R300.2

Automatic Coding.

In order to make coding easier the routine AUTOCODE has been constructed: it is an input routine that enables programmes to be written in a notation similar to the simple notations of algebra. In so doing, some of the subtleties of coding possible with the present system of direct coding are lost. This is the price that has to be paid for the desirable features of the AUTOCODE system.

In the AUTOCODE system there is a complete dichotomy between the arithmetical and organisational parts of the programme, so that the difficulties of each may be considered separately. The notation used for the arithmetical sections of the programme is an algebraic notation, whereas for the organisational section a description with a limited number of English words is used. The whole system is intended to provide an easy method of constructing programmes for individual problems where economy in the programmer's time is more important tham economy in the use of the machine's facilities. However the loss of efficiency (in the sense of the additional space taken by ~~rpu~~ routines made with AUTOCODE) is no more than about 10% . If economy in time and space is essential it is always possible to code the inner loops in the machine's code.

Part 1. The Arithmetical Description of the Programme.

1.1 Algebraic Equations.

The notation used in the description of the arithmetical sections of the programme is that of the "algebraic equations". These describe the operations of the machine using the accumulator. The left hand side of the equation represents the passage of information to the accumulator through the adder, subtractor or multiplier, while the right hand side represents a transfer of the accumulated result to the store. There is only one transfer of information from the accumulator in each algebraic equation and hence the left hand side can contain only additions and subtractions of single quantities or products of pairs, since these alone may be calculated without transferring information from the accumulator.

An example of the algebraic equation in this restricted sense is

$$+a \ +b \ +c \rightarrow d$$

representing a group of orders which form the sum of the contents of storage locations a b & c and send it to d. In this, as in all algebraic equations the first term must be signed; if the sign is omitted, the input routine will not accept the programme. Another example, with multiplication, is

$$+ab \ +cd \rightarrow e$$

It must now be remarked that such a notation is restricted, of necessity, to one system of number representation, which has been chosen to be the plus minus fractional convention, with numbers in the range $(+\frac{1}{2},-\frac{1}{2})$. A difficulty that arises is that of ensuring that the correct type of addition is used when the left hand side of the equation contains both additions and multiplications; in some contexts addition in the less significant half of the accumulator is required, in others addition into the more significant half. A convention is therefore established that any addition or subtraction that appears before multiplication in an algebraic equation uses the less significant half of the accumulator (L), while all operations after multiplication use the more significant half (M). If there is multiplication in the algebraic equation ~~th~~ the transfer specified by the right hand side is from M. (Subtractions from M may be specified ; ~~but~~ these take three orders in the resulting programme).

Thus in the equation

$+a +ab +b \longrightarrow c$

the content of a is added to L, the product ab is added and then b is added to M, the transfer to c being made from M.

It has been arranged that, whenever an addition or subtraction occurs as the first term, a direct transfer from the store to the accumulator is used so that the previous content of A is irrelevant. Where multiplication comes first there is no such provision possible but it is always possible to begin the equation with the round-off term which will be a direct transfer to L, but when sequences of algebraic equations occur with multiplication, the tranfers specified by the right hand side clears ~~the~~ M but not L, and so random round-off may be used in the usual ~~[illegible]~~ manner.

1.2. B-modification.

The foregoing description referred entirely to operations not effectively B-modified i.e. B 0 modified. Modification by other B-lines may be made by appending to the variable a suffix chosen from the letters ~~j~~, k, l, n, o, q and r corresponding to the B-lines 2 to 7 respectively. These letters may not be used to represent storage locations.(Note: since B1 is used in the R.C.S./B. system for use in changing routines, it is not normally used for other purposes. ~~A stunt character is used so that an alphabet letter may be saved for better use.~~)

1.3. Incomplete algebraic equations.

In cases where the sign of a quantity but not its magnitude is of interest, the algebraic equation may be used without its right hand side to form the ~~quati~~ quantity in the accumulator; so that its sign may be tested using the TESTA facility to be described in section 2.2. The fact that the accumulator is not left empty may lead to spurious results from the next algebraic equation. The avoidance of such effects will be described later.

