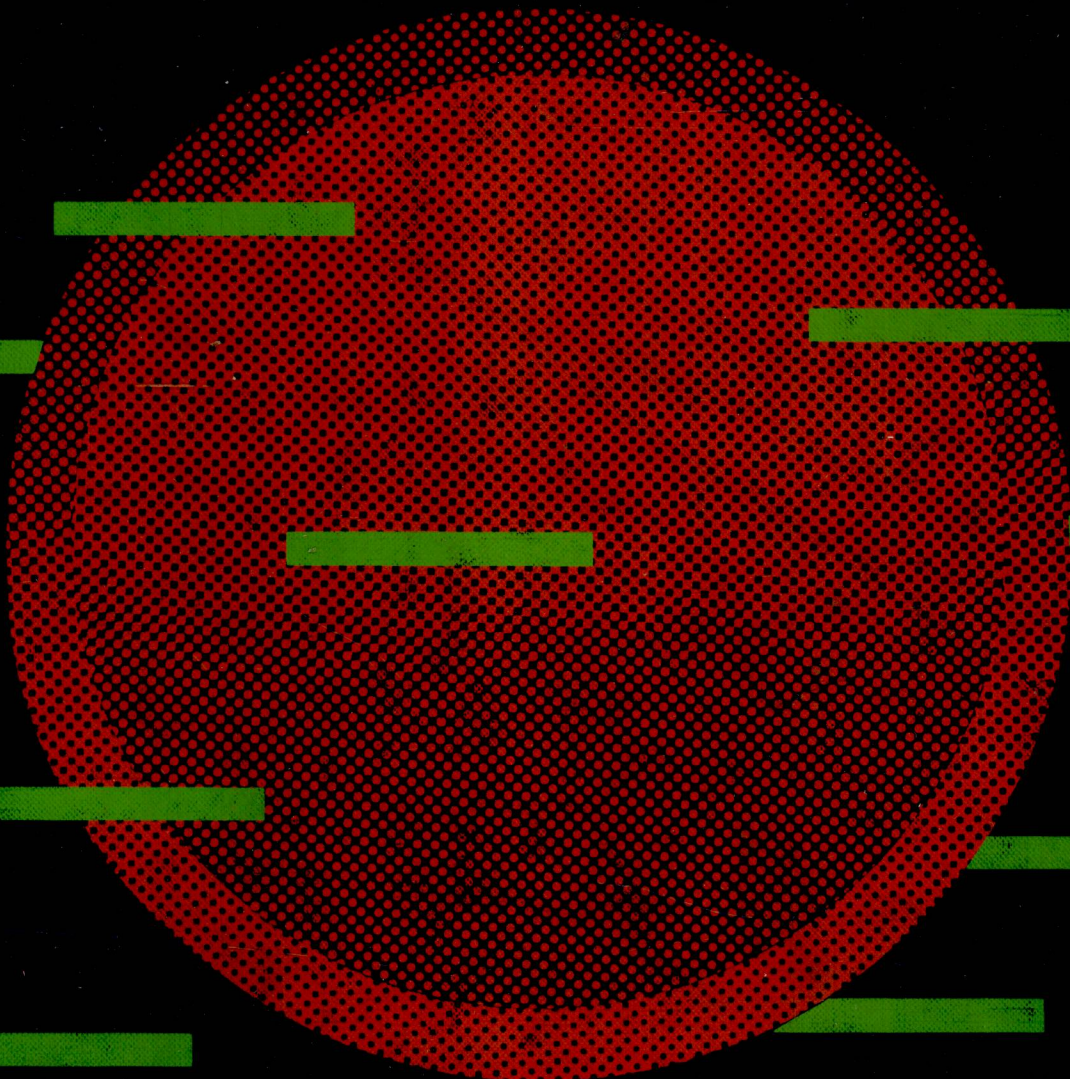


INTRODUCTION TO COMPUTERS

GORDON B. DAVIS

Third Edition



TP33
D2
E3

TP3
D2
E3

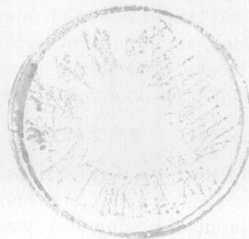
7961463
5

INTRODUCTION TO COMPUTERS

Third Edition

GORDON B. DAVIS

Professor, Management Information Systems
University of Minnesota



E7951463

McGraw-Hill Book Company

New York St. Louis San Francisco Auckland
Bogotá Düsseldorf Johannesburg
London Madrid Mexico Montreal
New Delhi Panama Paris
São Paulo Singapore Sydney Tokyo Toronto

INTRODUCTION TO COMPUTERS

Copyright © 1977 by McGraw-Hill, Inc. All rights reserved. Formerly published under the title of INTRODUCTION TO ELECTRONIC COMPUTERS, copyright © 1971, 1965 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.

2 3 4 5 6 7 8 9 0 VHVH 7 8 3 2 1 0 9 8

This book was set in Plantin by York Graphic Services, Inc. The editors were Peter D. Nalle, Matthew Cahill, and Annette Hall; the designer was Joseph Gillians; the production supervisor was Joe Campanella. New drawings were done by J & R Services, Inc. Von Hoffmann Press, Inc., was printer and binder.

Library of Congress Cataloging in Publication Data

Davis, Gordon Bitter.

Introduction to computers.

First-2d ed. (1965-71) published under title: Introduction to electronic computers.

Bibliography: p.

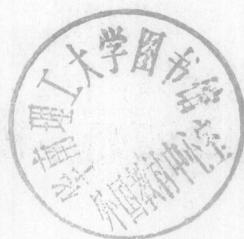
Includes index.

1. Electronic digital computers.
2. Electronic data processing.
3. Programming languages
(Electronic computers) I. Title.

QA76.5.D29 1977 001.6'4 76-52465

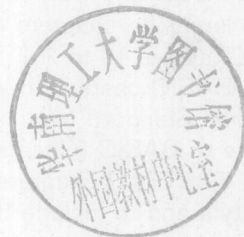
ISBN 0-07-015825-8

INTRODUCTION TO COMPUTERS



TP
C

PREFACE



In 1954 an important news item was the first installation of a computer by an industrial concern in the United States. Ten years later, when the first edition of this text was written, the number of nonmilitary computers was over 13,000. By the beginning of 1977, as the third edition is being printed, every organization of any size makes use of a computer. The computer is a central force in changes occurring in many fields—production, industrial management, accounting, mathematics, and information retrieval, to name a few. It is one of the most significant technological developments of our time. As such, it is a topic of interest to everyone. For the forward-looking student, a knowledge of computers is a necessity.

