The McGraw-Hill Computer Handbook

Editor in Chief
Harry Helms

The McGraw-Hill Computer Handbook

Editor in Chief
Harry Helms

Overview by
Adam Osborne

Foreword by
Thomas C. Bartee

McGraw-Hill Book Company
New York St. Louis San Francisco Auckland
Bogotá Hamburg Johannesburg London Madrid
Mexico Montreal New Delhi Panama Paris
São Paulo Singapore Sydney Tokyo Toronto

Library of Congress Cataloging in Publication Data
Main entry under title:

The McGraw-Hill computer handbook.

Includes index.
1. Computers – Handbooks, manuals, etc.
2. Programming (Electronic computers) – Handbooks, manuals, etc. 3. Programming languages (Electronic computers) – Handbooks, manuals, etc. I. Helms, Harry L. II. McGraw-Hill Book Company.
QA76.M37 1983 001.64 83-1044
ISBN 0-07-027972-1

1 2 3 4 5 6 7 8 9 0 KGP/KGP 8 9 8 7 6 5 4 3

ISBN 0-07-027972-1

The editors for this book were Patricia Allen-Browne and Margaret Lamb, the production supervisor was Teresa F. Leaden, and the designer was Mark E. Safran. It was set in Times Roman by University Graphics, Inc.

Printed and bound by The Kingsport Press.

Contributors

Bartee, Thomas C. *Harvard University*
Conroy, Thomas F. *International Business Machines Corp.*
Erickson, Jonathan *Radio Shack Technical Publications*
Gault, James W. *North Carolina State University*
Gear, C. William *University of Illinois*
Givone, Donald D. *State University of New York at Buffalo*
Hamacher, V. Carl *University of Toronto*
Hellerman, Herbert *State University of New York at Binghamton*
Helms, Harry L. *Technical Writer and Consultant*
Hohenstein, C. Louis *Hohenstein and Associates*
House, Charles H. *Hewlett-Packard*
Kohavi, Zvi *University of Utah*
Koperda, Frank *International Business Machines Corp.*
Miastkowski, Stan *Rising Star Industries, Inc.*
Newman, William M. *Queen Mary College, London*
Pimmel, Russell L. *University of Missouri*
Roesser, Robert P. *University of Detroit*
Sproull, Robert F. *Sutherland, Sproull and Associates*
Stout, David F. *Dataface*
Tucker, Allen, Jr. *Georgetown University*
Vranesic, Zvonko G. *University of Toronto*
Wiatrowski, Claude A. *Mountain Automation Corp.*
Wiederhold, Gio M. *Stanford University*
Zaky, Safwat G. *University of Toronto*

Overview

TRENDS IN THE MICROCOMPUTER INDUSTRY

Without a doubt, IBM's Personal Computer strategy cast the shape of the microcomputer industry in 1982— and probably for the rest of the decade. It was not simply sales volume or market share that made IBM's Personal Computer such a formidable factor in 1982. Rather, it was the marketing strategy IBM adopted. It is a strategy all other microcomputer industry participants who wish to survive will have to adopt.

Prior to the IBM Personal Computer, industry leaders (such as Apple, Commodore, and Radio Shack) all strove to build unique hardware which, whenever possible, would run programs that could not be run on any competing machine. Furthermore, these companies vigorously fought competitors attempting to build look-alike microcomputers.

That was the old minicomputer industry philosophy, adopted by too many microcomputer manufacturers. Well in advance of IBM's entry, the ultimate fallacy of this philosophy was evident. The reason was CP/M, an operating system standard on none of the three leading personal computers (Apple, Commodore, and Radio Shack). Nevertheless, CP/M not only survived but thrived. CP/M was kept alive by more than the flock of secondary microcomputer manufacturers. A large number of Radio Shack and Apple microcomputers were also running CP/M, even though it required additional expense for unauthorized software and additional hardware for the Apple.

Was there a message in the extremes to which customers would go to thwart the intent of microcomputer manufacturers and run CP/M? Indeed there was, and it was that de facto industry standards are overwhelmingly desirable to most customers.

That message was not lost on IBM. Few messages of validity are; that is why IBM has grown to be so successful.

And so when IBM introduced its Personal Computer, it went about making its entry a new de facto standard. Any hardware manufacturer who wanted to could build a product compatible with the IBM Personal Computer. Software vendors were given every encouragement to adopt the IBM standard. The first

independent microcomputer magazine dedicated to the IBM Personal Computer grew to be one of the most successful magazines in the business within six months of its first publication. People bought the magazine and advertised in it in unprecedented volumes.

