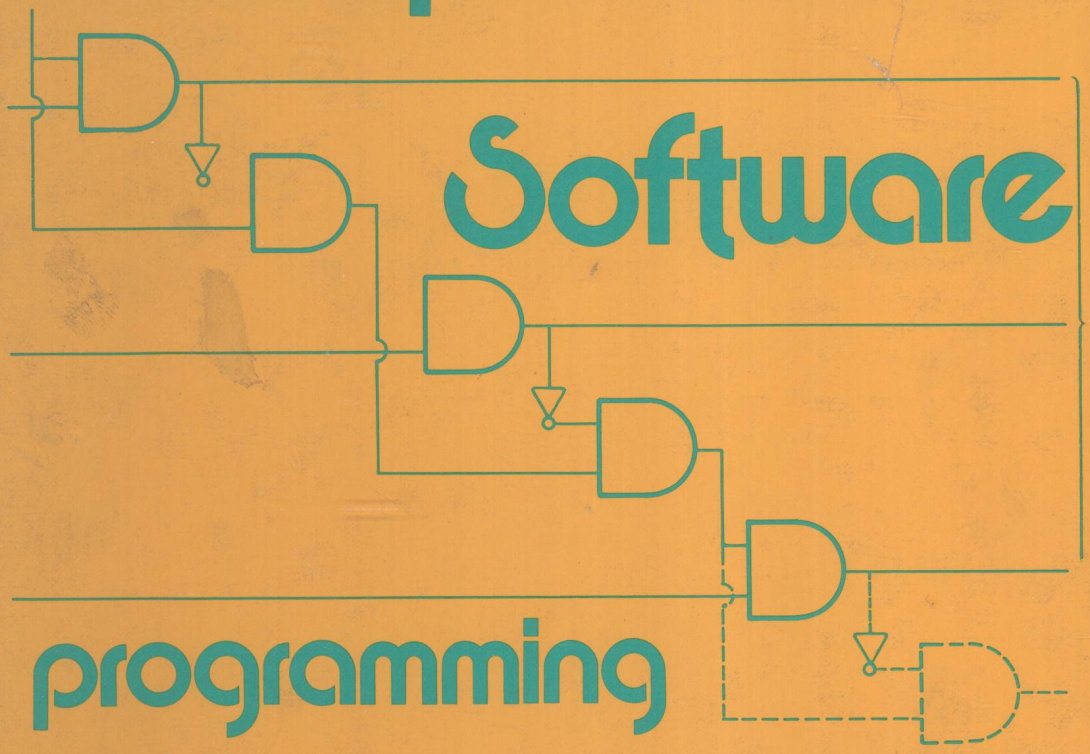
# Microprocessor Software



# programming concepts and techniques

Gene A. Streitmatter

# MICROPROCESSOR SOFTWARE: PROGRAMMING CONCEPTS AND TECHNIQUES

Gene A. Streitmatter

Associate Professor
Electronics Technology
Rock Valley College
Rockford, Illinois

Reston Publishing Company, Inc., Reston, Virginia
*A Prentice-Hall Company*

**Library of Congress Cataloging in Publication Data**

Streitmatter, Gene A
Microprocessor software.

1. Microprocessors—Programming. I. Title.
QA76.6.S79 001.64'2 80-28958
ISBN 0-8359-4375-5

© 1981 by Reston Publishing Company, Inc.
*A Prentice-Hall Company*
Reston, Virginia 22090

*All rights reserved. No part of this book may be reproduced in any way, or by any means, without permission in writing from the publisher.*

10 9 8 7 6 5 4 3 2

Printed in the United States of America

# PREFACE

The computer age has started. An expanding number of people are aware of the consequences of the statement just made. The progress of the involvement of the United States and other industrialized nations in the computer age accelerated tenfold just a few short years ago with the development of the microprocessor. Academic courses at all levels of sophistication are appearing daily. This text is aimed at the postsecondary school level as a classroom text. However, it could be useful to anyone wishing to learn the fundamentals of programming microprocessors. Let's examine this text to see how it is written and how it serves the reader in learning programming.

This text is written for the beginner. It makes no assumptions about previous knowledge of programming. It does assume, however, that the reader wants to learn programming and is willing to take the time and thought necessary to do so. At the beginning of each chapter, there is a partial glossary of terms introduced in that chapter. At the end of each chapter, there are questions and problems that help the reader to understand the material and that cover the main points of the chapter. The progression of the text is as follows:

Chapter 1 lightly discusses microprocessors and their use. It attempts to give an intuitive understanding of how the hardware and software aspects of this new technology relate to each other.

