TAB 1406

$11.95

MICROCOMPUTERS

BY A. J. DIRKSEN

MICROCOMPUTERS

What they are *and* how to put them to productive use!

BY A. J. DIRKSEN

TAB TAB BOOKS Inc.
BLUE RIDGE SUMMIT, PA. 17214

FIRST EDITION

FIRST PRINTING

Originally printed in the Dutch language
by Kluwer Technische Boeken BV - Deventer © 1978

Published in the English language in 1982 by TAB BOOKS Inc.

This edition not for sale outside the United States of America, its possessions, and the Phillippines.

Printed in the United States of America

Library of Congress Cataloging in Publication

Dirksen, A.J.
Microcomputers.

Includes index.
1. Microcomputers. I. Title.
QA76.5.D5623   001.64'04   81-18266
ISBN 0-8306-0063-9   AACR2
ISBN 0-8306-1406-0 (pbk.)

Introduction

This book was translated from the original Dutch text. During this process, the 8080 chip went out of production. There are, however, still many 8080 systems around.

Even though the illustrations and examples draw heavily on the 8080 chip, this book is by no means obsolete. The techniques explained here are applicable to any microcomputer system and, in most cases, directly translatable to the Z80, which remains the most widely used chip today.

Contents

Chapter 1

What Is a Computer?

As the name implies the word *computer* comes from the term to compute, meaning to calculate. Early computers were referred to as calculators. In this book we shall confine ourselves to the term "computer."

In addition to doing calculations a computer can perform processes which, at first, appear to have little to do with calculating. This can be translating texts, adding words to texts (or deleting them), transferring data, bookkeeping, and process control.

In this book a computer will be considered as a device with which data can be processed, using a program. We can represent this data as a series of binary digits. This series of digits (ones and zeros) is translated into electronic signals. We shall, therefore, be speaking about an electronic device, the *digital computer*.

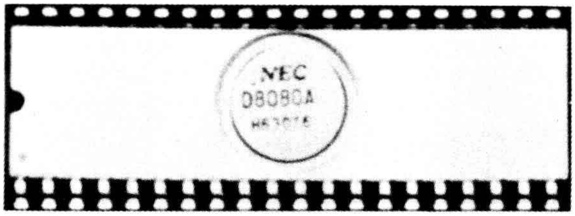

Fig. 1-1.

The definition of *microcomputer* refers to computers in which the greater part of the electronic circuitry is contained in one integrated circuit (IC). They differ from 'normal' computers only in their size and price. Such an IC has a surface area of 6.75 cm^2 (fig. 1-1). The circuitry is contained in an area of only 1 cm^2. Together, the electronic circuits perform a *series* of processes on the data fed to them. Because the electronic circuits are assembled on a very small area, the IC is called a *microprocessor.* The price of microprocessors is so low that by combining a microprocessor with a memory and an input/output device one can make an inexpensive microcomputer.

The purpose of this book is to provide the knowledge for understanding microcomputers. This knowledge will relate to the ***hardware***, i.e., the choice of equipment, and the *software*, i.e., programming as a whole.

In this chapter we will discuss a block diagram of a computer and the basic principles of computer programming.

THE COMPUTER IS AN ELECTRONIC DEVICE

The block diagram in fig. 1-2a is applicable to any electronic system. The input signal converter changes the given information into an electronic signal which is processed by the signal processing unit. The processed signal is then converted by the output signal converter into a non-electrical signal.

In fig. 1-2b an audio-amplifier system is shown using the same diagram as in fig. 1-2a. The signal processing unit is called the amplifier. The microphone is the input signal converter which changes the given information (sound) into an electrical signal (alternating current). The

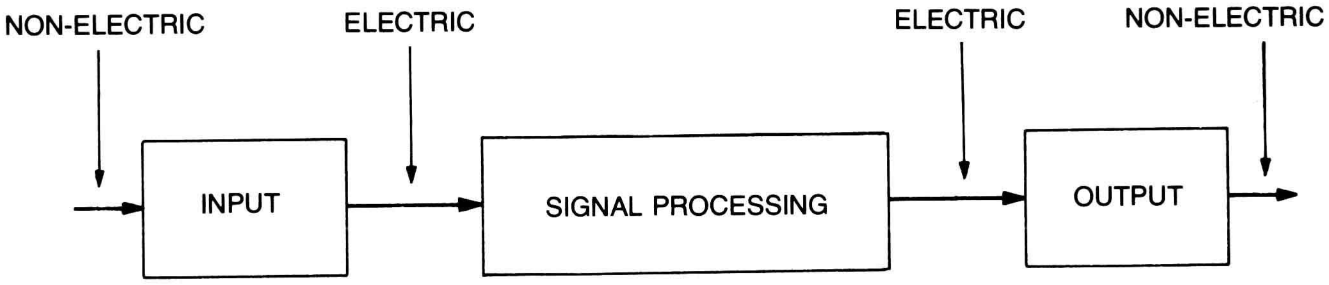

Fig. 1-2A.