The third edition represents a significant revision of the second edition. In addition to updating all chapters, the major changes are:

- 1 Introduction of material on the development life cycle for both data processing applications and mathematical-statistical-modeling applications (Chapter 3)
- 2 Greater discussion of alternative processing methodology (Chapter 3)
- 3 A new chapter on computer program structure and design, emphasizing programming discipline, structured design, and top-down development (Chapter 5)
- 4 Rearrangement of chapters to place those relating to development of computer applications and computer programs earlier in the text (Chapters 3 to 5)
- 5 Additional discussion of data preparation methods and alternatives for data preparation in system design (Chapter 8)
- 6 Addition of material on selecting application software (Chapter 12)

- 7 Rearrangement of all discussion of assembly language programming into a single chapter surveying the topic (Chapter 15)
- 8 Revision and reorientation of an introductory chapter on high-level languages to cover alternatives in language selection and the process of selecting a suitable language (Chapter 14)
- 9 Revision of the language chapters to reflect concepts of structured programming (Chapters 16 to 20)
- 10 Revision of the BASIC chapter to include both the 1977 proposed American National Standard Minimal BASIC and current developments in the use of BASIC (Chapter 16)
- 11 Revision of the FORTRAN chapters to present a more disciplined programming style and to include the FORTRAN changes of the proposed 1977 revision of American National Standard FORTRAN (Chapters 17 and 18)
- 12 Revision of the COBOL chapters to obtain a more disciplined programming style using structured programming and to reflect changes made in the newest (1974) American National Standard COBOL

Teaching a course about computers presents a recurrent problem—how much should be taught, and how much of what is taught should be tied to a single computer? A brief introduction leaves the student with little more than the impression that computers are “very, very fast.” On the other hand, a course that looks only at a single computer or at a single language may give the student too narrow a view of the field.

