

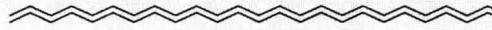
The background of the cover is a close-up photograph of a microprocessor chip mounted on a green printed circuit board (PCB). The chip is a square integrated circuit with numerous gold-colored pins extending from its edges. The PCB features intricate black circuit traces. A dark, diamond-shaped graphic is superimposed over the center of the chip, serving as a backdrop for the title text.

INTEL MICROPROCESSORS

Hardware, Software and
Applications

8085
8086
8088
80486

Roy W. Goody



INTEL MICROPROCESSORS:

Hardware, Software, and Applications

8085 8086/88 80486

Roy W. Goody

*Mission College
Santa Clara, California*

GLENCOE

McGraw-Hill

New York, New York Columbus, Ohio Woodland Hills, California Peoria, Illinois

To George and June Goody

IBM PC, IBM PC/XT, IBM PC/AT, IBM PS/2, and MicroChannel Architecture are registered trademarks of IBM Corporation. All references in this text to the following are registered trademarks of Intel Corporation: Intel, Intel 386™, i386™, 387™, Intel 486™, i486™, i860™, ICE, iRMX. Borland, Sidekick, Turbo Assembler, TASM, Turbo Debugger, and Turbo C++ are registered trademarks of Borland International, Inc. Microsoft, MS, MS DOS, Windows 3.0, CodeView, and MASM are registered trademarks of Microsoft Corporation. ® PAL is a registered trademark of Monolithic Memories, Inc. TRI-STATE is a registered trademark of National Semiconductor Corp. Other product names are registered trademarks of the companies associated with the product name reference in the text or figure.

Goody, Roy W.

Intel microprocessors: hardware, software, and applications / Roy W. Goody

p. cm.

"8085, 8086, 8088, 80486."

Rev. ed. of: The intelligent microcomputer. 2nd ed. 1986.

Includes index.

ISBN 0-02-801811-7. — ISBN 0-02-801812-5 (lab. manual). — ISBN 0-02-801813-3 (instructor's guide)

1. Microprocessors. 2. Intel 80xxx series microprocessors.

I. Goody, Roy W. Intelligent microcomputer. II. Title.

QA76.5.G627 1992

004.16—dc20

92-32084

CIP

Copyright © 1993 by the Glencoe Division of Macmillan/McGraw-Hill School Publishing Company. All rights reserved. Except as permitted under the United States Copyright Act, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the publisher.

Send all inquiries to:
Glencoe/McGraw-Hill
8787 Orion Place
Columbus, OH 43240

ISBN 0-02-801811-7

Printed in the United States of America.

2 3 4 5 6 7 8 9 042 04 03 02 01 00

P R E F A C E



INTRODUCTION

This text is divided into two units. The first half, Unit One, covers the world of the 8-bit 8085 microprocessor, and the second half, Unit Two, encompasses the 16-bit 8086/88.

Unit One is introductory in nature and is intended for those with little or no previous experience with microprocessors. For these students, the simpler 8-bit 8085 is the ideal learning vehicle. Unit One takes a gradual, deliberate, and “gentle” approach, and covers those fundamental concepts that are common to all computer systems. The writing style is relaxed and conversational, using many examples and analogies to stress the major points.

Unit Two builds on the lessons and knowledge of Unit One and gives the student the confidence and ability to enter the more complex world of the 16-bit 8086. Unit Two strives to “get up to speed” as fast as possible and to operate in an environment that is similar to that found in industry. Unit Two is more applications- and project-oriented, and introduces more advanced computer components and practices. All demonstration projects are based on the IBM PC line of computers.

Both Unit One and Unit Two give a balanced presentation between hardware, software, interfacing, and applications. Unit One is based purely on assembly-language and uses the 8-bit assembler from Avocet Systems, Inc. Unit Two combines both assembly and C, and uses the MASM and QuickC translators from Microsoft Corp.

INTENDED AUDIENCE

Although the text is introductory in nature, its depth of coverage and its gradual step-by-step approach make it appropriate for 2-year or 4-year technician, technology, or engineering students. The book’s modular design into 28 chapters makes it easy to customize a course’s content and approach to a particular audience.

APPROACH

Besides its overall content and purpose, as summarized above, a textbook also has a “personality”—that which makes it different from other texts in the field. What sets this text apart is its use of the *inventor’s approach*. That is, on successful completion of the material, you will be able to design and troubleshoot a simple microprocessor-based system starting completely from scratch. This real-world product development approach is deemed to be the most effective way to teach the “critical thinking” skills that are required of today’s technician and engineer. Also unique to this text is the use of an intelligent-machine (android) analogy throughout, designed to introduce the subject of artificial intelligence (AI) and to make the material more interesting, readable, and thought-provoking. . .and therefore easier to learn.

ORGANIZATION

Now that we have an overview of both the content and theme of the text, let us sharpen our focus in several areas.

Unit One

The first unit is divided into three parts and seventeen chapters. After a short introduction to the background and history of the microprocessor, we will enter the six chapters of Part I and concentrate on the hardware of the computer: its bus system, I/O ports, primary and secondary memory, and CPU. Part II takes us inside the 8085 for an X-ray look at program processing, timing, and waveforms. Part III covers software, and gives us a working knowledge of most of the 8085 instruction set. The final chapter in Unit One lets us apply all of our 8-bit knowledge to the interfacing, programming, and operation of Intel's SDK-85 single-board computer.

Unit Two

Unit Two is also divided into three parts. The four chapters of Part IV examine in detail the 16-bit 8086/88 and 32-bit 80486 microprocessors, and give us practice in assembly-language and C programming. Part V offers five chapters that look inside the IBM PC family of computers and see how programs are written for various subsystems, such as the keyboard, monitor, and printer port. Part VI concludes with two real-world applications that summarize much of the material presented within the text.

