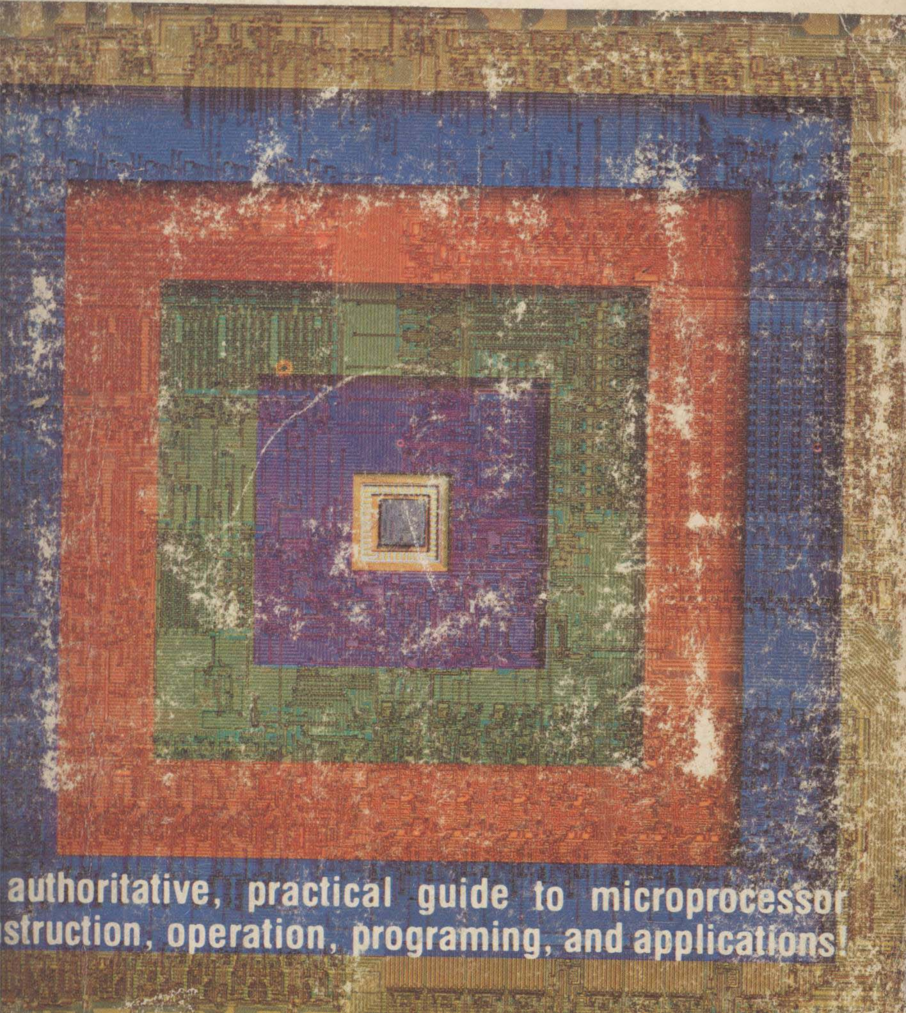
TAB BOOKS / No. 785 $8.95

microprocessor/ microprograming handbook

by brice ward



authoritative, practical guide to microprocessor
struction, operation, programing, and applications!

microprocessor/ microprograming handbook

Cover photo shows an extreme enlargement of National Semiconductor's PACE microprocessor.

microprocessor/ microprograming handbook

by brice ward

TAB BOOKS
BLUE RIDGE SUMMIT, PA. 17214

FIRST EDITION

FIRST PRINTING—DECEMBER 1975
SECOND PRINTING—JULY 1976
THIRD PRINTING—MAY 1977
FOURTH PRINTING—DECEMBER 1977
FIFTH PRINTING—JANUARY 1979
SIXTH PRINTING—MARCH 1980

Printed in the United States
of America

Hardbound Edition: International Standard Book No. 0-8306-5785-1

Paperbound Edition: International Standard Book No. 0-8306-4785-6

Library of Congress Card Number: 75-31466

Preface

The microprocessor has started a small revolution in the data-processing and machine-control fields—not by *adding* costly features, but by *eliminating* them. Let's face it, most computing and control applications do not require a modern, super-fast, super-expensive computer. Quite to the contrary, such applications would benefit most by using a small, inexpensive device that could be tailored to meet the specific requirements of each application. (You certainly do not need a million-dollar computer just to figure out sales tax!) And even the scaled-down computers—the so-called *minicomputers*—are still too expensive for most potential uses.

