NO. 1 MAY, 1964.

503 COMPUTER MANUAL

NEWSLETTER

SCIENTIFIC COMPUTING DIVISION

The purpose of this type of newsletter is to circulate recent information in a semi-official manner with minimum delay. It is not intended to bring out newsletters at regular intervals but rather as and when it is felt that they could fulfil a useful purpose. If anyone requires further information on any of the subjects raised they should contact the 503 Librarian.

IMPORTANT

Please note that whenever anyone receives provisional tapes they should replace them with the official tapes as soon as these are available; official tapes can be identified be legible headings.

----oOo----

The subjects of this newsletter are:-

(1) Notes on 503 Autocode.

(2) Notes on the use of RAP (Issue 2).

503 Autocode

The following points should always be borne in mind when using the 503 Autocode. Although the 503 Autocode is run under the control of RAP, the following restrictions apply:-

(1) The 503 Autocode Tapes 1 & 2 always occupy the same fixed area in the store.

(2) Autocode programs are always compiled from a fixed starting point.

Thus, when Tapes 1 & 2 or a compiled program are read in or when a program is compiled by the "Load & Go" method, any other programs already in the store will probably be overwritten.

However, once the compiled programs are in the store, the first-free and last-free pointers of RAP are set and other programs such as the Autocode post-mortem can be read in before the compiled program is run. If, however, at the end of the run another program were compiled, the post-mortem program would be lost and have to be read in again.

Note

As the Autocode Tapes 1 & 2 occupy a fixed area in the store it may not be possible to run them in conjunction with later versions of RAP which occupy more store space. Therefore the next (Issue 2) and future issues of the 503 Autocode Tape 1 will include a copy of RAP Issue 1 on the front, and a legible heading to this effect.

<u>RAP Issue 2</u>

The following description of how to use the IMASK facility in RAP Issue 2 is primarily for programmers wanting to allow interrupts on the basic 503 but PART I should be read by <u>all</u> programmers.

PART I

On any but the basic machine the interrupt facility is under the control of the Peripheral Control Programs (P.C.P.) which will be described in the document T.I.S. 503. 11.

On the basic machine a location referred to as "IMASK" is used to preserve a record of how the least significant eight bits of the Interrupt Permit Register were set (see 1.2.3 and 1.2.4 of the 503 Manual). Whenever RAP is read into the store the contents of IMASK are set to a standard value which, in the case of RAP (Issue 2), is zero. The RAP uses the IMASK facility whenever it transfers control to the main store program. As it is essential that interrupts be prevented whenever a RAP subroutine is called by a main store program, the overall inhibition is set during the execution of such a subroutine. Exit from the RAP subroutine resets the Permit Register in accordance with the contents of IMASK. The contents of IMASK are not affected by typing RESET (which clears the main store), or by pressing the RESET button (which clears the Permit register); they can only be set to the standard value by typing

CANCEL; IMASK.

In the case of programs not using interrupts it is recommended that

CANCEL; IMASK.

should be typed <u>before</u> the program is loaded; otherwise the contents of IMASK will remain as set by the previous programmer who may have permitted interrupts.

PART II

This need only be read by programmers wishing to use interrupts.

When the user wishes to change the setting of the Permit bit for an interrupt under his control, he must ensure that the setting of the remaining bits is preserved and that IMASK contains a record of this setting. The procedure to be adopted is best illustrated by examples:

A. To allow Tape Reader interrupt the following procedure is recommended:

	IMASK setting	Permit reg. setting
72 0	...ooooxxxx xxxx	oooo oooo
30 < + 191 >		
23 IMASK	...oooxo xx xxxx	unchanged
30 < + 64 >		
24 IMASK /	...ooox1xx xxxx	
72 256		x1xx xxxx

where x = original bit state.

B. To prevent Peripheral interrupts the following procedure is recommended:

	IMASK setting
72 0	...ooxxxx xxxx
30 < + 239 >	
23 IMASK /	...ooxxxoxxxx
72 0	

N.B. IMASK must be referred to by using the absolute value 7886 until SAP Issue 3 makes provision for mnemonic reference to it.

Whenever a routine has been entered through interrupt, the programmer usually wishes to inhibit other interrupts. If <u>all</u> other interrupts are to be prevented then, since the overall inhibition is <u>automatically</u> set on interrupt, no further action is necessary. But if another selected interrupt is required during the interrupt routine then, before returning to the main program, the Permit register must be set according to the contents of IMASK.

To ensure that the main program is re-entered <u>before</u> other interrupts are permitted, the 66 order and NOT the 72 256 order must be used to release overall inhibition, e.g.

```
67  IMASK
72  0
66 return.
```

If a different standard setting for IMASK is required, further information is available from the 503 Librarian.

NO. 2

503 COMPUTER MANUAL JULY, 1964.

NEWSLETTER

SCIENTIFIC COMPUTING DIVISION

The subjects of this Newsletter are:-

1) SAP, Mark I, Issue 2.
2) 503 Algol Issue 1.
3) Amendments to Volume 4 of the Computer Manual.

1) SAP, Mark I, Issue 2

The attached tape marked 'SAP, Mk I, issue 2' replaces that dated '1st January, 1964' and differs from it in the following respects:

(a) 3 errors have been corrected in the original 'assemble onto papertape' routines. These were:

(i) An error which caused assembly to stop before the end of the object tape.

(ii) An error which output an incorrect tape of any object program which contained a diamond bracketed reference to any label or blockname belonging to a block yet to come, or contained a diamond bracketed reference to any global identifier which had already been used in an identical diamond bracketed wholeword, in some previous block.

(iii) An error which output an incorrect tape if the program referred to a data name before its introduction.

(b) The replacement facility is now working; the modify facility will not be implemented, since EDITALL is a very effective substitute.

(c) Reduced identifiers (see the 503 Manual, 2.2.1.9, (RAP)). An identifier may be packed in reduced form by writing it as a wholeword preceded by the digit 9.

```
e.g.            30 < 9EX 1 >
        or      9here is a 2ND example
```

Note that spaces are packed, not ignored, but that every character after the 6th will be ignored.

(d) 4 standard identifiers are added to the list of those already available:

```
RAPprint
RAPlprint
RAPread
RAPsearch
```

Each of these wholewords represents a subroutine entry to RAP, and each of the subroutines will exit to the instruction immediately beneath the entry.

(i) RAPprint : the reduced identifier in the accumulator is displayed, on the same line.

(ii) RAPlprint : the reduced identifier in the accumulator is displayed, on a new line;

e.g. the instructions

```
30 < 9EX 1 >
RAPlprint
```

will cause

```
EX 1
```

to be displayed.

(iii) RAPread : an integer or identifier is read from the typewriter, placed in location 7914, with the terminator in location 7913 and the overflow register is set if an identifier was read.
Identifiers are packed in reduced form.

(iv) RAPsearch : the reduced identifier in the accumulator specifies a program which is to be searched for in the store. On exit the accumulator holds the address of the program head if the program has been found, but otherwise the content is zero.

(e) The halt code $\dot{\underset{\cdot}{H}}$ now has the same effect as ampersand & ; i.e. whenever the character is read a temporary pause occurs except when it is part of a comment or an alphanumeric group.