Chapter 2 introduces the fundamental concepts of programming. Methods of approach to programs are discussed, and the reader is given direction in thinking. That is, the author provides the reader with meaningful insight into the process of writing programs, establishes frames of reference, and illustrates the use of models in attacking problems.

Chapter 3 illustrates the difference between design and analysis. Most people assume that programming is a design function. The author, however, feels that the novice programmer

can learn more rapidly if he or she is capable of analyzing established programs. One can learn programming by doing, but the beginning programmer is so involved with keeping the basics in mind that designing a program that is really fun to work with is an immense task. It is usually easier and more satisfying to analyze and modify an existing program. This is also a necessary skill of the technician who maintains a microprocessor-based system.

Chapter 4 deals with the details of the models, addressing modes, and instruction sets of the two microprocessors under discussion. The hardware ramifications and requirements are dealt with in a general manner in hopes that the "trees will not distort the view of the forest." Where necessary, however, the hardware is dealt with as long as it does not become the dominant consideration.

Chapter 5 discusses common techniques used by microprocessor systems and most computers. The level of presentation is at the mnemonic and assembly level. Each program technique is documented with flowcharts and assembly level comments for clarity of operation. The concepts and main objectives of the techniques are presented first; then, specific methods are shown.

Chapter 6 deals with the concepts of utilities, routines, and I/O service routines.

Chapter 7 introduces assembly language conventions and establishes the ground rules for assigning symbols and labels. The basic procedures for manual assembly of programs, the concepts of macroinstructions, and the use of assembler editors are also covered.

Chapter 8 presents some standard methods used to troubleshoot a nonfunctioning program. Analysis tools available to the programmer are discussed. The reader is taught to debug through logical thinking.

Chapter 9 is a summary pertinent to the ongoing process of learning and developing the skills of programming.

This text is written in a progressive level approach. That is, the early chapters are less technical than the latter chapters. The fundamental structures of programming are presented early in the text. Subsequently, instructions and their use are explored, and some simple combinations of instructions are analyzed. The tools of thinking are presented as they are required by the material. The author has written the text in a linear manner so that it can be read continuously from front to back. It is desirable that the student have available a microprocessor that is programmable at the assembly level.

此为试读,需要完整PDF请访问: www.ertongbook.com

Practice and study are the keys to learning the material in this book. The author considered writing an additional chapter concerning a high-level language such as BASIC, but this thought was discarded because it would constitute a large amount of additional material that would not necessarily be more instructive in the fundamental skills put forth in the text. Many good texts on BASIC and other languages are readily available to the reader if he or she wishes to apply the skills learned here to a high-level language.

# CONTENTS

# INTRODUCTION 1

*GLOSSARY*

**BASIC.** Beginner's All-purpose Symbolic Instruction Code. A high-level programming language commonly used in small personal computers. A time-sharing programming language similar to FORTRAN II.

**Controller.** An element or a group of elements that takes data and processes it to cause active change in, or control of, an operational system.

**CPU.** Central Processing Unit. The portion of a computer system that provides the manipulation, control, and arithmetic and logical processing of system data in response to machine instructions usually residing in the system memory.

**Hardware.** The physical part or electronic circuitry of a computer system.

**Memory.** A device in which computer information and data are stored. Data can be entered, held, and retrieved at a later time.

**Microprocessor.** The central processing unit of a small computer, usually including the ALU (arithmetic logic unit) and control functions, on a single large-scale integrated circuit.

**Peripheral Device.** A device that works in conjunction with a computer, but is not part of it.

**Programmable Device.** A circuit or component of a system whose electrical or logical function can be altered by the application of control words or instructions usually in the form of a binary statement.

**Programming.** The act of preparing a list or sequence of instructions for a computer to accomplish the solution to a problem.

**Robot.** An independently operating system that simulates manlike activities and cognitive interaction with the environment in which it resides.

**Software.** The paperwork and documentation that describes and provides information about the programs within a particular hardware system.