This text arose from a course designed to avoid both the shallowness of the brief-introduction approach and the narrowness of the approach which ties the student to a single computer. The book is thus a general introduction to the concepts and basic features of computers (hardware, software, and systems). The basic elements of machine-oriented programming are explained, as are the three most popular machine-independent high-level languages (BASIC, FORTRAN, and COBOL).

Illustrations and examples in the text rely upon the equipment of many manufacturers. However, the popularity of the IBM System/370 made it appropriate to use it most frequently as the example computer. This is especially true in Chapter 15 (machine-level computer instructions), where the examples generally use the IBM Basic Assembly Language. The high-level language chapters (Chapters 16 to 20) are machine-independent, and the illustrations have been run using several different makes of computers.

A student can study the concepts and techniques related to computers without applying them to a specific computer. However, it is frequently useful to relate the general textual material to the specific computer available to the students. The text can therefore be used in two ways: (1) as a general introduction without reference to any particular computer, or (2) as a coverage of the field which is related to an actual

computer through lectures or a manual. Manufacturers generally have an introductory manual for each computer system. The instructor can extract parallel reading assignments from such a manual to show how the concepts described in the textbook are implemented on the computer system used by the students.

A problem which arises frequently is the depth of study a student should have in programming and the relative emphasis on machine-oriented programming versus machine-independent high-level languages. The trend in computers is clearly toward writing most programs in high-level languages. However, a general understanding of the elements of programming at the machine-language level is useful in understanding the computer field. This suggests that one approach to the field is to have the student obtain a general understanding of the basic features of machine-level assembly language programming *and* learn one or more of the high-level languages—FORTRAN, BASIC, or COBOL, all of which are covered in the text.

Since there are many high-level languages available for programming computers, Chapter 14 describes alternatives and discusses selection criteria. In selecting one or more of the high-level languages for the student to learn, the need of the student should be considered. FORTRAN and BASIC are algebraic languages best suited for programming solutions which can be stated in terms of arithmetic processes to be performed. The algebraic languages are therefore useful to the student in research or other course work. BASIC is simpler than FORTRAN but tends to be used primarily with timesharing. COBOL is suited to data processing applications. When FORTRAN or BASIC is the only language chosen, it is advisable to also read through COBOL for familiarity with this important language. In learning a language such as FORTRAN, BASIC, or COBOL, it is very helpful for a student to code, debug, and execute programs. Almost all computers of any size have both FORTRAN and COBOL compilers. BASIC is available on all timesharing systems and many minicomputers. If both an algebraic and a data processing language are to be studied, the algebraic language should usually be studied first because it is easier to get small problems to compile and run; the problems also can be shorter with an algebraic language than with a data processing language.

The text is suitable for either a one-semester (quarter) course or a two-semester (quarter) sequence, depending on the proficiency desired and the time devoted to a specific computer and programming languages. A one-semester course can cover all topics and can include the learning of one of the three high-level languages. A two-semester (quarter) sequence can develop topics in greater depth, give a better understanding, and develop better proficiency in FORTRAN, BASIC, and COBOL.

The text is somewhat modular. For example, the chapters in Part 3, which cover computer technology, may be assigned before the chapters in Part 2, which consider development of computer applications and com-

puter programs. An instructor may choose to skip the material on binary arithmetic in Chapter 6 and the material on internal operations in Chapter 7 if this is not appropriate for the students. Any of the language chapters (15 through 20) may be eliminated without affecting the use of the remaining chapters.

