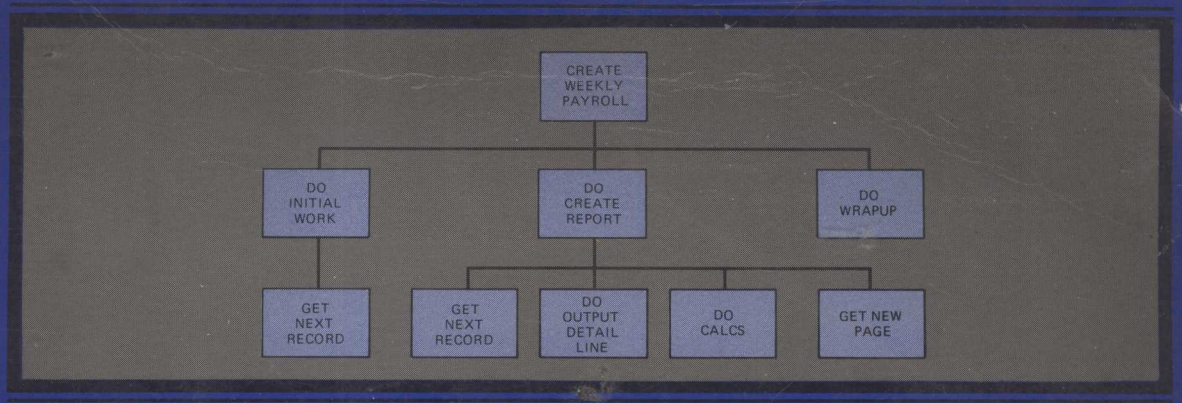
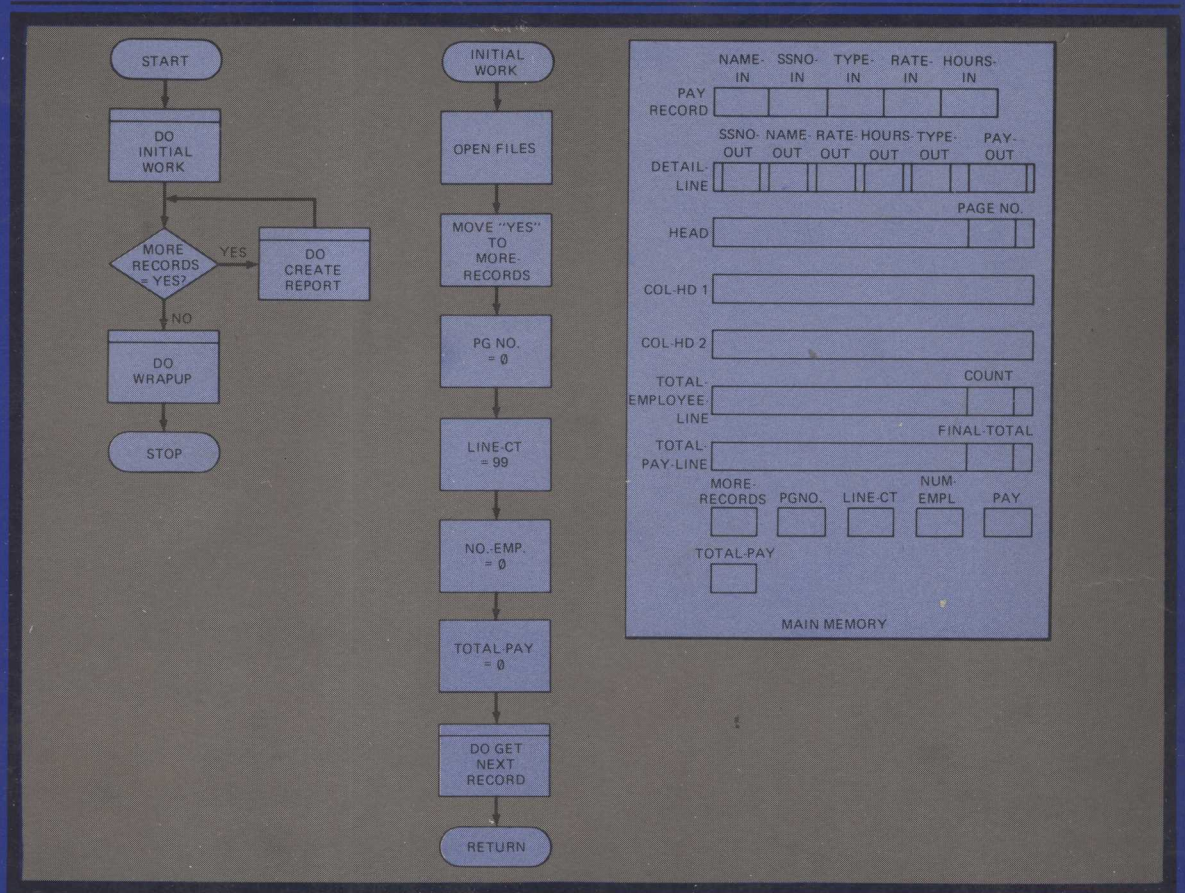


