Acknowledgement

The author would like to express his thanks to his project supervisors, Mr. H. A. Kemhadjian and Dr. D. W. Thomas, for their continual support and encouragement throughout this work and also wishes to acknowledge the financial support of the Science Research Council via an SRC Studentship.

Appendix 1 Examples of threshold gate designs

1.1. Threshold = 2

No. of Inputs = 6.

Input variables are denoted by ABCDEF.

$$T = (A + B + C)(D + E + F) + (A + B + D)(C + E + F) + (A + E)(B + F)\ .$$

1.2. Threshold = 3

No. of Inputs = 12.

Input variables are denoted by ABCDEFGHJKLM.

$$\begin{aligned} T = {} & (A + B + C + D)(E + F + G + H)(J + K + L + M) \\ & + (A + B + E + F)(C + D + J + K)(G + H + L + M) \\ & + (A + B + L + M)(C + D + G + H)(E + F + J + K) \\ & + (A + C + E + G)(B + D + J + L)(F + H + K + M) \\ & + (A + C + K + M)(B + D + F + H)(E + G + J + L) \\ & + (A + B + E + J)(C + F + G + K)(D + H + L + M) \\ & + (A + E + H + L)(B + F + J + K)(C + D + G + M)\ . \end{aligned}$$

References

Amodei, J. J., Winder, R. O., Hampel, D., and Mayhew, T. R. (1967). An Integrated Threshold Gate, *Dig. Proc. 1967 International Solid-State Circuits Conference*, University of Pennsylvania.

Hurst, S. L. (1971). *Threshold Logic*, Mills and Boon.

Hurst, S. L. (1970). Universal Threshold Gates for 3-, 4- and 5-Variable Linearly Separable (Threshold) Functions, *Electronics Letters*, Vol. 6, pp. 324-326.

Book review

APL Congress 73, edited by P. Gjerløv, H. J. Helms and Johs Nielsen, 1973; 506 pages. (*North-Holland*, Dff 75.00.)

Since birth, APL has had its proponents and its opponents and, as in other areas, reactions generated in any discussion quickly become emotional to the extent of a love/hate relationship. Naturally, the book under review, which represents the proceedings of the APL Congress 1973 in Copenhagen, reports only the views and attitudes of lovers. But this reviewer was pleased to note that he is apparently not alone in developing a schizophrenic attitude to the language.

The conflict stems from the fact that, as demonstrated also by many papers in the book, APL is an extremely useful language when one wishes to get correct results in specific application areas. That is, it is a valuable tool for the researcher in almost any area and an aid whereby one may develop information processing tools that may subsequently be used effectively by the uninitiated. Pragmatically, APL has proved itself in many situations.

On the other hand, when APL is viewed from the point of view of the programming theorist or the software engineer, it may be held up as a model of what a programming language should *not* be. Richness of operators, a single machine-oriented control statement, outstanding interactive attitudes, are examples of language and system features that facilitate the creation of unintelligible, poorly structured and therefore, unmaintainable software. APL is just too good to be put into the hands of mere humans.

Perusal of *APL Conference 73* reveals that others share this uneasiness. In the first paper 'Program Writing, Rewriting and Style' for example, Phil Abrams in his references to APL pornography and to syntactic glue clearly outlines the traps one may so easily fall into in developing large APL programs. By the time it is completed the program will be unintelligible even to the author. And intelligibility of a program is perhaps its second most important attribute.

The most important property is, of course, 'correctness' and this topic is addressed in related papers by Feldbrugge and by Vervoort. But practical program verification makes demands on the programming language used, on programming style and on programming methodology. Papers in this volume discussing this issue suggest, correctly of course, that freely used APL *as such* is inadequate from this point of view and requires augmentation, particularly with user-oriented primitives for control structuring. Equally important—though the papers do not say this—is the fact that in practice, protocol limiting the use of language primitives and features would have to be adopted if confidence in the correctness of a proof is to be greater than confidence in the correctness of the program in the first place.

The proceedings contain some 67 papers, most of which are application oriented; many with exotic titles and even more exotic content. The breadth of coverage is remarkable and almost everyone should find at least one paper of interest, though papers will usually be understandable only to those who are already familiar with APL.

In general then this is an interesting book to browse through for those who know APL. It is not a book to buy. And we may look forward to even greater things from the (at the time of writing) forthcoming Second APL Conference in Pisa.

M. M. Lehman (London)

Field-effect Transistors in Integrated Circuits, by J. T. Wallmark and L. G. Carlstedt, 1974; 153 pages. (*Macmillan*, £4·50.)

This book is well written and well translated, well structured and well thought-out by authors who are truly authorities on the subject. It introduces the principles of field effect devices adequately and describes and demonstrates their use thoroughly in several relevant applications. It is also uncompromisingly dull; it has no real alternative to being so. The information with which it is packed is valuable and much of it is essential knowledge for computer hardware men if they are to be masters of the tools of their trade. But it is hard reading for anybody who is not an enthusiast about electronic devices for their own sake. It should provide a valuable library book for teaching and engineering organisations in the computer, field, since it contains the answers to many questions that computer engineers must want to know as the use of field effect devices becomes increasingly widespread. It is not a book for reading by a warm fireside with the TV on, after a hard day—but the wise hardware man should find time to read it.

B. S. Walker (Reading)