2) <u>503 Algol Issue 1</u>

<u>Known errors</u>

tape 1. a) Leading '+' sign in complex arithmetic expression may be rejected
E.g. a:=+(b)+(c); will give spurious error message.

b) The check function checkb is not accepted: it must be checkB.

tape 2. a) When a program overwrites the system, the message 'Reinput tapes 1 and 2' is displayed instead of 'Reinput tape 2' and vice versa.

b) Using checkB, 'false' is displayed instead of 'true' and vice versa.

<u>Fame</u> Because the entry "FAME" sets up the system for output to papertape, and then suppresses output, if a very large program is tested in this way you will get 'error 49' when the length of compiler list plus dictionary exceeds 250 locations.

These errors will be corrected in the next issue of 503 Algol.

3) The following errors have been found in the specified sections of the 503 Computer Manual:-

Bull Section 4.8.2.

FIG.	BOARD	CHANGE
6	RD15	BRPvS to BRP+S
7	RD17	Rx 1/P of $\overline{\text{ACT 72}}$ $\ast$ 11
8	RD19	remove arrow for p O/P of C2
10	RD567 & 8	add 7.3 µS to all monostables
11	RD10	delay A5/A3 to 110 µS and 11 µS above
	RC1	NX234 should read NX235
12	PC16	$\overline{\text{APTP}}$ to APTP and add $\ast$ RD2/1/W $\ast$ PC18/1/C $\ast$ PC17/1/C $\ast$ I/C
24(a)	RB5	D5 O/P from f to p
40(2)		C1 not electrolytic
26(a)	RB4	D4 and D5 O/P f
29	RC16	Bistable C5/C5 input 't' on right hand element, should read 'b'.

Main Store Section 4.5.1.

Fig. 41

Change reference no. MR12a to MR13a
Change reference no. MR13a to MR12a.

No.3 AUGUST, 1964.

503 COMPUTER MANUAL

NEWSLETTER

SCIENTIFIC COMPUTING DIVISION.

The subjects of this newsletter are:-

(1) SAP replacement statements

When determining the effect of a replacement which itself contains a replacement, the text should be expanded one'layer' at a time, applying the appropriate set of parameters to it.

Thus

```
replace  HOP [A,B; SUB [B]
                   40 A] , SUB [X; 05X];
```

with a call of

```
         HOP [error, < + 1 >]
```

becomes

```
         SUB [< + 1 >]
         40 error
```

and then

```
         05 < + 1 >
         40 error
```

This rule excludes such a case as

```
replace  HOP [A,B;    ], SKIP [;HOP]
```

with a call

```
         SKIP [error, < + 1 >]
```

since SKIP has been given actual but not formal parameters.

(2) SAP references to unintroduced data items

It is sometimes useful to be able to refer to an unintroduced data item by means of a compound address, as in, for example

```
data        A,B,C(15);
                .
                .
                .
22A/
26 C-1
                .
                .
                .
```

where the location C-1 falls outside the range of data space which has been reserved by the data introduction (see Technical Manual 2.1.2.6.1). References like this are assembled correctly into store by both issues of SAP, but result in incorrect assembly on to paper tape.

The following text is always assembled correctly, however, since location C-1 now falls inside the allocated data area:

```
data        A,C(15),B;
                .
                .
                .
22A/
26 C-1
                .
                .
                .
```

Future versions of SAP will be programmed to reject examples of the first type if outputting on to tape. The current issue of EDITALL does not assemble correctly on to paper tape because of an array reference of this kind; a new mnemonic tape will be issued.

(3) Errors in SAP Mk.I, Issue 2

(a) An error exists in the 'reduced-identifier' facility (see Newsletter No.2 June): spaces appearing in wholewords which begin with the character 9 are stored as zeros.

(b) Issue 2 will not accept a label at the side of an instruction in a replacement text if that label has been specified as an actual parameter, and if it has not yet been referred to in any previous instruction in the object program.

For example,

```
replace   mult {A,B; A) 04 <+B>
                        22 count
                        41 A};
                 .
                 .
          mult [label 1,5]
```

causes the incorrect indication

```
          ERR i     A
```

to be displayed, unless label 1 has been used in some instruction previous to

```
          label 1)  04 <+ 5>
```

These errors will be corrected in the next issue.

(4) Error in CBSTST

An error has been found in the core backing store test program tape (c.f. Technical Manual 2.2.4.2.)

The location x displayed in the Phase 2 error indications is, due to a program error, displayed as x + 1024, i.e. 1024 greater than the location which actually failed. This will be corrected in Issue 2 of CBSTST.

(5) Errors in ALGOL Issue 1

(a) In the standard procedures arccos and arcsin there is an error in that

as $x \rightarrow 1, \mathrm{arcsin}(x) \rightarrow \frac{\pi}{2}, \mathrm{arccos}(x) \rightarrow 0$ (which is correct)

but at $x = 1, \mathrm{arcsin}(x) = \frac{\pi}{2}, \mathrm{arccos}(x) = \pi$ (which is an error).

(b) A call of the standard procedure "storemax" causes the program to go into a loop. For the time being, therefore, this procedure should not be used.

(c) If an identifier begins with A,B or C, then its first seven characters are significant. Thus, A12345 and A123456 are treated as different identifiers.

(d) If a real number contains a subscript ten and the exponent is equal to the number of digits after the decimal point, then a spurious message "Error No.1" occurs during translation. Thus, $1.7_{10}1$ and $1.276_{10}3$ are not accepted, but $1.27_{10}1$ and $1.716_{10}5$ are accepted.

(e) The incorrect assignment statements A:= B(x) or A:= B[i], where A is an arithmetic variable and B is a Boolean procedure or Boolean array, are accepted by the translator instead of being rejected as a case of Error No.31.

These errors will all be corrected in due course.

(6) ALGOL and the RAP list

When the ALGOL system tapes are read into the 503, the following typewriter output occurs:-

(a)	Tape 1	IN. ALGOL1
	followed by	Swait
	Tape 2	ALGOL 2
		Swait
(b)	Tape 2 alone	IN. OWNIN ALGOL2
		Swait

However, if the RAP command LIST. is typed when ALGOL is in store, the lists are as follows (using RAP Issue 2 - with Issue 1 the Freestore size is 15 locations greater).

Tape 1 only	Tapes 1 and 2	Tape 2 only
Swait	DRPRE*	DRPRE*
REPEAT	DUMP3	DUMP3
PRE3	DUMP2	DUMP2
PRE2	DUMP1	DUMP1
FAME	Dwait	Dwait
OWNOUT	PUBLIC	PUBLIC
REIN2	ALGOL2	ALGOL2
ALGOL	Swait	Swait
ALGOL1	REPEAT	REPEAT
2776	PRE3	OWNIN
	PRE2	5974
	FAME	
	OWNOUT	
	REIN2	
	ALGOL	
	ALGOL1	
	1194	

The asterisk appears against DRPRE if the system has been used before LIST. is typed. It is not a sign of error.

If a program has been read into store before the ALGOL system, its name appears at the end of the list, and the freestore size is correspondingly less.

Some of the extra names in these lists correspond to non-standard functions of the system (see 2.1.3.5.4.2) and may be typed by the operator when required. The other names belong to headers which are used to provide communication between the two parts of the ALGOL system. Their names must NEVER be typed.