There are already several microcomputers built by companies other than IBM that are compatible with the IBM Personal Computer. Already there are probably more software companies devoting themselves to the IBM standard than any other with the possible exception of CP/M.

Within a short time, I predict there will be two de facto industry standards for microcomputers: CP/M running on the Z80 for 8-bit systems and IBM (MDOS and CP/M 86) running on the 8086 or 8088 for 16-bit systems. Companies not supporting one or both of these standards have a tortuous, uphill fight ahead of them.

One can well argue that 8-bit microprocessors are generally obsolete and that the 8086 and 8088 are among the least powerful 16-bit microprocessors. But what has that to do with anything? If these microprocessors are adequate for the tasks they are being asked to perform, then any theoretical inadequacies will not be perceived by the end user. And even if the de facto standards are not the best in whatever area they have become standard, what does it matter providing their shortcomings are not apparent to the user?

The difference between the microcomputer and minicomputer industries is that microcomputers are rapidly becoming consumer products. It will be far more difficult for microcomputer industry managers to impose their will on a mass market of consumer buyers than it was for minicomputer manufacturers to manipulate a relatively small customer base (which was commonly done in the early 1970s).

This message has not been learned by many present participants in the microcomputer industry. But this message will determine more than anything else who will be the survivors when the inevitable industry shakeout occurs.

Adam Osborne
President
Osborne Computer Corporation
1983

Foreword

Since the 1950s the digital computer has progressed from a "miraculous" but expensive, rarely seen, and overheated mass of vacuum tubes, wires, and magnetic cores to a familiar, generally compact machine built from hundreds of thousands of minuscule semiconductor devices packaged in small plastic containers.

As a result, computers are everywhere. They run our cash registers, check out our groceries, ignite our spark plugs, and manage the family bank account. Moreover, because of the amazing progress in the semiconductor devices which form the basis for computers, both computers and the applications now being developed are only fragments of what will soon be in existence.

The Computer Handbook which follows presents material from the basic areas of technology for digital computers. The Handbook progresses from hardware through software to such diverse topics as artificial intelligence, robotics, and voice recognition. Microprocessors, the newest addition to computer hardware, are also discussed in some detail.

Computers are beginning to touch the lives of everyone. If you are ill, the hospital will be full of computers (there could be more computers than patients in a modern hospital; medical instrument designers make considerable use of microprocessors). Schools have been using computers for bookkeeping for years and now use computers in many courses outside computer science. Even elementary schools are beginning to use computers in basic math and science courses. Games designed around microprocessors are sold everywhere. New ovens, dishwashers, and temperature control systems all use microprocessors.

The wide range of applications brings up some basic philosophical questions about what the future will bring. System developers can now produce computers which speak simple phrases and sentences reasonably well—bank computers probably give the teller your balance over the telephone in spoken form and some of the newer cars produce spoken English phrases concerning car operation. Computers also listen well but have to work hard to unscramble what is said unless the form of communication is carefully specified. This is, however, a hot research area and some details are in this book.

The speech recognition problem previously described is a part of a larger problem called *pattern recognition.* If computers can become good at finding patterns, they can scan x-rays, check fingerprints, and perform many other use-

ful functions (they already sort mail). The human brain is, however, very good at finding patterns even in the presence of irrelevant data (noise), and research in this area faces many challenges if computers are to become competitive. If, however, computers can become good at recognizing speech and returning answers verbally, it might be possible even to enter programs for computers and data verbally. This would make it possible literally to tell the computer what to do and have the computer respond, provided the directions were clear, contained no contradictions, etc.

While verbal communication might facilitate the programming of a computer, there would still be the problem of what language to use. There are many proponents of English, but English need not be precise and lacks the rigidity now required by the computer and its translators. Certainly much has been done in this area and the steady march from machine-like assemblers to today's high-level programming languages testifies to the need for and emphasis on this area. Much more is needed, however, and the sections of this Handbook fairly represent the state of this art and point to the direction of future work.

Robotics also presents an outstanding area for future development. Factories now use many robotic devices and research labs are beginning to fill with robots in many strange and wonderful forms. Waving aside the possibility and desirability of robots for maids, waitresses, waiters, ticket sales persons, and other functions already exploited by television and movies, there are many medical operations and precision production operations which can and will be performed by computer-guided robots (often because they are better than humans).

