Ken Barbier

CP/M SOLUTIONS

PRENTICE-HALL, INC.,
Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging in Publication Data

Barbier, Ken.
CP/M solutions.

"A Spectrum Book."
Includes index.
1. CP/M (Computer operating system) I. Title.
QA76.6.B35737 1985 001.64 85-526
ISBN 0-13-188186-8
ISBN 0-13-118178-7 (pbk.)

© 1985 by Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632.
All rights reserved. No part of this book may be reproduced in any form
or by any means without permission in writing from the publisher.
A Spectrum Book. Printed in the United States of America.

This book is available at a special discount when ordered
in bulk quantities. Contact Prentice-Hall, Inc., General
Publishing Division, Special Sales, Englewood Cliffs, N.J. 07632.

10 9 8 7 6 5 4 3 2 1

Bookware® is a registered trademark of Prentice-Hall, Inc.

Editorial production/supervision by Jane Zalenski
and Rhonda K. Mirabella
Cover design © 1985 by Jeannette Jacobs
Manufacturing buyer: Frank Grieco

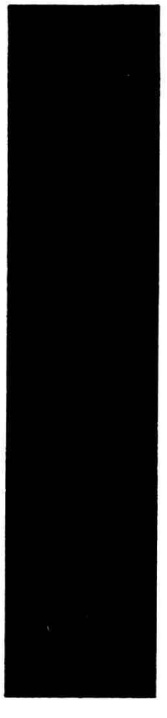
CP/M is a registered trademark, and CP/M Plus is a trademark
of Digital Research, Inc. Centronics is a registered trademark of
Centronics Data Computer Corp. Intel is a registered trademark
of Intel Corporation.

ISBN 0-13-188186-8

ISBN 0-13-188178-7 {PBK.}

Prentice-Hall International (UK) Limited, *London*
Prentice-Hall of Australia Pty. Limited, *Sydney*
Prentice-Hall Canada Inc., *Toronto*
Prentice-Hall Hispanoamericana, S.A., *Mexico*
Prentice-Hall of India Private Limited, *New Delhi*
Prentice-Hall of Japan, Inc., *Tokyo*
Prentice-Hall of Southeast Asia Pte. Ltd., *Singapore*
Whitehall Books Limited, *Wellington, New Zealand*
Editora Prentice-Hall do Brasil Ltda., *Rio de Janeiro*

CP/M SOLUTIONS



Ken Barbier has more than thirty years of experience in electronics and computers. In addition to writing numerous articles on computer hardware, software, and applications, he has been involved in the system integration of microcomputers and in the designing, constructing, and programming of real-time data acquisition and control systems.

试读结束，需要全本PDF请购买 www.ertongbook.com

Preface

Intended for the experienced user of CP/M–based computers, this book answers questions that arise when new peripheral devices are connected to an existing computer, or when new systems are first assembled. Some hardware topics are included, but the reader does not need any previous knowledge of electronics or peripheral interfacing.

The reader is expected to be an experienced assembly language programmer. If the experience is with some other computer and assembler, the confident reader can use this text to learn about the CP/M operating system and 8080 assembly language programming, although a beginner's text is a recommended prerequisite.

Programming style and practices, and some of the subroutines in this book, are identical to those previously contained in my books *CP/M Assembly Language Programming* (1983) and *CP/M Techniques* (1984), both published by Prentice-Hall, Inc. The first two chapters here, in particular, are largely a review of topics previously covered.

I encourage you to read the first two chapters, even if they are a review of lessons previously learned, because they establish the nomenclature used and basic techniques followed throughout the rest of the text. Beginning at Chapter 3, discussions target the problems encountered in interfacing computers to peripheral devices and to other computers. Both hardware and software solutions to those problems are presented in a depth and level of completeness not found in other, more theoretical, presentations.

Chapter 3 looks at the Centronics parallel printer interface and its variations and explains why even changing the cable can solve computer-to-printer incompatibilities. The next chapter expands on printer interface problems and includes a PRINT program that can solve format and timing differences. An install program for PRINT is included to permit painless modification of the program.

