```
T4: 'SET' RESULTST,MCANDST;
    !COPY ADDER OUTPUT TO RESULT REGISTER AND
     COMPLETE SECOND HALF OF MULTIPLICAND SHIFT;
    'WAIT FOR' RESULTDN&MCANDDN;
    'SET' RESULTSET;
    ! SET LOCAL CONTROL TC INDICATE COMPLETION;
    *;
    !TERMINATE CONTROL PATH;

T5: 'SET' DECODEST;
    !START DECODING OF TRIPLE;
    'WAIT FOR' DECODEDN & RESULT SET;
    !WAIT FOR DECODING DONE AND THE RESULT TO
     BE IN THE CORRECT REGISTER;
    'SET' DECODED;
    'DECODE' DECODEOUT -> [T6,T7,T8,T9,T10];
    !SELECT MULTIPLE OF MULTIPLICAND TO ADD;

T6: 'FLOW'   ADDINB <- MCANDOUT,
             ADDINC <- 0;
    'SET' ADDST;
    !ADD 1 TIMES MULTIPLICAND TO RUNNING TOTAL;
    'WAIT FOR' ADDDN & SHIFTDN;
    -> T4;

T7: 'FLOW'   ADDINB <- MCANDOUT[0:14] * 0,
             ADDINC <- 0;
    'SET' ADDST;
    !ADD 2 TIMES MULTIPLICAND;
    'WAIT FOR' ADDDN & SHIFTDN;
    -> T4;

T8: 'FLOW'   ADDINB <- MCANDOUTI,
             ADDINC <- 1;
    'SET' ADDST;
    !ADD -1 TIMES MULTIPLICAND;
    'WAIT FOR' ADDDN & SHIFTDN;
    -> T4;

T9: 'FLOW'   ADDINB <- MCANDOUTI[0:14]*1,
             ADDINC <- 1;
    'SET' ADDST;
    !ADD -2 TIMES MULTIPLICAND;
    'WAIT FOR' ADDDN & SHIFTDN;
    -> T4;

T10: ! ADD 0 TIMES MULTIPLICAND;
     'WAIT FOR' SHIFTDN;
     -> T4;

T11: 'SET' MPLIERXST;
     !HALF SHIFT MULTIPLIER;
     'WAIT FOR' MPLIERXDN  & DECODED;
     !WAIT FOR DECODING DONE TO COMPLETE SHIFT;

T12: 'FLOW' MCANDXIN <-  MCANDOUT[2:15] *00,
            MPLIERIN <-  MPLIERXOUT[0] * MPLIERXOUT[0] *
                         MPLIERXOUT[0:14];
     'SET'  MCANDXST,MPLIERST;
     !FINISH MULTIPLIER SHIFT AND HALF SHIFT MULTIPLICAND;
     'WAIT FOR' MCANDXDN & MPLIERDN;
     'SET' SHIFTDN;
     'IF'  /'EQV' MPLIEROUT  -> D1;
     !IF NO SIGNIFICANT BITS IN MULTIPLIER EXIT;
     -> (T5,T11);

T14: 'SET' INT;
     !SET INTERRUPT - OVERFLOW;
     'WAIT FOR' INTAK  & RESULTSET;
     !WAIT FOR ACKNOWLEDGE AND LAST ADD CYCLE DONE;
     -> T1;

T15: !WAIT FOR RESULT TO APPEAR IN CORRECT REGISTER;
     'WAIT FOR' RESULTSET & OUTBUFFAV ;

T16: 'SET' OUTBUFFST;
     !LOAD DATA TO OUTPUT BUFFER;
     'WAIT FOR' OUTBUFFDN ;
     'SET' OUTBUFFGO;
     ! START THE OUTPUT CYCLE;
      -> T1;

T17: 'SET' OUTBUFFAV;
     ! THE OUTPUT BUFFER IS FREE;
     'WAIT FOR' OUTBUFFGO;
     !WAIT UNTIL THE OUTPUT CYCLE IS STARTED;

T18: 'SET' MOUTAV;
     !INDICATE THAT THE DATA IS AVAILABLE;
     'WAIT FOR' MOUTTK;
      ->  T17;

'END';
```


References

ADDYMAN, A. M. (1969). Some Aspects of the Design of a B-Arithmetic Unit, M.Sc. Thesis, University of Manchester.

BURSTON, A. K. (1975). The Development of a Computer Logic Design Language, M.Sc. Thesis, University of Manchester.

Computer, December 1974.

GIUMALE, C. (1975). *Languages for the Description and Design of Logic*, Ph.D. Thesis, University of Manchester.

IBBETT, R. N. (1972). The MU5 Instruction Pipeline, *The Computer Journal*, Vol. 15, No. 1.

Book reviews

Microcomputers/Minicomputers/Microprocessors 77, Proceedings of the ISCM conference held at Geneva; 368 pages. (*IPC*, £19)

This is a good quality bound reprint of the papers given at a commercially organised conference held in May 1977. The general theme of the conference was microprocessors, although there was the usual tendency to stray towards minicomputers.

As with any such conference, the papers are a ragbag and a reviewer's nightmare. There are only four papers of any real merit: a review of available microprocessor software by Michael Rooney, an elementary exposition of reliability by D. Popovic, an elementary exposition of multiprogramming by Dan Prener and a discussion of a possible pattern of development for GSI (the next in thing after LSI) by Dan Gajski of Illinois. If your library buys a copy, it might be worth reading that last one.

IANN M. BARRON (Redbourn)

Computing with FORTRAN: A practical course, by D. M. Monro, 1977; 242 pages. (*Edward Arnold*, £3·95 paper)

FORTRAN was pronounced dead within IBM as early as 1961 and four years later the head of a well known software house, later to become president of the BCS, was quoted as saying that FORTRAN would never catch on in Europe because it was so clearly inferior to other languages already in use. FORTRAN has survived many announcements of its imminent demise and indeed is about to enter a new phase of development. Having been essentially unchanged for a dozen years it has attracted during this time the attention of many authors and well over fifty introductory texts have been published. As the market became saturated, successive writers attempted to optimise sales by identifying in their titles potential readers whose needs were not apparently catered for; these trends were parodied in *Fortran for Humans* in 1974.

This addition to the literature is a self-teaching course which is intended to appeal to those who might, or might not, already know BASIC. The many exercises assume familiarity with polynomials, differential equations and linear algebra, revealing their origin in the electrical engineering department at Imperial College. The order of introduction of statements is somewhat unusual, with arrays and common blocks being delayed as long as possible. The author is not averse to airing his prejudices—he is for assigned GO TO's and subroutines but against computed GO TO's, arithmetic IF's and logical variables—and he displays the traditional engineer's regard for numerical precision.

There are many good passages to balance the not infrequent loose descriptions and the paucity of effort made to distinguish universally available and implementation-dependent features, but as a text in the 1960's FORTRAN tradition this book is unlikely to lead anyone seriously astray and is relatively cheap.

D. T. MUXWORTHY (Edinburgh)