STRUCTURED PROBLEM ANALYSIS AND LOGIC DESIGN



Victor E. Broquard
J. William Westley

STRUCTURED PROBLEM ANALYSIS AND LOGIC DESIGN

Victor E. Broquard

J. William Westley

Illinois Central College

PRENTICE-HALL, INC., Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging in Publication Data

BROQUARD, VICTOR E.
Structured problem analysis and logic design.
Includes index.
1. Electronic digital computers—Programming.
I. Westley, J. William. II. Title.
QA76.6.B756 1985 001.64'2 84-18022
ISBN 0-13-854712-2

Editorial/production supervision and
interior design: *Esther S. Koehn*
Cover design: *20/20 Services, Inc.*
Manufacturing buyer: *Ed O'Dougherty*

To our families for their patience and indulgence.

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-854712-2 01

Prentice-Hall International, Inc., *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*

PREFACE

Data processing has emerged as one of the most popular and largest professional fields that a person can enter today. Since its infancy in the 1950s, there has been a continual change in the nature of the profession and the kind of knowledge and skills it requires. In addition, there has been a continued growth in the quantity of knowledge required of neophytes as they embark on their quest for membership in the profession. Unfortunately, however, the span of time within which this knowledge must be acquired has not expanded correspondingly. This is a problem that is particularly significant for community colleges.

In order to meet the challenge presented, two courses of action have emerged. The first has been the development of a greater degree of specialization within the profession. This is a phenomenon that occurs in nearly all professions as they mature. The second course of action is for educators and professionals to study and develop new ways of packaging the volume of knowledge so that it can be learned within the time available.

In order to raise the quality of programmer instruction, a course, Datpr 115, Computer Logic and Problem Solving Techniques was developed in the late 1960s as a part of the data processing curriculum at Illinois Central College, East Peoria, Illinois. The objective of the course was, and still is, to provide students with the ability to solve programming problems prior to learning a particular programming language. Over the years this course has provided a solid foundation for the programming students who have graduated from the data processing program.

As is so often the case, the unavailability of an appropriate textbook has made the teaching of this course somewhat difficult. A number of books have been written, but none has quite met the objectives of our course. As a result, much of the content of the course has been based on each instructor's perception of what constitutes good problem-solving logic.

Since the course was first implemented in 1967, a large volume of notes, practice problems, quizzes, and teaching techniques necessary to supplement the text used in the course has been accumulated. A careful weeding out process was conducted and much of the material that was out of date was discarded and new

material added. From a detailed outline, the materials were arranged into an easy-to-understand and logical sequence to form the basis for this textbook.

The philosophy upon which this textbook is based is that computer programming is a unique form of problem solving that uses the computer as the means of implementing a properly defined solution. From this philosophy, we see *programming* as consisting of two distinct activities: *problem solving* and *coding*. This means that the problem *must* be solved and tested first; then an appropriate language will be selected for implementing the solution on a computer. For this reason it is believed that a solution should be "generic" in nature. That is, the solution to a programming problem should not be developed with a particular computer or language in mind. From a practical viewpoint, however, it is recognized that students in many classes will be "gearing up" for a particular language and possibly for a specific computer. Their solutions will probably reflect that bias.

The text and examples are written using English words and phrases so that they can easily be understood by students as they study the development of programming solutions. It may appear that much of the text is COBOL-oriented; however, this is not the case. It would be impossible to write such a textbook without using the student's native language. Regardless of the computer language used to code the solution, students will develop their problem-solving skills using English words and phrases that describe the action they want the computer to perform.

We have elected to use flowcharting with ANSI Standard Flowchart Symbols as the best technique for developing and presenting solutions. Our experience has shown that students relate to and understand the use of symbols far more quickly and easily than they do pseudo-code or other techniques. This is especially true during the early stages of the learning process. After refining their problem-solving skills and with on-the-job experience, most programmers abandon flowcharting and develop their own problem solving techniques. It is interesting to note, however, that experienced programmers frequently revert to flowcharting when they are faced with a complex programming problem.