Part 2 is devoted to a discussion of the RS-232 serial interface, why any two devices connected by it may not work together, and how to use it to communicate with terminals, printers, other computers, and the outside world.

Part 3 discusses problems that arise during the course of implementing solutions to other problems. And in the final chapter, the reader is provided with an example and is encouraged to make use of all of the program modules in the book to solve future problems.

The solutions presented are based on many years of experience in connecting devices that may have plugs and jacks that are similar in appearance but that often don't perform as expected when first connected. Many examples from real life are discussed so that the reader can learn both canned solutions and the approach necessary to tackle interfacing problems as they occur in the future.

Problems will always crop up in the computer world, and solutions can always be found. With the examples in this book, the reader should find many solutions to existing problems, as well as guidelines that will help solve future problems as they arise.

Contents

part 1

Utility solutions

试读结束，需要全本PDF请购买 www.ertongbook.com

chapter 1

CP/M computer solutions

You've got troubles. I've got troubles. We've all got troubles. When you find the solution to a problem, pass it on.

The channels of communications for problems and their solutions are often too limited. Computer magazines want short articles to attract the reader—much shorter than some of the chapters in this book. Sometimes subjects are so confusing that an in-depth treatment is necessary. I have tried to provide that degree of detail in this book—details that will help you troubleshoot problems that are often real stumbling blocks.

Serial communications via the RS-232 "standard" is an example of a simple interface that can and has generated an incredible number of problems for the computer user. Some solutions are simple, whereas others are complicated by the differences between two connected devices. Sometimes the engineers who designed the two devices interpreted the "standard" in two different ways. As a result, too often when we hook up an "RS-232" device to the "RS-232" port on a computer, nothing intelligible gets through. Many hours or sleepless nights later a solution is found.

A problem arose, and human intellect found a solution. Pass it on.

PROBLEMS AND SOLUTIONS

That is the theme of this book—problems and solutions. Obviously, it is impossible to cover all the problems that could arise during the normal use of any computer. We have to begin somewhere, so we will start with the most common problems of the most common computers: those micros running the CP/M operating system and communicating with the outside world through RS-232 serial ports and Centronics standard parallel ports.

Each time we solve a problem, we create an experience that can be used in the solution of other problems in the future. Unfortunately, too often the solution of a computer programming problem results in source code that is not in a form suitable for use in future programming efforts.

Portable Solutions

One of the goals of this book is to show how to structure program building blocks that can be easily included as parts of future programs. Subroutines and modules that can be used in many programs are scattered throughout the examples that follow. I have used all these modules in other programs in various microcomputer systems. They have proved to be transportable.

By applying consistent register-use rules and by routing disk and I/O access through the built-in interface provided by CP/M, we can ensure that all our programs can operate on any computer running any version of CP/M. This is one of the nice things about CP/M. It provides consistent interface protocols between the operating system and application programs.

Of course, there are times when we want to write a test or diagnostic program that is hardware-specific so that we can look past CP/M and test the computer hardware directly. CP/M can provide the facilities to construct programs to do this, but these programs will include the absolute addresses of memory locations and I/O ports. We have to remember that they will not run on another computer with a different hardware configuration. These programs are sometimes necessary, but we should write them only when there is no alternative.

WHAT IS CP/M?

CP/M was the first standard operating system for microcomputers, which once were all 8-bit machines. When CP/M was introduced, memory was expensive, and 16 Kbytes was a lot. The floppy disk was new, and it made the early microcomputers practical for the general computing tasks they were really not designed for.

The first version of CP/M was tiny, to fit the existing hardware. The latest 8-bit version, bank-switched CP/M Plus, is itself larger than that original 16 Kbytes. It is the first really user-friendly version, making use of multiple 64 Kbyte banks of memory to provide features that were undreamed of a decade ago.

A Familiar Environment

In spite of their differences in size and features, all versions of CP/M provide the programmer with a familiar environment. If properly written, assembly language programs for the various 8-bit versions can be moved from computer to computer without change, and most can be easily translated into 16-bit CP/M-86 source code, to be assembled and run on the newest microcomputers.