ADDITIONAL COVERAGE

To back up the material and to enhance the software development skills of the student, each chapter contains a "Questions and Problems" section. The solu-

tions to all questions and problems, as well as additional suggestions on the use of the text, are given in the *Instructor's Guide*. All Unit One programs listed in this text and in the *Instructor's Guide* can be run without modification on the Intel SDK-85 single-board computer; all programs developed in Unit Two can be run on any 8086/88-based single-board computer or directly on the IBM PC.

PREREQUISITES

The prerequisites for this text are a basic background in dc/ac and semiconductor electronics, experience in the use of the hexadecimal number system, and a working knowledge of digital circuits and gates. No previous knowledge of microprocessors or personal computers is assumed.

ARTIFICIAL INTELLIGENCE

Within the theme of artificial intelligence (AI), the text makes continuous reference to the evolutionary parallel between human and computer. This recurring theme is used primarily as a vehicle to motivate your interest and perhaps to make the material more exciting and fascinating. No attempt is made to rule out alternative beliefs or to convince you that a machine will ever be the equal of a human being. On the other hand, the similarities between computers and people should in no way be construed as mere fantasy; moreover, it is predicted that after completing this book you will come to regard your home computer a little more like your family pet (a living creature) and a little less like your family car (a simple tool).

To the reader: Good luck and good success.

Roy W. Goody
Computer/Electronics Technology
Mission College

C O N T E N T S

UNIT ONE—THE 8-BIT WORLD

| | | | |
|--|-----------|--|-----------|
| Chapter 1 The Microprocessor: An Overview | 2 | Chapter 3 Input and Output Ports | 19 |
| The Microprocessor Revolution | 2 | The Input Port | 19 |
| Following the Revolution: The Invasion | 2 | The Output Port | 22 |
| What is a Microprocessor? | 3 | Intelligent-Machine Update | 23 |
| Microprocessor Applications | 3 | <i>Questions and Problems</i> | 24 |
| A Brief History | 4 | Chapter 4 Introduction to Memory/Memory Hierarchy | 26 |
| The Fundamental Principles of Computer Action | 6 | Memory Hierarchy | 26 |
| Beyond the Fundamental Concepts | 8 | The Microcomputer Memory Spectrum | 27 |
| <i>Questions and Problems</i> | 9 | Virtual Memory | 28 |
| | | The Memory Map | 29 |
| | | Intelligent-Machine Update | 29 |
| | | <i>Questions and Problems</i> | 30 |
| | | Chapter 5 Primary Memory | 31 |
| | | RAM vs ROM | 31 |
| | | Static RAM (SRAM) | 32 |
| | | Dynamic RAM | 41 |
| | | Read-only Memory | 42 |
| | | A RAM/ROM System | 50 |
| | | Application-Specific ICs | 50 |
| | | Intelligent-Machine Update | 54 |
| | | <i>Questions and Problems</i> | 55 |
| Chapter 2 The Bus System | 11 | | |
| The Bus Concept | 11 | | |
| Bus System Categories | 12 | | |
| Tristate Circuits | 13 | | |
| Examples of Tristate Circuits | 13 | | |
| Buffering the Bus Lines | 14 | | |
| Common Bus Standards | 16 | | |
| Intelligent-Machine Design | 17 | | |
| <i>Questions and Problems</i> | 18 | | |


PART I HARDWARE: THE ANATOMY OF A COMPUTER 10

Chapter 6 Secondary and Backup Memory 56

Thin-Film Magnetic Technology 56
Secondary Memory Systems 62
Optical Storage 65
Backup Storage Devices 67
Selecting the Right Secondary/Backup
Combination 69
Intelligent-Machine Update 69
Questions and Problems 70

Chapter 7 The Central Processing Unit: Introduction to Processing Action 71

A Computer Analogy 71
Processing Action 76
Intelligent-Machine Update 76
Questions and Problems 78



PART II BASIC PROCESSING ACTION: THE METABOLISM OF A COMPUTER 79

Chapter 8 Introduction to Programming and Program Processing 80

Writing a Program 80
Writing the Simple Mimic Program 81
Teaching the Mimic Task 85
Performing the Mimic Task 86
A Problem with Syntax 86
Intelligent-Machine Update 88
Questions and Problems 88

Chapter 9 Timing and Multiplexing 89

The System Clock 89
8085 Timing Cycles 89
The Wait and Hold States 91
Multiplexing 94
Intelligent-Machine Update 96
Questions and Problems 98



PART III SOFTWARE: THE SPARK OF LIFE 99

Chapter 10 The Data-Transfer Group 100

Data-Transfer Verbs 100
Data-Transfer Addressing Modes 101

A Data-Transfer Example 107
Additional Study 108
Intelligent-Machine Update 108
Questions and Problems 109

Chapter 11 The Arithmetic Group 111

The Group 2 Verbs 111
Positive and Negative Numbers 112
Arithmetic Instruction Examples 113
The Increment and Decrement Instruction Types 114
An Arithmetic Program 116
Direct vs Indirect Addressing 116
The Addition Process 116
High-Level and Low-Level Flowcharting 119
Multiple-Precision Numbers 120
Multiplication 121
Division 122
BCD Mathematics 122
ASCII Mathematics 123
Fractions 124
Floating-Point Numbers 126
Intelligent-Machine Update 127
Questions and Problems 127

Chapter 12 The Logical Group 129

The Elements of Logic 129
The Group 3 Verbs 129
Boolean Operations 130
Solving the Mystery 131
Bit Manipulation 133
Bit Change of State 134
Intelligent-Machine Update 134
Questions and Problems 135

Chapter 13 Loops and Jumps: Introduction to Assembly-Language Programming 137

Loops and Jumps 137
Assembly-Language Programming 138
Program Assembly 140
Intelligent-Machine Update 143
Questions and Problems 144