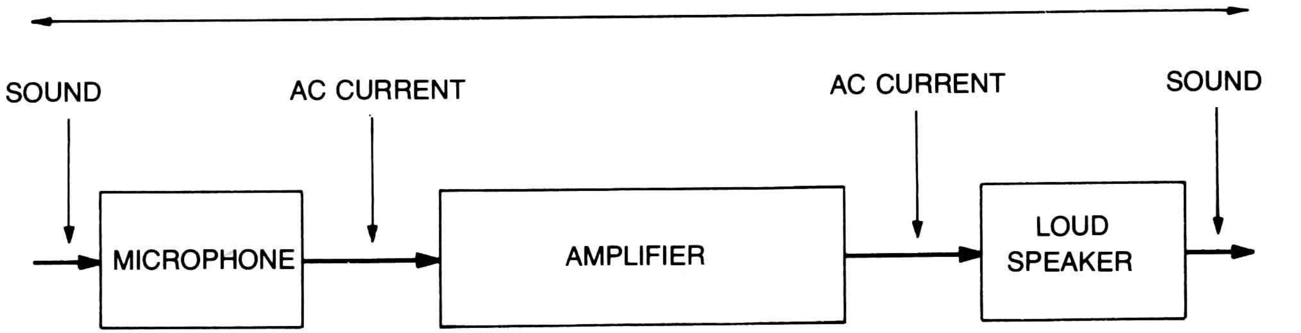

Fig. 1-2B.

loudspeaker does the opposite. For computers the terms given in fig. 1-3 are used.

The *Input* device converts data; for example, a *paper tape reader* converts a code consisting of holes on a punched tape into electrical impulses. This data is processed in the *Central Processing Unit*, the CPU. The *Output* device, for example a line printer, translates these electrical impulses into text. These Input and Output devices collectively are called I/O devices.

HOW IS INFORMATION PROCESSED?

A computer processes information exactly as we do. We will first see an example of how a *person* does this, in order to understand better how a computer works. This example can be seen in fig. 1-4.

If we want to process information we have to use our memory. The following is a simple example. You are given the following information: the next 3 numbers are the data—2, 5, and 3. Multiply the first number by the third and store the answer. Give the result.

To complete this task you have had to continually use your memory. While feeding in the data, you had to store the numbers, 2, 5, and 3 in your memory. You also had to store the result of the multiplication in your memory. The *data* (2, 5, 3) and the *program* (multiplying the first and third numbers together) were fed in (input) through your sense of sight. In order to feed-in this information, light was changed into electrical impulses which reached your brain through the nervous system and then stored in your memory. Your brain then processed this information under the direction of the program. The result which is now present in your memory is fed back through your mouth - the output device. This output device says, 'six.'

The program which has been described here is so simple that you could execute it immediately. In other cases you might not have sufficient knowledge of the facts to complete the task given. You might then refer to a card file and take data from there. You would read this data into your memory in order to be able to return to the processing of the program.

BLOCK DIAGRAM OF A COMPUTER

Refer to fig. 1-5 throughout this section.

Information

When we refer to information or data in this context we are talking about a task which is given to a computer. It can be in the form of letters, punctuation marks, numbers, or impulses related to a given action. This information is stored in the *main memory.*

Program

A program comprises a number of instructions. All the instructions that a computer can execute are contained in the *instruction set*. (See Appendix A.) The data which is to be processed *and* the program to be executed are stored in the main memory.

Input

Data and the program are changed into electric signals using the Input device. These electronic signals activate certain sections of the main memory so that their condition reflects the data and the program which has been stored. Input devices will be studied later in this chapter. A more detailed description will be given in chapter 20.

Control Unit

The control unit directs *every activity* which takes place between the various components of the computer. The control unit may make use of a number of *registers* for temporary storage. One of these registers is the instruction register, in which the instruction to be executed, after being taken from main memory, is stored. The instruction which, at a given moment, is in the instruction register tells the control unit:

- The location in which the data is stored. It could be in main memory or a register.
- Where this data must be transferred. Again, to main memory or a register.
- What operation must be executed?

