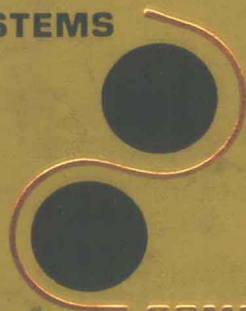


OPERATING SYSTEMS



**COMPUTER
SCIENCE
SERIES**

Operating Systems

Stuart E. Madnick

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ALFRED P. SLOAN SCHOOL OF MANAGEMENT
and Project MAC

John J. Donovan

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ALFRED P. SLOAN SCHOOL OF MANAGEMENT*
and Project MAC

*Formerly with the Department of Electrical Engineering, M.I.T.

McGraw-Hill, Inc.

New York St. Louis San Francisco Auckland Bogotá
Caracas Lisbon London Madrid Mexico City Milan
Montreal New Delhi San Juan Singapore
Sydney Tokyo Toronto

OPERATING SYSTEMS

Copyright © 1974 by McGraw-Hill, Inc. All rights reserved.
Printed in the United States of America. No part of this publication
may be reproduced, stored in a retrieval system, or transmitted, in any
form or by any means, electronic, mechanical, photocopying, recording, or
otherwise, without the prior written permission of the publisher.

22 23 24 25 26 27 BKMBKM 9 9 8 7 6 5 4

Library of Congress Cataloging in Publication Data

Madnick, Stuart E
Operating systems.

(McGraw-Hill computer science series)

Bibliography: p.

1. Electronic digital computers--Programming.

I. Donovan, John J., joint author. II. Title.

QA76.6.M33 001.6'42 74-2040

ISBN 0-07-039455-5

This book was set in Press Roman by Allen Wayne Technical Corp. The
editor was Kenneth J. Bowman; the designer was Allen Wayne Technical
Corp.; the production supervisor was Bill Greenwood.

To all those who have supported me when I most needed encouragement or help, from my high school teacher, Mr. Thornton; my college advisors, Professors Shan Kuo and Conrad Wogrin; my wife, Marilyn; my co-author, friend, and colleague Stuart Madnick; Professor Licklider; my friend James Nichols; to my M.I.T. friend and critic, Professor Paul MacAvoy.

DONOVAN

To my wife, Ethel, and my children, Howard, Michael, and Lynne.

MADNICK

Preface

Purpose of Book

This book presents advanced software techniques, especially focusing on operating systems. It presents material that will enable the reader to design, use, and analyze not only current operating systems but future ones as well. The presentation covers a spectrum ranging from specific implementation details to relevant theoretical results. The book is written as a text, with relevant problems, examples, and exercises.

Approach of Book

Our approach is to develop a framework based upon the concept of resource management. Within this framework, specific case studies are introduced as examples and relevant theory is presented where appropriate.

The book is unique in that it presents a comprehensive framework for the design, study, and implementation of operating systems. Most other texts either present miscellaneous theoretical results or use a case study approach. In the first case, the student may not learn where or how to apply the theoretical results. In the second, he may find it difficult to differentiate between fundamental concepts and design issues and arbitrary decisions.

The features we attempt to provide in this book are the following:

- 1 It is a package course. This is not just a collection of articles but an integrated textbook.
- 2 It is a text for upper undergraduate or graduate courses in computer science or management.
- 3 Questions appear at the end of each chapter.
- 4 A teacher's manual is available with solutions to the questions and possible course syllabuses.
- 5 The book is based on a class-tested course given at M.I.T. for several years. It has been used at other universities, such as the University of Rhode Island and the University of Pittsburgh, and at several industrial companies, for example, The Foxboro Company, U.S. Naval Underwater Systems Center.
- 6 The text relates to existing operating systems. We draw examples particularly, though not exclusively, from IBM's System/370.

- 7 The book uses a framework approach that our students find conceptually enjoyable and understandable.
- 8 Emphasis is given to relevant material. We have found that most students cannot appreciate theoretical results without some practical motivation.
- 9 A complete sample operating system is presented. Studying an operating system is similar to riding a bicycle—you can write the equations to show the bicycle will balance but you can never ride the bicycle until you actually try.
- 10 A comprehensive and annotated bibliography is provided to enable the reader to delve further into the topics introduced in the text.
- 11 Teaching aids are available. In addition to this book, we have made the following available through the M.I.T. computation center:
 - a A 360 machine simulator written in PL/I.
 - b Grading programs for simple assembly language problems.
 - c Sample PL/I problems and a facility for running them.
 - d Sample I/O and interrupt problems.
 - e An environment simulator for testing and running different process schedulers (dispatchers).
 - f An I/O environment simulator for testing and evaluating different I/O scheduling algorithms.
 - g A fast PL/I compiler—this uses the same source as IBM's PL/I(F) and produces the same object code, but in less time.
 - h A submonitor system for running student jobs.