We often complain that others do not understand us, and at present computers do not understand us; for a while we will have to be content with computers which will simply follow our directions. Computer memories are making gains on human memories, however. Largely due to the ingenuity of memory device designers, the memory capacity of large machines now competes with our own but the different organization of the brain seems to give it substantial advantages for creative thought. Artificial intelligence delves into this area. In some areas formerly relegated to "human" thought, computers do quite well, however. For example, in such straightforward mathematical systems as Euclid's geometry, computers already perform better than might be expected; in a recent test a computer proved all the theorems in a high school test in minutes.

I think that to be really comfortable with computers it is necessary to have some knowledge of both hardware and software. In order to make computers more widely used, there is a tendency to make consumer-oriented personal computers appear to be "friendlier" than they really are. This limits their flexibility and presents users with a mystery element which needs to be and can be dissolved by a little knowledge of actual computer principles. A handbook such as this can be very helpful to users in dissolving some of the mystery. At the same time, such a handbook can open new doors in exploration and serve as a continuing reference.

Thomas C. Bartee
Harvard University
1983

Contents

sec. 1

Computer History and Concepts

Herbert Hellerman

1-1 INTRODUCTION

The modern general-purpose digital computer system, which is the subject of this book, is the most versatile and complex creation of mankind. Its versatility follows from its applicability to a very wide range of problems, limited only by human ability to give definite directions for solving a problem. A **program** gives such directions in the form of a precise, highly stylized sequence of statements detailing a problem-solution procedure. A computer system's job is to reliably and rapidly execute programs. Present speeds are indicated by the rates of arithmetic operations such as addition, subtraction, and comparison, which lie in the range of about 100,000 to 10,000,000 instructions per second, depending on the size and cost of the machine. In only a few hours, a modern large computer can do more information processing than was done by all of mankind before the electronic age, which began about 1950! It is no wonder that this tremendous amplification of human information-processing capability is precipitating a new revolution.

To most people, the words "computer" and "computer system" are probably synonymous and refer to the physical equipment, such as the central processing unit, console, tapes, disks, card reader, and printers visible to anyone visiting a computer room. Although these devices are essential, they make up only the visible "tip of the iceberg." As soon as we start to use a modern computer system, we are confronted not by the machine directly but by sets of rules called **programming languages** in which we must express whatever it is we want to do. The central importance of programming language is indicated by the fact that even the physical computer may be understood as a hardware interpreter of one particular language called the **machine language.** Machine languages are designed for machine efficiency, which is somewhat dichotomous with human convenience. Most users are shielded from the inconveniences of the machine by one or more languages designed for good man-machine communication. The versatility of the computer is illustrated by the fact that it can execute translator programs (called generically **compilers** or **interpreters**) to transform programs from user-oriented languages into machine-language form.

It should be clear from the discussion thus far that a computer system consists of a computer machine, which is a collection of physical equipment, and also programs, including those that translate user programs from any of several languages into machine language. Most of this book is devoted to examining in some detail theories and practices in the two great themes of computer systems: equipment (hardware) and programming (software). It is appropriate to begin, in the next section, by establishing a historical perspective.

1-2 HISTORICAL PERSPECTIVE

Mechanical aids to counting and calculating were known in antiquity. One of many ancient devices, the abacus, survives today as a simple practical tool in many parts of the world, especially the East, for business and even scientific calculations. (A form of the abacus was probably used by the ancient Egyptians, and it was known in China as early as the sixth century B.C.) In the hands of a skilled operator, the abacus can be a powerful adjunct to hand calculations. There are several forms of abacus; they all depend upon a positional notation for representing numbers and an arrangement of movable beads, or similar simple objects, to represent each digit. By moving beads, numbers are entered, added, and subtracted to produce an updated result. Multiplication and division are done by sequences of additions and subtractions.

Although the need to mechanize the arithmetic operations received most of the attention in early devices, storage of intermediate results was at least as important. Most devices, like the abacus, stored only the simple current result. Other storage was usually of the same type as used for any written material, e.g., clay tablets and later paper. As long as the speed of operations was modest and the use of storage also slow, there was little impetus to seek mechanization of the control of sequences of operations. Yet forerunners of such control did appear in somewhat different contexts, e.g., the Jacquard loom exhibited in 1801 used perforated (punched) cards to control patterns for weaving.

Charles Babbage (1792–1871) was probably the first to conceive of the essence of the general-purpose computer. Although he was very versatile, accomplished both as a mathematician and as an engineer, his lifework was his computing machines. It is worth noting that Babbage was first stimulated in this direction because of the unreliability of manual computation, *not* by its slow speed. In particular, he found several errors in certain astronomy tables. In determining the causes, he became convinced that error-free tables could be produced only by a machine that would accept a description of the computation by a human being but, once set up, would compute the tables and print them—all without human intervention. Babbage's culminating idea, which he proposed in great detail, was his Analytic Engine, which would have been the first general-purpose computer. It was not completed because he was unable to obtain sufficient financial support.