The term *structured programming* is more than just a popular buzzword. Experience has shown that by the use of a structured approach to problem solving, there is a basic set of logic principles and guidelines that, if followed carefully, can yield a reasonably satisfactory solution to a programming problem. Through examples, we describe these principles; and through class exercises, we attempt to reinforce them.

All problems, solutions, and exercises are representative of today's common business problems. We have included as well a chapter on interactive or real-time business applications, a topic not covered in other textbooks although it is becoming an increasingly important part of the programming environment.

We see this text as being suitable for problem-solving courses similar to the one for which the text material was developed. It would also be helpful to students who are learning to program in a language syntax course as a supplement to their language text. In addition, we believe that those who have acquired personal home computers will find it helpful as they learn to program.

We believe that our approach to problem analysis, although it is quite simple, describes the function of problem analysis in such a way that students can quickly develop an understanding of programming problems. Further, our top-down-design approach is compatible with the "structuring" that is so prevalent in the computer field today. The guidelines and checklists that are provided should be useful to beginners as they develop their problem-solving skills. These guides will serve to keep their solutions well organized and consistent with industry standards.

CONTENTS

chapter

1

THE COMPUTER: AN INTRODUCTION

Computer technology is changing at an accelerating rate. In the early years, a small computer would fill a gymnasium. Today, however, comparable computers are not much larger than a typewriter. And the cost of computers has shrunk from hundreds of thousands of dollars to just a few hundred dollars for a small computer. At first, only large companies could afford computers to help them solve their business problems, but today a computer is within the reach of even the smallest business.

During the past few years, the increased use of computers by business has become obvious. Think for a moment of all the establishments you have seen recently that had computer-related equipment in evidence. Undoubtedly, you've noticed them in such places as grocery stores, department stores, banks, gas stations, hospitals, and shopping centers. The list gets to be quite lengthy! Since computers have become so commonplace in our society and we are affected by them daily, it is necessary for us to know something about them—what they are, how they work, what they can be used for, and how to make them work for us.

What is a computer? The dictionary tells us that a computer is "a calculator especially designed for the solution of complex mathematical problems; specifically, an automatic electronic machine for performing simple and complex calculations." This definition gives us only a limited idea of what a computer is and does. It is not a good "action definition." Perhaps a better definition of a computer would be "an electronic machine that receives instructions it can recognize, processes and/or performs the required calculations using those instructions, and presents the results to its user quickly, accurately, and in a usable form."

A user is anyone who makes use of the output from a computer. For example, accountants use a printed general ledger, billing clerks mail out the results from a monthly billing program and the payroll department distributes the checks produced by a payroll program. These are only three examples of the many users of computer output.