CP/M relieves us of the necessity of knowing very much about the hardware configuration of the computer on which our programs will run. However, we still have to start by defining some hardware-related terms so that when reference is made to the LST: or AUX: we will all know what these cryptic designations mean.

In looking at some of the problems that confront the computer user—

problems with hardware, software, and their interactions—we will use techniques designed to make our programs easy to maintain and transport from version to version. To understand how this is done, you have to know a little about the organization of CP/M and a lot about how to write programs that make use of its standard interface "port," the BDOS function call through memory location 5.

In this introductory chapter, which may be a review for many of you, we will begin with a quick look at the hardware of a CP/M computer. Then we will examine the organization of CP/M and follow this with a close look at the BDOS interface and what it means to programmers.

Hardware and Terms

So that we start out with some degree of mutual understanding, let's look at a minimum configuration of a microcomputer running the CP/M operating system and see what we call the various devices that are connected to it. Figure 1–1 shows a typical small, single-user CP/M computer. It includes one CRT-type (TV tube) *console terminal* (CON:), a *line printer* as the hard-copy list device (LST:), from one to 16 *disk drives,* (A:, B:, . . . ,P:), and a *modem* connected to telephone lines to provide communication, message handling, and software sharing with the rest of the computer world.

Some people use modems to transfer programs from computer to computer, but of course *we* wouldn't do that without paying for the programs. If the modem is connected to a CP/M Plus-based computer, it is accessed through the AUX: logical device, which can both send and receive. Other versions of CP/M call the auxiliary *output* device the punch (PUN:), a term left over from the long-ago days of the paper tape punch. Its complement, the auxiliary *input* device, was the RDR:, which read data from previously punched paper tape. The tape is gone, but the nomenclature lingers on.

Logical vs. Physical

There is no physical difference between an AUX: device and the combination of PUN: and RDR: devices. CON:, LST:, and the other terms ending in ":" are *logical device* names applied to *physical devices;*, the actual computer peripheral hardware. We don't want our portable programs to worry about whether they are punching holes in paper tape or sending funny noises out over telephone lines, so CP/M provides interfaces to logical devices, and you connect whatever physical device you want at any time. Bet you never even saw any punched paper tape, did you?

Now that we have defined some terms that relate to parts of your computer, let's try to standardize a set of verbs that describe communications with those devices. It is difficult to do this because there are too many variations in common use. We will do the best we can and never try to "print" on the console screen, but there are too many synonyms for some