As Western industrial civilization developed, the need for mechanized computation grew. As the 1890 census approached in the United States, it became clear that if new processes were not developed, the reduction of the data from one census would not be complete before it was time for the next one. Dr. Herman Hollerith applied punched cards and simple machines for processing them in the 1890 census. Thereafter, punched-card machines gained wide acceptance in business and government.

The first third of the twentieth century saw the gradual development and use of many calculating devices. A highly significant contribution was made by the mathematician Alan Turing in 1937, when he published a clear and profound theory of the nature of a general-purpose computing scheme. His results were expressed in terms of a hypothetical "machine" of remarkable simplicity, which he indicated had all the necessary attributes of a general-purpose computer. Although Turing's machine was only a theoretical construct and was never seriously considered as economically feasible (it would be intolerably slow), it drew the attention of several talented people to the feasibility of a general-purpose computer.

World War II gave great stimulus to improvement and invention of computing devices and the technologies necessary to them. Howard Aiken and an IBM team completed the Harvard Mark I electric computer (using relay logic) in 1944. J. P. Eckert and J. W. Mauchly developed ENIAC, an electronic computer using vacuum tubes in 1946. Both these machines were developed with scientific calculations in mind. The first generation of computer technology began to be mass-produced with the appearance of the UNIVAC I in 1951. The term "first generation" is associated with the use of vacuum tube as the major component of logical circuitry, but it included a large variety of memory devices such as mercury delay lines, storage tubes, drums, and magnetic cores, to name a few.

The second generation of hardware featured the transistor (invented in 1948) in place of the vacuum tube. The solid-state transistor is far more efficient than the vacuum tube partly because it requires no energy for heating a source of electrons. Just as important, the transistor, unlike the vacuum tube, has almost unlimited life and reliability and can be manufactured at much lower cost. Second-generation equipment, which appeared about 1960, saw the widespread installation and use of general-purpose computers. The third and fourth gen-

erations of computer technology (about 1964 and 1970) mark the increasing use of integrated fabrication techniques, moving to the goal of manufacturing most of a computer in one automatic continuous process without manual intervention.

Hardware developments were roughly paralleled by progress in programming, which is, however, more difficult to document. An early important development, usually credited to Grace Hopper, is the symbolic machine language which relieves the programmer from many exceedingly tedious and error-prone tasks. Another milestone was FORTRAN (about 1955), the first widely used **high-level language,** which included many elements of algebraic notation, like indexed variables and mathematical expressions of arbitrary extent. Since FORTRAN was developed by IBM, whose machines were most numerous, FORTRAN quickly became pervasive and, after several versions, remains today a very widely used language.

Other languages were invented to satisfy the needs of different classes of computer use. Among the most important are COBOL, for business-oriented data processing; ALGOL, the first widely accepted language in the international community, particularly among mathematicians and scientists; and PL/I developed by IBM and introduced in 1965 as a single language capable of satisfying the needs of scientific, commercial, and system programming.

Along with the introduction and improvements of computer languages, there was a corresponding development of programming technology, i.e., the methods of producing the compiler and interpreter translators and other aids for the programmer. A very significant idea that has undergone intensive development is the **operating system,** which is a collection of programs responsible for monitoring and allocating all systems resources in response to user requests in a way that reflects certain efficiency objectives. By 1966 or so, almost all medium to large computers ran under an operating system. Jobs were typically submitted by users as decks of punched cards, either to the computer room or by **remote-job-entry** (RJE) terminals, i.e., card reader and printer equipment connected by telephone lines to the computer. In either case, once a job was received by the computer, the operating system made almost all the scheduling decisions. A large computer could run several hundred or even thousands of jobs per 24-hour day with only one or two professional operators in the machine room.

The 1960s saw a great intensification of the symbiosis of the computer and the telephone system (**teleprocessing**). Much of this was RJE and routine non-general-purpose use, such as airline reservation systems. Considerable success was also achieved in bringing the generality and excitement of a general-purpose computer system to individual people through the use of **timesharing** systems. Here, an appropriate operating-system program interleaves the requests of several human users who may be remotely located and communicating over telephone lines using such devices as a teletype or typewriter terminal. Because of high computer speed relative to human "think" time, a single system could comfortably service 50 to 100 (or more) users, with each having the "feel" of his own private computer. The timesharing system, by bringing people closest to the computer, seems to have very great potential for amplifying human creativity.