Chapter 14 Reasoning: The Conditional Jump 145

The Elements of Reasoning 145
A Decision-Making Example 148
Machine-Assembly Update—I 150
Greater-Than/Less-Than Decisions 150

Multipronged Forks in the Road 151
 Additional Decision-Making Examples 152
 Machine-Assembly Update—II 157
 Intelligent-Machine Update 157
Questions and Problems 160

Chapter 15 Calls, Subroutines, and Stacks 162

Subroutines 162
 The CALL Instruction 163
 A Subroutine Example 167
 Nesting Subroutines 168
 The PUSH and POP Instructions 168
 When to CALL 170
 The CALL and RETURN Conditions 171
 Parameter Passing 172
 Machine-Assembly Update—I 172
 Machine-Assembly Update—II 173
 Putting It All Together 177
 The SDK-85's Special Monitor Subroutines 178
 Intelligent-Machine Update 178
Questions and Problems 178

Chapter 16 Interrupts 181

An Everyday Example 181
 8085 Interrupt Processing 182
 SDK-85 Interrupt Processing 185
 A Simple Interrupt Test Program 186
 Additional Interrupt Considerations 188
 Machine-Assembly Update—I 191
 Multitasking via Interrupts 191
 Machine-Assembly Update—II 193
 Intelligent-Machine Update 193
Questions and Problems 196

Chapter 17 Programmable Peripheral Chips: The SDK-85 Single-Board Computer 197

Computers as a Hardware/Software Blend 197
 The SDK-85 Single-Board Computer 198
 The 8155A 256 × 8 RAM with I/O Ports and
 Timer 199
 Interfacing Additional Programmable Chips to the
 SDK-85 System 205
 Intelligent-Machine Update 212
Questions and Problems 215

UNIT TWO—THE 16-BIT WORLD

PART IV ADVANCED PROCESSORS: THE NEWEST GENERATION 218

Chapter 18 The 8086/88 16-Bit Microprocessor 219

The 8086/88 Architecture 219
 Multiprocessing 222
 Interrupts 223
 The 8086/88 Instruction Set 224
 The 8086/88 Addressing Modes 226
 Instruction Coding—An Example 227
 The 8087 Math Coprocessor 229
 Intelligent-Machine Update 230
Questions and Problems 234

Chapter 19 The 80486 Microprocessor 235

Introduction to the 80486 235
 80486 Architecture—A Quick Look 236

The 80486 Major Features 239
 Intelligent-Machine Update 242
Questions and Problems 243

Chapter 20 8086/88 Assembly-Language Programming 244

Top-Down Structured Design 244
 The Development Process 244
 Ten Demonstration Projects 245
 The Instruments of Troubleshooting 265
 Intelligent-Machine Update 272
Questions and Problems 274

Chapter 21 The C High-Level Language 275

Top-Down Structured Design 275
 Ten Demonstration Programs 275
 Intelligent-Machine Update 287
Questions and Problems 287



PART V THE IBM PC FAMILY 288

Chapter 22 The IBM PC Family—A Brief Overview 289

Hardware 289
The Operating System Software 290
Applications Programs 294
Writing Programs for IBM PC Execution 294
Intelligent-Machine Update 296
Questions and Problems 297

Chapter 23 Keyboard/Video Systems—Animation 298

Video Systems 298
Keyboard Systems 300
Project—Computer Animation 301
Intelligent-Machine Update 306
Questions and Problems 308

Chapter 24 The 8253 Timer/Clock Chip—Computer Music 309

The 8253 Programmable Peripheral Timer 309
Project—Computer Music 309
Intelligent-Machine Update 314
Questions and Problems 316

Chapter 25 Parallel I/O—A Printer/Typewriter 317

Synchronous vs Asynchronous 317
Requirements of Asynchronous Transmission 318
Handshaking 318
Programmed I/O vs Interrupt I/O 319
Memory-Mapped I/O vs Isolated I/O 321
The IBM PC Parallel Port 322
Project—Computer Typewriter 322
Intelligent-Machine Update 324
Questions and Problems 328

Chapter 26 Serial I/O—Computer Talk 329

Synchronous vs Asynchronous 329
Simplex/Duplex Transmission 331
The RS-232C Standard 331
The 8251A Programmable Communication Interface 331
Local-Area Networks (LANs) 335

Telecommunications 340
Fiber-Optic Digital Highways 344
The IBM PC Serial Port 345
Project—Computer Talk 347
Intelligent-Machine Update 350
Questions and Problems 351



PART VI SPREADING OUR WINGS—TWO REAL-WORLD APPLICATIONS 352

Chapter 27 Signal Processing—A Low-Pass Filter 353

Signal Processing Basics 353
Digital-to-Analog Conversion 353
Analog-to-Digital Conversion 354
A Low-Pass Filter 359
Project—Low-Pass Filter 362
Intelligent-Machine Update 367
Questions and Problems 371

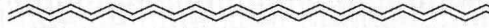
Chapter 28 A Robotic Control System—Light Tracking Radar 372

Cybernetics 372
Stepper Motor 373
Microcontrollers 374
Robotics 376
Project—Light Tracking Radar 376
Intelligent-Machine Update: A Final Word 379
Questions and Problems 387
Glossary 388

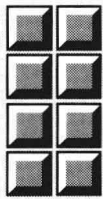
Appendixes

A 8085 Instruction Set 395
B SDK-85 Operation 403
C 8085 Instruction Set Machine-Cycle Analysis 405
D The 10 Secret Op Codes 410
E 8080/8085 Assembly-Language Reference Card 411
F 8086/88 Instruction Set 412
G ASCII Code Table 421
H Program Development Using Microsoft Software 422
Index 427

U N I T O N E



THE 8-BIT WORLD



Before studying the complexities of the human body, a medical student first uses simpler creatures to master the characteristics that are common to all animals. When studying computers, the same is true, and therefore Unit I of this text is based on Intel's 8-bit 8085 micro-processor, and stresses those aspects of computer technology that are common to all computers. We will find that the 8-bit 8085 is the ideal vehicle to begin our studies, and that all the major principles found in the more advanced 8086 and 80486 of Unit II are also found in the 8085— but in a scaled-down, easier-to-learn package.