All the programs listed in 11 above are described in our book, *Software Projects: Pedagogical Aids for Software Education and Research* (see page 581), which describes all the programs listed in 11 above. Specifically, the description for each program consists of three sections:

- 1 A student guide—sample of student handouts for the package.
- 2 A teacher's guide—descriptions of how a teacher would operate the program.
- 3 A maintenance manual—a complete description of the implementation of the program.

Uses of Book

As noted above, this book is primarily a text on an upper undergraduate or graduate level. As a prerequisite to the material presented, the student should be familiar with the concepts of an assembler, loader, compiler, and should have programming experience in some high-level language. Assembly language programming is desirable. A good background can be found in the text *Systems Programming* by John J. Donovan.

We see four ways in which this book may be used as a text:

- 1 As an intensive one-semester course in operating systems.
- 2 As a two-term sequence:
 - a First term: course introducing the concepts and practice of I/O programming and interrupt processing. The course would provide the conceptual frame-

- work necessary for further study in operating systems as well as the practical experience for programming real-time systems, minicomputers, and specialized applications (e.g., patient-monitoring computers).
- b Second term: high-level course focusing on the concepts, design, implementation, and relevant theory of operating systems.
- 3 To formulate courses using individual sections of the book, e.g.,
 - a Chapter 2—a course devoted to applications of real-time programming, I/O programming, and interrupts.
 - b Chapter 7—a workshop or lab course constructing sample operating systems.
 - c An overview course on operating systems, skipping the details of Chapter 2 and focusing on the theory and concepts of Chapters 3, 4, 5, and 6.
 - 4 As a course for professionals or for self-study using the problems and solutions manuals.

The book is aimed at helping the student in the following areas:

- a Designing operating systems
- b Understanding relevant theory
- c Using techniques presented in operating systems in other applications
- d Using operating systems
- e Buying or evaluating operating systems

As a text, the book covers the topics mentioned in the recent study by the ACM COSINE Committee, "An undergraduate course in operating systems." The text further covers material contained in two courses of Curriculum 68 as described by the ACM Curriculum Committee in Computer Science¹: (A2) Advanced Computer Organization and (A8) Large Scale Information Systems. The book covers material as reported by ACM Curriculum Committee on Computer Education for Management focusing on Curriculum for Graduate Professional Programs in Information Systems.² These are (C10) Introduction to Computer Systems and Programming; (C11) Information Structure and Files (Chapter 6 of this book); (C2) Computer Systems; (D2) System Design; and (D3) Systems Development Projects (Chapter 7).

In Chapter 2 we present the basic material of machine structures, machine language programming, I/O programming, and interrupt handling. We feel that a person knowledgeable in operating systems must not be intimidated by "pushing bits." This chapter uses the IBM System/370 for detailed examples (this is the only chapter that is heavily machine-dependent).

In Chapters 3, 4, 5, and 6 we develop the framework for the analysis and design of operating systems. In our view of an operating system, the system is a manager of resources. (We differentiate four resources: memory, processors, devices, information.) These chapters discuss the techniques and issues involved in managing these resources and develop the framework for managing them through examples, starting with the simplest and moving to the most complicated. The maximum function (taking the most complicated example of each resource) of these chapters

¹ Communications of the Association of Computing Machinery (CACM), Vol. 11, No. 3, p. 151 (March, 1968).

² Communications of the Association of Computing Machinery (CACM), Vol. 15, No. 5, p. 363 (May, 1972).

is a system like MULTICS, whereas the minimum function is a system such as IBM's OS/360 Primary Control Program (PCP).

In Chapter 7 we design an operating system using the framework developed in previous chapters. We start with the general strategy, develop descriptions of the databases and routines, and finally present the code for an operational implementation.

When the management of all these resources is put in one system, we note that they are not independent. For example, the processor management technique may depend upon whether or not the system uses a paged or swapping memory management technique. In Chapter 8 we explore the interrelationships among the managers of these resources and develop techniques for performance evaluation.

