wade his way through the material and come out with sufficient understanding to complete the remainder. Truly experienced programmers wishing only to learn Pascal will almost certainly find the book too slow, too verbose, but certainly not useless. Medium level programmers and those with their own home computer who have mastered BASIC will best find the book suited for their needs.

First, a few of the faults. The coding conventions used throughout the text are abysmal and even inconsistent. They can easily be tolerated since most of the program examples are short and not complex, but they definitely should not be adopted. Occasionally a programming function or concept is used before it is introduced and explained. Also, some of the author's stylistic attitudes are highly questionable and are not to be recommended.

As for good points, the book is well written, explanations are well worded (if not always lucid) and there is an abundance of material. The organization is highly conceptual – a concept such as looping is introduced and then given a thorough and detailed examination. One problem with this approach when accompanied by the author's dual purpose is that programming constructs often get introduced painfully late. The author states that his solution to this problem is skipping around the chapters as needed. Readers unfamiliar with the text's contents and purpose may not readily be able to do this on their own. A wealth of examples accompany each concept, and each chapter contains a comprehensive list of challenging exercises. Concepts covered include different data types, loops, procedures and functions, program construction, scalar types, and advanced data structures. A set of short, useful appendices is included.

How To Build Your Own Working Microcomputer
By **Charles K. Adams**
Published by **Tab**
$9.95, 308 pages
Reviewed by **Carl E. Whitney**

This book is ill-conceived, and its audience poorly targeted. Beginners would be better off with a kit, since much of the information presented here is in a condensed, formal style. Experienced hobbyists will not be very interested in an 8080 machine with hexadecimal input and LED display. Adams should devote more thought to his potential audience and their information needs – and he should insist that his publisher not print a cover depicting a typewriter-ASCII input and video output, when the project includes neither.

There is some useful material here on the hardware and software required to program EPROM's – but periodic typographical errors make one wonder how carefully the book has been proofread, and how bug-free the programs are.

The Dartmouth Time Sharing System
By **G. M. Bull**
Published by **Halsted Press (John Wiley & Sons)**
$65.00, 240 pages
Reviewed by **Edward T. Ordman**
New England College

It is hard to understand the high price of this book. Does the publisher feel there will be enough library sales to pay for the publishing costs, and sales to individuals can be neglected? It is an excellent book, and could be very widely recommended at one-third of the price. At this cost, it can be recommended only for the libraries of schools or companies which actively use the Dartmouth Time-Sharing System (DTSS) or which are engaged in writing a time-sharing system large enough to require man-years of effort. Perhaps others can read it by interlibrary loan.

DTSS, while physically installed at about a dozen sites, is far more widely used: Dartmouth College is tied into all the major networks, and DTSS was a major formative influence in eduational computing. The commands LIST, RUN, SAVE and others common nowadays first existed (about 1964) as system commands in DTSS.

The book is a very technical, but very readable, distillation of a great many manuals and technical documents about DTSS. It is not an introductory user's manual, and does not replace the standard manuals; it is very useful reading for someone who has a modest acquaintance with the system and would like to become an expert. More important, for the person interested in internals of operating systems, it gives a great deal of detail about the parts of DTSS and how they communicate with one another. Organizations of buffers, flags, job tables, details of fault handling, communication between jobs, and techniques used in deadlock avoidance are described in detail. There is an excellently documented list of parameters used in adjusting scheduling and of the consequences of changing parameter values. There is a good explanation of swapping algorithms (determining which jobs should be in core at any moment). There is a discussion of how it looks to the operator, jobs to be performed by the operator and by the computer in starting the system, stopping the system, modifying the operating system.

DTSS is definitely a "big computer" system – at least in 1981. As the operating systems of small computers grow, more small computer users will want to know about big operating systems and the techniques they use. DTSS is conceptually simpler and more user-oriented than many large systems, and this book is full of ideas that will be valuable to someone writing a large operating system. It does not replace a textbook on operating systems, since it refers purely to DTSS and does not discuss other possible solutions to the problems faced by operating systems; but it discusses the solutions used in DTSS in far more detail than can be done in a more general text. ■■