The programming assignments require sufficient elapsed time that an instructor may wish to make programming-language-chapter assignments in parallel with earlier chapters. The order of chapter use in such a case might be Chapters 1 to 3, Appendix 1, Chapters 14, 4, and 5, and then language chapter assignments alternating with Chapters 6 to 13.

The fact that the text is not tied to any particular subject area makes it suitable for students in all fields. The text is suitable for self-study. Of the 20 chapters, 8 contain self-testing quizzes either in the chapter or as part of the exercises at the end. The technique of using self-testing quizzes as an aid to the learning process is applied intensively in the BASIC, FORTRAN, and COBOL chapters.

An alternate version of this text is available for those who do not desire the six programming language chapters (Chapters 15 through 20). The alternate text, *Introduction to Computers and Information Systems*, McGraw-Hill, 1978, contains the material covered in Chapters 1 through 14 of this text plus a chapter summarizing characteristics of programming languages.

Gordon B. Davis

Note to the second printing. The proposed 1977 FORTRAN standard has now been adopted, with minor changes from the proposal on which Chapters 17 and 18 were based. These changes have been reflected in the second printing.

CONTENTS

PREFACE	vii
PART 1 INTRODUCTION	
1 An Overview of Computers	3
2 The Uses of Computers	32
PART 2 THE DEVELOPMENT OF COMPUTER APPLICATIONS AND COMPUTER PROGRAMS	
3 The Development of a Computer Application	55
4 Tools for Use in Design of Computer Applications and Computer Programs	84
5 Computer Program Structure and Design	116
PART 3 COMPUTER TECHNOLOGY	
6 Computer Arithmetic and Data Representation	151
7 Internal Operations and Storage in a Computer	180
8 Data Preparation, Input, Output, and Data Communications	210
9 Secondary Storage and Computer Data Structures, Files, and Databases	247
PART 4 EFFECTIVE OPERATION AND USE OF THE COMPUTER	
10 Operating an Inhouse Computer and Using Outside Computer Services	289
11 Quality Control and Security in Computer Processing	311

- 12 Acquisition of Computer Hardware and Software 337
- 13 Current and Prospective Developments in Computer Hardware,
Software, and Applications 360

PART 5 COMPUTER PROGRAMMING LANGUAGES

- 14 Selecting a Language for Computer Programming 374
- 15 Machine-oriented Symbolic Assembly Language 394
- 16 The BASIC Programming Language 441
- 17 FORTRAN—Elementary Features 473
- 18 FORTRAN—Additional Features 529
- 19 Elementary COBOL 603
- 20 Additional Features of COBOL 674

APPENDIXES

- 1 How to Use the Card Punch 737
- 2 Guide to United States Computer Organizations and
Periodicals 746
- 3 The Certificate in Data Processing 751

**SELECTED
REFERENCES 755**

GLOSSARY 763

INDEX 773

PART



The chapters in Part 1 provide an introduction to the study of computers. The first chapter gives an historical overview and surveys the equipment and other elements to be found in computer systems. Chapter 2 describes some of the uses for computers. The variety of uses provides a background for the study of computer application development, computer technology, computer operations, and programming languages.

INTRODUCTION

CHAPTER



AN OVERVIEW OF COMPUTERS

Definition of a Computer

Historical Events in the Development of Computers

Babbage and the Analytic Engine
Boolean Algebra
The Machine-readable Punched Card
The Automatic Sequence Controlled Calculator—Mark I

The First Generation of Automatic Computers

ENIAC (Electronic Numerical Integrator and Calculator)
EDVAC, EDSAC, and IAS
UNIVAC (UNIVersal Automatic Computer)

Computers since the First Generation

Elements of a Computer Processing System

Hardware
System Software
Application Software
Procedures
Personnel

Processing Methods

The Equipment in a Computer Installation

The Central Processing Unit
Secondary Storage
Data Preparation Equipment
Input Devices
Output Devices