Unit I is divided into three parts and seventeen chapters. Before entering Part I, an introductory chapter gives us an historical background and reviews the most fundamental principles of computer action. In Part I we look at the anatomy (hardware) of the 8085, and we will see an elegant and simple design that was later expanded to create the next generation 8086. In Part II we study the metabolism (processing action) of the 8085, and we will see a clean and efficient process that was later streamlined to yield the higher processing speeds required by the 8086. In Part III we will learn the vocabulary (instruction set) of the 8085, and we will see a complete and powerful set of commands that was later refined and expanded for use in the 8086.

CHAPTER 1

The Microprocessor: An Overview

The microprocessor has flourished because—like its animal kingdom counterpart—it found a fertile niche in the electronic environment and survived by being the fittest of the species.

In this chapter we will look at the historical development of the microprocessor and examine its fundamental nature. As you will see, we can learn a great deal about the computer by studying ourselves.

THE MICROPROCESSOR REVOLUTION

By the turn of this century the first industrial revolution, which began in England around 1720, was well under way, and machine power was replacing muscle power at an ever-increasing pace. Since the early 1970s a second and far more profound industrial revolution has gathered strength. This time, however, machine power will enhance and replace not the muscle power of the human species but the *brain* power.

The machine we are talking about is, of course, the computer. But computers are not all that new. Why then has the microprocessor—essentially a computer on a chip—ushered in the second industrial revolution?

Strangely enough, the answer to this question can be found on the Salisbury Plain in southwest Eng-

land—the site of Stonehenge. This curious arrangement of 30-ton stones, each hewn from a distant quarry and transported hundreds of miles, is actually an ancient neolithic computer, constructed 500 years after the Egyptian pyramids to predict eclipses and other celestial events. Clearly, if today's computers were of Stonehenge design—requiring 30-ton components and hundreds of years to design and construct, fashioned from hard-to-get materials, dedicated to a single purpose, slow, inaccurate, and very limited in power—the second industrial revolution would not be taking place. But today's microprocessor-based computers are just the opposite. They are inexpensive, ultrasmall, lightweight, multipurpose, highly accurate, breathtakingly fast, and incredibly powerful—and the second industrial revolution is under way.

FOLLOWING THE REVOLUTION: THE INVASION

Although the computers of the 50s and 60s were successful, they were like lumbering dinosaurs, awaiting the appearance of systems that were smaller and quicker. That day came in 1971, when Intel Corporation developed the 4004. Originally marketed as a simple calculator chip, it emerged as the world's first

general-purpose information processor. Like an invading virus, this single chip of 3,000 transistors multiplied, evolved, and flourished into a diverse family of silicon systems.

These intelligent systems on a chip quickly spread from California's "Silicon Valley" to all corners of the world. They have been incorporated into a wide variety of consumer products, and they formed the brain of today's popular desktop personal computers (PCs). Soon after, these chips evolved into full-fledged workstation computers, with advanced math and graphics capabilities. Eventually they will invade the last stronghold—the world of the mainframe computer—and from there they will break new ground into parallel processing and artificial intelligence.

Looking at the Intel "family tree" of Figure 1.1, we see the many similarities between the silicon-based microprocessor and the carbon-based family of living creatures. We see the evolution of complexity in the general-purpose branch, the advent of specialization with the many single-chip microcontrollers, and the driving force of diversity with the "new world" RISC (reduced-instruction-set) and parallel processor branches.

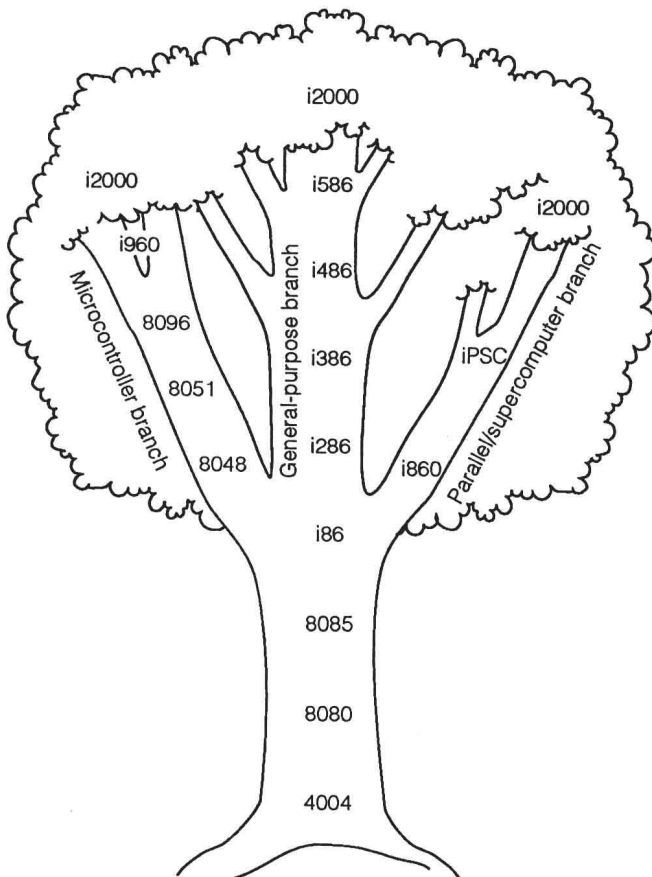


Figure 1.1 The Intel family tree.