Cancelling ALGOL

The ALGOL system may be removed from the store in the following way:-

Tape 1 only in store Tapes 1 and 2 in store	CANCEL; ALGOL1
Tape 2 only in store	CANCEL; OWNIN

It is not permitted to qualify the CANCEL instruction by any other of the names in the ALGOL lists. However, if a program X has been input before the ALGOL tapes the message CANCEL; X. can be used to delete X and the ALGOL system at the same time.

(7) Autocode Film Functions on the 503

At present the 503 Autocode cannot be used to run programs that use magnetic film. If the word FILM occurs in a program, the translator treats it as an error.

However, now that magnetic films have been added to the range of peripheral equipment available on the 503, the Autocode is being modified, and the next issue will accept FILM instructions written in the same way as for the 803 Autocode.

FILM instructions can be used in an 803 Autocode program that is being run under the control of the 803 Operator.

(8) Form in which SAC programs are stored

(see Technical Manual 2.1.2.6.1)

SAC programs translated by SAP Mk.I are stored as follows:-

At the low addressed end of the freestore, and progressing up to the high addressed end:

(i) A 5 word program head (see RAP 2.2.1). The left hand instruction of the 2nd word in this head is an absolute jump to the 1st word of the stored program's trigger block.

(ii) A single word, which is used as the program's common-program-link (LINKCP).

(iii) Then follows the program proper, stored in exactly the form and with exactly the layout specified by the programmer; but note

(a) The first instruction of each block is put in a fresh location; it never shares a location with the last instruction of the previous block.

(b) After each block there may be stored a number of diamond bracket constants, (these are constants which have been used in instructions of the form "FN < ... > "). Each diamond bracket constant used in a program is stored at the end of one of the program blocks; if the same diamond bracket constant appears in more than one block, it is stored only at the end of the block in which it is first used, unless the constant quotes an unallocated identifier, when it is stored only at the end of the block in which this identifier becomes allocated.

(c) STOP, OSTOP, ZSTOP & NSTOP are single instructions (jumps), and are treated like any other instruction; COMP,EXITCP,SUBR,EXIT are all wholeword instruction pairs -

```
COMP,X,n              73 LINKCP of X : 40 nth entry to X
EXITCP,n                 00 LINKCP /40 n
SUBR, A*B             73 LINK*B: 40 A*B
EXIT, n                  00 LINK /40  n
```

(iv) 4 + n words of trigger block, which are

```
     30   RAPsep  :  05 <+31>
     42   p       :
     Entry to RAPread
     02   0       :  17  RAPword
  p) 42   E1      :  05  RAPword
     42   E2      :  05  RAPword
        .
        .
        .
     42   En      :  jump to RAP to output 'NOPROG'
```

where E1,..., En are the trigger points.

The above trigger block is stored at the end of each program assembled by SAP Mk.I Issue 2. Issue 1 stores

```
42 p : 51 1
entry to RAPread if accumulator zero
```

as the 2nd and 3rd words of this block.

At the high addressed end of the freestore, and progressing down to the low addressed end:

(i) A single location for each block name read, which serves as the LINK location for the block. These LINKS are allocated down the store, in the order in which their associated blocks appear in the block introduction.

(ii) A single location for each data item introduced, in the order they appear in the introduction lists.No data location reserved in this way is ever associated with any other data name used in the program, or with one of the data names used in any other of the programs in the store.

(9) Calling Up a SAP Common Program from within an ALGOL program

If a SAP common program has been read into store before the ALGOL system, then it can be used as a subroutine by an ALGOL program.

One method of entering the common program is as follows:-

1. Include in the declarations of the program the declaration of the procedure enter CP (given below).

2. To achieve the same effect as the portion of SAP code:

```
COMP, CPname, n
      param 1
      param 2
         .
         .
         .
      param k
```

write in the ALGOL program

```
enterCP (CPinteger, n, NOPROG);
elliott (param 1 )
elliott (param 2 )
         .
         .
elliott (param k)
```

the parameters being in pseudo-order form.

CPinteger is the integer representation of CPname (note that only the first six characters of the name are significant and that the name is left justified so that the binary representation contains 36 bits). NOPROG is a label to which a jump is made if the program CPname is not in store.

```
procedure enter CP(CPinteger,n,NOPROG);
value CPinteger,n;integer CPinteger,n; label NOPROG;
begin switch S:=noprog;
             elliott(0,0,0,0,3,0,CPinteger);
             elliott(7,3,7932,0,4,4,8082); comment RAPsearch;
             elliott(4,2,noprog,0,2,0,CPinteger); comment get address of head;
             elliott(7,3,4,1,2,7,8189);
             elliott(2,0,4,1,7,7,8191); comment get procedure link;
             elliott(0,0,CPinteger,1,2,0,5); comment plant in LINK*CP;
             elliott(3,0,n,0,2,0,7914);
             elliott(0,0,CPinteger,1,3,0,1);
             elliott(5,1,20,0,2,0, CPinteger);
             elliott(0,0,CPinteger,1,0,0,3); comment jump into trigger block of
                                                                       CPname;
noprog:goto NOPROG
end enterCP;
```

NOTES

(1) enterCP cannot be used if CPname expects to find its data in the accumulator. However, CPname can be modified to pick up this data from a workspace location whose address is given as the first parameter word. The call of CPname then takes the form

```
plant data in A;
      enterCP (CPinteger, n, NOPROG);
      elliott (0,0,0,0,0,0,A);
                    .
                    .
                    .
```

(2) If NOPROG labels the statement "elliott (4,4,8064,0,0,0,0", then the RAP message "NOPROG" is output if CPname is not in store.

(3) If CPname sets the overflow register, then the statement "elliott (7,3,4,1,4,3,1)" must be written immediately after the last parameter of CPname, so that overflow is cleared as soon as CPname is exited. Otherwise a spurious "Int Oflo" message occurs.

(4) CPname must be a proper SAP common program. That is to say, it must expect to find its link in LINKCP and exit must be made by an instruction EXITCP,n. CPname will then be stored in the manner described in Section 8 of this Newsletter.

(5) There is not sufficient room in the store for SAP and ALGOL together. Therefore, CPname must be read in as a binary tape by the RAP instruction IN.

(6) More than one common program can be used by the same ALGOL program. All the common programs that are required must be read in <u>before</u> the ALGOL systems tapes.

Example

This example uses the library program PTSREAD as a common program. Its function is to read numbers from the typewriter, alternately real and integer, and to print each one out after it is read. If PTSREAD is not in store, the message "Try Again" is displayed.

When the program is entered, it uses the RAPread subroutine to obtain the integer equivalent of "PTSREAD" and assigns this to the integer variable PTSREAD. This is just a way of making the computer do the work of converting the alphanumeric into an integer.