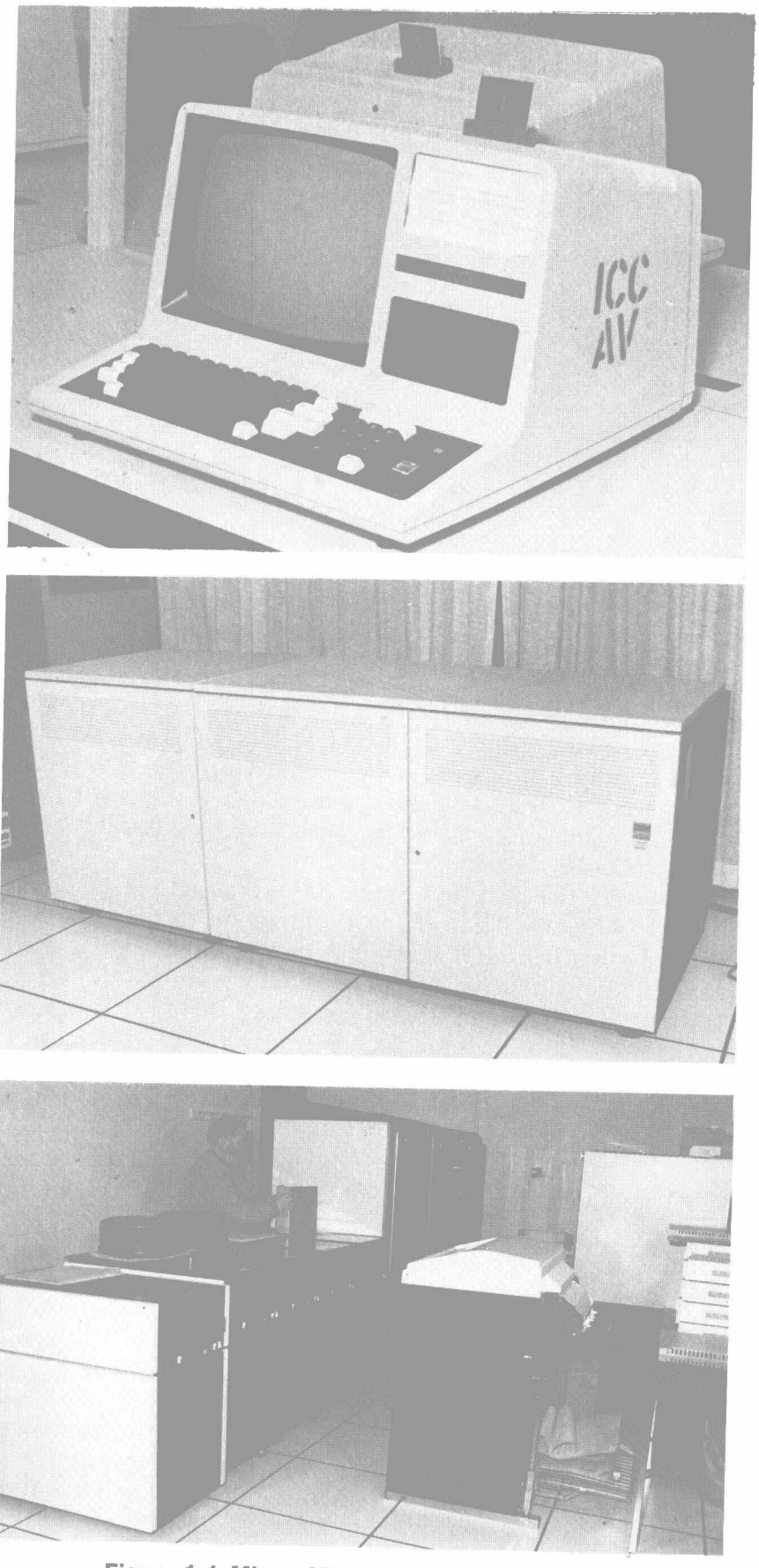


Figure 1-1 Micro, Mini, and Maxi Computers

What is the need for computers? Why are they used? Computers are known for their speed and accuracy, but these are only secondary reasons for using computers. The main reason and justification is that they solve a wide variety of problems for people quickly and inexpensively. In businesses, the computer center is the place where many of the data processing needs of its users are met. If these needs are not met, the center can run into countless difficulties.

Think of a time when you came face to face with a computer—for example, an automatic teller machine connected to a bank's computer. As you approached it, did it do anything? Not likely! When did the machine go into action and do something? Only when you took the initiative and gave it some data did anything happen. You either pushed a button, inserted a card, or identified yourself in some manner. An automatic teller can perform a number of tasks—for example, it can find your current balance, accept your deposit, or process your withdrawal. As soon as it has completed its task, it communicates with you by telling the status of your account or by giving you the correct amount of money. Only when a computer has been given data or instructions can it do anything.

To the user of a computer, each part of the definition is important. Have you ever approached a computer-driven system and been told, "Sorry, the computer is down—you'll have to wait 10 minutes"? Have you ever entered your identification number only to have the computer do nothing? Have you ever received a garbled, undecipherable printing of your bank statement? Have you entered your identification code only to have the bank terminal fail to respond? Have you ever asked for $20 and received only $10 when using your bank card? You probably answered "no" to all these situations. If you gave a "yes" response to any one of them, it indicates a possible problem with the system, or perhaps a problem with the way you communicated with the system. For the users of computers, a failure or breakdown of any part of the expected service yields instant frustration.

BASIC COMPUTER FUNCTIONS

From an analysis of the definition and the descriptions given, it can be seen that computers perform four basic functions:

1. They input data or information and output data or information.
2. They perform arithmetic operations (addition, subtraction, multiplication, division).
3. They perform logical operations (such as determining if one number is larger or smaller than another).
4. They store or retain data and instructions in the computer memory.

Whether the computer is a large one processing airline reservations for thousands of passengers or a tiny one operating as an arcade game, it can perform only these four functions. These may appear to be minimal and not particularly powerful; however, it is the ability of the computer to do these things quickly and reliably that makes it unique when compared to manual processing. All these operations are performed by electronic circuits that rarely fail and, depending on the specific computer, can execute thousands to millions of additions per second.

DATA PROCESSING

This leads us to the question, What is *data processing?* First, data may be defined as a collection of facts, information, concepts, ideas, or instructions presented in a form that can be used by both humans and machines. For example, data could be a person's name, Social Security number, insurance-policy number, salary, or grade earned in a course. Data processing is, then, the processing or handling of data. More specifically, data processing is the procedure for getting the right information to the right person at the right time.