As we enter the 21st century, we envision the i2000 (Intel's mythical microprocessor of the year 2000). Of course the i2000 will be a family of processors, from general-purpose mainframe powerhouses to those dedicated to specific (*embedded*) applications. For supercomputer applications, we will see the complexity increase from 2 million transistors on a chip the size of a dime, to over 100 million transistors on a chip a mere square inch in area. The clock speed will jump from 50 MHz to over 250 MHz, and it will perform vector (parallel) processing at the rate of 2000 MIPS (million instructions per second). It will dazzle our senses with such feats as real-time graphical animation, made possible with the help of a floating-point processor capable of 1 billion operations per second.

Another version of the mythical i2000 will bring a complete PC on a single chip, integrating such functions as hard-disk control and networking. It will be smarter, faster, and "friendlier," incorporating such features as advanced voice recognition and multimedia presentations which combine television, stereo, telephone, optical storage, and computing. Most exciting of all, these chips of the 21st century are not destined for some future generation—they will be part of *our* world in just a few short years!

WHAT IS A MICROPROCESSOR?

A *microprocessor* is a VLSI (very large scale integrated) programmable logic device on a single silicon chip, less than 1 inch on a side. Under control of a stored program in memory, the device performs mathematical, logical, and decision-making operations, and communicates with the outside world through *input/output (I/O) ports*. The many varieties and types of microprocessors usually fall into one of two categories: the *general-purpose microprocessor*, containing on one chip all necessary computer components except memory and I/O ports; and the *microcontroller* that offers on-chip memory and a variety of I/O ports.

A *personal computer* (sometimes called a *desktop computer*) is an entire computer system, including a microprocessor, external disk memory, and standard I/O devices such as keyboard, monitor, and printer port. Add high-resolution graphics and high-speed floating-point math operations and we enter the realm of the *workstation*.

MICROPROCESSOR APPLICATIONS

Like the great inventions of the past, the microprocessor will prove its worth by what it can do—its *applications*. Certainly, few inventions have pervaded our lives as completely and quickly as the microprocessor.

The applications are so vast that the microprocessor invasion has already spawned an electronic age, in which an army of willing servants watches over us from morning to night. We wake up to a microprocessor-controlled alarm clock, read a newspaper that was edited by a microprocessor-based word processor, and watch the morning news on television as a microprocessor fine-tunes the picture. We leave our homes guarded by a microprocessor "watchman," drive to work as a microprocessor instantly adjusts the car's timing and fuel/air mixture for optimum performance, and converse over a microprocessor-controlled car phone. Our commute is speeded up by microprocessor-based traffic-control systems, our on-the-job productivity is increased by computerized inventory systems, and our workrooms are environmentally controlled by a microprocessor. We shop at a store where an intelligent (microprocessor-controlled) cash register inputs data to a central computer system that automatically inventories and orders merchandise. We go out for dinner and have our meal cooked under micro-

processor control, and we finish our day watching the stunning computer graphics of the latest movie release. Then we go to bed and dream.

When we wake up it will be the future and our alarm will talk to us in perfect English. Our home computer will accurately forecast the weather and diagnose our ailments. We will realize how much we have come to rely on our computer for its *judgment* as well as its knowledge, and it will be hard to tell whether we are still dreaming.

A BRIEF HISTORY

Like most inventions, the microprocessor resulted from the gradual blend of many scientific trends. Those most important to the development of the microprocessor were the mathematical, electronic, and computer trends. As shown in Figure 1.2, several important milestones finally led to today's advanced microprocessor.