Under the direction of the program the control unit regulates the flow of information in the computer. When an instruction has been completed, the *program counter* in the control unit sends a signal. This signal "fetches" the next instruction from the main memory and stores it in the instruction register.

Arithmetic and Logic Unit

The Arithmetic and Logic Unit, generally called the ALU, should be viewed as an assistant to the control unit. The ALU can perform *arithmetic* operations such as adding and subtracting and *logical* operations. It can manipulate numbers, using the AND, OR, or EXOR functions. The ALU also has a register at its disposal usually called the *accumulator.*

When the instruction that is in the instruction register of the control unit demands that a fixed arithmetic or

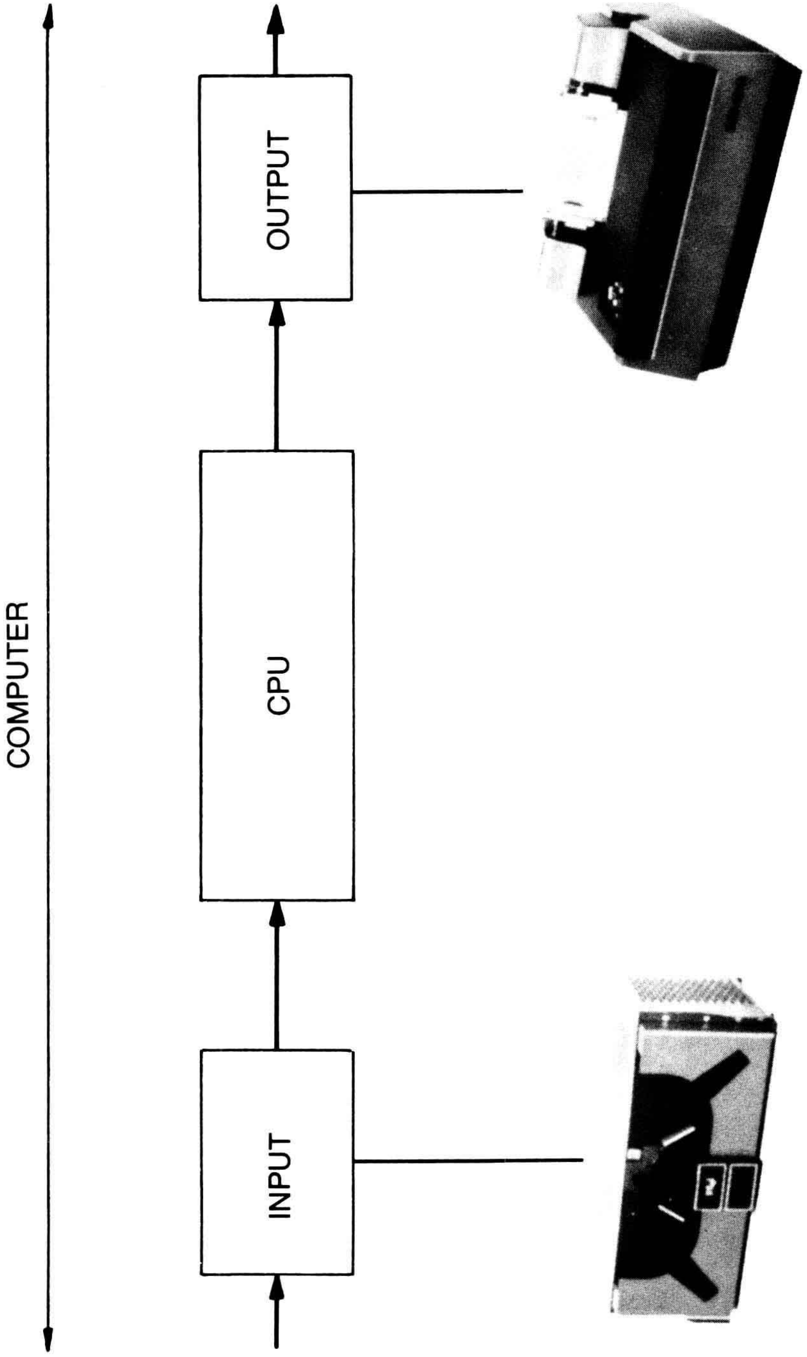

Fig. 1-3.

logic operation is to be performed on certain data, this data is taken out of the main memory or register and presented to the ALU. The ALU then processes the data as instructed by the control unit. After the operation has been performed, the result - new data - is stored in the accumulator.