此为试读,需要完整PDF请访问: www.ertongbook.com

## MICROPROCESSORS

A microprocessor is a small electronic device that is causing a revolution within the electronics industry that will eventually affect the lives of everyone. Historically, the electronics industry has been involved with hardware solutions to problems. That is, when signals or data had to be moved, processed, or applied, a specific circuit was developed to accomplish the objective. The user of such a circuit is usually not concerned with why the circuit works the way it does, but rather with how to use it or how to make it operate. For example, when a woman pushes the buttons on her microwave oven, washer/dryer, or blender, she is not concerned with what circuitry is being activated, modified, or otherwise used. She is usually only concerned with whether or not the device acts as expected, reliably and predictably. Consequently, manufacturers constantly attempt to make the operational use of systems as simple as possible.

As consumer demand for more and more complex devices has increased, the circuit complexity required to maintain simple operational procedures has increased. The trend has been, and continues to be, toward more sophisticated and prestructured programmed operations, which are initialized by relatively few and simple operator activities. These activities usually consist of pushing particular buttons one at a time or in sequence to initiate an operation. The device then takes over the operations, leaving the operator free to do other things.

As the trend toward more sophisticated functions with less operator involvement developed, the technology to produce smaller circuits became necessary to bring the systems within a manageable size. This trend also forced a more universal design approach to circuits. It is desirable to have a circuit that can be adapted to many applications with only a few minor electrical modifications. The search for universally applicable circuits has relentlessly led to programmable devices—devices that can be told what function is to be done by the application of a combination or sequence of signals. The outgrowth of all of these trends was the development of the microprocessor.

### What Microprocessors Are

Microprocessors are extremely compact and complex electronic circuits capable of highly sophisticated functional operations.

Large microprocessors are from 5 mm to 7 mm (¼ inch) square and can contain up to 70 thousand electronic components. They are usually packaged in a flat, multiple connection box made of metal, ceramic, or plastic, and in many cases, they look like insects or caterpillars. The packages vary in size from 1 to 5 inches long and from ¼ to 1 inch wide, with connections or leads exiting from the package in a variety of directions and configurations. The arrangement of leads is sometimes called the *footprint* or *pinout*.

Recently developed microprocessors are built to operate using a single power source of only one voltage level. There are a variety of reasons for this, including system cost. The general rule is that the more power supplies required, the more the system into which the processor is to be placed costs. Microprocessors are all digitally based and operate with binary data. That is, they deal with strings or groups of electrical signals that are of consistent single voltage level and pulsing in character. The information dealt with must be formatted or coded in numbers consisting of 0's and 1's only. The binary number system is discussed in detail in Appendix A.

The microprocessor itself is only a small part of an overall system that usually consists of several parts—a main or central processing unit (CPU), a memory section, and a variety of peripheral devices that provide communication between the CPU and the real outside world. Without the memory and the support units, the CPU is quite helpless.

## What Microprocessors Do

Microprocessor applications are growing daily. Most of these applications are very direct and visible—small business computers, small home computers, complex TV games, and devices in cars that calculate miles per gallon and cruising distance.

Another application, which is not as readily observable, is the hidden controller—kitchen appliance controllers, household timers, burglar alarms, thermostat controllers, industrial machine controllers, communication equipment controllers, and supermarket point-of-sale terminals. It is in this application of hidden service that the microprocessor will have its greatest impact on the lives of people. It can relieve human beings of many tasks done everyday that are simple monitoring functions. No one will be tied to the kitchen while cooking. One can simply program the start cooking times of various items, the

temperatures at which they are to cook, and the stop cooking times. The assembly-line worker will be relieved of much of the monotonous nut-and-bolt activity of assembly and will be more involved in the production process. An exciting and innovative application is robotics. The age of the cognitive function machine (robot) is just around the corner. As electrical engineers design more universally applicable electrical and mechanical machines, the microprocessor applications for control will become much more commonplace. The key to this variability and flexibility of application is the ability to program different operations into the machine. Once the general capabilities of a machine have been established by the design engineer, the specific operation and application are accomplished by the programmer.

Universality and flexibility of application is the main attractive feature of the microprocessor. During the design, prototyping, and testing phases of a new product, the low cost and fast turnaround characteristic inherent in microprocessor-based design is a tremendous cost-saving factor to the manufacturer.