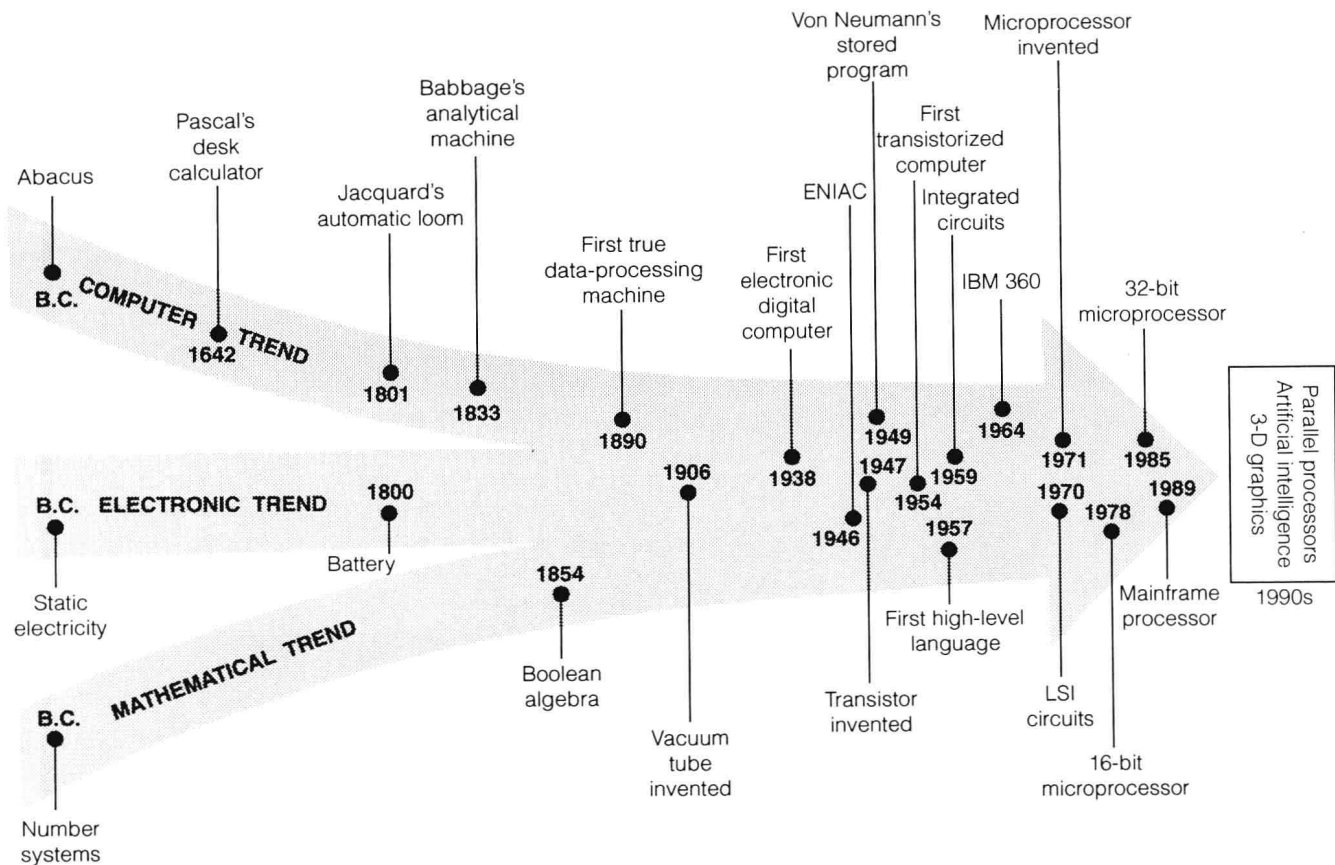


Figure 1.2 Technological trends and milestones in microprocessor development.

- *The early days:* The first calculations were done on the human hand. From this simple beginning the familiar decimal, binary, and hexadecimal number systems eventually evolved. The first mechanical device to make use of number systems was the *abacus*, a calculating device that dates back before the birth of Christ and is still used today.
- *1642—Calculating machine:* Blaise Pascal invented the first “desk calculator.” It was strictly a mechanical device, using systems of gears to add and subtract. Since the precision machining of parts was still many years away, the idea slowly died. But Pascal’s name did not die away, for a popular high-level computer language is named after him.
- *1801—Automatic loom:* Joseph Jacquard’s idea revolutionized the weaving industry and resurfaced many years later in the computer industry. It was an automatic loom that used punched cards (IBM cards!) to control the pattern.
- *1833—“Analytical engine”:* Charles Babbage, more than any other computer pioneer, deserves the title “father of modern digital computers.” His “analytical engine,” developed to calculate and print mathematical tables, incorporated many of the principles of modern digital computers. Babbage was the first to envision the stored-program concept, in which all numbers *and* instructions were read before calculations began. In other words, once programmed, the machine worked without human intervention. Unfortunately, for a number of practical reasons that plague all inventions ahead of their time, it was never developed beyond the prototype stage.
- *1854—Boolean algebra:* Can formal logic be described mathematically? George Boole discovered that it could, and he developed a symbolic form of logic called *Boolean algebra*, a subject familiar to every student of digital electronics. The door to computer design was now wide open.
- *1890—Electric tabulating machine:* Herman Hollerith developed the first true data-processing machine. Using his machine, the task of tabulating the 1890 census was reduced from 20 to 3 years.
- *1906—Vacuum tube:* The electronic pathway began in earnest with the application of the triode vacuum tube, invented by Lee De Forest. Mathematical manipulations could now be done electronically rather than mechanically, with a quantum jump from seconds to milliseconds in processing speed.
- *1938—Electronic digital computer:* John V. Atanasoff formulated the basic ideas for computer memory and associated logic, and built the first electronic digital computer. Based on vacuum tubes, it paved the way for all work to follow.
- *1946—The large-scale electronic digital computer:* Prompted by the wartime need to calculate ballistic tables to produce trajectories for artillery and bombing, the U.S. Army funded the Electronic Numerical Integrator and Calculator (ENIAC) project. Completed in 1946, ENIAC was the first large-scale electronic digital calculating machine. By today’s standards it was a monster. Composed of 18,000 vacuum tubes, it weighed in at 30 tons, occupied 1,500 square feet, and consumed 130,000 watts of power. However, it could multiply two numbers in about 3 milliseconds—a thousand times faster than ever before—and without the use of a single moving part. It was turned off for the last time in 1955.
- *1949—Stored-program computer:* Although ENIAC could perform individual mathematical operations at high speed, it had to wait for each instruction to be entered by its human operators. Intent on removing this human factor, John von Neumann picked up on the stored-program concept first conceived by Babbage and proposed placing computer instructions as well as data in the computer’s memory. Whenever a sequence of instructions was to be performed, the computer could read each instruction from memory without waiting for human intervention. Storing the program inside the computer along with the data allows today’s computers to operate at high speed (and distinguishes a computer from a calculator).
With the stored-program concept, the last major hurdle to modern computer design was crossed, and in May of 1949 the first digital computer based on the stored-program concept went into operation. Named the Electronic Delay Storage Automatic Calculator (EDSAC), it set the stage for all computers to follow.
- *1954—Transistorized computer:* Although based on electronics, ENIAC and EDSAC were still of “Stonehenge” design—far too big, bulky, and power consuming to command widespread attention. In 1947, however, a breakthrough was made that can only be described in fairy-tale terms, for like Alice in Wonderland, the solid-state research it set in motion was destined to shrink the size of computers a thousandfold and more. Invented by John Bardeen, W. H. Brattain, and W. B. Shockley at Bell Laboratories in 1947, the transistor was the seed from which the second industrial revolution sprouted.

The first product of this seed flowered in 1954 with the introduction of the TRAnsistor Digital Computer (TRADIC). By 1960 hundreds of transistorized computers were in operation, processing data faster and at lower cost than ever before. The days of the vacuum tube were numbered.