Since both RAPread and PTSREAD can set the overflow, the "clear overflow" instruction appears after each entry.

```
Test enter CP;
begin integer PTSREAD,i; real A; switch S:=error,L;

comment insert enter CP here;
            elliott(7,3,7932,0,4,0,8020);
            elliott(7,3,4,1,4,3,1);
            elliott(3,0,7914,0,2,0,PTSREAD);
            comment enter RAP to read the name PTSREAD from the typewriter,and
                    assign its integer representation to the variable
                    PTSREAD;
                punch(3);
       L:   enterCP(PTSREAD,1,error);
            elliott(0,0,0,0,0,0,4096);
            elliott(7,3,4,1,4,3,1);
            elliott(2,0,A,0,0,0,0); comment read real;
            print A;
            enterCP(PTSREAD,1,error);
            elliott(2,0,0,0,0,0,4096);
            elliott(7,3,4,1,4,3,1);
            elliott(2,0,i,0,0,0,0); comment read integer;
            print i;
            goto L; error:print ££l?Try again

end;
```

503 Librarian.

No.4 September, 1964.

503 COMPUTER MANUAL

NEWSLETTER

SCIENTIFIC COMPUTING DIVISION

1. **SAPTEST Issue 2**

A tape of SAPTEST Issue 2 is attached. The operating instructions for this (which are the same as for SAPTEST Issue 1) are in Appendix 2 of 2.1.2 in the Technical Manual.

2. **Replacement Identifiers in SAPTEST**

The value of an identifier which has been replaced may be output, but the identifier in the replacement text and not the original replacement identifier must be specified on the control tape.

e.g. In order to output the value of the replacement identifier "number", and when at label "LX" "number" has been replaced by "one", part (v) of the control tape should read:

```
LX  *  <blockname> ; one ; I.
    .
    .
    .
end
 %
```

(where I specifies the format in which the value of the data variable will be output; see Appendix 2 of SAP Issue 2).

3. **Volume Binder 4D of the 503 Computer Manual**

Extract and destroy from Section 4.4.2 the Supplement page 117. This being replaced by Issue 2 of 4.4.2 recently issued by Amendment 4/0016.

503 Librarian.

503 COMPUTER MANUAL

PROGRAM NEWSLETTER

SCIENTIFIC COMPUTING DIVISION

In future, two forms of Newsletters will be distributed, one for Volumes 1, 2 and 3 of the Manual and one for Volume 4.

The Newsletters for Volumes 1, 2 and 3 will be known as Programming Newsletters and numbered P5, P6 etc. The four Newsletters already distributed should be regarded as Programming Newsletters; hence this Newsletter is numbered P5.

The Newsletters for Volume 4 will be known as Engineering Newsletters and numbered E1, E2, etc.

The subjects of this Newsletter are:-

(1) Error in SAP Mk.I issue 2.

(2) Error in TIS.503.6 (Conversion of 803 Programs to Work with the 503 Systems Programs).

(3) Error in the telephone number of the paper tape suppliers, Volume 3.1.1.

1) Error in SAP MK.I

Both issues of SAP mishandle forward references which occur in the following circumstances:-

The object program is being output to paper tape, and the forward reference occurs as an address in the last instruction pair of a block in which no new diamond bracketed constants are used; the reference is to a point in the block which is about to be input from the object tape.

In these circumstances, no forward reference to this same point will be assembled correctly.

The error will be corrected in issue 3.

2) Error in T.I.S.503.6

Page 5 section (A)(a)(iii) of TIS.503.6 (Conversion of 803 Programs to Work With the 503 Systems Programs) states that a TI code block which has been converted to SAC may be entered at any of its 2nd half instructions by means of the pair

73 X : 44 X + n

where X is the name of the block.

This is incorrect, of course, because the assembler treats blocknames as if they were 1st half labels, and so converts 44 X + n to 40 X + n.

If a block contains 2nd half entry points, at least one of the entry points must be introduced as a global label: this means introducing it at the head of the program and then by the side of the appropriate instruction.

3) Volume 3.1.1. Paper Tape Suppliers

Please note that the telephone number of Waterlow Automation Services Ltd., given on page 1 of the section should read

DUNSTABLE 63196

FEBRUARY, 1965.

503 COMPUTER MANUAL
PROGRAMMING NEWSLETTER
SCIENTIFIC COMPUTING DIVISION

The subjects of this Newsletter are:-

(1) Errors in 503 ALGOL Mark I.

(2) Additions to the operating instructions for 503 Autocode.

(3) Programmer's error in SAP Mark I.

(4) Errors in SAP Mark I, Issue 2.

(5) Printing errors in 503 Manual 1.1.3., 1.2.4., 1.4.3., 2.2.4.2., 2.2.3.24.

1. <u>Errors in 503 ALGOL Mark I</u>

Where practicable the following errors will be corrected in the next issue.

(a) Instring does not stop if it finds a Ⓗ before the leading £.

(b) The conversion of real numbers greater in magnitude than 2^{37} to type integer is liable to give integer overflow.

(c) The occurrence of identifiers of type 'string', 'label' or 'switch' in arithmetic expressions is not always detected as an error.

(d) The assignment of a Boolean procedure to an arithmetic variable is not detected as an error.

(e) A unary plus sign immediately followed by an open bracket causes a spurious error indication. (Unary minus signs are treated correctly).

e.g. A:= +(B - C); will not be accepted.

(f) arcsin (1) is wrongly evaluated as $-\frac{\pi}{2}$
and consequently arccos (1), which is obtained from the relationship
$\arccos(x) = \frac{\pi}{2} - \arcsin(x)$, is wrongly evaluated as π. The correct values are 0 and $\frac{\pi}{2}$ respectively.

(g) An elliott instruction must not contain a jump out of the block in which it occurs. This error is not detected at present.

(h) In an array declaration, an array bound which is an integer constant must lie in the range $-8191 \leqslant \text{suffix} \leqslant +8191$.
Failure to observe this restriction will lead to a corrupt program, and will usually cause a spurious subscript overflow message to be displayed. The restriction can be evaded by writing the suffix as a floating-point number.

(i) An assignment statement of the form

n:= A [------]

where (i) the first element of the right hand side expression is the identifier of an array, A, of the same type as the left hand side and

(ii) the last operand (i.e. identifier or constant) occurring in the list of subscripts of A is the identifier, F, of a type procedure or name parameter,

is compiled as if it were

n:= F........

and, moreover, each time the statement is obeyed F is evaluated twice.
The fault can be avoided by writing

n:= +A [........]

which is compiled correctly.

(j) If in a block head an illegal array declaration of the type

<u>begin</u> <u>integer</u> A; <u>array</u> B [1 : A];

is made, spurious error indications may be displayed.

(k) When making a syntactic check (i.e. using FAME without actually assembling the program), error 49 (NO ROOM) may be displayed spuriously.

(l) When using checkB or checkb 'true' is output for 'false' and 'false' is output for 'true'.

2. Additions to operating instructions for 503 Autocode (503 Manual, 2.1.4)

(a) On page 31, paragraph 23.2. the 'Result' column of (b) Stage 2 should read:
"As for Stage 3 of 23.1. except that translation is to paper tape and, when translation is complete, the machine cycles in a keyboard loop."

The 'Result' column of (b) Stage 4 should read:
"The tape is read in at speed and sum-checked. ERRSUM is displayed if sum-check fails. The tape can be rechecked by placing the leading end in the tape reader and typing

2204; S.

If the check still fails, then repeat from (b) Stage 2. Otherwise END is output."

(b) At the end of paragraph 3(a)(i) of Appendix 4, insert:
"Note: The subroutines must be punched on 5 hole tape."

