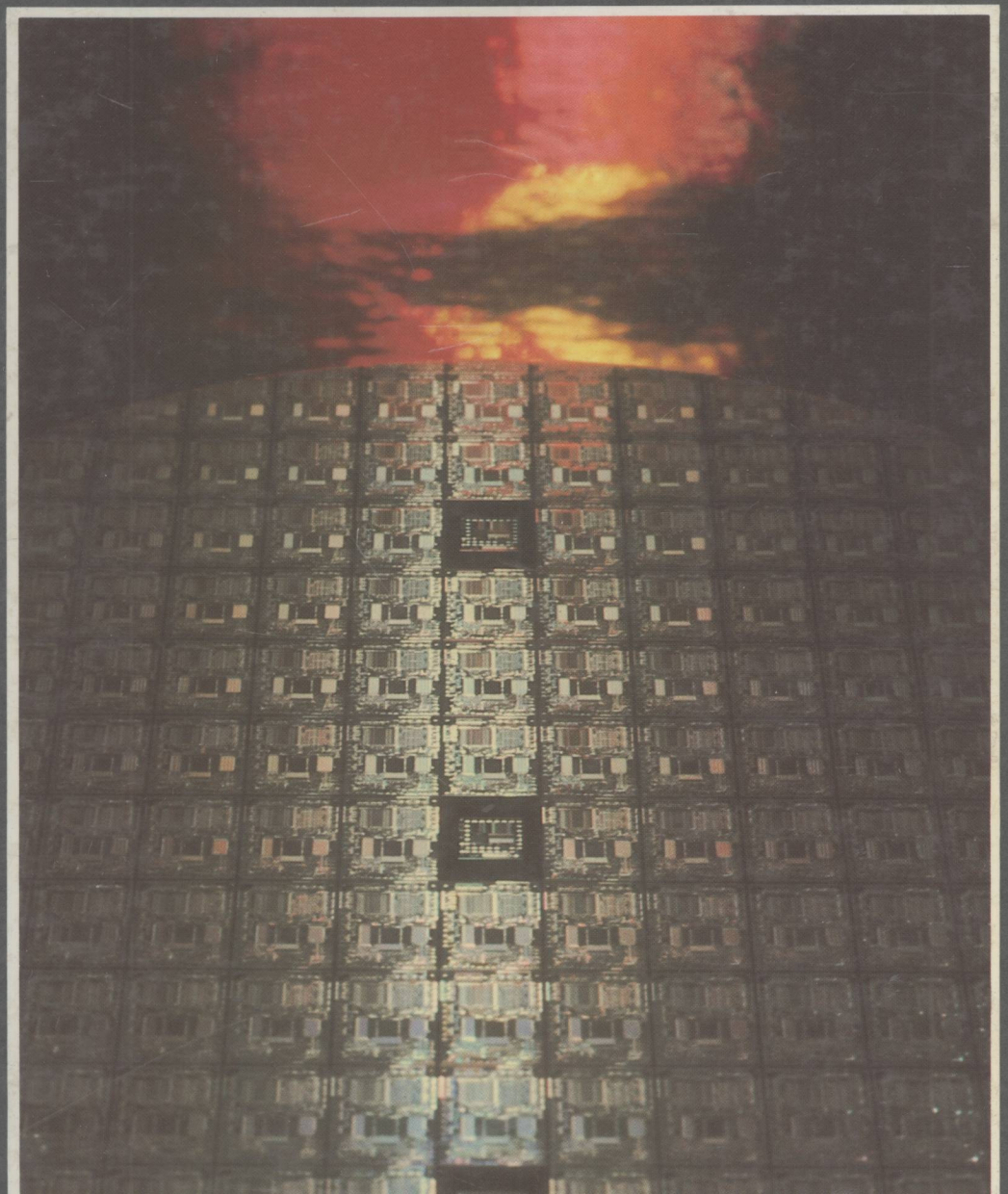
GAONKAR

MICROPROCESSOR ARCHITECTURE, PROGRAMMING, AND APPLICATIONS WITH THE 8085/8080A



Microprocessor Architecture, Programming, and Applications with the 8085/8080A

Ramesh S. Gaonkar
ONONDAGA COMMUNITY COLLEGE

Charles E. Merrill Publishing Company
A Bell & Howell Company
Columbus Toronto London Sydney

