themselves largely calls to other PROCs and there is no doubt that anyone who had enough patience to study the design would learn a lot. Like line 6210 above there are a few aberrations, including two separate definitions of the same PROC in one program, but BBC BASIC fortunately does not mind this.

This set of programs is possibly good value for money if there is a need to sketch a lot of patterns, and with a good colour monitor the coloured pictures would be quite pleasing.

W H P Leslie
Glasgow, UK

LOGO as a programming environment

Martin Lesser

'LOGO for micros' Newnes Books, Feltenham, UK (1985) £7.95 pp 186

Within the past 12 months there has been a blossoming of books dealing with LOGO. This being the case, it is now valid to ask what a writer is seeking to do when a new book is published and how it adds to the state of the literature. With *LOGO for micros* Lesser sets out to explain for adult readers 'what LOGO is and what can be done with it'.

Lesser is chairman of the Department of Mechanical Engineering and Applied Mechanics at the University of Pennsylvania, a background which hints at possible differences in this book, when compared with others. Unlike many other LOGO texts, this book is not meant to be a 'hands-on' manual, to be read by the side of the computer. Rather it sets out to be an intellectual explanation of how LOGO is structured, and how, within this structure, it operates. From this it should be clear that turtle graphics forms only a small and incidental part of the content of the book. Instead most of the example material is concerned with using the words and lists which are the building bricks of LOGO.

There is also a difference of emphasis in this book as compared with others in the way that it sets out to reveal LOGO as a programming environment, as distinct from the potent learning environment which is the burden of most other books. In other words, the book is emphasizing LOGO as a means, rather than an end.

The first six chapters of the book are concerned with familiarizing the reader with what LOGO is. Although within this section there is some use of turtle graphics, its role is really to illustrate the modularity of programming in LOGO. Most illustrations involve operations on words and lists. These progress from the simple definition of procedures and assignment of variables, the use of output commands such as PRINT and SHOW, through to more complex examples involving list dissection and generation with FPUT, LPUT, LIST SENTENCE and WORD using nested FIRSTs and BUTFIRSTs.

There is an explanation of the looping structure of REPEAT in LOGO; what I find less desirable is the equal prominence which is afforded to LABEL and GO, the LOGO

Faults in hardware designs and how to find them

Prag K Lala

'Fault tolerant and fault testable hardware design' Prentice-Hall, Englewood Cliffs, NJ, USA (December 1984) £24.95 pp xii + 263

This hardback book is intended as a reference volume for graduate engineers and computer scientists currently practising in industry, and also claims to be suitable as a final-year undergraduate text. It would be most useful to the hardware designer involved in high-reliability circuit or systems design, particularly in the computer industry, whether concerned with small or large systems. It is some 263 pages in length, which makes it a fairly manageable size, and each of the seven chapters is followed by a very comprehensive list of references, typically 40-80 per chapter, making the volume a useful index to many other related works.

Additionally, the author includes an annotated bibliography covering journals, annual conferences and classic texts in this field.

Both the layout and illustrations in the text are good, with a large number of diagrams, and the author's style of writing is clear and easily digested (except where the mathematics precludes this possibility).

After an introduction which defines the basic terms of reliability theory such as MTBF, the most commonly occurring digital circuitry faults are covered. This includes classical 'stuck-at' faults and intermittent and transient faults. A chapter is devoted to the detection of faults in combinational and sequential logic, and the difficult area of testing sequential logic is handled and some practical methods described.

Fault-tolerant and detection methods in computer circuit design are examined, including hardware and software redundancy. Several real computers embodying these techniques are described in detail. Selfchecking and fail-safe logic design and methods of designing for testability are given close scrutiny.

As a relative novice to the fairly specialized world on which this book focuses, I felt that this text has a lot to offer and is well written and presented, but its appeal will be to a select group of designers. In my opinion it meets its stated objective and would be well worth buying if one had a particular interest in this area. The author expounds upon the subject in depth, rather than superficially, and clearly knows his subject well.

Ian P Leslie
Logica, UK

equivalent of GOTO in the spaghetti programs of BASIC. When recursion is dealt with, the control conditionals of IF and TEST are explained. As in Abelson's *LOGO for the Apple II*, there is an illustration of how new conditionals such as WHILE can be created as extensions to LOGO.