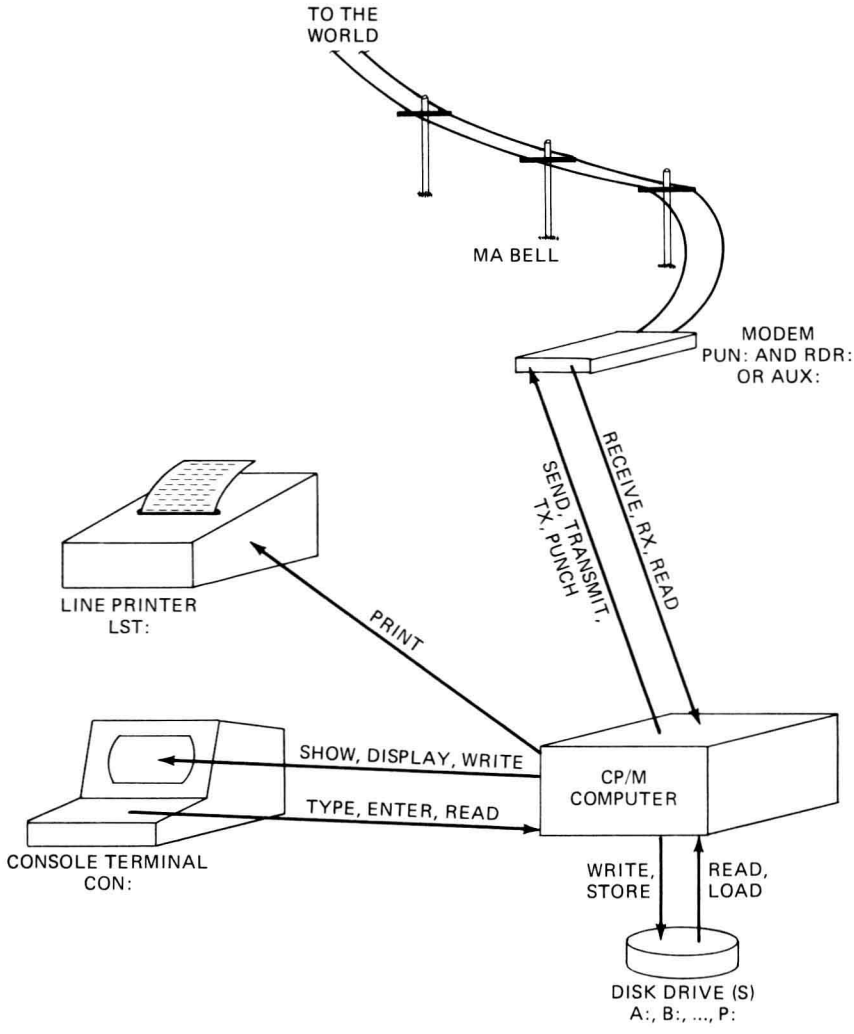


Figure 1–1. A typical CP/M computer. The common names (LINE PRINTER, etc.) and logical unit names (LST:, etc.) of the computer are defined, along with terms that refer to data flow between the units. While many other types of peripheral equipment can be connected to a CP/M computer, the examples in this book are limited to this subset of all those possible.

operations. We have included some common ones in Figure 1–1. Your computer programs can send data to your modem, transmit it (abbreviated TX), or punch it to the PUN:. But please don't punch the CON:, even if you feel like it sometimes.

ORGANIZATION OF CP/M

The internal organization of CP/M, as is true for all operating systems, has been dictated partly by the constraints imposed by the computer hardware

and partly by the logical breakdown of its functions. Some aspects of CP/M were forced upon it by the environment in which it was originally developed: the Intel MDS development system. This computer was the one that had the paper tape punch and reader. Some characteristics of its internal organization have been passed down to us through CP/M.

Dedicated RAM

The lowest 256 bytes (100 in hexadecimal) of main memory in a CP/M computer have been set aside for dedicated uses. This is partly a hand-me-down from the MDS and partly dictated by the organization of the 8080 CPU and its descendants. The functions of the first eight memory locations are examples. The first three contain a jump instruction into the warm start entry in CP/M. The next contains an 8-bit value (IOBYT) defining what physical device is attached to each logical device. The fifth location records which disk drive is the currently selected default drive. The last three of the bottom eight bytes contain a jump instruction into CP/M's BDOS, which we will discuss in detail very shortly.

Following those first eight memory locations are 56 bytes reserved for hardware interrupt vectors, as dictated by the architecture of the 8080. From just above the highest interrupt vector location and on through the first 256-byte "page" of memory, CP/M sets aside some standard usage buffer workspaces.

These buffers have been organized to make the most efficient use of the first memory page so that user application programs can all start at the same easy-to-remember location: hexadecimal 100 (100H), the start of the transient program area (TPA). These buffers also permit the Console Command Processor (CCP) of CP/M to leave data behind so that CCP itself can be overwritten by transient programs, giving the user more workspace.

Console Command Processor

This operating system (OS) functional block is the software that communicates with the computer operator. CCP inputs a command line, provides simple editing functions as the line is typed in, sets up buffers with data derived from the command line, and instructs BDOS to load and execute named programs, unless the operator requests a built-in CCP function.

The built-in functions for most versions of CP/M are DIR, ERA, REN, SAVE, TYPE, and USER. Any other command (and some of these, in CP/M Plus) input by the operator as the first word in the command line is assumed by CCP to be the name of a transient program, and CCP will tell BDOS to load and execute the named program as soon as the rest of the command line has been decoded.

The buffers set up by CCP include a copy of the command line typed by the operator so that transient programs can be aware of instructions that the operator has placed on that line following the program name. If the operator has specified the names of disk data files that the transient pro-

试读结束，需要全本PDF请购买 www.ertongbook.com