To Shivram K. Gaokar—
my father, my teacher, my inspiration

Published by Charles E. Merrill Publishing Company
A Bell & Howell Company
Columbus, Ohio 43216

Production Coordinator: Gnomi Schrift Gouldin
Cover Design: Tony Faiola
Cover Image: Courtesy of Intel Corporation
This book was set in Times Roman and Lubalin

Library of Congress Catalog Card Number: 84-60073
International Standard Book Number: 0-675-20159-4

Printed in the United States of America
1 2 3 4 5 6 7 8 9 10—91 90 89 88 87 86 85 84

Microprocessor Architecture, Programming, and Applications with the 8085/8080A

Merrill's International Series in Electrical and Electronics Technology

BATESON: Introduction to Control System Technology, 2nd Edition, 8255-2

BOYLESTAD: Introductory Circuit Analysis, 4th Edition, 9938-2
Student Guide to Accompany Introductory Circuit Analysis, 4th Edition, 9856-4

BOYLESTAD/KOUSOUROU: Experiments in Circuit Analysis, 4th Edition, 9858-0

BREY: Microprocessor/Hardware Interfacing and Applications, 20158-6

FLOYD: Digital Fundamentals, 2nd Edition, 9876-9
Electronic Devices, 20157-8
Essentials of Electronic Devices, 20062-8
Principles of Electronic Circuits, 8081-9
Electric Circuits, Electron Flow Version, 20037-7

GAONKAR: Microprocessor Architecture, Programming, and Applications with the 8085/8080A, 20159-4

NASHELSKY/BOYLESTAD: BASIC Applied to Circuit Analysis, 20161-6

ROSENBLATT/FRIEDMAN: Direct and Alternating Current Machinery, 2nd Edition, 20160-8

SCHWARTZ: Survey of Electronics, 2nd Edition, 8554-3

SEIDMAN/WAINTRAUB: Electronics: Devices, Discrete and Integrated Circuits, 8494-6

STANLEY, B. H.: Experiments in Electric Circuits, 9805-X

STANLEY, W. D.: Operational Amplifiers With Linear Integrated Circuits, 20090-3

TOCCI: Fundamentals of Electronic Devices, 3rd Edition, 9887-4
Electronic Devices, 3rd Edition, Conventional Flow Version, 20063-6
Fundamentals of Pulse and Digital Circuits, 3rd Edition, 20033-4
Introduction to Electric Circuit Analysis, 2nd Edition, 20002-4

WARD: Applied Digital Electronics, 9925-0

Preface

This text is intended primarily for undergraduate students in technology and engineering curricula. The treatment of the microprocessor is comprehensive, covering both theoretical concepts and practical applications using the 8085/8080A microprocessor family for illustrations. The text assumes that the student has completed a course in digital logic; however, it does not assume any background in programming.

The microprocessor is a general purpose programmable logic device. The thorough understanding of the microprocessor demands the concepts and skills from two different disciplines: hardware concepts from electronics and programming skills from computer science. Hardware is the physical structure of the microprocessor and programming makes it come alive — one without the other is meaningless. Therefore, in this text, the contents are presented with an integrated approach to hardware and software in the context of 8085/8080A microprocessor. Part I focuses on the microprocessor architecture and related hardware; Part II introduces programming; and Part III integrates hardware and software in interfacing and designing microprocessor-based products. Each topic is covered in depth from basic concepts to industrial applications and the topics are illustrated by numerous examples with complete schematics. The material is supported by assignments using practical applications.

Part I, dealing with the hardware of the microcomputer as a system, has four chapters presented with the spiral approach, in a format similar to the view from an airplane that is getting ready to land. As the plane circles around, one observes a view without any details. As the plane descends, the view begins to include more details. This approach allows the students to use a microcomputer as a system in their laboratory work during the early stages of the course, before they have an understanding of all aspects of the system. Chapter 1 presents an overview of the computer systems and discusses the microcomputer and its assembly language in the context of the entire spectrum of computers and computer languages. Chapters 2, 3, and 4 examine microprocessor architecture, memory, and I/O, each chapter increases in depth — from registers to instruction timing.