Where the book does well is in its illustrative material dealing with recursion. Many texts use turtle graphics in trivial examples which, more often than not, serve to confuse rather than to explain recursion. Recursion becomes clearest when seen operating upon lists; it is only then that the 'spiralling down' and back up again can be properly recognized. It is also only through the correct use of the OUTPUT command that true recursive output can be created. It is in this area that I think the book has a particular strength.

Although the last six remaining chapters of the book are by implication concerned with developing programming applications, this is not wholly the case. There is one further chapter whose function is to reveal more of the structure of LOGO. This is the one dealing with error trapping. There is an explanation of the use of CATCH and THROW. Also buried in this chapter is a short section of property lists. I find this to be the least satisfactory chapter of the book. First, it seems a curious 'rag-bag' combination of topics. Secondly, I find the explanation of property lists far too brief and superficial. For a book which aims to explore LOGO as a programming medium, I think the very least that should have been done with property lists was to illustrate their use with an applications program. After all, they are one of the more distinctive characteristics of recent LOGO implementations.

Overall I feel *LOGO for micros* does add something new to the literature of LOGO. I think the examples which are given are sound and of particular interest to engineers and mathematicians. Where I do have reservations is the extent to which an intelligent computer-literate adult would really come to a meaningful understanding of the nature and power of LOGO by merely using this text. I am sceptical of this; I think that, to be really effective in its aims, the book alone is not enough. On this subject of experience at first hand, I am also critical of the book's structure in the way that it reflects little of the LOGO learning philosophy. Now, this brevity may seem legitimate, given that the book sets out to explore LOGO as a programming medium, but I would disagree. I see no reason why the form of presentation should not embody the problem-solving and exploratory characteristics of the LOGO learning environment. I feel that, by not doing so, the book is the poorer for it.

However, these criticisms are small when set against what the book adds to the literature of LOGO. I would recommend this book. What greater recommendation can I give, than to say that I had already bought it before I was asked to review it?

Derek Radburn
British LOGO User Group,
Milton Mowbray, UK

Linear integrated circuit design still holds many pitfalls for the unwary

J Michael McMenamin

'Linear integrated circuits: operation and applications' Prentice-Hall, Englewood Cliffs, NJ, USA (January 1985) £39.30 pp 293

It can be argued that the design of analogue circuitry in this day and age is merely a matter of selecting appropriate integrated circuits and connecting them together with the addition of a few discrete components where necessary. Obviously this somewhat simplistic point of view ignores the fact that there will be a host of specifications to be considered before device selections can be made and, in many cases, a plethora of calculations to ascertain optimum discrete component values. Successful design also demands a good understanding of the capabilities and limitations of a range of different circuit types, as there are still many pitfalls (eg unintentional positive feedback) to trap the unwary. *Linear integrated circuits* sets out to explain the necessary facts on modern analogue electronics design to the student, technician or engineer, and assumes some prior exposure to mathematics and simple electronic circuitry.

Where analogue (linear) electronic circuits are concerned, one of my favourite reference texts to date has been Horowitz and Hill's *The art of electronics*, which gives a broad coverage of both linear and digital design techniques using an excellent intuitive approach. For the most part in *Linear integrated circuits*, the field of study has been restricted to purely linear integrated circuit design. As a result, McMenamin's scope and intensity of coverage of the major building blocks in analogue circuit design is considerable. A reader armed with a reasonable knowledge of algebra and calculus should find that a substantial depth of understanding is possible that cannot really be attained using the simple intuitive nonmathematical approach. Where digital circuitry is considered, it is in areas where the two differing technologies interface, eg analogue-to-digital and digital-to-analogue conversion, and data acquisition for digital computers.

The book runs to over 400 pages. Roughly one-quarter of the space is devoted to appendices of reference material including manufacturers' data sheets for the devices featured in the circuit design examples, resistor and capacitor data, and guidelines for setting up laboratory experiments using the circuits. Enough material is included to ensure the book's usefulness both as a student primer or teaching text and as a circuit designer's reference. As mathematical derivations are an essential part of the design process, an appendix featuring common integrals and differentials is usefully