Even small businesses can have sufficient amounts of data to justify processing by computer. Customer invoices with items, quantities, prices, and dates; stock location with items, quantities, plant locations, reorder quantities, part descriptions; payroll information with hours worked, rates, names, Social Security numbers, federal and state taxes, and pay dates—all must be maintained and kept up to date. Regardless of size, a business will have a data processing department or someone who is responsible for handling all these data. Since customers must be billed correctly and on time, parts must be ordered before the stock is depleted, and employees must be paid accurately and on time, a computer may be desirable. The larger the business, the greater the task of data processing. Hence the need for computers.

COMPUTER INSTRUCTIONS

In order to perform these functions, all computers have sets of instructions. These include instructions that input data into the memory, those that output data from the memory, those that perform mathematical operations, those that give the computer its ability to make logical decisions, those that move data from one location in memory to another, and finally, those that serve in a control capacity. In addition, each computer has one or more languages that are used by the programmer to implement the solutions to his or her programming problems.

A computer instruction is made up of two parts. The first is the *operations code,* which identifies the specific operation to be executed by the computer. The second is the *address,* which gives the location in memory where data are to be obtained for processing, where the result is to be placed after execution of the instruction, or where the next instruction to be executed is. Depending on the specific instruction and the language involved, addresses may be explicit or implied, or may reference data in a special-purpose area of the computer called a *register.* Some instruction formats may even require more than one address. Consider the following instruction: ADD HOURS-WORKED TO TOTAL-HOURS.

The word ADD is the operation that you want to have executed (the operations code). HOURS-WORKED is the name given to the location in memory (the address) where the data to be added will be found. TOTAL-HOURS is the name of the location in memory (the address) where the other factor in the ADD instruction will be found and where the result of the calculation will be placed.

The format and syntax in which the instructions are written depend on the specific language selected for coding the solution. Nevertheless, all language instructions will contain these two elements, an operations code and one or more addresses which may be either explicit or implied. The study of specific languages makes this evident.

When selected and combined into a logically organized sequence, these instructions make up a *computer program.* They tell the computer what to do, how to do it, and the sequence to follow when executing the instructions. *Software* is a general term that refers to all programs that run on a computer.

A *programmer* is the person who selects, organizes, and assembles these instructions (writes the program). For example, the accounting department of a company decides it would like the computer to print out the customers' bills each month, thereby saving a lot of manual labor. The task is given to a programmer, who studies the problem, plans the logical steps to be followed, and codes the computer program to accomplish the goal.

FORMS OF DATA

Throughout our discussion, we have made frequent references to the *data* being processed. Data take on a number of different names, depending on the size of the data item and its relationship to other data items. The smallest, a *character* of data, is a single digit, 0–9; a letter, A–Z; or a special character, such as a $ or %. Seldom does one character of data make much sense all by itself, except perhaps when referring to a grade in a course or a child's age.

A *field* of data is one or more characters of data that represent a single entity. An account number; a person's name, age, sex, or hours worked; and a part number are all examples of fields of data. Each field is given a name by which it may be referenced and a length indicating the number of characters associated with it and the kind of data (numbers, letters, or special characters) it represents. An account number may be given the name ACCOUNT-NUMBER and be defined as being nine characters in length; a name field called NAME may be 15 characters long; and so on. (*Note:* The naming of data fields is usually language-dependent and must conform to the syntax of the language. The names of the data fields in the examples in this book are intended to be descriptive of the data themselves and are not related to any specific language.) Fields of data are quite useless unless they are grouped into a meaningful set. Such a meaningful set is called a *record.*

A record of data is made up of one or more fields that are related and refer to one whole person, place, or thing. For example, a customer invoice record will contain all the fields of data that relate to that one invoice—customer name, account number, date, item ordered, description, and price. It makes very little sense to have a record that contains unrelated data, such as the name for Customer 1 with the address for Customer 2 and the order for Customer 3.

In the case of a customer invoice, once the computer has read the record, it has all the fields of related data for Customer 1. It can then do the required processing and create a billing statement for that customer. Now, this process handles the billing for one customer, but what about the other customers? These records must likewise be grouped into a logical set, called a file.

A *file* of data is a collection of all the related records of data organized for a specific use. If your purpose is to produce monthly customer statements, then you would have the computer read and process the customer-accounts file of data. Each record on the file contains all the data for one customer. (*Note:* It is possible to have a file that has no records in it yet. This occurs when you first begin operations. If you haven't sold anything yet, you have no customer records.)

A company will typically have a number of files in order to carry on its business. You might expect files for customer accounts, orders, inventory, general business accounting, and so forth. In today's business world, the concept of files, records, and fields is one you must understand and be able to use.