The Computer Industry

Computer System Configurations

Large-Scale Computer
Medium-Scale Computer
Small-Scale Computer
Small-Business Computer
Minicomputer Systems
Microprocessors

Summary

Exercises

The modern computer represents a fundamental advance in computation on a par with other advances such as the development of the zero or the discovery of calculus. The computer is a result of, and a major contributor to, the current technological explosion. No one can escape some contact with computers. This contact may result from such mundane computer applications as the preparation of payroll checks and department store charge account billings, or it may be from such esoteric activities as predictions of the outcome of elections or analysis of the probable authorship of ancient documents. Since the present and the future are often best understood in terms of the past, it will be valuable to an understanding of computers to briefly review their history.

Definition of a Computer

Strictly speaking, a computer is any calculating device. The name is derived from the Latin *computare*, meaning “to reckon” or “to compute,” and can be applied as properly to an abacus or an adding machine as to the modern computer. However, the term “computer” has come to mean a special type of calculating device having certain definite characteristics.

There are basically two types of computers—*analog* and *digital*. *Analog computers* use electronic circuitry to represent physical processes, with changes in electric current representing the behavior of the system being studied. *Digital computers*, on the other hand, are based essentially on counting operations. A *hybrid computer* is a combination of an analog and a digital computer. Most modern computers are digital computers, and usually digital computers are meant when the word “computer” is used. For this reason, the explanations in the chapters to follow apply only to digital computers.

Several characteristics typically found in digital computers differentiate them from mechanical and electronic calculators. These features are speed, internal memory, and stored program. The speed of the electronic computer is the result of electronic circuitry. Mechanical counters are severely limited by the speed with which mechanical devices can start, move, and stop, while the speed of electronic circuitry is limited only by the speed at which electricity is transmitted (which is the speed of light). Electronic computers can hold data and instructions in an electronic representation in an internal memory unit. This feature not only adds to the speed of processing but also forms the basis for a stored program. In contrast to a mechanical or electronic calculator, which requires human direction at each step in a computational routine, the computer is automatic in the sense that the stored program will direct the performance of a long and complicated sequence of operations without operator intervention. For this reason, early writers often spoke of the automatic computer as contrasted to the existing nonautomatic calculating devices. The stored program usually includes logical tests to determine which of many possible program steps should be taken at important junctures in the program. Thus, the stored program is not just a sequence of steps to be followed in

computation but typically includes all possible paths that computation might take within the scope of the problem.

Historical Events in the Development of Computers

While modern computers were not developed until the late 1940s, some important prior developments were the algebra of logic by George Boole, the punched card by Herman Hollerith, and the calculator built by George Aiken. Also important historically is Charles Babbage. His work did not directly influence the design of the first modern computer, but certain basic ideas of the stored computer program can be traced to this remarkable nineteenth-century inventor.

Babbage and the Analytic Engine

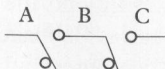
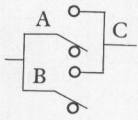
In 1812, Charles Babbage, professor of mathematics at Cambridge University, devised a machine called a “difference engine” to automatically perform simple computations needed for trigonometric and logarithmic tables. Babbage’s difference engine carried out a set sequence of operations one at a time, but Babbage also conceived of an analytic engine which would execute an arbitrary sequence of operations and would have internal storage for data. Babbage’s description of the analytic engine’s stored program is remarkably close to the concept of the stored program of modern computers.

Boolean Algebra

The switching networks in a computer can be very complex, and the logic of a complicated program can be very difficult to analyze. Boolean algebra provides a systematic method of representation and analysis. It is named after the English mathematician George Boole, who pioneered in the field of symbolic logic. His book *The Laws of Thought*, published in 1854, represents logic in mathematical symbols and provides rules for calculating the truth or falsity of statements.