Chapter 9 analyzes various existing systems within the framework developed, and Chapter 10 contains an annotated bibliography of the field.

Stuart E. Madnick

John J. Donovan

Acknowledgments

We list here the people who assisted us in writing this book.

We thank Charles A. Ziering, our head teaching assistant, and particularly acknowledge his reliability. We especially note his contributions to Chapters 4 and 8, as he confirmed (and corrected) most of our figures, and his work on the solutions manual.

We thank our teaching assistants, Paul Gregory, Judy Piggins, Harry Forsdick, Jeff Bunza, Mike Knaur, and Suhundra Umarji, for contributing to the questions, David Schwartz for his work on the bibliography, Chad Carpenter and Jeff Buzen for proofreading, and Anne Beer for her assistance with the index.

Chapter 7 evolved from the thesis work of John DeTreville and was further refined by Richard Swift.

We give special note to those M.I.T. students who went through early versions of our course and survived—we thank them for their constructive criticisms.

We thank Ellen Nangle, who has once again edited, proofread, and typed a book singlehandedly.

We especially appreciate the fine editing of Irene Gunther, the perseverance of Nora Braverman, and the innovative styling of Irving Weinstock, all of Allen Wayne Technical Corp.

We thank Professors Michael Scott-Morton and John Little of the Sloan School for providing a healthy environment within the Management Science group for producing this book.

*Stuart E. Madnick
John J. Donovan*

Contents

PREFACE **xiii**

ACKNOWLEDGMENTS **xvii**

1 **FRAMEWORK OF BOOK** **1**

Importance of Operating Systems **2**

Basic Concepts and Terminology **4**

 Computer Hardware Structure Terminology/**4** Programming Terminology/**4** Operating System Terminology/**5**

An Operating System Resource Manager **8**

 Memory Management Functions/**9** Processor Management Functions/**9**
 Device Management Functions/**9** Information Management Functions/**10**
 Resource Management Summary/**10**

An Operating System—Process Viewpoint (Where These Resource Managers Are Activated) **11**

Operating Systems—Hierarchical and Extended Machine View **15**

 Extended Machine Concept/**15** Hierarchical Machine Concept/**16**

An Example Using Our View—OS/MVT **20**

 Memory Management—OS/MVT/**21** Processor Management—OS/MVT/**21**
 Device Management—OS/MVT/**21** Information Management—OS/MVT/**22**

Other Views of an Operating System **22**

 Historical View/**22** Functional View/**24** Job Control Language and Supervisor Services Interface View/**25**

General Design Considerations **26**

 Software Design Procedure/**26** Implementation Tools/**26**
 Documentation/**27**

Prelude of Things to Come **27**

Summary **28**

Questions **29**

2 **I/O PROGRAMMING, INTERRUPT PROGRAMMING, MACHINE STRUCTURE** **35**

Machine Structure **36**

Assembly Language Programming **37**

 General Approach to a New Computer/**38** Machine Structure—360 and 370/**39** Machine Language/**46** Assembly Language/**47** A Sample Assembly Language Program/**49**

I/O Programming	52
Types of I/O Channels/53	I/O Programming Concepts/54
I/O Processor Structure—360 and 370/55	Examples of I/O Programs/57
Communications between the CPU and the Channel/60	I/O Example Using Single Buffering/62
	I/O Example Using Double Buffering/64
	Multiple Card Buffering/66
Interrupt Structure and Processing	66
Interrupt Types/67	Interrupt Mechanism/68
Interrupt Handler Processing/69	Example of Exceptional Interrupt Processing/70
	Example of Asynchronous Interrupt Processing/74
Examples of I/O and Interrupt Processing Programs	78
IPL Program Example/78	I/O Buffering Example/82
Summary	93
Questions	94
3 MEMORY MANAGEMENT	105
Single Contiguous Allocation	107
Hardware Support/108	Software Support/108
	Advantages/108
	Disadvantages/108
Introduction to Multiprogramming	109
Example of Multiprogramming/110	Measures of System I/O Wait Percentage/111
	Relevance of Multiprogramming to Memory Management/112
Partitioned Allocation	114
Hardware Support/114	Software Algorithm/115
	Advantages/123
	Disadvantages/123
Relocatable Partitioned Memory Management/	124
Hardware Support/125	Software Algorithm/127
	Advantages/129
	Disadvantages/129
Paged Memory Management	130
Hardware Support/132	Software Algorithm/135
	Advantages/138
	Disadvantages/138
Demand-Paged Memory Management	139
Hardware Support/142	Software Algorithm/144
	Performance and Program Behavior Considerations/160
	Advantages/164
	Disadvantages/165
Segmented Memory Management	165
Hardware Support/169	Software Algorithm/170
	Advantages/180
	Disadvantages/181
Segmented and Demand-Paged Memory Management	181
Hardware Support/181	Software Algorithms/182
	Advantages/185
	Disadvantages/185
Other Memory Management Schemes	185
Swapping/185	Overlays/186
Future Trends in Memory Management	187
Large Main Memories/188	Storage Hierarchies/188
	Hardware Support of Memory Management/189