1.4. Equations of definition.

The algebraic equations of previous sections must be defined in terms of the lines of the electronic store. This is done by ~~menas~~ the "equations of definition" which are written like

a@/C

for example, defining the variable a to be the number stored in the line /C . The equations of definition must prcede any feference to the variables when the programme is fed into the machine. It is possible to make routines without refernce to the the storage space ultimately used, which may be changed by alterring the equations of definition.

1.5. Operations on B-lines.

Operations on the B-lines may be made using the B-lines equation which has the following forms.

r→a , a transfer from the B-line to the line a.
a→r , a transfer from the store to the B-line.
r–a, a subtraction from the B-line.

Note that these are not preceded by a sign.

1.6. The allocation of symbols .

In the section on B-modification, mention was ~~be~~ made of the exclusive use of symbols for the B-suffix notation. The range of any translation routine such as AUTOCODE is seriously restricted by the lack of symbols in the 5 hole teleprint code, since the information is represented on a lineal medium and positional notations ~~a~~ can be represented only by the exclusive use of different symbols corresponding to the different positions, as here in the suffix symbols.

In addition to their use in the arithmetical sections, the alphabet letters have uses in the descriptive word technique where letter groups are ~~x~~ used, as opposed to the single symbols of the previous sections.

The algebraic variables are	a,b,c,d,e,f,g,h,i,j, s,t,u,v,w,x,y,z ~~and~~ : and ½
The B-suffices are	~~½~~,k,l,n,o,q and r .
The plus sign is	+ (also p).
The minus sign is	– (also m).
The transfer sign → is	" .
The defining symbol is	@

The other symbols, / and £, have special uses.
The £ sign has no effect if it occurs in the algebraic equation or in blank tape but will cause trouble if used elsewhere. It may be used to erase punching mistakes, ~~i-~~ either of single symbols in algebraic equations or whole words.
The / symbol (blank tape) is used as a warning character showing the end of descriptive words or terms in the algebraic equations. It is otherwise meaningless but should not appear in B equations, equations of definitions or descriptive words.

Thus with the single exception of the transfer sign the symbols of the written and punched programmes correspond.

Part 2. The Organisational Part.

Since the use of single symbols has been fully allocated in the previous sections, the specification of any further operations such as transfers of control must be by groups of symbols which have been made for convenience English words (or something like them). The input routine is able to detect the presence of words since they are not preceded by a sign, as are the algebraic equations, nor do they have -, ~~m~~ @ or → as their second letter, like the equations of definition of the B-lines equations. The symbols forming the English words are read and used ~~e~~ in the formation of a 20-digit number which is then compared with the numbers in a "vocabulary" of words corresponding to the words used. If through mis-spelling or other mischance no correspondance is found the input routine will cease to function. If correspondance is found the apprppriate ~~ordser~~ orders are synthesized. The facilities are described below.

To most of these English words are attached "labels". These are single teleprint chatacters that follow, at an interval of blank tape, the English words; they are used for purposes of identification, and ~~except for the words LOOP and REPEAT~~ should be chosen from the same letters as are used for algebraic variables

ENTRY
TESTA
TESTB
CONTROL

2.1. ENTRY.

This word~~i~~ is used to signify at which point in the routine control enters after a transfer of control instruction. It is labelled to identify which transfer of control is meant.

2.2 TESTA.

This word specifies a transfer of control conditional on the accumulator sign, control being transferred if the accumulator is non-negative. It is labelled to indentify it with its cotrresponding ~~e~~ ENTRY point.

2.3. TESTB.

This word specifies a transfer of control conditional on the sign of the B line last used. It is used as TESTA.

2.4. CONTROL.

Thisrhas an effedt similar to TESTA but with an unconditional transfer of control.