Central Processing Unit

In large computers, used for administrative and scientific applications, the central processing unit is considered to be a combination of the control, arithmetic and memory sections.

With microcomputers, the central processing unit may be seen as a combination of the control unit and arithmetic and logic unit, together with the associated registers that serve as memory space. This terminology has been developed because in microcomputers the control unit and arithmetic and logic unit are contained on one chip and are physically separate from the main memory.

Because this course concerns itself with the working of microcomputers, we will confine ourselves to the terminology in current use. In other words, the term CPU includes the control unit, the arithmetic and logic unit and its associated registers.

Output

The Output devices present the result of operations in legible form. On the instruction of the computer, or the computer operator, the signals representing the results of the operations are either printed or displayed.

MEMORY

To be able to process data according to a given program, a computer must have a memory at its disposal in which the instructions and the data to be processed are stored. The memory in a computer system can be divided as follows:

The Main Memory

The following are stored in the main memory:

●The instructions, which together form the program.

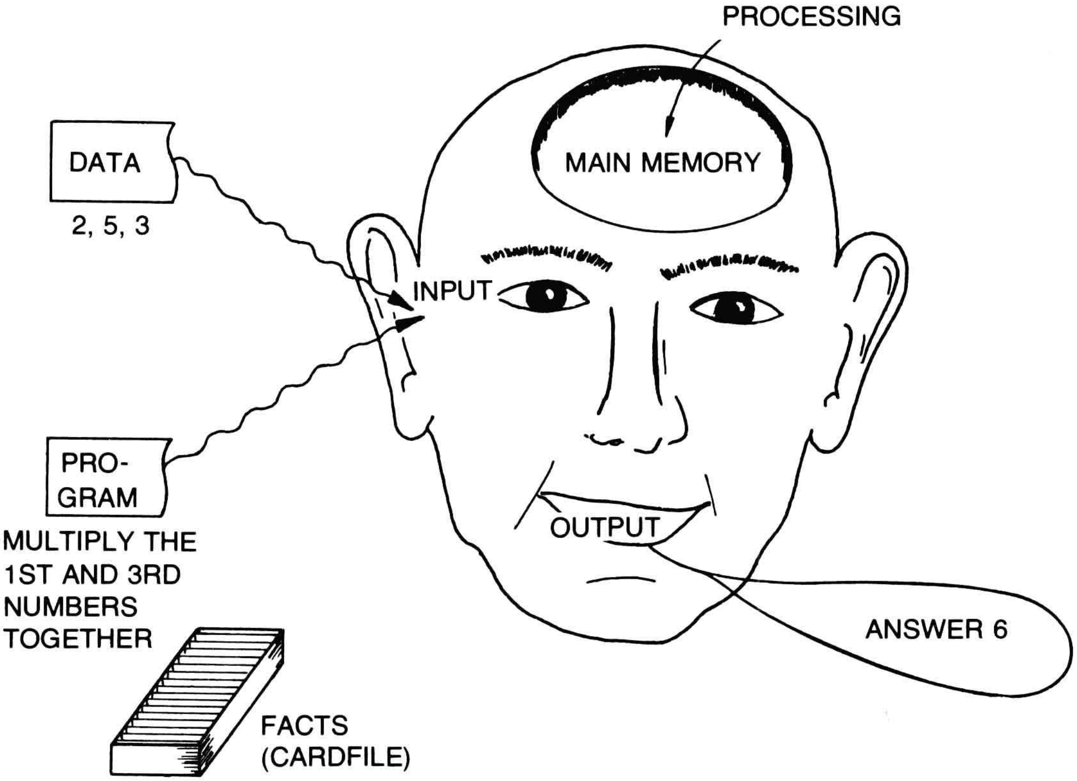

Fig. 1-4.