Summary	197
Questions	199
4 PROCESSOR MANAGEMENT	209
State Model	211
Job Scheduler/212	Process Scheduling/212
Synchronization/213	Job and Process Structure of Processor Management/213
Job Scheduling	215
Functions/215	Policies/216
Environment/217	Job Scheduling in Nonmultiprogrammed Environment/217
Job Scheduling Summary/234	Job Scheduling in Multiprogrammed Environment/220
Process Scheduling	235
Functions/236	Policies/237
Scheduling/238	Process State Diagrams for Evaluation of Round-Robin Multiprogramming Performance/240
Multiprocessor Systems	244
Separate Systems/245	Coordinated Job Scheduling/245
Scheduling/245	Master/Slave Homogeneous Processor Scheduling/246
Process Synchronization	247
Race Condition/248	Synchronization Mechanisms/251
Embrace/255	Deadly Synchronization Performance Considerations/261
Combined Job and Process Scheduling	266
Summary	267
Questions	268
5 DEVICE MANAGEMENT	283
Techniques for Device Management	284
Dedicated Devices/284	Shared Devices/284
Generalized Strategies/285	Virtual Devices/285
Device Characteristics—Hardware Considerations	285
Input or Output Devices/286	Storage Devices/286
Channels and Control Units	297
Independent Device Operation/297	Buffering/298
Paths/299	Multiple Block Multiplexing/300
Device Allocation Considerations	300
I/O Traffic Controller, I/O Scheduler, I/O Device Handlers	301
I/O Traffic Controller/301	I/O Scheduler/303
Handlers/303	I/O Device Device Management Software Overhead/307
Virtual Devices	307
Motivation/307	Historical Solutions/308
System/314	Design of a SPOOLing SPOOL System Performance Considerations/323
Future Trends in Device Management	324
Summary	324
Questions	326

6 INFORMATION MANAGEMENT 337

Introduction 337

A Simple File System 339

File Directory Database/340 Steps to Be Performed/342

General Model of a File System 343

File Directory Maintenance/344 Symbolic File System/344 Basic File System/346 Access Control Verification/347 Logical File System/347 Physical File System/348 Allocation Strategy Module/349 Device Strategy Module/350 I/O Scheduler and Device Handler/350 Calls and Returns between File System Modules/350 Shortcomings of Simple File System Design/351

Symbolic File System 351

Directory Files/352 Sample Hierarchical File Structure/353

Basic File System 355

Access Control Verification 355

Access Control Matrix and Access Control Lists/356 Passwords/357 Cryptography/357 Summary/358

Logical File System 358

Sequentially Structured Fixed-length Records/358 Sequentially Structured Variable-length Records/359 Sequentially Structured Keyed Records/361 Multiple-Keyed Records (Inverted File Structure)/363 Chained Structured Records/363 Relational or Triple-structured Records/364

Physical File System 364

Minimizing I/O Operations/365 Allowing Logical Record Size Independent of Physical Block Size/365 Allowing Noncontiguous Allocation of File Space/367

Allocation Strategy Module 369

Automatic Allocation/369 Dynamic Allocation/369

Device Strategy Module, I/O Initiator, Device Handler 371

Trends in Information Management 371

Questions 374

7 DESIGN OF A SAMPLE OPERATING SYSTEM 381

Introduction 381

Overview of the System 382

Design Overview 383

Extended Machine Approach/384 Summary of Concepts Used/385

Levels and Layers of the Sample Operating System 385

Process Management, Lower Module/386 Memory Management Module/387 Process Management, Upper Module/388 Device Management Module and Processes/388 Supervisor Module and Process/389 The User Program and Process/389

