interrupt Line Assignments for the PC *71*
Table 4-2 End Of Interrupt Summary *75*

Table 4-3 RS-232-C Signals Relevant to PC *91*
Table 4-4 I/O Port Addresses of 8250 ACE Registers *95*
Table 4-5 Line-Control Register Definition *97*
Table 4-6 Baud Rate Generator Divisors *98*
Table 4-7 Line Status Register Definition *99*
Table 4-8 Interrupt Identification Register Definition *100*
Table 4-9 Interrupt Enable Register Definition *101*
Table 4-10 Modem Control Register Definition *102*
Table 4-11 Modem Status Register Definition *103*

Table 5-1 I/O Ports for 8250 Asynchronous Communications Element *114*
Table 5-2 Interrupt Control Functions *117*

Table 7-1 Interrupt Organization For the Multitasking System *166*

Table 9-1 Cursor Movement Utility Function Requirements *253*
Table 9-2 Special Character Handling *272*

此为试读，需要完整PDF请访问： www.ertongbook.com