## How Microprocessors Work

A natural question that arises out of the preceding discussion is "How is all of this flexibility and adaptability accomplished?" This question is perhaps best answered with an analogy. The analogy is that of a piano. A piano has all the hardware potential to produce a song by Beethoven, Liberace, or Jerry Lee Lewis. It has a sounding board, strings, pedals, keys, and so on. It is flexible in that various combinations of the hardware items can produce different sound effects that, when put together properly by a pianist, can result in beautiful and harmonious sounds. Also, just as there are various name brand pianos that have different hardware characteristics, there are various name brand microprocessors that have different hardware characteristics. In terms of hardware features, perhaps an electronic organ would better fit the analogy in that one can purchase built-in rhythms, chording functions, and sounds. Some organs are more difficult to use than others, but they are all produced with one aim, providing a basic instrument that can be made to produce satisfactory music. The quality of the music and the smoothness of the delivery, however, depend on the skill of the player rather than on the availability of the features. There are basic hardware minimums for a piano or organ that must exist

or it will not work, such as keys, strings, and oscillators. This is also true for microprocessors. The minimums for microprocessors are memory, CPU, I/O ports, and connecting lines. Without these minimums the system will not work, but features can be added to operate beyond the minimum capabilities. The analogy also holds for the programming realm. Once a microprocessor system is built, the use of that system depends on the programmer.

## PROGRAMMING

### What Programming Does

Keeping the analogy in mind, programming is to a microprocessor system what songwriting is to a piano or organ. There are basic rules to programming just as there are basic rules to songwriting. Programs can be transposed from one machine to another just as songs can be transposed from a Lowery organ to a Thomas organ. The organs may have different features, but both can be made to play the same songs.

Programming consists of putting together basic instructions into a complete functional program. To do that, a thorough understanding of the basic instructions for the processor in which the program is to operate is required. One must also have a basic understanding of the rules of programming. Learning to write programs takes time, study, dedication, and much practice and inspiration.

### How Programming Is Done

There are different levels of accomplishment to programming just as there are different levels of accomplishment to songwriting. In order to put oneself in the right frame of mind for this text, one should approach the microprocessor as a beginning songwriter approaches the piano.

To begin the study, an overview of how programming is done must be kept in mind. Primarily, a programmer engages in three activities:

1. Analysis of a particular problem and an assessment of the functional capabilities of the machine in which the program is to operate.

2. Design of the solution in terms of sequence of accomplishment, use of peripheral systems, and so on.

3. Writing, coding, and debugging to verify that the program functions properly.

The end result of these activities is a finalized written document that describes the program at various levels of complexity. The document should provide all the basic information needed for another person to use the program in the system for which it was designed.

The activities of programming are both physical and mental. The mental portion is by far the most time consuming. It has been demonstrated that the average programmer can produce only ten good program lines per day. Therefore, approximately 45 minutes of thought go into every sequential decision in a good program. A programmer may spend five to ten days thinking, analyzing, and trying out various combinations of instructions before, on the last day, he or she finally completes writing 50 to 100 lines of programming.

## Who Does Programming?

You might ask "Who does programming, and can I learn it?" The answer to the second question is yes. Anybody can learn the basics of programming; however, not everyone is capable of being a Roger Williams or a Liberace. There are two answers to the first question, one for today, and one for the near future.

Today, programming is primarily done by groups or individuals hired by corporations to produce programs for equipment and machines manufactured for sale. These people are, in effect, the early pioneers of the computer age. The programs and techniques being developed today are the fundamental building blocks that will be used by almost everyone in the near future. In the not too distant future, the use of computers and processors in everyday life will be as commonplace as driving a car is today. Almost everyone will know the fundamentals and will be able to write programs.

The consumer of tomorrow who cannot write programs will be as limited as the consumer of today who cannot read or write. The programmers of today are developing layman languages that can be easily used by everyone. Consequently, what is complex today will be fundamental tomorrow.

1. What is a microprocessor?
2. What is hardware?
3. What is software?
4. Why is it important that a person should learn programming today?
5. Name the two major application categories of microprocessors.
6. In your own words, briefly explain the procedure a programmer goes through when writing a program.
7. Why are microprocessors important to us today and in the future?
8. Name some applications that you have seen that use microprocessors.
9. What is a peripheral device?
10. What is meant by the term *robot*?

*Self-Analysis Questions*

1. Do you think you can learn programming?
2. Do you think it will be difficult or easy?
3. Are you a logical thinker?
4. Do you want to learn programming?
5. Are you a hard worker?
6. Will you take time to study?
7. Do you have access to a working computer system?



此为试读,需要完整PDF请访问: www.ertongbook.com