Note: The use of the words TESTA, TESTB ~~and~~, CONTROL and ENTRY has an important effect on the subsequent interpretation of algebraic equations, in that any algebraic equation following tham is treated as a new equation and, in particular, if the first order is addition or subtraction, ~~adirect~~ a direct transfer is used. This has the effect that the content of the accumulator is not carried across these words. The exception to this is the case of a multiplication following one of these words. If the incomplete algebraic equation preceding the word does not have a multiplication in it this will not matter much as the content of the accumulator will be in L; if ther previous algebraic equation had multiplication then if the subsequent equation commences with a multiplication term the content of the accumulator will be in M and will be carried across the word. In this last case it is possible to discard the previous contant of the accumulator by starting with a round-off term.

2.5. SUBROUTINE.

This word translates into a group of orders for calling down a subroutine on the R.C.S./B. system. The word must be labelled with the label of the routine required. Thus if the n'th subroutine is required the word SUBROUTINE is followed by the teleprint character equivalent to n.

2.6. ADROUTINE.

This is used as the word SUBROUTINE, but calls the routine down as an adroutine on the R.C.S./B. system.

2.7. CLOSE.

This translates into the order NS/P, required to close the routine.

2.8. LOOP and REPEAT.

These two words are used to specify a loop in the programme with counting in a B line (in steps of 2). They provide a notation analagous to the usual summation (Σ) and product (Π) notations of analysis. They as used as follows.

Suppose that a process (or sequence of orders) written in AUTOCODE notation is to be repeated n+1 times with counting in the B tube in the usual manner, so that the content of the B-line is initially ~~2n~~ 2n and is reduced by 2 on each traverse of the lopp, until the last loop is traversed with the B-line equal to zero, then the process is preceded by the word LOOP and ended with the word REPEAT. The word LOOP is followed, ~~a tan~~ at an interval of blank tape, by the number n in decimal form immediately followed by the suffix of the B-line used for the count; the word ~~repeat is~~ REPEAT is also followed by the suffix (with the usual interval of blank tape).

As an example of the use of the LOOP and REPEAT facilities consider the coding of a summation of the numbers stored in a column of the electronic store whose first line is defined to be a,

```
    LOOP  15r
        +a_r
   REPEAT  r
```

There is no limit ~~to~~ to the complexity of the processes inside the loop, even the the extent of inner loops, but these ~~however~~ must use different B-lines. Subroutines inside the loop are also permissible , but in all cases the corresponding LOOP and REPEAT words must be in the same routine. Another feature of the word LOOP is that it is never followed by a direct order in the synthesis of the routine, so that the accumulator contents are preserved on each traverse of the loop. The example shown above illustrates the necessity for this: as a corollary to this care must be taken that the content of the

ac

accumulator is zero at the beginning of the loop (or if not, some known number according to the programmer's intention.)

The word LOOP should always occur before the word REPEAT appropriate to it.

2.9. ERASE.

This word is used to erase from the AUTOCODE routine knowledge of the transfer of control specified by the character used as the label. This is to ensure that the number of transfers of control is not~~t~~ limited and to prevent confusion in the use of microroutines (q.v.).

2.10. FRACTIONS.

The following symbolism is used to send decimal fractions to the store.

FRACTIONS $+01 \rightarrow a$

This has the effect of sending +0.01 to the (long) line defined by a, which must be an address in S 0. or S 1. The use of the word FRACTIONS causes an ~~indefinite-sequ~~ indefinitely long sequence of numbers to be sent to the store, provided that not more than four rows of blank tape intervene between the ~~1~~ algebraic variable and the ~~n~~ sign of the next number . The number sequence is ended by leaving more than four rows of ~~balk~~ blank tape after the variable.

Any number of decimals may be punched. The maximum error is about 6.2^{-40} and the standard deviation is 2.2^{-40}. When a number is ~~pal~~ placed into S 1, the subsequent WRITE transfer to the magnetic store will be a two-page transfer. The wrong punching of the decimals will be noticed by the input routine, as will an attempt to place the numbers in stores other than S 0 or S 1.