THE DATA PROCESSING CYCLE

When processing data, most computers in business follow a three-stage sequence called the *data processing cycle.*

Take the task of creating a customer's monthly bill. First, the computer *inputs* one customer's data (record). Then it *processes* that customer's data by calculating the balance due and moving the result and other data to the output area. Next, it *outputs* (prints) the customer's statement. The computer continues the cycle by inputting the next customer's record, processing and outputting it. This cycle is repeated until the entire file of customer records is processed or the program is stopped by the operator.

Figure 1-2 Data Processing Cycle

A COMPUTER SYSTEM

A *computer system* consists of four components that fulfill the functions of the data processing cycle. The heart of the computer system is the *main processor.* A wide variety of devices can be attached to the main processor for the purpose of inputting data, and numerous devices are used to output data. The last component is the *console.* It is through the console that the person operating the computer can enter instructions, make inquiries, determine the status of the processing being done, or respond to messages outputted by the central processor to the console. A fifth component, *auxiliary storage,* may be added to expand the storage capabilities of a computer system. All these devices are classified under the general term *hardware.*

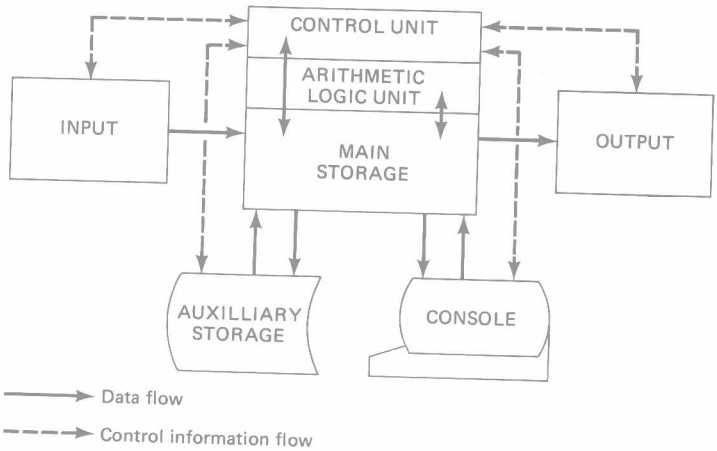


Figure 1-3 Computer System Functional Diagram

Figure 1-3 shows the functional relationship among the various components that make up a computer system. Hardware is all of the electronic and mechanical devices collectively that make up a computer system. We will next examine some of the devices that constitute a computer system.

Input, output, and auxiliary devices are many and varied. The ones chosen for use on a particular system depend in part on the form of the data to be processed. Only a few of the most commonly used devices will be described here.

INPUT MEDIA AND DEVICES

For years, the industry standard for input has been the *punched card.* Letters and numbers are represented by unique combinations of holes punched into