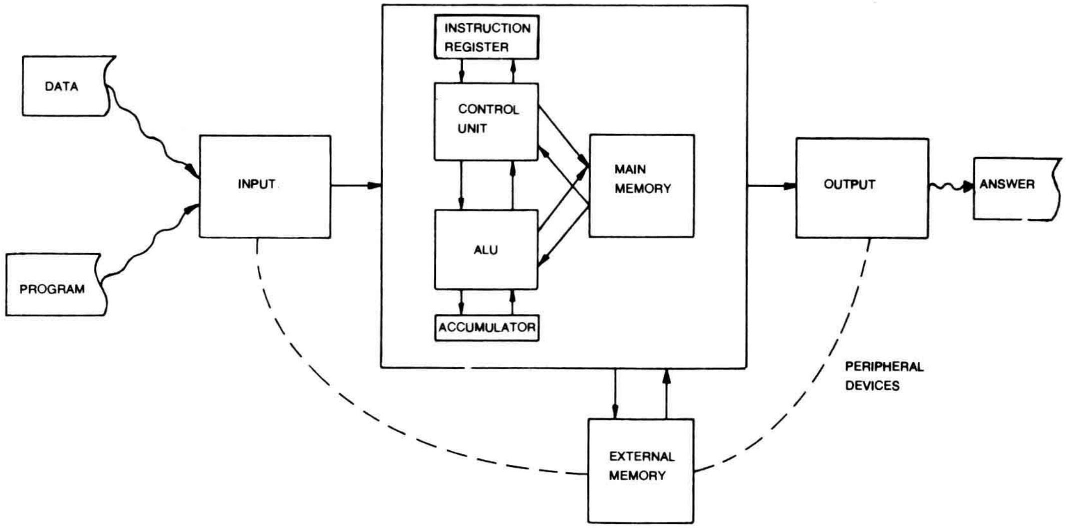

Fig. 1-5.

●The data which is to be processed in a given situation.

The part of the main memory in which the instructions are stored is called the *program memory*. When a computer is to be used only for a special purpose, the program can be permanently stored in the main memory. Such a computer is called a special-purpose or *dedicated computer*. This is often the case with microcomputers. According to the requirements of the user, the manufacturer provides a preprogrammed memory.

If the computer is general purpose, the program must be exchangeable. This, for example, is true when a computer is used for salary administration and stock control for different firms. In order to execute a certain program, that program has to be *read in* to the main memory.

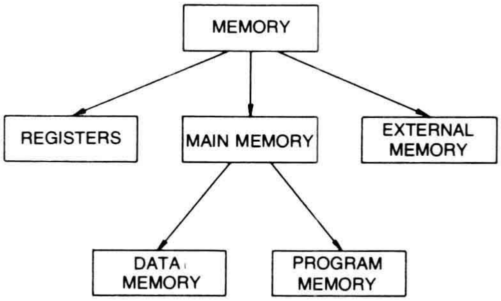

Fig. 1-6.

Note

Preprogrammed memories can also be used with general-purpose computers as long as the number of programs is limited and they don't occupy too much memory space. The various main programs can then be stored in the remaining memory and executed as required.

The section of the memory where data is temporarily stored is called the *scratch-pad* memory. This part of the memory can be seen as fulfilling the same function as a scratch-pad. It only retains the results of an operation until they can be transferred to the output device. Even a dedicated computer must have some scratch-pad memory.

External Memory

Whenever data or programs must be stored for future processing or reference, *external memory* is used. In most cases this is in the form of a magnetic tape or disc. Figure 1-7 shows an example of a magnetic tape unit. In microcomputers, this magnetic tape usually comes in the form of a cassette. This cassette is similar to the cassette found in modern audio cassette recorders.

The magnetic disc is called a floppy disc. It is made from flexible magnetic material and is stored in a sort of envelope. A disc closely resembles a flexible stereo record, except for the fact that it can be erased and used again just like a magnetic tape. The unit that "plays" the disc is shown in fig. 1-8.

Fig. 1-7.

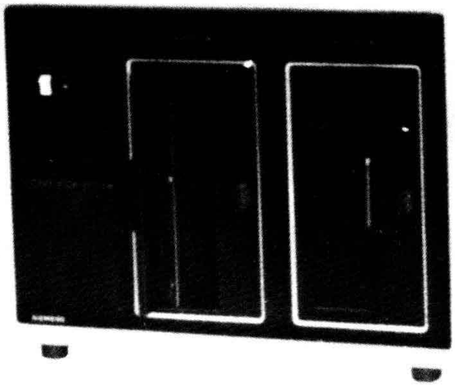

Fig. 1-8.

Registers

Registers to store data temporarily are spread throughout the entire CPU. In addition to the control unit and the ALU, sometimes the Input/Output devices also contain such registers.

Note: The word computer is sometimes *not* meant to represent the installation as a whole, but rather that part which is *exclusive* of the peripheral devices. The system as a whole, including the peripheral devices, is called a computer system or computer installation. In this book, in order to avoid confusion, we shall be referring to the system as a whole when we say computer.