3. Possible programmer's error when using SAPTEST and SAP Mark I

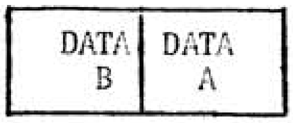
There has been some difficulty with programs which appear to run correctly when translated by SAPTEST but which are obviously wrong when translated by SAP. This is not a fault in the compiler but may well originate from an error on the programmer's part, as follows:-

If reference is made to an element of an array outside the bounds given in the data declaration, it may not be apparent from the results when the program is translated by SAPTEST. This is because with SAPTEST the dictionaries are not overwritten by data but are preserved in between blocks of data:

DICTIONARY B	DATA B	DICTIONARY A	DATA A

Thus, after a declaration data B(20); reference to B(21) causes a number to be picked up from DICTIONARY A; this may be quite a reasonable number, it will not alter during the run and unless the results are checked carefully, may not lead the programmer to suspect his error.

However, when SAP is used to translate, the data blocks overwrite the dictionaries and are adjacent in the store:



Reference to B(21) would now cause the content of A(1) to be picked up, which may change quite frequently during the run. The results would obviously be wrong and thus indicate a programming error.

4. Errors in SAP Mark I, Issue 2

The following errors have been found in SAP Mark I, Issue 2, and where practicable will be corrected in the next issue.

(a) The size of a data vector cannot be determined by means of a replacement statement. Thus,

```
replace A [; 100];
data  B(A);
```

is not equivalent to data B(100);

(b) Replacement statements cannot refer to elements in begin or data introductions.

Thus,

```
replace  A [; B];
begin  A;
```

is not equivalent to writing

```
begin B;
```

(c) If, when the object program is being assembled onto paper tape, there are less locations between the initial value of the FF pointer and 8192 than there are words in the program, the trigger block will be output incorrectly.

(d) SAP will not assemble a program correctly if both the following conditions occur:

(i) A direct forward reference is made to the block name of a block which has not been read in yet.

(ii) All the references in a later block (i.e.a block after the direct forward reference) to the block name of the block not yet read in are indirect.

Thus, in example 1 below the 40 C instruction would be assembled incorrectly, but all the instructions in examples 2 and 3 would be correct.

```
     Ex.1                 Ex.2                  Ex.3
begin A;             begin A;              begin A;
     40 C                      40 C                  40 C
end ;                end;                  end;
begin B;             begin B;              begin B;
     30 <+C >                  30 <+C >              40 C
                               40 C                  30 <+C>
end;                 end;                  end;
begin C;             begin C;              begin.C;

end;                 end;                  end;
```

(e) A particular distribution of identifiers and negative diamond bracket constants within a block, may cause SAP to stop assembling onto paper tape. This can be cured by adopting the following procedure:-
Mark the point at which the tape stops - this will always occur at the end of a SAC block. On a print-up of this block mark

every instruction which uses a particular negative diamond bracket constant for the first time in the program. Label each of these instructions; these labels are not necessary for the structure of the program but will allow SAP to make a correct assembly.

(f) If SAP finds that the block about to be assembled has been given the same name as a block already assembled, it will skip characters until it finds the underlined word <u>end</u>, and further until it finds a semicolon. However, it will mistakenly accept <u>end</u> and a semicolon if they occur as part of a comment.

5. <u>Printing errors in 503 Manual</u>

(a) 1.1.3 On the diagram showing layout of a computer room, the scale should be given as ¼ in. to 1 ft.

(b) 1.2.4 On page 1 (Issue 3), last sentence of paragraph (e) should read

"Digit 39 of this location is made zero."

(c) 1.4.3 On page 2 (Issue 3): line 16 should say that the control loop is searched for a punching in "Channel 1" <u>not</u> "Channel 8"; on line 22 "+3" should be deleted.

(d) 2.2.4.2 On page 3 (Issue 2), paragraph (c) should read:

"(c) Function Test Error. If an error occurs in the function test then

ERROR FUNCTION TEST N

is displayed, where N is an integer denoting the number of the individual test within the function test that failed.

(e) 2.2.3.24. On page 3, Note 1: U and V should be shown as the underlined characters <u>U</u> and <u>V</u>.

JULY, 1965

503 COMPUTER MANUAL

PROGRAMMING NEWSLETTER

SCIENTIFIC COMPUTING DIVISION

The subjects covered in this newsletter are:-

1. ERRORS IN SAP MARK I, ISSUE 2

The following errors have been found in SAP Mark I, Issue 2. They will be corrected, together with several other errors mentioned in previous newsletters, in the next issue (3) of SAP to be distributed shortly.

(a) Programs assembled by SAP 1, Issue 2 will not run correctly if <u>both</u> the following conditions occur:-

(i) the assembled program takes up more than half the store, even without its data space

and (ii) there is an identifier in the program which is referred to while still unallocated, the first such reference being made at some point where the size of program has already exceeded 4096 words.

The effect of this is that a 'random' number of instructions in 'random' positions of the store have their address part altered so that it becomes the address of the above identifier.

(b) In SAP 1, Issue 2 replacement introductions do not affect identifiers quoted in SUBR and COMP instructions; this facility will be included in issue 3.

2. ADDITIONS TO SAP 1 DESCRIPTION, 2.1.2.

(a) Insert at the end of section 4.3. the sentence:-

"N.B. A comment may extend over more than one line."

(b) The last paragraph of section 2.7. is ambiguous. It should read:

"The contents of a block's local data locations are not overwritten or destroyed on exit from the block since local data space is not shared between blocks. For example, if an identifier A is declared locally in two different blocks, two separate locations are reserved."

3. 503 ALGOL MARK 1

(a) Note that the standard subscripted variable 'location' may not be used as the actual parameter of a procedure where the formal parameter is called by name.

e.g.

```
procedure  increment (a); real a ;
           a : = a + 1 ;
increment  ( location [ i ] ) ;
```

is not allowed.

(b) If the same label is used twice in the same block, the error is not detected; instead, the first label is treated as the legitimate one and the second one ignored.

4. ALTERATIONS TO THE DESCRIPTION OF PCP, 2.5.3.

(a) On page 22, the instruction 'delete' should be inserted as the next to the last line of the example:

```
           consider [ < + Mine > ]
           delete
       end example ;
```

(b) On page 29, the information on errors is held in the auxiliary register *not* as

00 Device Number : 00 Error Type No.

but as

39	28 ---- 25	24 --- 21		3 --- 1
0 ------- 0	Device Class	Device No.	0 ---------- 0	Type

(c) On page 41 the mode (M) parameter for the card reader/punch should be changed to read:

M = 1	Read and Punch on both Main and Sub. tracks
M = 2	Read only
M = 3	Punch only
M = 4	Read and Punch

These modes are now compatible with those specified in IMP (2.5.4.).

5. <u>ALTERATIONS TO DESCRIPTION OF MACHINE CODE, 2.1.1.1</u>.

(a) On page 7, the result of dividing 0/0 is <u>not</u> -2^{-38} as shown but is zero. Therefore the fifth line of the table should read:

0/0 0 set.

This should also be altered on page 12 to Divide 56 0/0 = 0.