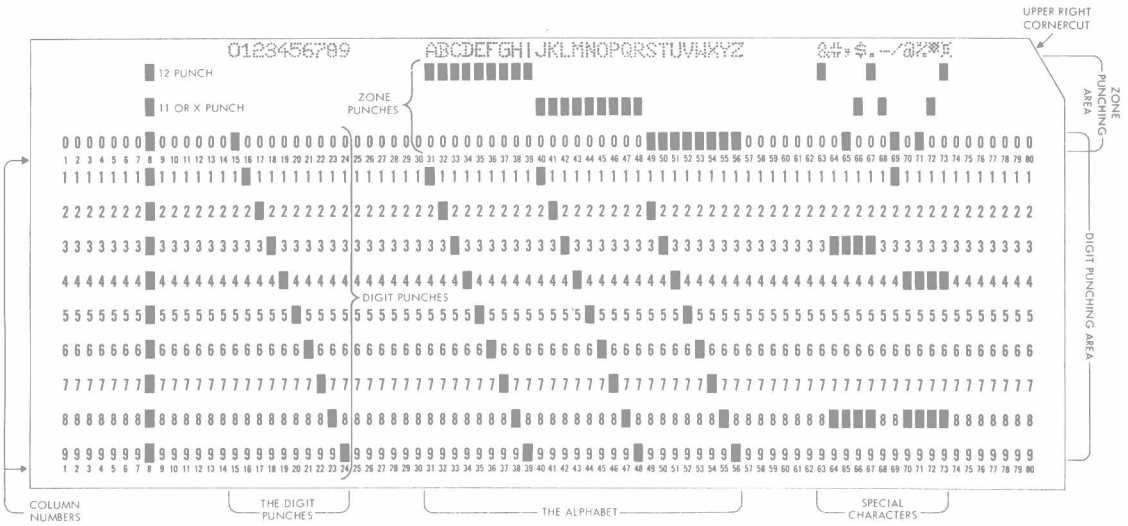


Figure 1-4 Punched Card and Card Reader

a card. A device called a *card reader* is needed to input (read) data from the punched cards. The card reader senses the holes in the card and converts them into electronic signals that represent the character or number. The most common card, measuring approximately 3 by 8 inches, can hold a maximum of 80 characters. The speed of card readers ranges from 200 to nearly 1,200 cards per minute. This translates to a maximum input speed of approximately 1,600 characters per second. This may seem fast, but in terms of the speed at which today's computers can handle data, it is very slow. Another drawback to using cards is the necessity to correct errors by punching an entirely new card. Punched cards are used as output only when a special need arises and will not be discussed further.

An alternative to the punched card is the *cathode ray tube* (*CRT*), also called an input terminal. This device looks much like a small television set with a typewriter keyboard attached. When you type the data, they are stored as electronic impulses and are simultaneously displayed on the screen. When all the data have been entered and verified for correctness, the operator depresses another key, thereby sending the data from the CRT to the computer at extremely high speeds. The speed at which the data are entered into the terminal is limited only by the speed and accuracy of the typist. A principal advantage to this device is that the data are entered into the computer immediately, rather than indirectly through the punched card or some other medium. In addition, the correction of errors is much easier and faster.

Figure 1-5 Cathode Ray Tube (CRT)

OUTPUT MEDIA AND DEVICES

Just as there are a number of devices for getting data into a computer, there are also a number of devices for getting output from the computer. The primary consideration when selecting the output device is how the output is to be used. Will it be used by people or by the computer?

The printed report is the most common form of human readable output. Whether letter-sized reports or premium-due notices, printed reports are written by a device called a *printer*. Printers come in a wide range of speed, price, and quality of printing. Some print just a few characters per second, some print a line at a time, and others can print an entire page at a time.

Although there are a variety of criteria for selection, the choice of printers is frequently based on the volume of printing to be done. The greater the volume of printing, the faster the printer should be. Slower printers have the advantage of being lower in price and may have a higher quality of print.

CRT terminals can be used for output as well as input. The computer, under the direction of a program, causes the results of the processing to be displayed on the screen, where they are viewed by the operator. When the operator has read the material presented, he or she may then instruct the CRT to display another screen of data or to take other action. The operator continues until the required information has been viewed or the end of file has been reached.

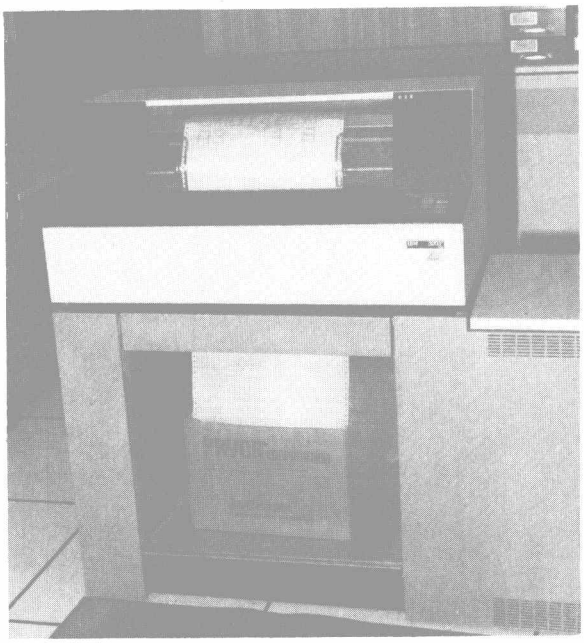


Figure 1-6 Line Printer

AUXILIARY STORAGE MEDIA AND DEVICES

Auxiliary storage serves as an extension of the computer's memory by providing storage for large volumes of data in a relatively permanent form. On them you will find many key master files of a business, *i.e.*, customer account files, parts inventory files, employee files, and so forth. When it is necessary to store data, they are written onto the storage medium under the direction of the computer. When it is necessary to retrieve the data, the computer accesses them and causes them to be returned to the memory of the computer. In addition, auxiliary storage devices are used for input and output. When you take a tour of a data center, you can expect to see one or more of these devices.

Magnetic tapes which run on tape drives serve in a dual capacity, storing data from the computer as well as inputting it to the computer. Reels of magnetic tape

come in various sizes, from "mini" reels with 800 feet of tape to large reels holding 2,400 feet. Computer magnetic tape is much like the tape used in music cassettes. It differs in that it is wider (1/2 inch) and is much heavier. Characters of data are stored as magnetic spots arranged according to a standardized coding pattern. A 2,400-foot reel of tape can store as many as 2 million characters, or 22,000 punched cards. Data can be read into the computer at rates up to 1 1/4 million characters per second. (Surprisingly enough, this is still not as fast as the computer's capability to accept data.)