Part II has six chapters dealing with 8085/8080A instructions, programming techniques, program development, and software development systems. The contents are presented in a step-by-step format. A few instructions that can perform a simple task are selected. Each instruction is described fully with illustrations of its operations and its effects on

the selected flags. Then, these instructions are used to write programs, along with programming techniques and troubleshooting hints. Each illustrative program begins with a problem statement, provides the analysis of the problem, illustrates the program, and explains the programming steps. The chapters conclude by reviewing all the instructions discussed within them. The contents of Part II are presented in such a way that, in a course with heavy emphasis on hardware, students can teach themselves assembly language programming if necessary.

Part III synthesizes the hardware concepts of Part I and the software techniques of Part II. It deals with the interfacing of I/Os and contains numerous examples from industry and other practical applications. Each illustration analyzes the hardware and software and describes how the two work together to accomplish a given objective. Chapters 11 through 16 include various types of data transfer between the microprocessor and its peripherals such as simple I/O, interrupts, interfacing of data converters, I/O with handshake signals using programmable devices, and serial I/O. Chapter 14 discusses programmable devices used in the Intel SDK-85 system (such as 8155, 8755, and 8279), while Chapter 15 discusses general purpose programmable devices (such as 8255A, 8253, 8259A, and 8257). Chapter 17 deals primarily with the project design of a single-board microcomputer that brings together all the concepts discussed previously in the text.

A WORD WITH FACULTY MEMBERS

This text is based on my teaching experience, my development of courses, and my association with industry engineers and programmers during the past five years; it is an attempt to share these experiences in and out of the classroom. Some of my assumptions and observations are as follows:

1. Software (instructions) is an integral part of the microprocessor and demands the emphasis equal to that of the hardware.
2. In industry, for the development of microprocessor-based projects, 70 percent of the effort is devoted to software and 30 percent to hardware.
3. Technology and engineering students tend to be hardware oriented and have considerable difficulty learning programming.
4. Students have difficulty in understanding mnemonics and realizing the critical importance of flags.
5. Introductory microprocessor courses should be based on a specific microprocessor. It is irrelevant which microprocessor is selected as an illustration; the concepts transfer easily from one device to another device.

This text is written with the concerns and assumptions stated above. The text should be able to meet the objectives of undergraduate courses with various areas of emphasis. For a one-semester course whose emphasis is evenly divided between hardware and software, the following chapters are recommended: Chapters 1 through 4 for hardware lectures and Chapters 5 through 8 and selected sections of Chapter 9 for software laboratory sessions. For an emphasis on interfacing, initial sections of Chapters 11, 12, and 16 (concepts in peripheral and memory-mapped I/O, and introduction to interrupts and serial I/O) are recommended. If the course is heavily oriented towards hardware, Chapters 1 through 4 and Chapters 11 through 16 are recommended and the necessary programs selected from Chapters 5 through 9. Interfacing laboratory sessions can be designed around the illustrations in those chapters or the experimental assignments given at the end of the chapters. If the course is heavily oriented towards software, Chapters 1 through 10 and selected portions of 11 and 12 can be used. The entire text can be covered in a two-semester course. The instructor's manual includes course designs, a suggested weekly lecture and laboratory schedule, solutions, and selected figures to produce transparencies.

A WORD WITH STUDENTS

Microprocessors are exciting and challenging, and the field is growing. These microcomputers will pervade industry for decades to come. To meet the challenges of this growing technology, you will have to be well conversant with microprocessor programming. Programming is a process of problem solving and communication through the strange language of mnemonics. Most often, hardware oriented students find this communication process very difficult. One of the questions, frequently asked by a student is, How do I get started on a given programming assignment? One approach to learning programming is to examine various types of programs and imitate them. You can begin by studying the illustrative program relevant to an assignment, its flowchart, its analysis, program description and, particularly, the comments. Read the instructions from Appendix F when you need to and pay attention to the flags. This text is written in such way that simple programming of the microprocessor can be self-taught. Once you master the elementary programming techniques, interfacing and design will become exciting and fun.