- **1957—High-level language:** Primarily because of their awesome size, the early vacuum-tube computers quickly acquired a public image of “giant brains,” fearsome machines to be viewed with apprehension and mistrust. This image was largely undeserved, of course, for these early computers were very crude, and in one area in particular—languages—they were downright primitive. The only language these early machines understood was machine language—the language of ones and zeros—that made programming a cumbersome, error-prone, and difficult endeavor.

The first major breakthrough in language development was made by an IBM research team headed by John Backus. Primarily interested in developing a language to solve mathematical calculations, the team devised a way of writing a program using mathematical notation instead of machine language. Using common typewriter symbols to write and enter the program, the computer would then translate (compile) the sequence of symbols into the required machine-language instructions.

Introduced in 1957, the language was called **FORTRAN** (for FORMula TRANslation) and is still in widespread use today. By 1960 numerous high-level languages were in use, including COBOL (COMMON Business-Oriented Language), which gave the business community many of the same advantages that FORTRAN gave the scientific community.

- **1959—Integrated circuit:** By applying the principles of photolithography to flat surfaces of silicon, and by developing the method of solid-state diffusion for introducing the impurities that create *p* and *n* regions, engineers found they could construct entire circuits, consisting of many transistors, on a single chip of silicon. This was the basis of the integrated circuit, a technology that was to show the same explosive growth in sophistication as took place in the human brain during the end of the last ice age.
- **1964—Integrated-circuit computer:** On April 7, 1964, IBM introduced the standard mainframe computer, the System/360 (called 360 because the system was said to encompass the full range of scientific and business applications). Using highly reliable, mass-produced, integrated circuits, it could

perform in 1 second nearly a half million computations at a cost of less than 10 cents. The System/360 also introduced a number of new input/output and auxiliary storage devices.

- **1970—Large-scale integrated (LSI) circuit:** From the early 1960s to the early 1970s, the maximum complexity of integrated circuits doubled approximately every 18 months. By 1970 more than 15,000 transistors could be etched onto a single chip of silicon, an achievement that made the handheld calculator feasible.
- **1971 to present—era of the microprocessor:** In 1971 the computer, electronic, and mathematical trends came together in a unique way, and the microprocessor emerged on the scene—initially with little fanfare. In a little over 20 years, though, it went from a simple 4-bit LSI device developed for calculators (the Intel 4004), to the iAPX586, a 32-bit SLSI (super large scale integrated) enhanced microprocessor family that forms the heart of a system resembling a mainframe computer.
- **Some time in the future—first true intelligent machine:** In the field of microprocessors, the future blends with the past so quickly that no historical survey would be complete without a word about what lies ahead. Most far-reaching of all is a fourth scientific trend that is now merging with the ongoing development of the microprocessor. This fourth trend, known as *artificial intelligence*, will combine with the new parallel array processors to produce the true intelligent machine. Unlike the human brain, which must depend on the relatively slow process of biological change, the intelligent machine made of silicon is under no such restrictions. We can only wonder where the new technology will lead us. Soon a true learning machine will emerge, *one able to modify its program based on learning experience*. What would be the result if we taught two such learning machines to play chess, and what if they were pitted against each other, playing games at the speed of light for a month or a year? At the end of the time, what would we find? Perhaps a new way of thinking, or a new philosophy, or a new mathematics—or perhaps something we would not be able to understand at all!

THE FUNDAMENTAL PRINCIPLES OF COMPUTER ACTION

The apparent similarity between people and computers presents us with an exciting possibility: If computers and humans do things in a similar way, then to

understand the more basic concepts of computer action, perhaps we should begin by studying ourselves.

What separates us and other members of the animal kingdom from the world of inanimate objects? The answer is very straightforward: we perform tasks, we do things.

To use an everyday example, consider a major-league outfielder catching a baseball. Even the most cursory analysis of this simple task reveals that it is composed of individual steps, taken in sequential order:

Run under ball
Raise glove
Catch ball

Each step in the sequence commands a specific operation. In computer terminology, each command is called an *instruction*. To complete the task of catching a baseball, then, we simply go through the instructions in order. *A computer performs a task in precisely the same way*. Therefore, by studying our own actions, we already have uncovered perhaps the first and most fundamental of all computer concepts.

1. *A computer performs a task by processing a sequential list of instructions.*

Continuing on, we find that two major items are required to catch a baseball: the list of instructions and the collection of physical components (player, baseball, and glove).

2. *For a computer, the list of instructions is called software and the physical circuitry is called hardware* (see Figure 1.3).

When we look at the physical requirements in greater detail, we find that the following parts of our anatomy are used to catch a baseball: an *eye* to input data about the ball's position, a *voice* to call for the ball, a *memory* to hold the stored program, a *computation and logic* area of the brain to compute the ball's trajectory, and a *central nervous system* to coordinate all activities.

3. *A computer consists of five basic hardware blocks: input, output, memory, arithmetic/logic unit (ALU), and control unit* (see Figure 1.4).

As demonstrated below, we find that each of the three instructions required to catch a baseball is composed of two parts—a verb, or action portion, and a noun, or object portion:

| Verb | Object |
|-----------|--------|
| Run under | ball |
| Raise | glove |
| Catch | ball |

4. *Each instruction processed by a computer consists of two parts: the action part is called the operation code and the noun portion is called the operand.*

When each of our three instructions is carried out, we find that the action portion (such as “raise”) directs the activities of the object portion (such as “glove”).