(b) Page 9, Group 6 functions : if a B-digit follows the instruction

65 4096 i.e. standardize,

the result of the modified instruction is unspecified. Normal modification does not take place. (This note should also be inserted on page 5 of 1.2.1.).

6. <u>ERROR IN 503 AUTOCODE DESCRIPTION, 2.1.4</u>.

Appendix 4, page 2, 3.(a)(iii) - the typed message on the first line should read

AUTO3; 1

and <u>not</u> AUTO3.1. as shown.

7. ADDITIONS TO FUNCTIONAL SPECIFICATION, 1.1.2. - INTERFERENCE AND SHOCK

The second paragraph of 1.1.2., page 3, should be deleted and replaced by the following paragraphs:-

"Details of earthing requirements are given on Figure 2 of Section 1.1.3.

Interference filters are fitted and are sufficient to remove the high-frequency noise on the power supply in all except the most arduous industrial environments. If the levels of interference, either on the power supply or radiated, conform to British Standards, Specification Number 800, 1954 Edition, computer operation will not be impaired. Above these levels, the effect will depend on the type of interference and the layout of the installation. It is unlikely that noise on the power supply of less than 1 V peak or a radiated interference of less than 1 mV/m will result in malfunctioning. Installations are known to have operated correctly in appreciably worse conditions.

The standard system will withstand an interruption in the power supply of up to 5 ms i.e. one quarter cycle. If the supplies are still outside tolerance at the end of this period, a controlled shut-down as detailed below will be automatically initiated."

The following paragraph should be added at the end of section 1.1.2.

"Vibration and Shock.

It is expected that the computer will operate satisfactorily when subjected to vibration at a level of 2g over the range 20 c/s to 2000 c/s, or shock loads equivalent to 2g applied for up to 20 ms.

These figures are for guidance only and if particularly severe conditions of shock or vibration are expected, these should be referred for investigation, especially as regards alternative installation which would overcome the source of vibration or shock."

8. CORRECTION TO 1.2.1.

In the 8-channel paper tape code table on page 7, note that the binary values for 95 and 127 have been misprinted. They should be exchanged to read

10101 . 111	95 %	11111 . 111	127	**E**

9. CORRECTION TO 1.2.2. (CONTROL STATION)

(a) In the list of bits in a control word at the bottom of page 5, only bit 39 should be specified as the sign bit; bits 38-34 are not, of course, sign bits and on these lines "(sign bit)" should be deleted.

(b) On page 8, the colour of the TRANSFER lamp should be specified as being green.

10. CORRECTION TO 1.4.1. (CORE BACKING STORE)

The last line of page 2 should say that the program hesitates for 9.6 μ secs, not 10 μ secs as stated.

11. INSERTION FOR 1.4.6. (CARD READER)

The following paragraph should be inserted on page 1 after (f) Engineers' switches .....

"It is not possible to detect the absence of a card from the sensing platform until some 10 msecs after the last instruction of the previous card has been read. Consequently, even if no new card has been fed to the platform, bit 1 will not be set to 1 until 10 msecs after the reader became non-busy. If cards are read at full speed, the subsequent 76 and 77 instructions will be issued before bit 1 is set to 1. The apparent affect is to read a blank (unpunched) card. If this is unacceptable, the 76 instruction must be delayed until at least 15 msecs after the reader becomes unbusy.

12. CORRECTION TO DIAGRAMS IN 3.1.2. (BASIC 503 SYSTEM)

On FIG. 9. the RUNOUT P1 and P2 buttons should be changed to read LOAD R1 and R2. The MODE P1 and P2 buttons should be changed to read MODE R1 and R2.

On FIG. 12. the LOAD R1 and R2 buttons should be changed to read RUNOUT P1 and P2, and the MODE R1 and R2 buttons to read MODE P1 and P2.

13. RIBBONS FOR THE LINE PRINTER, 3.1.1.

In addition to Anelex, the following company can supply inked ribbons of the correct specification for the

503 line printers :

Silk Finish Ltd.,
Ormond House,
Barnet By-pass,
Borehamwood,
Hertfordshire.

14. OPERATING MAGNETIC TAPE HANDLERS, 3.1.5.

A note should be added between steps 4 and 5 of the unloading procedure warning operators that failure to observe the following precautions may result in a damaged reel of tape:

> After placing the packer arm in the withdrawn position, sufficient time should be allowed for the vacuum in the vacuum chamber to ease off <u>before</u> operating the thread lever; this can usually be determined by ear.

15. ALTERATIONS TO MAGNETIC TAPE DESCRIPTION, 1.4.2.

(a) Under the section 'Special Effects' on page 5, a fourth point should be added: 5

(4) <u>Noise Blocks</u>

(i) Noise blocks can be caused by switching the writing heads on or off; any noise blocks caused in this way will be less than 13 characters long. They will occur in an inter-block gap, and will usually not be read back into the computer since reading is suppressed in a normal inter-block gap.

If, however, the inter-block gap is lengthened by one or more erases, any noise block will be read.

The standard methods of dealing with noise blocks, which are incorporated in the PCP Magnetic Tape routines, are capable of dealing with those noise blocks in addition to those caused by imperfections of the tape surface.

(ii) Noise blocks may be produced if an attempt is made to read blocks at a density other than that at which they were written.

(iii) The use of a damaged tape may also cause noise blocks.

(b) The following changes have been made in the recommended methods of <u>programming</u> for magnetic tape handlers:

(i) Before starting to write on a tape, issue either 2 erase instructions or none at all. Do <u>not</u> issue a single erase instruction.

(ii) Before retreating after writing a block, do <u>not</u> issue an erase instruction since this may cause noise blocks. This point also affects the description of PCP (2.5.3.), Chapter 20.6.2.

(iii) This point is really a clarification of the method recommended, rather than a change:

Before issuing <u>any</u> instruction to a handler (72 or 77) ensure that the handler is not in an error state by issuing the appropriate 75 instruction.

DECEMBER, 1965.

503 COMPUTER MANUAL

PROGRAMMING NEWSLETTER

SCIENTIFIC COMPUTING DIVISION

The subjects covered in this newsletter are:-

(1) Modifications to the Mark 2 Software System

The changes to the Mark 2 software system as detailed in the recent Sales Document (September, 1965) have resulted in the following general changes to the form of Volume 2 of the manual:

> The enclosed re-issue of the CONTENTS LIST shows that PART 5, section 4 now refers to 'The Controlling Programs'. This section is issued together with this newsletter and explains how the functions formerly to be performed by the Executive program are now to be undertaken.
>
> Section 4 on DSP and IMP as at present issued should be destroyed. Section 1 now refers to the operation of the system with special reference to the appendix to section 2 on SPAN which is also attached to this newsletter.

The sections on PCP, SAP2 and SPAN all require detail changes in the light of the modifications and the pages affected will be re-issued shortly.

(2) Notes on operation of the Mark 2 system

The following is a brief description of the basic parts of the system as far as their operation is concerned. Much use is made of the ease of handling provided by the system DUMP, BRING and LOAD (2.2.3.25) for holding program batches on magnetic tape. A batch is usually created after a standard sequence of operating instructions, e.g. 1-11 below.