ACKNOWLEDGMENTS

My students provided the primary incentive for writing this text. Their difficulties, questions, and frustrations shaped the book's design and its development; however, several individuals have made valuable contributions to this text. I would like to extend my sincere appreciation to James Delaney, a colleague who made valuable contributions to the book throughout its various phases. I would also like to thank two other colleagues, Charles Abate and John Merrill, and Dr. Jack Williams of General Electric, who offered many suggestions throughout the project. Similarly, I appreciate the efforts and numerous suggestions of my principal reviewers: John Morgan from DeVry Institute of Technology — Dallas, Peter Holsberg of Mercer County Community College, (N.J.), David M. Hata of Portland Community College (Ore.). Other reviewers were Leslie Sheets of Southern Illinois University at Carbondale — School of Technical Careers, William B. Oltman of Atlas Electric Devices Co., Richard T. Burchell of Riverside City College (Calif.) Jon LeGro of S.U.N.Y. Agricultural and Technical College at Alfred, and Manuel Lizarzaburu of Suffolk County Community College. If this text reads well, the credit goes to Dr. Kathy Forrest of the English Department, who devoted painstaking hours editing the rough draft. Finally, I would like to thank Chris Conty for undertaking and coordinating this project through all its phases and Gnomi Schrift Gouldin for all her efforts through the production process; she made this project more enjoyable than I imagined.

Contents

I Microcomputer Systems and Hardware

Part I of this book deals primarily with the microcomputer as a system. The system is discussed in terms of the four components — microprocessor, memory, input, and output — and their communication process. The role of the programming languages, from machine language to higher-level languages, is presented in the context of the system.

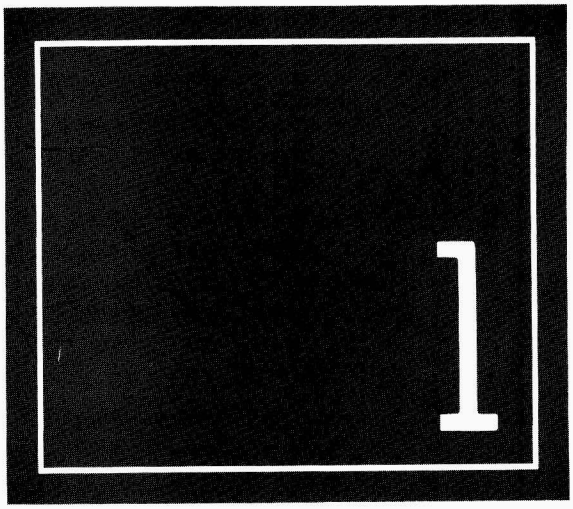
The material is presented in a format similar to the view from an airplane that is getting ready to land. As the plane circles around, a passenger observes a view without any details. As the plane starts descending, the passenger begins to see the same view but with more details. In the same way, Chapter 1 presents a general computer as a system, various types of computers (from large computers to microcomputers), and their processes of communication. Chapter 2 provides a closer look at a microcomputer system in relation to the 8085/8080A microprocessor, while Chapter 3 examines the details of the components of the system and their interfacing. Chapter 4 provides an overview of the instructions — the tasks the microprocessor can perform — and their timing relationship. (Chapters 2 and 3 focus on the hardware aspect of the 8085/8080A microcomputer system; Chapter 4, on the other hand, focuses on the 8085/8080A's software capabilility.)

PREREQUISITES

The reader is expected to know the following concepts:

- ☐ Number systems (binary, octal and hexadecimal) and their conversions.
- ☐ Boolean algebra, logic gates, flip-flops, and registers.
- ☐ Concepts in combinational and sequential logic.