2.11 WRITE.

This word causes the routine to be written to the magnetic store, either as a single page or ~~twoe~~page routine according to the storage space used for the orders and constants. The word is labelled and the ~~the~~ ~~information~~ routine is written so that it may be called down as a subroutine or adroutine (see 2.5 & 2.6.) by using the same label. In the directory (Tr 34 L) the user must provide, on the odd lines, the numbers of tracks that may be used for the routines to be construsted by AUTOCODE. If the label n is used, the directory line 2n + 1 contains the track used. This information is used to construct a one or two-page transfer as required and the line 2n is made ££££ so that the control enters the subroutine at the top of the page. This is then written on the directory , the routine is written up and the routine AUTOCODE is ready to receive the next routine. The equations of definition for the routine will still be effective for the new routine, but all knowledge of transfers of control will be lost, so that it is impossible to have transfers of control directly between two different pages.

~~START~~

2.12 START.

The word start causes the routine R.C.S./B? to be brought down and the routime specified by the label to be used as the master routine on the zeroeth ~~lae~~ level. If the label is not used or more than eight rows of blank tape intervene the master routine will be the routine ~~whoeed~~ whose cue is in the zeroeth line of the directory. The master routine commences with the electronic store somewhat cluttered up with the remains of AUTOCODE, but S 3,4,5 are cleared if the word previous to START was WRITE, as it would normally be. AUTOCODE contains a dummy stop $/G before the entry to R.C.S./B.

Part 3. Examples.

3.1. Example 1 : a division microroutine.
(a microroutine is a small routine that may be incorporated in a larger routine; thus this routine may be copied in a routine where division is required.)

Division using the repetition $x_{n+1} = -x_n z_n + x_n$

$z_{n+1} = -z_n^2$

where $z_0 = 1 - y$, $x_0 = x$

and x_n tends to x/y as z_n tends to 0.

```
          +y         TESTA X
         -y→y
         -x→x
                     ENTRY X
       +x +x→x
       +y +y→y
          +y         TESTA X
          w@IS       ENTRY Y
  →w -xy  +x→x
         -yy→y
          w@A:
       -w -y         TESTA Y
                     ERASE X
                     ERASE Y
```

Explanation;-

If the divisor y is negative the signs of both x and y are reversed. They are then doubled till y goes negative and forms z_0. The repetitive formulae are then used, with w used first as the round-off variable and secondly as a convergence test on z_n. The word ERASE is used to erase the information about the labels X & Y used for the transfers of control so that if another microroutine is used there will be no confusion about transfers of control.

This routine could be converted into a closed division subroutine and written to the magnetic store by adding the words CLOSE and WRITE (with a label).

Example 2. A complete programme.

Tabulate the function $f(x) = x^{\frac{1}{2}}(0.12 + 0.022x)$ for the range 0.00 (0.002) 0.20 .

The following subroutines are available,:-

Subroutine 2, $L^{\frac{1}{2}} \rightarrow L$

Subroutine 3, print L, 10 decimals with carriage return and line feed.

```
a@VA b@MA c@GA x@IC y@½C z@RC
FRACTIONS +12→a   +022→b   +002→c
               →x
               →x
LOOP  100n
                +x
SUBROUTINE 2
               →z
      +bx  +a→y
         +zy →y
             +y
SUBROUTINE 3
               →y
      +x  +c →x
REPEAT   n
ENTRY A CONTROL A WRITE 1 START 1
```

Explanati9n.

The constants requred are placed in the store using the word FRACTIONS. The routine itself operates as follows:- The variable x is made zero by twice emptying the accumulator to it. The loop is then entered, subroutine 2 is used to find the square-root of x, the argument being carried in the accumulator. The square-root is then seht to z, and the factor bx+a calculated and sent to y. The required function is then formed and sent to y and is then printed by bringing it to L with +y. After the operation of the subroutine the accumulator is cleared by sending its content to y. The loop is then repeated with x increased by 0.002 . When the loop has been traversed 101 times, the programme comes to a dynamic stop.