A simple illustration will show why mathematical logic is important in the design of computers. If an electronic switch is open, we shall assign the value 0, and if it is closed, we shall assign the value 1. These two states of the switch will correspond to the values of false (0) and true (1) in the symbolic logic of Boolean algebra. Two switches which operate independently such that there is current at the output line C only if both switches A and B are closed can be described by the Boolean statement A AND B (written symbolically as $A \wedge B$). A circuit in which there will be current at C if either A or B is closed is described by the Boolean statement A OR B ($A \vee B$). The results from these circuits can be described by examining all possible combinations of open and closed states for switches A and B and determining the effect on output C. The resulting tables (Table 1-1) are identical to the truth tables derived from symbolic logic. The work of Boole and others in the mathematics of symbolic logic thus provided an analytical basis for computer design.

TABLE 1-1 ANALYSIS OF SWITCHING CIRCUITS

GIVEN THESE COMBINATIONS OF SWITCH POSITIONS		AND CIRCUIT DESIGN DESCRIBED BY $A \wedge B$		AND CIRCUIT DESIGN DESCRIBED BY $A \vee B$	
A	B	Diagram	Output at C	Diagram	Output at C
Closed	Closed		On		On
Closed	Open		Off		On
Open	Closed		Off		On
Open	Open		Off		Off

The Machine-readable Punched Card

Although not a component of the electronic computer, the punched card has become an integral part of input to data processing using electronic computers, and as such, it should be mentioned in a history of electronic computers. The use of punched cards began in 1745 when a Frenchman, Joseph M. Jacquard, designed a method for using holes in cards to control the selection of threads in weaving designs. Babbage also proposed cards with holes as input for data into the analytic engine.

The originator of the modern machine-readable punched card was Herman Hollerith. The need for mechanical tabulating equipment was seen by Hollerith while working with the 1880 United States census. During the 1880s, he developed the idea of the punched card and designed a *census machine*. The Hollerith method was chosen for tabulating the 1890 census, and the tabulation was done quickly and at a substantial savings in cost over the previous hand method. In 1896, Hollerith organized the Tabulating Machine Company to manufacture and market the machines and cards. This company merged with others to eventually become International Business Machines Corporation (IBM). James Powers, also connected with the Bureau of the Census, designed some card-processing equipment with slightly different features. This equipment was used to tabulate the 1910 census. In 1911, Powers formed the Powers Accounting Machine Company, subsequently acquired by Remington Rand (now a division of Sperry Rand Corporation). The cards designed by Hollerith used 80 columns and rectangular punches, while those designed by Powers used 90 columns and round punches. Sperry Rand has discontinued the 90-column card, and the field is dominated by the 80-column Hollerith card (Figure 1-1). A 96-column small card is used primarily in connection with the IBM System/3 computer introduced in 1969 (Figure 1-2). The small size is made possible by use of very small holes and a more compact coding scheme.

It is helpful to understand how data is encoded in punched cards. In the 80-column card there are 80 columns, and each column has 12 rows or punching positions numbered (from top to bottom) 12, 11, 0, 1, . . . , 9. A numeric digit 0 to 9 is encoded in a column by punching the corresponding row position (see Figure 1-1). An alphabetic character is encoded by a combination of a zone punch