Microcomputers, Microprocessors, and Assembly Language



The microcomputer is making an impact on every aspect of our lives; and soon it will play a significant role in the daily functioning of all industrialized societies. The basic structure of the microcomputer is no different from any other computer. In the 1960s, computers were accessible and affordable only to large corporations, big universities, and government agencies. Because of advances in semiconductor technology, the million-dollar computing capacity of the 1960s is now available for less than ten dollars in an integrated circuit called the **microprocessor**. A computer designed using the microprocessor is called a **microcomputer**. This chapter introduces the basic structure of a computer and shows how the same structure is applicable to the microcomputer.

Computers communicate and operate in the binary numbers 0 and 1, called **bits**. Each computer has a fixed set of instructions in the form of binary patterns called a **machine language**. Since it is difficult for people to communicate with computers in the language of 0s and 1s, the binary instructions are given abbreviated names, called mnemonics, which form the **assembly language** for a given computer. This chapter provides an explanation of both the machine language and the assembly language of a microcomputer designed with the microprocessor known as the 8085 (or the 8080A—the 8085 is the enhanced version of the 8080A). In addition, the advantages of assembly language in comparison with such English-like languages as BASIC and FORTRAN are discussed. Later in the chapter, microcomputer applications in an industrial environment are presented in the context of the entire spectrum of computer applications.

OBJECTIVES

- ☐ Draw a block diagram of a computer and explain the functions of each component.
- ☐ Explain the functions of each component of a microcomputer: microprocessor, memory, and I/O, and their lines of communication, called the bus.
- ☐ Explain the terms SSI, MSI, and LSI.
- ☐ Define the terms bit, byte, word, instruction, software, and hardware.
- ☐ Explain the difference between the machine language and the assembly language of a computer.
- ☐ Explain the terms low-level and high-level languages.
- ☐ Explain the advantages of assembly language over high-level languages.

1.1 DIGITAL COMPUTERS

A digital computer is a multipurpose, *programmable* machine that reads *binary* instructions from its *memory,* accepts binary data as *input* and processes data according to those instructions, and provides results as *output*. The physical components of the computer are called **hardware**. A set of instructions written for the computer to perform a task is called a **program**, and a group of programs is called **software**. (This definition of the computer includes several new concepts and technical terms that will be explained in the following paragraphs.)

BINARY

The computer recognizes and operates in binary digits, 0 and 1, also known as bits. A bit is the abbreviated form of the term **binary digit**. These digits are represented in terms of electrical voltages in the machine: generally, 0 represents one voltage level and 1 represents another. The digits 0 and 1 are also synonymous to **low** and **high**, respectively.

PROGRAMMABLE

The computer is **programmable**; that is, it can be instructed to perform tasks within its capability. A toaster can be cited as an example of an elementary programmable machine. It can be programmed to remain on for a given time period by adjusting the setting to "light" or "dark." The toaster is designed to understand and execute one instruction given through a mechanical lever. On the other hand, digital computers are designed to understand and execute many binary instructions. A computer is a much more sophisticated machine than a toaster. It is a multipurpose machine that can be used to perform sophisticated computing functions, as well as to perform such simple control tasks as turning devices on and off. One of the primary differences between a toaster and a computer is that a toaster executes only one function, and that function cannot be changed. On the other hand, the person using a computer can select appropriate instructions and ask the computer to perform various tasks on a given set of data.

The engineer who designs a toaster determines the timing for light and dark toast, and the manufacturer of the toaster provides the necessary instructions to operate the toaster. Similarly, the design engineers of a computer determine a set of tasks the computer

should perform, and design the necessary logic circuits. The manufacturer of the computer provides the user with a list of the instructions the computer will understand. Typically, an instruction for adding two numbers may look like a group of eight binary digits, such as 1 0 0 0 0 0 0 1. These instructions are simply a pattern of 0s and 1s. Now, the question is: Where does one write those instructions and enter data? The answer is *memory*.

MEMORY

Memory is like the page of a notebook with space for a fixed number of binary numbers on each line; however, these pages are generally made of semiconductor material. Typically, each line has space for eight binary bits. In reality, a line is an 8-bit register that can store eight binary bits; several of these registers are arranged in a sequence called memory. These registers are always grouped together in powers of two. For example, a group of 1024 (2^{10}) 8-bit registers on a semiconductor chip is known as 1K byte of memory; 1K is the closest approximation in thousands. The user writes the necessary instructions and data in memory, and asks the computer to perform the given task and find an answer. This statement raises several questions: How does one enter those instructions and data in the computer's memory? And where does one look for the answer? The answers are *input* and *output* (I/O) devices.