Nucleus Databases and Routines 390

Supervisor Call (SVC) Handler/390 Listing of Nucleus Routines/390

Processor Management (Lower Module) Databases 392

Process Control Block/392 Save Areas/395 Semaphores/395

RUNNING/396	NEXTTRY/396	NEXTTRY_MODIFIED/396
SYSTEM_SEM_SAVE_AREA/396	Sample Databases/396	
Processor Management (Lower Module) Routines 396		
Traffic Controller/396	Routine XP (P Operation)/399	Routine XV (V Operation)/399
Routine XPER (Enter Traffic Controller)/400	Routine XEXC (Enter SMC)/400	Routine XCOM (Leave SMC)/400
Memory Management Databases 400		
Free Storage Block (FSB)/400	Free Storage Block Pointer (FSBPTR)/401	Free Storage Block Semaphore (FSBSEM)/401
Memory Semaphore (MEMORY)/401	Sample Databases/401	
Memory Management Routines 402		
Routine XA (Allocate a Block of Memory)/402	Routine XB (Link a Block onto the FSB Chain)/403	Routine XF (Free a Block of Memory)/403
Processor Management (Upper Module) Databases 404		
Messages/404	Sample Databases/404	
Processor Management (Upper Module) Routines 405		
Routine XC (Create a Process)/405	Routine XD (Destroy a Process)/406	
Routine XH (Halt Job and Signal Supervisor)/406	Routine XI (Insert PCB into Chains)/406	Routine XJ (Remove PCB from Chains)/407
Routine XN (Find PCB by Name)/407	Routine XR (Read a Message)/407	Routine XS (Send a Message)/407
Routine XY (Start Process)/408	Routine XZ (Stop Process)/408	Routine XQUE (Abnormal Termination)/409
Device Management Databases 409		
Unit Control Block/409	CAW Semaphore (CAWSEM)/410	
Device Management Routines and Processes 410		
Reader Handler Routine and Process/410	Printer Handler Routine and Process/411	General Device Handler Routine (EXCP) and Process/411
Input/Output Interrupt Handling Routine/411		
Supervisor Routine and Process 412		
Job Stream Processing Routine/412	Supervisor Initiation/413	
User Programs and Processes 418		
User Termination/418	User Execution/418	
Partial Trace of SVC Flow 418		
Sample Operating System Program Listings 423		
Questions 478		
8 INTERDEPENDENCIES: PERFORMANCE EVALUATION 483		
Memory Management 484		
Processor Management 485		
Device Management 486		
Queue Reordering/487	Blocking/Buffering/488	Data Packing Techniques/488
Information Management 489		
Tradeoffs over Which the User Has Control 490		

Influences 490

Influence of a Multiprogrammed System on User/490 Psychological Tradeoffs/491

Swapping versus Paging 491

Sample Program and System Case 1: Time Quantum = ∞ /492 Case 2: Time Quantum = 500 ms/495 Case 3: Time Quantum = 50 ms/496 Conclusion/497

Thrashing 498

Description of the Sample Job/498 Analysis of Total CPU Time Spent Running the Sample Program/499 Conclusions/504

Conclusions 506

Questions 507

9 CASE STUDIES 511

Introduction 511

IBM System/360 and System/370 Operating Systems 512

Philosophy/512 Memory Management/514 Processor Management/516 Device Management/519 Information Management/520

The Compatible Time-Sharing System (CTSS) 523

Philosophy and History/523 Memory Management/527 Processor Management/528 Device Management/532 Information Management/533

Multiplexed Information and Computing System (MULTICS) 534

Philosophy and History/534 Hardware/535 Memory Management/537 Processor Management/543 Device Management/546 Information Management/548

Virtual Machine/370 (VM/370) 549

Philosophy and History/549 Hardware/554 Memory Management/555 Processor Management/557 Device Management/559 Information Management/560

Questions 564

10 ANNOTATED BIBLIOGRAPHY 569

Cross References to Annotated Bibliography 569

Annotated Bibliography 572

APPENDIX A IBM 360/370 SPECIFICATIONS 605

APPENDIX B INTRODUCTION TO MICROPROGRAMMING 621

INDEX 625

Framework of Book

1

This book has two major objectives: to teach the fundamentals of advanced software engineering (including asynchronous operation, interrupt processing, operating system interfaces, and hardware-software tradeoffs), and to enable the reader to design and analyze operating systems.