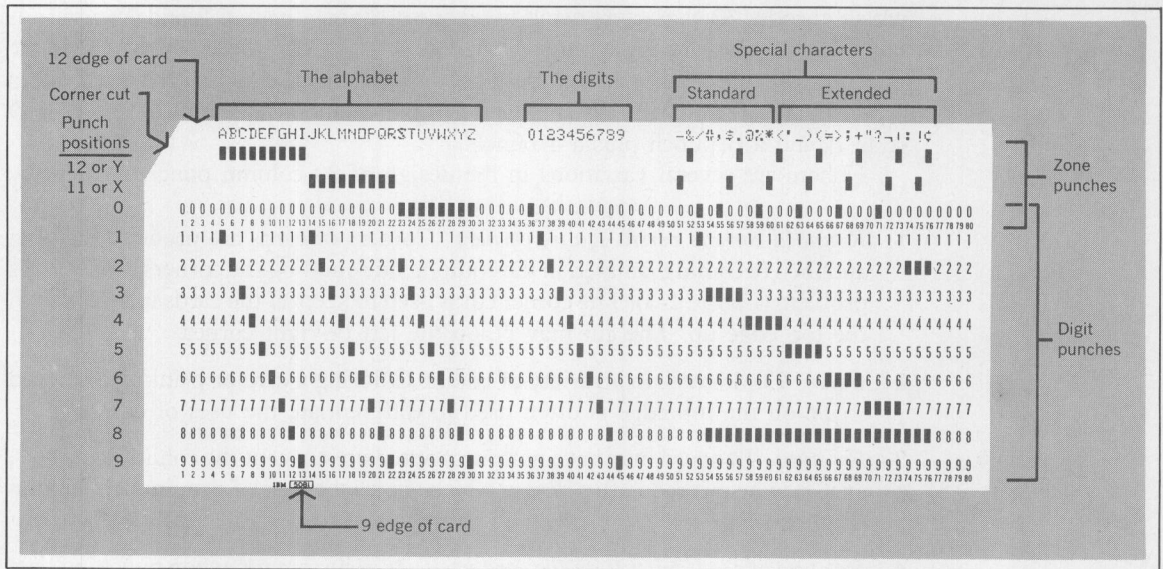


FIGURE 1-1 80-column Hollerith punched card.

in row 12, 11, or 0, and a numeric punch in rows 1 through 9. For example, the letter A is a 12 punch plus a 1 punch; the letter Z is a 0 punch plus a 9 punch. Special characters such as # and \$ are encoded by a zone punch and one or more numeric punches. The 96-column card is divided into three separate areas each of

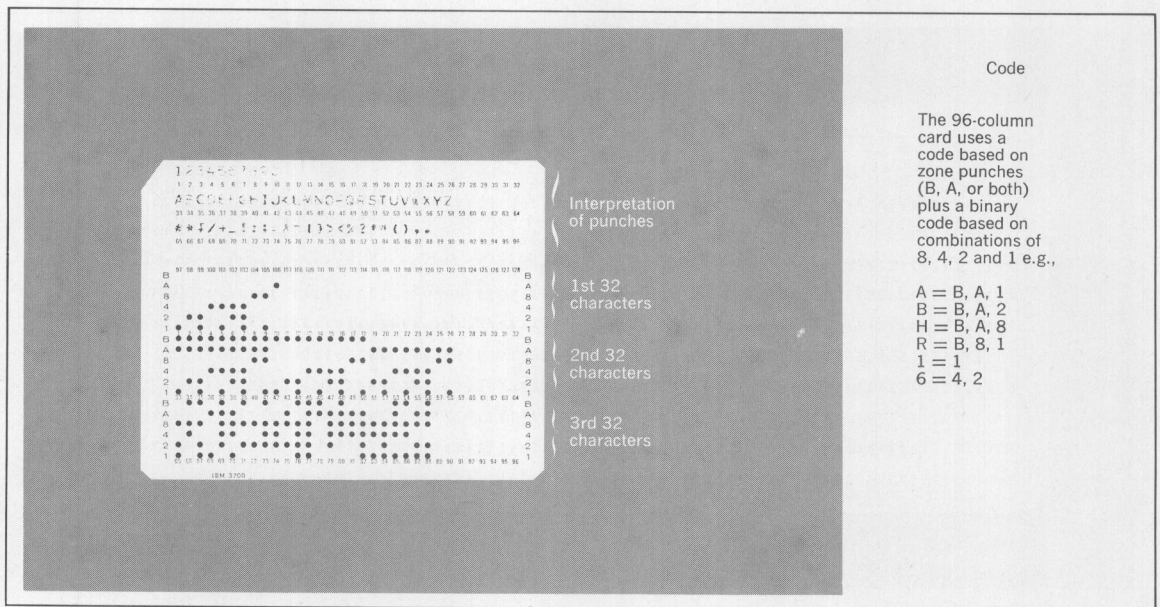


FIGURE 1-2 96-column card for IBM System/3.