A fuller discussion of the operating system will be found in section 2.5.4 of the Manual. The full reasons for the appearance of two versions of SPAN, span (basic) and SPAN (full), are detailed in the appendix to SPAN attached to this newsletter. This also accounts for the programs SETCBS and SPANSWAP also mentioned below and issued with the basic set of systems programs.

Loading the basic systems tapes

The steps taken below are those to be taken in preparing the basic systems batch on magnetic tape. All further batches may be built up from the stage reached below.

1. Type RESET.
2. Input SPAN (full) by typing IN.
3. Input SETCBS by typing IN.
4. Type SPAN;5.
5. Type SETCBS.N. , where N is the number of units (16,000 words per unit) of core backing store fitted. SPAN is now stored in the core backing store.
6. Type RESET.
7. Input span (basic) by typing IN.
8. Input SAP2 by typing IN.

9. Input SPANSWAP by typing IN.

10. (optional) Input other binary programs required at run-time.

11. Type span;N. where N has the <u>same value</u> as in step 5.

At this stage a batch may be dumped on magnetic tape.

Each remaining system program is coded in SAP2 form and may be assembled to store using the basic batch created above. A further batch may then be created by adding SAP2 coded user programs and/or systems programs.

<u>Running assembled programs</u>

Before running the assembled program the basic version of SPAN must be replaced by the full version. If issue 1 of both span (basic) and SPANSWAP are used, then the operator must type SPANSWAP. before attempting to enter the program. If issue 2 of the above programs are used, this exchange takes place automatically as soon as the program is entered.

All programs which make use of PCP <u>must be triggered via PCP.</u> To run one program, type:

PCP. "PROGNAME"; "TRIGGER NUMBER".

To run two programs, type:

PCP. "P1NAME"; "T1"; "P2NAME";"T2".

Further information on the operation of programs using PCP will be formed in the appendix to PCP attached to this newsletter.

(3) Description of machine code, 2.1.1.1

All references to the mnemonic form of the instruction code are merely provided in order to further clarify the effect of that instruction. These mnemonics may not occur in a SAC program since the SAP compiler does not allow for their use.

Immediately below the table on page 3, the reference to 04, 05, 06 should read : 04, 05, 07.

On page 5, for +.25 read +.23.

(4) B-line modification

A note should be added at the bottom of page 14 of this section to the following effect:

> If the B-digit is present following a "standardise" instruction (65 4096) the result of the subsequent modification is unspecified. Normal modification does not take place.

(5) Correction to 2.2.3.15 (multke)

On page 2, $d_i = 1 \times 2^{-38}$
should read $d_i = -1 \times 2^{-38}$

The restriction on page 1 could be interpreted wrongly inasmuch as the user may think that the routine is corrupted beyond repair after the first entry to it. This is not so, the subroutine may be called as often as required. Even if the program containing this subroutine has a sumcheck, provided it is called as a common program by a further master program then it in turn may be called as often as required.

This applies equally to the subroutines swsort (2.2.3.13) and gensor (2.2.3.14).

(6) Correction to SAP Mark 1 (2.1.2.3)

On page 7 for 40 15. read 40 15,

(7) Note on Owncode to Backing Store Mods

The method of writing the Algol system to backing store, and of reading the system back into main store is described in Appendix 1 of Section 2.1.3 of the 503 Technical Manual.

However, additional action must be taken so that the compiler can use the backing store to hold the owncode when a program is too large to be translated normally. Whenever the system is read into main store (either from paper tape or from backing store), and has reached the systems wait that precedes the translation of a program, press the message button and read in the tape "Owncode to backing store mods" by typing IN.

If the system is read in from backing store and the "Owncode to backing store mods" tape is not input, then a program that is too large to be translated normally will cause ERRINT 4 to occur.

(8) AMENDMENTS TO SAP2 PROGRAMMING GUIDE - 2.4.1

The following alterations should be made:

PAGE 3

3.2 The third sentence should read:

"It is not the same as the relocatable input available on the 803 or the RAP relocatable binary input used in the Mark 1 system on the 503 where the position of the program is decided on input and cannot be altered without re-input".

PAGE 17

8.2 The last sentence on the page should read:

"This is done by the use of the control word array, which may only be introduced universally, and the SPAN allocation macro".

PAGE 19

8.4 The first line of the example should read;

" array Mine; (introduced universally)"

⋮

PAGE 24

The following paragraph should be added at the end of chapter 8:

"The SPAN macro 'consider[X]' cannot be used with reference to blocks of program since there is no way of naming a block. Therefore the SAP2 macro 'chapter[A]' is provided where A is a global or universal label. The effect is the same as 'consider', e.g.

```
chapter[SPIDER*SPIDER*WEB]
fast
```

brings down to mainstore the chapter of the program WEB which contains the block SPIDER".

PAGE 27

9.3 The second sentence of the second para. should read:

"However, the values of the parameters must be stored in the relevant locations."

The 2nd line of the example
"formal X,Y,Z;"
should be deleted.

From the "Note:" to the end of the page should read:

"Note: that we write

consider[X]

because X contains the address of a codeword. To access a location of the matrix we write Tkl[X,I]. This macro differs from the macro tkl in that X is taken to contain the address of a codeword and not the codeword itself. Likewise there is a macro Stl which has the same relationship with stl."

PAGE 28

The 2nd line of the example

"formal X,Y,Z;"

should be deleted.

The instructions tkl[X,I]

```
                  tkl[X,I]
                  tkl[Y,I]
                  stl[Z,I]
```

should be replaced by the instructions

```
                  Tkl[X,I]
                  Tkl[Y,I]
                  Stl[Z,I]   respectively.
```

PAGE 29

The last paragraph should read:

"The programmer must declare all arrays universally; these are introduced by the control name array immediately after the program introduction.

Thus:

```
    program compute;
    array all;
    universal many;
    block alpha,beta(one);
```

In this case, 'compute', 'all' and 'many' are universal identifiers. 'alpha' and 'beta' are global identifiers".

PAGE 41

The third item of the list at the top of the page should read:

"(iii) Another identifier. 3-part identifiers e.g. A*B*C may not be used as replacement parameters (except in the standard macros enter, trans etc)".

<u>PAGE 46</u>

All references to the introduction <u>formal</u> and the global <u>array</u> introduction should be deleted.

<u>PAGE 47</u>

The following paragraph should be added at the end of Ch.15:

"If the trans and enter macros are used to transfer to some point in another program, it is important that the program referred to in the macro is already in store when the calling program is assembled".

The sections mentioned below should be ignored for the present, they refer to programs which are under review:-

<u>PAGE 6</u>

The first paragraph.

(PCP is now available, but at the moment PTSREAD and PTSPRINT should be used for binary/decimal conversion).

<u>PAGE 7</u>

Sections 5.2, 5.3 and 5.4.

<u>PAGE 29 ff</u>

All chapter 10.

<u>PAGE 45</u>

In 14.2 (3) the phrase

"the standard input/output devices to be used and".

February, 1966.

503 COMPUTER MANUAL
PROGRAMMING NEWSLETTER
SCIENTIFIC COMPUTING DIVISION

The subjects covered in this newsletter are:-

(1) Corrections to SPAN description (2.5.2).

(2) Correction to Algol procedure "enterCP".