The trouble with the modern computer is that it is a *general-purpose* machine, specifically designed to handle any conceivable computing chore. To accomplish this feat, the modern computer must be *fast*, have a *large memory* to store both data and instructions, and be quickly *reprogramed* to solve new problems. Today, however, there are ever-increasing numbers of applications that simply do not require high speed, or a large memory, or even the need to be reprogramed. In fact, most of these applications are *single-purpose*, rather than general-purpose, and so would normally be designed with an unalterable, built-in program.

All of the essential processing elements of a computer can be built into one or more tiny integrated circuits, comprising a *microprocessor*. For doing the actual arithmetic and logic operations, the microprocessor contains a *central processing unit*. So that the microprocessor always remembers what functions it is to perform, its instructions are stored permanently in a *read-only memory*. And to remember changing data and input/output information, the microprocessor has a small *random-access memory*. The result is a single-purpose or *dedicated* microprocessor that can do just about anything you want it to do—cheaply!

As a designer, you can now define your special requirements and, possibly with the help of a general-purpose computer, develop a unique program—a *microprogram*—for your microprocessor system to enable it to perform the desired operations. Your program can then be written permanently into a read-only memory, which then becomes an integral part of the microprocessor. Or, for added flexibility, several read-only memories can be used to make major design changes as simple as plugging in a new chip. But this is, in essence, the simplicity in design that has been brought about by the modern microprocessor system.

For your convenience, the addresses of a large number of manufacturers who are making microprocessors and microprocessor components are listed in the appendixes to this book. The author wishes to thank three manufacturers that have been most helpful in the preparation of this book: Motorola Semiconductor Products (with the MC6800 family of microprocessor parts); Intel Corporation (with the MCS families of microprocessors, interfaces, and memory devices); and National Semiconductor Corporation (with the PACE microprocessor). Their help in supplying extensive literature and information is acknowledged with thanks.

Brice Ward

Contents

microprocessor/ microprograming handbook

1

Introduction To Microprocessors

While it is easy to envision a microprocessor as being a handful of integrated circuits, the term actually refers to a concept rather than to any specific device. A microprocessor is a small processor and a processor is a special machine that has been devised for the express purpose of processing information—of performing specific tasks. And therein lies the great potential behind the microprocessor, for it represents an extremely powerful and inexpensive design approach to a wide variety of industrial and commercial applications.

MICROPROCESSOR OR MICROCOMPUTER?

The microprocessor is often referred to as a microcomputer, since it contains all of the arithmetic and logic capabilities that are normally associated with a computer. However, the microprocessor is not really a computer in itself, though it may be used to construct minicomputers and even may find application in full-size computers and their peripheral equipment. The mere fact that microprocessors will be found in such diverse applications speaks highly of the microprocessor's ability to take over many tasks that were previously assigned to hard-wired, discrete-logic circuitry.

The truth is that the microprocessor can be just about anything you want it to be, and therefore is able to do just about anything you want it to do. Manufacturers of microprocessor components have made this possible by designing microprocessor circuits and peripheral-support devices that are at once both very inexpensive and very flexible. That is, the microprocessor system possesses the inherent capacity to handle a wide variety of processing chores and control operations without posing a great expense.

The modern full-size computer is certainly able to cope with an equal variety of processing functions. However, the modern computer is also very large and very costly. The trouble with the modern full-size computer is that it is specifically designed to handle any conceivable processing job, and to provide this flexibility by including quick and easy methods of changing the program that controls its operation. Consequently, the modern computer is a general-purpose computer; and the major cost factor is due to this general-purpose capability.

Most applications are not general-purpose in nature, but rather are single-purpose. For example, the microprocessor contained in a modern cash register terminal is required to perform only those functions generally required of a cash register: totaling sales, computing sales tax, making change, and possibly maintaining a running inventory of stock. There is no need in such a *dedicated* application to provide other costly functions as found in a general-purpose computer. In fact, the ultimate desire in such a single-purpose application is to perform all necessary operations at as little cost as practicable. And here is the type of application in which the microprocessor excels.