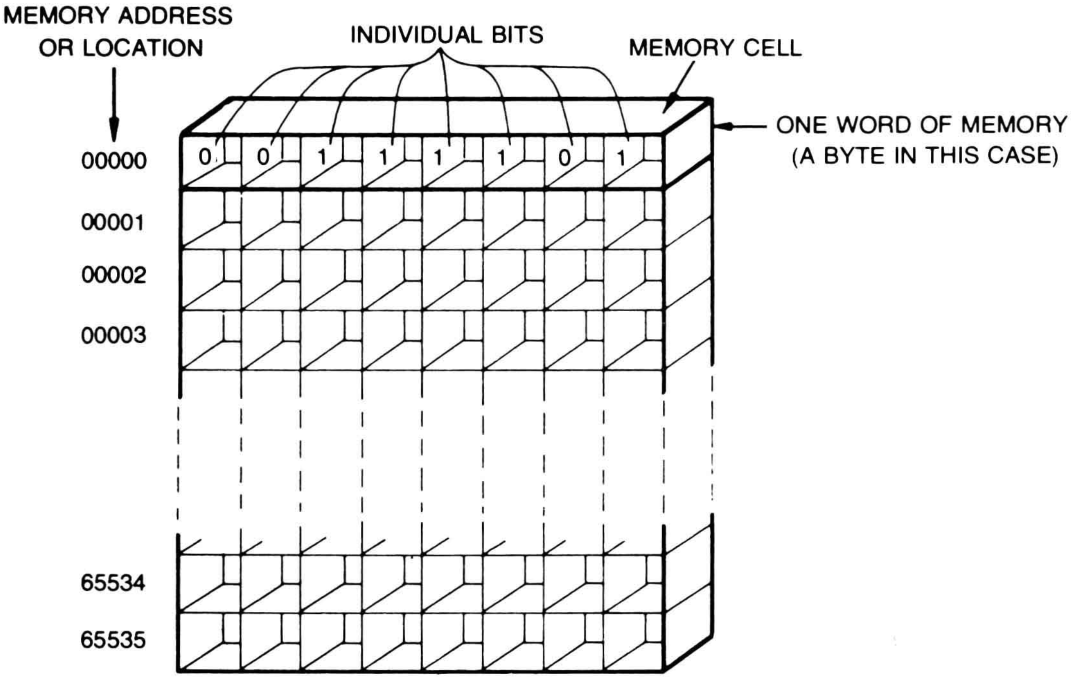

Fig. 1-9.

ORGANIZATION OF THE MAIN MEMORY

The instructions and data stored in the main memory must be easily retrievable. The memory is thus divided into blocks of equal size (fig. 1-9). These blocks are called *memory locations, memory addresses,* or *byte*. A memory location is divided into *memory cells.*

In most microcomputers, the memory location contains eight memory cells. In such a case we speak of an 8-bit computer. Each cell can contain a single binary number (1 or 0). Each memory location can thus contain 8 bits (*b*inary dig *its*). Because 8 bits = 1 byte, we say that the word length of a microcomputer is 1 byte.

The zeros and ones in the 8 memory cells together form the *contents of the memory location.* Each memory location has an *address*. Most microcomputers can address a maximum of $2^{16} = 65.536$ addresses, so there is no point in having a larger main memory, since extra memory locations couldn't be addressed and therefore couldn't be used.

INPUT/OUTPUT DEVICES

You will see these abbreviated as I/O Devices.

Paper Tape Reader and Paper Tape Punch

Punched tape is often used to feed data to microcomputers. This punched tape is a continuous thin strip of paper on which data is recorded by means of round or square holes. A character on the punched tape consists of a number of holes in line across the breadth of the tape. Depending upon the code used, this could be 5, 6, 7 or 8 holes. Figure 1-10 is an example of a tape with a maximum of 7 holes. Using these 7 holes we can code a total of $2^7 = 128$ different characters.

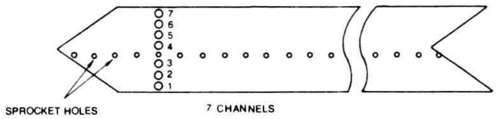

Fig. 1-10.

For transport the tape has a row of smaller holes (sprocket holes) along the middle. Fig. 1-11 shows a combination paper tape reader/punch. Used as a reader, it converts data on the tape into electrical signals for the computer. Used as a punch, it converts electrical signals from the computer into holes on the tape.

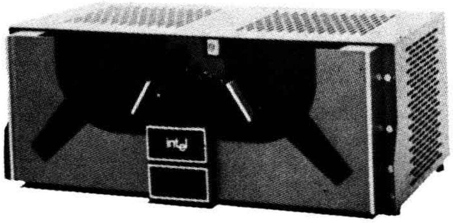

Fig. 1-11.