INPUT/OUTPUT

The user can enter instructions and data in memory through such devices as a keyboard or simple switches, which are called input devices. The computer reads the instructions from the memory and processes the data according to those instructions. The result can be displayed in several ways, such as by seven-segment LEDs (Light Emitting Diodes) or printed by a printer. These devices are called output devices.

This still does not explain how and where the computer processes data. It processes data by using the group of logic circuits called its Central Processing Unit (CPU).

THE CPU

The central processing unit of the computer consists of various registers to store data, the arithmetic/logic unit (ALU) to perform arithmetic and logical operations, instruction decoders, counters, and control lines. The CPU reads instructions from the memory and performs the tasks specified. The CPU communicates with input/output devices to accept or to send data. The input and output devices are known also as **peripherals**. The CPU is the primary and central player in communicating with various devices such as memory, input, and output; however, the timing of the communication process is controlled by the group of circuits called the **control unit**.

The description of a computer given here is traditionally represented by the block diagram shown in Figure 1.1. The block diagram shows that the computer has five components: ALU, control unit, memory, input, and output. The combination of the ALU and the control unit is known as the CPU. Before exploring the details of these components of the computer, we should examine what changes have occurred in semiconductor technology and how those changes have affected computers.

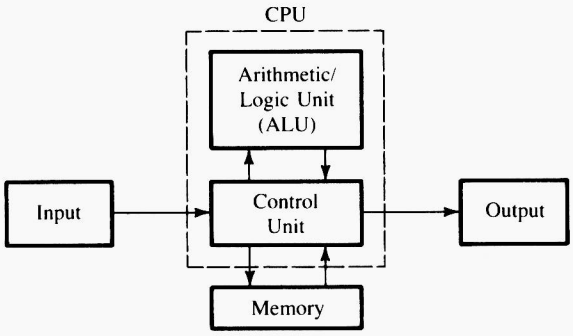


FIGURE 1.1
Traditional Block Diagram of a Computer

1.11 Computer Technology

In the last twenty-five years, semiconductor technology has undergone unprecedented changes. Integrated circuits (ICs) appeared on the scene at the end of the 1950s, following the invention of the transistor. In an integrated circuit, an entire circuit consisting of several transistors, diodes, and resistors is contained on a single chip. In the early 1960s, logic gates known as the 7400 series were commonly available as ICs, and the technology of integrating the circuits of a logic gate on a single chip became known as Small-Scale Integration (SSI). As semiconductor technology advanced, more than a hundred gates were fabricated on one chip; this was called Medium-Scale Integration (MSI). A typical example of MSI is a decade counter (7490). Within a few years, it was possible to fabricate more than a thousand gates on a single chip; this came to be known as Large-Scale Integration (LSI). Now we are in the era of Very-Large-Scale Integration (VLSI) and Super-Large-Scale Integration (SLSI). The lines of demarcation between these different scales of integration are ill-defined and arbitrary.

As the technology moved from SSI to SLSI, the face of the computer changed. Initially, computers were built with discrete logic gates (SSI). As more and more logic circuits were built on one chip using LSI technology, it became possible to build the whole CPU with its related timing function on a single chip. This came to be known as the **microprocessor**, and a computer built with a microprocessor is known as a **microcomputer**. This distinction may soon disappear, however, as the computing power of the microprocessor approaches that of the CPUs of the traditional large computers. Early microcomputers were built with a 4-bit microprocessor. Now a 64-bit microprocessor is being used in some prototype computers. Even if they are built with a microprocessor, it is meaningless to classify them as microcomputers.

1.12 Microcomputer Organization

Figure 1.2 shows a simplified but formal structure of a microcomputer. It includes four components: microprocessor, input, output, and memory (read/write memory and read-only memory). These components are organized around a common communication path called a *bus*. The entire group of components is called a *system* or a *microcomputer system,* while the components are called *subsystems*.