Another auxiliary storage medium is the *magnetic disk*. On larger computer systems, magnetic-disk devices are commonly used. A *disk pack* (the medium upon which the data are stored) looks somewhat like one or more LP records stacked on a center spindle, with each platter spaced about an inch above the next. The flat surfaces are smooth and covered with a magnetic oxide coating. Data are recorded in a coded pattern of magnetic spots and stored in concentric circles on the disk surface. Disks spin at approximately 3,600 revolutions per minute.

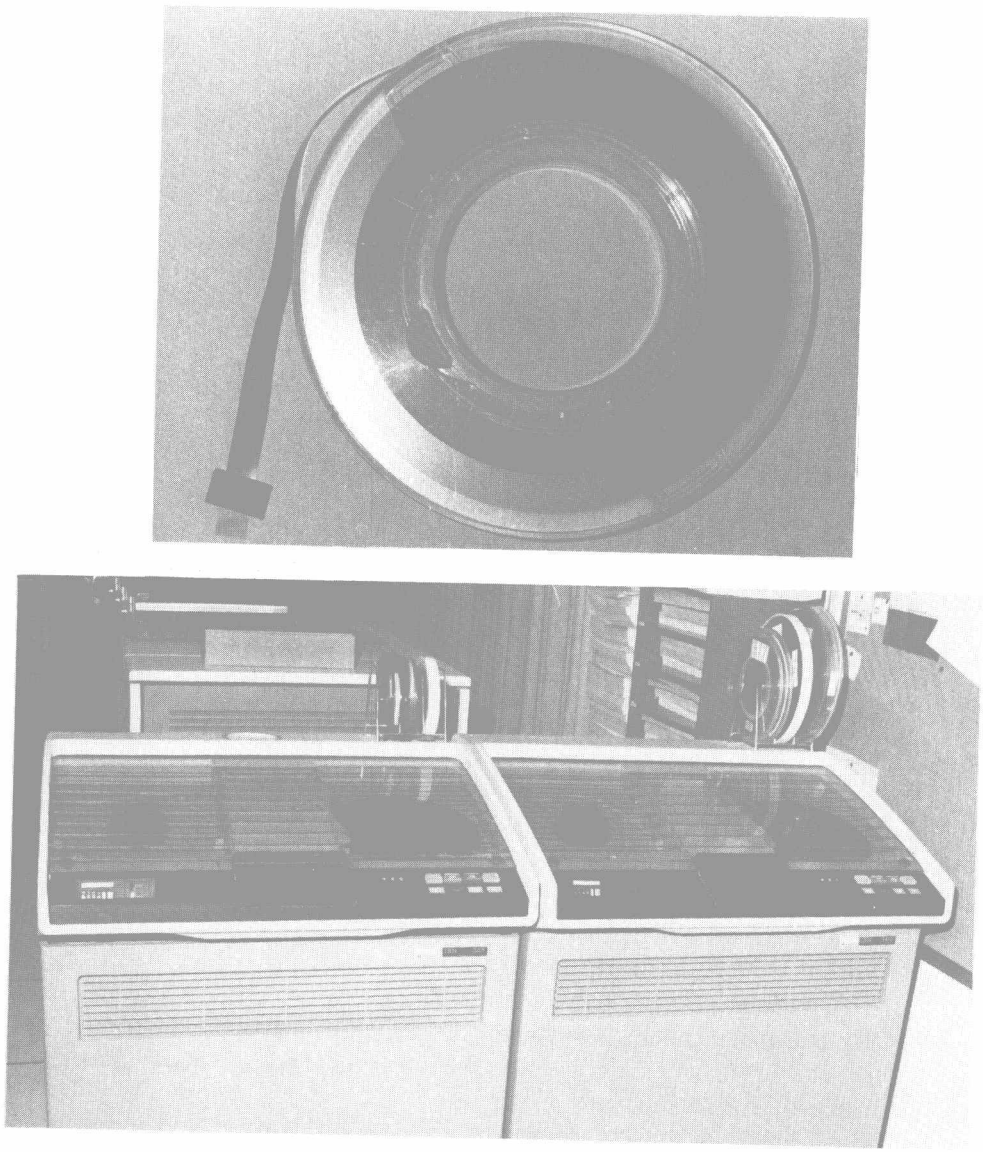


Figure 1-7 Magnetic Tape and Drive