Line Printer

The line printer (fig. 1-12) is the output medium most frequently used. Line printer paper is usually folded in the form of an accordion. This is sometimes called continuous form, or form fold paper.

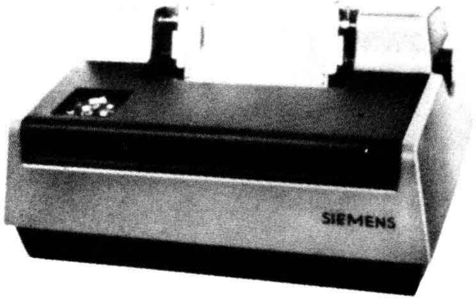

Fig. 1-12.

Display

A display is, strictly speaking, merely a video screen which displays data in the form of words, numbers or graphics. In most cases the display is combined with a keyboard which acts as an input device, and such a combination is also referred to as a display (fig. 1-13). Using the

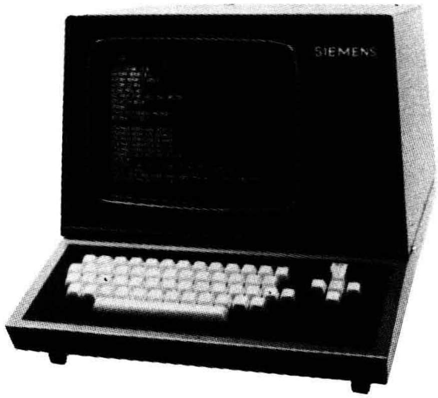

Fig. 1-13.

keyboard, information can be requested from the computer and be displayed on the screen.

Teletype

A teletype (fig. 1-14) is suited for the input as well as the output of data. A teletype is a typewriter which serves the programmer (input) as well as the computer (output). In addition, it may have a paper tape reader and punch.

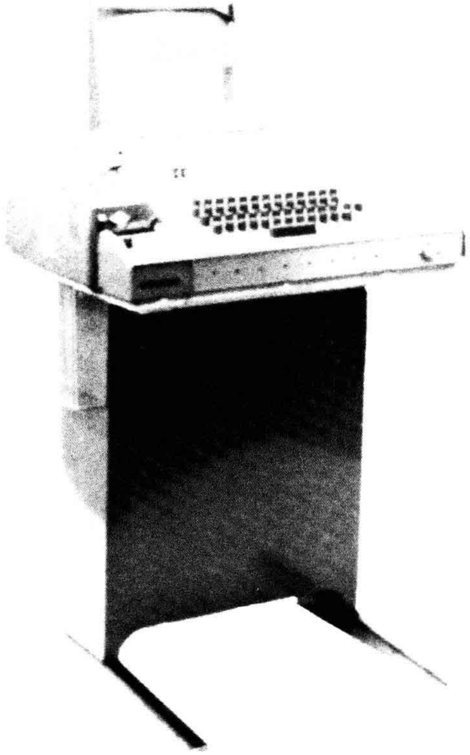

Fig. 1-14.

THE FLOWCHART

When properly instructed, a computer can perform calculations very quickly. The use of a computer is therefore most profitable when the same procedure must be carried out repeatedly using variable data. The computer only needs instructing once. The procedure which the computer must carry out is caled the *program*. The separate steps used to carry out this program are called the *instructions*.

To program a computer for a specific problem we must schematically define the consecutive operations that lead to the solution. This schematic definition is called a *flowchart* or *flow diagram*. When making a flowchart one has to take into consideration the properties of the computer in question. The form of a flowchart depends on the person who made it. Making a flowchart demands not only knowledge and experience, but also creativity. Computer programming is as much an art as it is a science. People's characters differ both in creativity and in depth. By means of an example we will try to help you to become familiar with flowcharts. This topic will be discussed in more detail in Chapter 14.

Problem

Find the sum of a column of consecutive, positive, whole numbers, beginning with P and ending with N. P and N can be chosen at random. We will choose P = 2 and N = 5.

Note: We must bear in mind that:

- A computer can only add 2 numbers at a time.
- That the amount of data to be read in should be kept to a minimum.

The amount of data to be read in can be kept to a minimum by reading in only the initial value P and the final value N. Starting from the initial value the computer can then fill in the values between. The initial P is supplied to a memory location to which we shall designate the *symbolic address* NUMBER (fig. 1-15a).