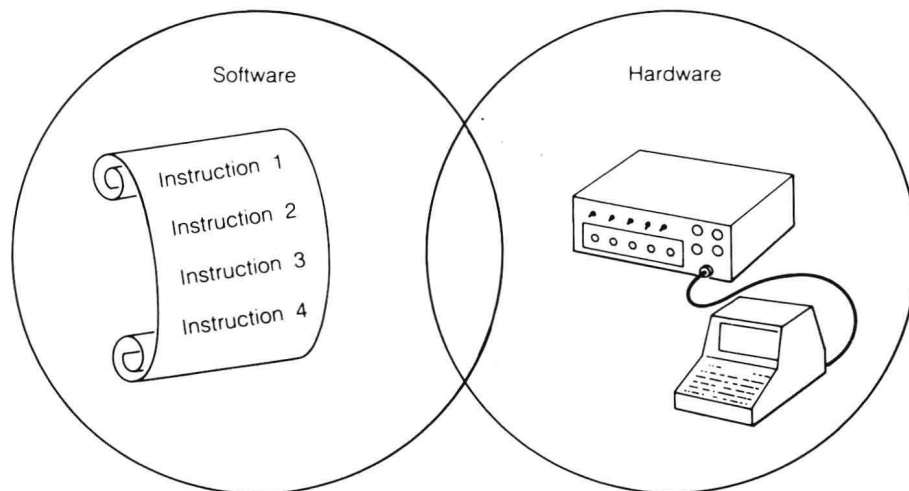


Figure 1.3 A computerized solution to a problem—both software and hardware required.

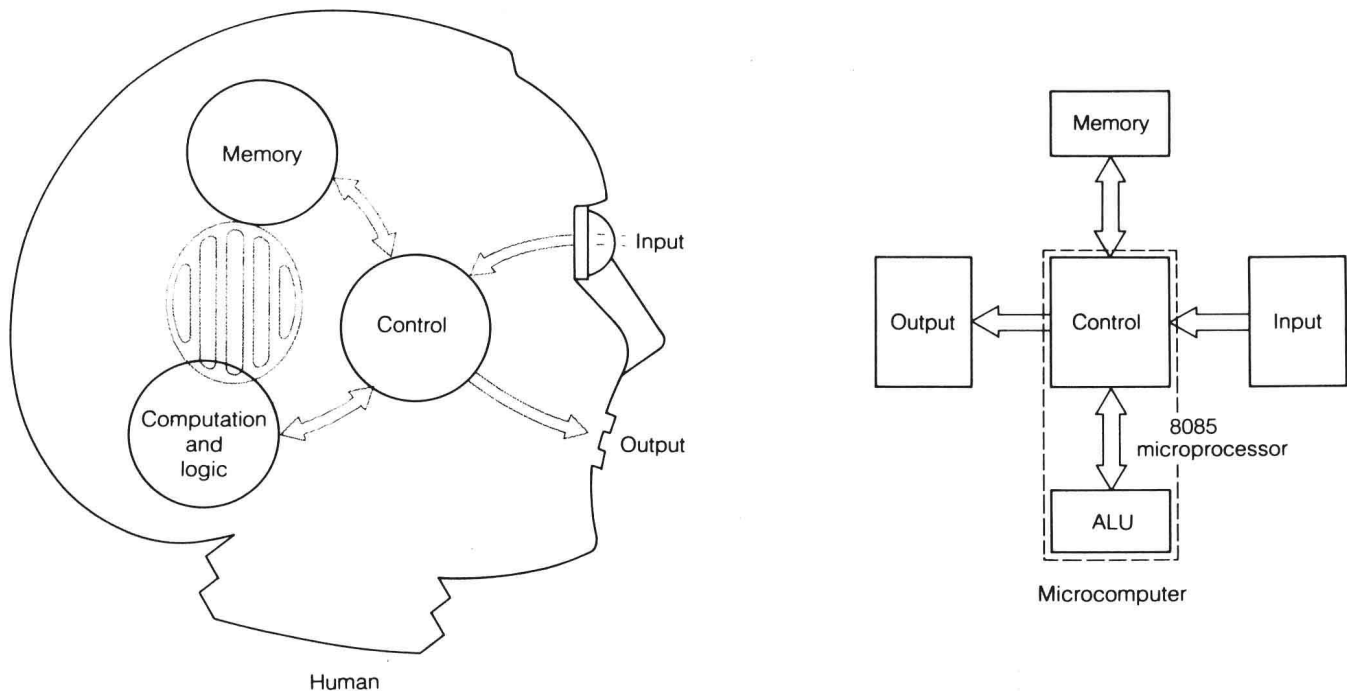


Figure 1.4 The five basic hardware blocks are the same for both human being and computer.

5. *When a computer instruction is executed, the operation code directs the activities of the operand.*

During training, we find that a baseball player quickly commits to memory the sequence of instructions for catching a baseball. In other words, the player has *learned* the sequence of steps.

6. *To program a computer for a specific task, we enter the proper sequence of instructions into its memory.*

To catch a baseball, we find that it involves a three-step process: we take in information, manipulate it mentally, and output the result.

7. *The basic processing cycle of a computer system consists of input of data, manipulation of data, and output of display of result.*

BEYOND THE FUNDAMENTAL CONCEPTS

When described in their most basic terms—as we have done in this chapter—the actions that human beings take to catch a baseball or perform other common tasks do not seem complex. It appears we use our marvelous mental machines with little regard for the intricate operations involved. Unfortunately, we cannot

take computers so lightly, for to design and troubleshoot computer systems we must understand precisely how every component functions and interacts, and how every signal and waveform carries forward the processing cycle. The tasks before us are clear:

- The five blocks of the computer system must be opened up and the inner workings revealed.
- The processing action itself must be studied in detail.
- The computer's vocabulary (all the instructions it can follow) must be learned, and all instructions must be converted to a form the computer will understand (English will not do). In addition, the computer must be taught (programmed) to perform simple tasks.
- Newer-generation processors and languages must be introduced.
- The operation of a computer system (such as the IBM PC) must be studied.
- The computer applications in a real-world environment must be performed.

Each of these tasks corresponds closely to Parts I, II, III, IV, V, and VI of this book.