The routine is written as routine number 1 and is then obeyed using the word START.

Part 4. Miscellaneous.

4.1. The estimation of storage space taken by the routines.

The following table shows the number of lines taken by each of the operations of Parts 1 and 2.

(a)	algebraic equations :	1 for each +,- or $\rightarrow$ sign, except for - after multiplication for which allow 3.
(b)	B-line equation :	1.
(c)	ENTRY, TESTA, TESTB CONTROL & CLOSE :	1 for each word.
(d)	SUBROUTINE & ADROUTINE :	3
(e)	LOOP :	3
(f)	REPEAT :	22
(g)	FRACTIONS :	2 for each constant.
(h)	All other words :	0, since they are instructions for the input routine.

4.2. Tape punching.

The rules are

(a) Punch as written, except $\rightarrow$, which is punched " .

(b) Leave blank tape before and after English words, one row is sufficient except for FRACTIONS & START (q.v.) .

(c) Mistakes may be corrected immediately by punching £ either over single symbols in algebraic equations or over whole groups elsewhere. £ is not meaningless as a label.

(d) Blank tape may be left before + - and $\rightarrow$ signs in algebraic equations.

~~[illegible]~~ To be inserted at head of section 4.3.

4.3

Before the tape AUTOCODE is fed into the machine the odd lines YA to £A of the directory must contain 6 different magnetic half-cues of the form ab@/ (where ab is the track ~~[illegible]~~ address of a full track of satisfactory performance). The corresponding even lines are to be left blank. B Input must be used

AUTOCODE may be started in two ways

(i) by cue on ~~tape~~; QGA with B input

~~(ii) by hand t H = ab@/~~
~~where ab@/ is the same as~~
~~line [» A] of the DIRECTORY.~~
~~Then press [illegible]~~ Clear con

(ii) By hand, transfer with H = ab@/ which is the content of Directory line [» A]. Then clear control and switch on completion signals.

The Routine AUTOCODE and its Use.

15

4.3. Operating the machine.

A rough tape sending AUTOCODE to tracks 10,11,12,13 14 and 16 is availibible ; it also sends R.C?S./B. to 96R. To start AUTOCODE, set H = F/@/ , clear ABC and D, switch off /L and switch on.

If the tape contains no misspellings or other things inconsistent with the AUTOCODE system, then the programme tape will be taken in without hindrance till the word START has e been read, when the /G stop will operate. Since the writing transfers to the magnetics are checked in AUTOCODE a repeat of the input with the write power switched off may be used to chedk the operation of AUTOCODE. If a directory has not been provided, the routine will probably come to a stop in its endeavours to write the programmes on to track 0.

4.4. Examples of the translation produced.

Some examples are given to illustrate the action of the routine, especially on algebraic equations.

```
(a)   +a ->b            :   a T½
                            b TA
(b)   +a  +ab  +a->b    :   a T½
                            b /K
                            a /F
                            a /J
                            b /A

(c)   +ab  -b           :   b /K
                            a /F
                            DSTJ
                            b /J
                            DSTJ
```

```
(d)  +a  TESTA A              :    a T½          s
      +b → c                    s+2 /H           s+1
          ENTRY A                 [illegible]
                                s+4 ££           s+2
                                   b T½          s+3
                                   c TA          s+4

(e)  +a  ENTRY A              :    a T½          s
     ------------              s+1  ££           s+1
          TestA A               :    :           :
                               s+1  /H           s+n

(f)  LOOP 10r                 : s+2  QO          s
      +a_r                         E:/Q          s+1
     REPEAT  r                    [illegible]
                                20  //           s+2
                                   a  QC         s+3
                                   A: QG         s+4
                                   s  /T         s+5

(g)     +y                    :    y T½          s
     SUBROUTINE 3              s+1 QO            s+1
       +x  +c → y                 GS/P           s+2
                                  I//V           s+3
                                  x TC           s+4
                                  c TC           s+5
                                  y TA           s+6
```

Note that the interpretation is the same as in the equation +y +x +c → y .