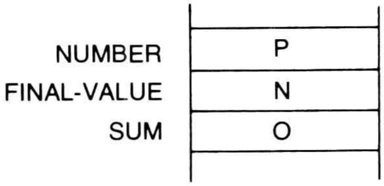

Fig. 1-15A.

Note

In subsequent program-writing lessons we shall see that this is something that we do repeatedly. Symbolic addresses are given in the form of a name. The computer then calculates the appropriate address numbers. We don't have to concern ourselves with this. Calculating the address is done by an assembly program. This will be discussed fully in Chapter 18 and to a certain extent in

Chapter 3. When the number P has been processed, the address NUMBER is filled with the succeeding number. We indicate this as:

$$\text{NUMBER} \leftarrow (\text{NUMBER}) + 1$$

This notation must be read as follows:
The memory location with the address NUMBER is filled with the sum of the present contents of the memory address plus the value 1.
When we wish to indicate the content of a word in memory, we put the address of the word in parentheses.

When the content of the memory location with the address NUMBER is equal to N the computer must stop the program. We must therefore continually compare the content of the address NUMBER with the content of the memory location in which we have put the value N. N has been put in the memory location with the symbolic address FINAL-VALUE. The check to see if the content of the address NUMBER is equal to N is shown in fig. 1-15b. If the content of the address NUMBER is the same as N, the program continues in the direction 'yes.' If the contents are not equal to N, the program continues in the direction 'no.'

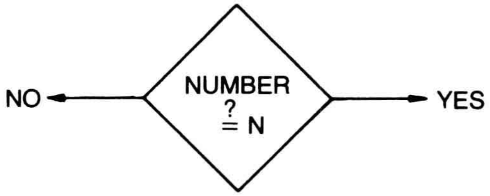

Fig. 1-15B.

Because a computer can only add 2 numbers at a time, we assign the memory location in which the running total is stored the symbolic address SUM. This memory word is filled with the value 0. We then add to the content of SUM step-by-step.

We first add P to the content of location SUM. The content of the address SUM then becomes

$$\text{SUM} \leftarrow (\text{SUM}) + \text{P}$$

We then add the second number to the new content of SUM. The content now becomes P + (P + 1). The third number is added to this new content. Since P is stored in the memory location symbolically addressed by NUMBER, we define this process as:

$$\text{SUM} \leftarrow (\text{SUM})+(\text{NUMBER})$$

This must be read as follows:
The memory location with the address SUM is loaded with the sum of its former content and the content of the word in memory which has the address NUMBER.

It should be clear that after each addition the content of the memory word with the address NUMBER must be incremented by 1.

Solution

The method for solving the problem may be summed up in a flowchart (fig. 1-16). The command START indicates that the program must begin. The values required to continue are then read in. In this case the memory locations NUMBER, FINAL-VALUE, and SUM are filled with 2, 5 and 0, respectively (fig. 1-17a). We now proceed to the processing stage.

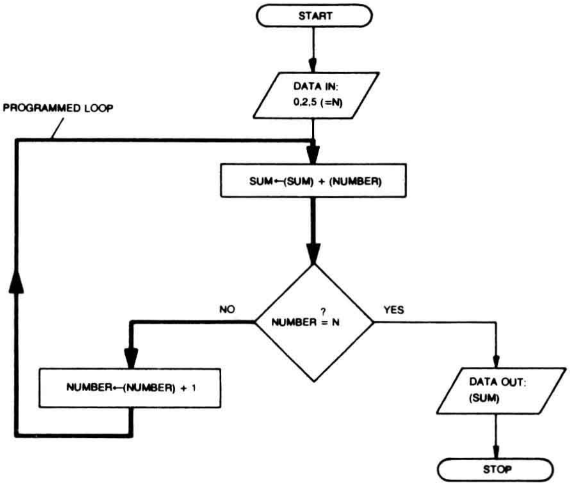

Fig. 1-16.

$$\text{SUM} \leftarrow (\text{SUM}) + (\text{NUMBER})$$

The content of the address SUM becomes $0 + 2 = 2$ (fig. 1-17b). We must then check to see if the content of the address NUMBER is equal to 5. The content of the address NUMBER is 2. The result of the comparison is thus 'no.'

We now find ourselves in what is called a *programmed loop*. (This loop is executed continually until the content of the address NUMBER is equal to 5). Subsequently, the calculation NUMBER←(NUMBER)+1 is performed. Because of this calculation the content of the address NUMBER becomes $2 + 1 = 3$. The contents of the memory locations are shown in fig. 1-17b.