Modern computer hardware is very powerful (see Figure 1-1a) and can be used for many purposes. But the hardware often provides an awkward or difficult-to-use interface to the world. In order to tame this “bare machine,” we have developed operating systems that can manage the basic hardware resources and provide a more hospitable interface to users and their programs (see Figure 1-1b). Operating systems have become so essential to efficient computer operation that many people view them as inseparable from the hardware.

There are many important reasons for studying operating systems; the most notable are: (1) for special-purpose usage you may have to design your own operating system or modify an existing one; (2) the selection of the operating system and its options is a major decision for most computer installations; (3) the user must interact with the operating system in order to accomplish his task since it is his primary interface with the computer; and (4) many concepts and techniques found in operating systems have general applicability in other applications. The framework presented in this book is especially suited to addressing these issues.

The term *operating system* denotes those program modules within a computer system that govern the control of equipment resources such as processors, main storage, secondary storage, I/O devices, and files. These modules resolve conflicts, attempt to optimize performance, and simplify the effective use of the system. They act as an interface between the user's programs and the physical computer hardware.

These modules are sometimes collectively called the *Control*, the *Monitor*,¹ the *Executive*, the *Supervisor*, or the *Operating System*. Applications-oriented modules,

¹ Not to be confused with a *Monster*.

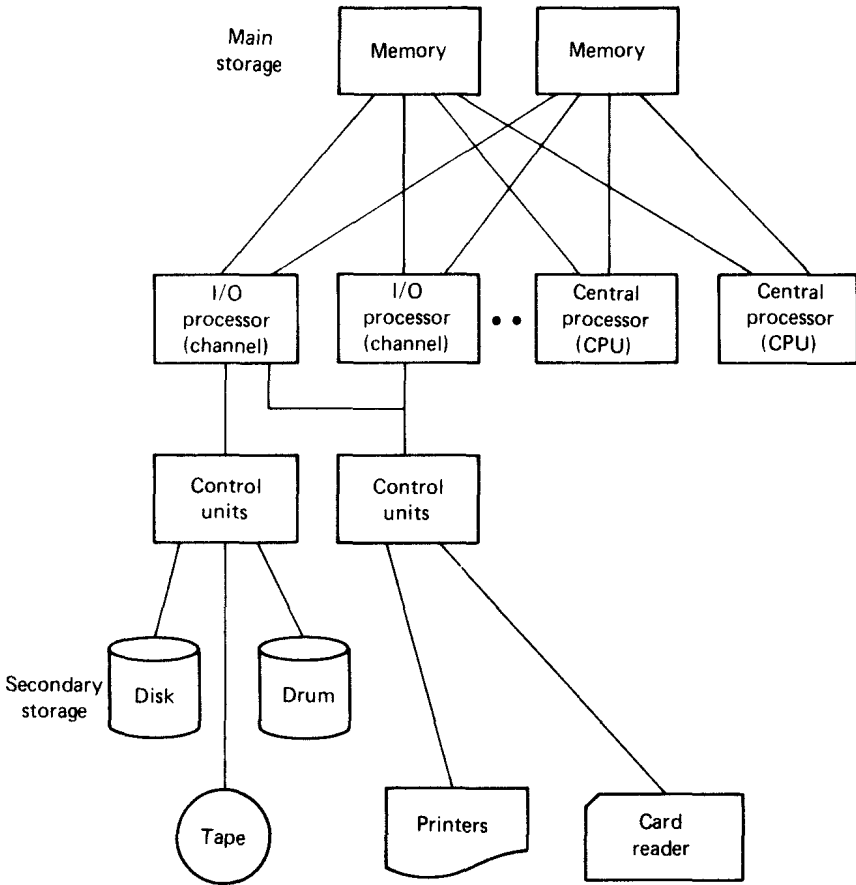


Figure 1-1a Basic computer hardware (bare machine)

language processors, library routines, and debugging aids are not included in our definition of an operating system. These are merely users of the operating system and are covered adequately elsewhere (Donovan, 1972; Knuth, 1968).

1-1 IMPORTANCE OF OPERATING SYSTEMS

Computers have become essential to society in such traditional applications as payroll and banking where the volume of operations performed makes them indispensable. For example, in the mid-1950s the Bank of America pioneered the use of computers in banking after realizing that, in the foreseeable future, it would require the entire adult population of California to handle checks manually (Fano, 1972). Computers are now being used in new fields, such as the design of automobiles, improved medical care, and space exploration; future applications are being envisaged in the areas of learning, acquiring knowledge, and artificial intelligence.