The microprocessor differs from the full-size computer primarily in that the program which governs its operation becomes "frozen" into the design at the time of manufacture, and so is not alterable as in the programmable computer. Unlike discrete-component designs, however, the microprocessor is not programed using hard-wired connections that are expensive to manufacture and nearly impossible to change. Instead, the microprocessor's program is usually stored permanently in a read-only memory (ROM), which is both a very simple and inexpensive way to store the program. And reprograming is equally simple—just plug in a different ROM! Thus, in the design stage, the microprocessor retains the computer-like flexibility and programing simplicity that enable the designers to incorporate complex processing functions with little effort. And in the production stage, the microprocessor also retains the low cost and operational proficiency that make it very desirable for an ever-increasing number of applications. Since the microprocessor represents the best possible compromise for so many applications, is it any wonder that it is hailed as one of the most revolutionary concepts of our modern age?

The modern microprocessor is the product of two technological developments: large-scale integration (LSI) and low-cost semiconductor memories. LSI has made it possible to manufacture sophisticated microprocessors that are able to routinely handle complex processing chores. On the other hand, semiconductor memories, consisting of read-only memories (ROMs) and random-access memories (RAMs), have provided an inexpensive and compact storage medium for the microprocessor's program instructions and data. As a result, it is now possible to design a complete microprocessing system utilizing only a small handful of integrated circuits. However, the exact function of the microprocessor still remains to be established by the design engineer or technician, who must know both the internal operations of the microprocessor and the programing techniques required to implement the desired functions. It is the purpose of this book to provide sufficient information that you will understand the operation of a microprocessor and be able to program it.

THE EVOLUTION OF THE MODERN COMPUTER

This is not an effort to trace and date various machines, but simply an indication of the rough transition these machines have gone through in the last 30 to 40 years. It was realized from the days of Babbage that a machine for making fast arithmetic calculations would be a useful device. This was particularly necessary for astronomy and other sciences that required many tedious hours of calculation to arrive at desired results.

During World War II, machines were developed that used mechanical cams, gears, and differentials to calculate gunnery fire-control problems and handle certain types of navigation systems. These devices were not the most accurate devices in the world, but they did save time.

There was, after the war, a growing need in another area also. This was the world of business in which International Business Machines (IBM) took a leading role. The required business machines had to sort, alphabetize, and maintain lists of addresses and similar information related to the business world. Mathematical calculations were required, but were a only small part of the total load.

Over a period of time, these areas were roughly divided into *scientific* machines and *business* machines. Early IBM equipment used the IBM-type punched card with the Hollerith code. These cards could be handled nicely in the sorting and collating machines developed by IBM. They ultimately became one of the major systems for storing raw data and programs of all sorts.

Early sorting machines used conducting brushes to "read" the punched holes in the cards; they were programed using "plug" boards, into which jumper wires were inserted to make up programs for sorting. Many early computers used similar boards for programing and storing data on cards. This separate-storage technique for programs and data has become known as Harvard architecture.

All of the machines were developed to do one thing—manipulate data. This manipulation might have been addition, subtraction, multiplication, or division, but it still fell into the category of data manipulation. The requirement for data manipulation is still one of the major requirements of any data processing system.

The machines became more and more refined. They moved from mechanical systems to relays, from relays to tubes, from tubes to transistors, and from transistors to integrated circuits. But the basic requirement still existed—to manipulate data.

Other things happened. Early machines attempted to retain decimal operations, but it was soon realized that for cost reasons it was preferable to do *all* operations in binary arithmetic. Binary was much easier to represent in electrical *on/off* terms than was decimal. As a result the machines, and the engineers and technicians that worked on them, became committed to using binary or natural binary codes such as octal and hexadecimal.

It also became apparent that for real speed the routine required to do a particular problem had to be built *into* the machine. Not only the additions (for example) but the program that determined the order and manner in which the additions were done had to be electrical or electronic to take maximum advantage of the inherent speed of the equipment.

CALCULATOR VERSUS COMPUTER

Suppose it is necessary to add 341 + 24 + 419. On a calculator you would *enter* the first number. The act of entering the number places it in the *accumulator* register and displays the number in a *readout* device. The [+] key is pressed to indicate that an *add* is to take place. The second number is then entered. Pressing the [+] key again results in the last add instruction being *executed*. The result of the addition is again stored in the accumulator and simultaneously displayed.