(3) Correction to section on Peripheral Control (1.3.1).

(4) Modifications to Sections of Volume 3.

(5) Amendments to the SAP2 Programming Guide (2.4.1).

(6) Note on errors in newsletter P8.

(7) Magnetic Tape programming note.

(8) Note on Chapters 31 and 32 of the program library.

(9) Note on PCP description (2.5.3).

(10) Corrections to FORTRAN description (2.4.3).

(11) Restriction in SAP Mark I.

(1) Corrections to SPAN description (2.5.2)

Page 1

In part 1.3, in the last line of this paragraph replace 'one' by 'some'.

Page 2

Bottom of page:
For "4095", read "8191".

Page 7

This page is re-issued together with this newsletter.

Page 10

Add, after the last line of this page, the following:

'reserve' takes the absolute address 18, so that when the programmer uses 'reserve' he should include the appropriate replacement declaration in his program.

Page 11

Paragraph 2, line 3.
Substitute:
"....... 'reserve' is an identifier naming location 18 which is used by SPAN......."

Page 14

Section 8.2. :

Delete the sentence beginning: "The programmer must however....."

Page 15

This page is re-issued together with this newsletter.

Page 19

Delete the third paragraph which begins: "A tagged codeword.........".

Insert a note referring to the following text which should be added to page 22:

The 'check subscripts' marker may be set in the headerword by adding 4 to the parameter 'type' when using 'ALLOC', or by placing the marker in the headerword specified in an 'alloc' macro (see page 19 description). The effect is such that whenever reference is made to the block, the index of the element referred to is checked to ensure that it lies within the lower and upper subscript bounds. The codeword of the block will take the special 'tagged' form which it also takes when the 'retire' operation is carried out.

Page 20

Section 13:

In part 1, delete the reference to formal.

Delete part 2 and replace by the following:

"Note that there are indirect equivalents to the macros tk 1 and st 1 (see page 7, issue 2). There are, however, NO indirect equivalents to tk 2 and st 2." C

(2) Correction to Algol procedure "enterCP"

(2.1.3.3. page 28).

1. Remove the switch declaration.
2. Remove the label 'noprog' form the text of the procedure.
3. Replace

```
        "elliott (4,2, noprog, 0,2,0, CPinteger)"
by
        "elliott (7,3, 4 , 1,4,2,  9  )
         elliott (2,0,  CPinteger, 0,0, 0, 0 )"
```
C

(3) Correction to section on Peripheral Control

(1.3.1., page 2)

On line 3, for 'handler 2', read 'handler 3'.

(4) Modifications to section of Volume 3

(Section 3.1.1., "Materials to be Obtained")

Page 11

Delete the Part Number of the liquid-ink pen set and also the whole of the next six lines.

Insert, as the Part Number of the liquid-ink pen set, "20-300".

Insert, in place of the six lines deleted:

"Full details of the liquid-ink pen set, which includes an ink cartridge, a cleaning kit and a carrying case, can be obtained from the suppliers."

(Section 3.1.6., "Line Printer Operating Instructions")

Page 25

Add the following new sub-paragraph:-

The test tape must be run daily:
details of this are given in Section 3.1.3.,
"Test Programs and First Line Maintenance."

(Section 3.1.8)

Page 10

Add the following new sub-paragraph:

(7) Run the daily test program (see section 3.1.3).

(5) Amendments to the SAP2 programming guide

(3.3.1., 2nd paragraph)

Page 4

Note that a programname may never be quoted by itself in any program: it may only be used to form a relative identifier. e.g. an identifier such as LABEL*BLOCK*PROGRAM would be permitted, but the corresponding identifier PROGRAM would be rejected.

(3.4., last sentence)

Page 4

The reader is referred to the note in 13.4.1. on page 42 (see below)

(6.3.)

The phrase "or by a mnemonic followed by a comma" should be deleted.

(6.6.)

After the last sentence insert "though the first of these should, of course, not be the underline character."

(6,7., comments to the example)

Page 10

for "form $2y_{u+1}$ ($= y_u + \frac{x}{y_u}$)"

read "form $2y_{u+1} = (y_u + \frac{x}{y_u}$)"

(6.8.1.)

Page 10

for " -274,877,906,444" read " -274,877,906,943".

(Note that the number -274,877,906,944 must be represented as 400: or as 84000 000 000 000).

(6.8.2.)

Page 12

for "-21 is 7 777 777 777 753"
read "-21 is 87 777 777 777 753"

(7.4. example)

Page 16

replace the line "end exit [1]" by "end)exit [1]".

(8.2.)

Page 17

At the end of the page insert "Storage space booked in this way should be carefully deleted when no longer required, as this cannot be done by SPAN. Deletion is effected by the consider and delete macros, (see 8.4.1.)".

Page 18

Insert, at the end of the paragraph "Note that the control word array may be positioned only after the word program (see note given in previous paragraph, and also Chapter 10)".

Page 22, example :

The second comment should refer to "length c[N] +1" instead of to "length N+1".

(9.1. 1st example)

```
        for the lines   "20y
                          63x "
                  read  "20y
                          63y "
```

(9.2.)

Page 26 9th paragraph should read:

The exceptions to the convention are the macros tk 1 [A,I], st1[A,I] , tk2[A,I,J], st2[A,I,J].

(11.1.)

Page 30

The second paragraph should be deleted.

(11.2., 11.3., 11.3.1., 11.3.2.)

These sections should be ignored.

(Chapter 12)

This Chapter should be ignored.

(13.4.)

Page 42

In the last paragraph of note 1, for "Once such an identifier is found to be a standard macro name" read "Once such an identifier has been used as a standard macro name."
(This should be borne in mind when reading the second note in 3.4).

(14.2., 3rd part)

Page 45

Remove the phrase "standard input/output devices to be used and the "

(14.3.)

Page 45

for "blocks" read "block"

(6) Note on errors in newsletter P8

1. Amendment 6 should have been:

Correction to SAP Mark 1 (2.1.2.3)

On page 7 for "40 15."
read "40 15,"

2. Amendment 8, Page 10.

The reference to PAGE 29 should be ignored. This should have referred to certain parts of Chapter 11. The correction now appears in Amendment 5 of this newsletter.

(7) Magnetic Tape programming note

The user's attention is drawn to the warnings outlined in issue 4 of section 1.4.2. of the 503 Manual.

The user is recommended to employ a block addressing system on magnetic tape for the following reason:

> The use of the 'erase' instruction may cause a spurious character to be written which will be treated as a block by 'advance' or 'retreat' instructions. Thus, recovery techniques previously recommended are not suitable and the programmer should therefore introduce block addressing in order to be able to verify the tape position.

(8) Note on descriptions of chapters 31 and 32 of the program library

(2.2.3.31, 32)

The parameter B is described as the 'codeword' of the array. This term is somewhat misleading. The user is recommended to substitute the word 'identifier'.

(9) Note on PCP description (2.5.3.)

Page 16

The reference (using an asterisk) to the high-speed character printer should be deleted as this peripheral is no longer included in the 503 catalogue.

(8) Note on description of chapters 31 and 32 of the program library

(2.2.3.31, 32)

The parameter B is described as the 'codeword' of the array. This term is somewhat misleading. The user is recommended to substitute the word 'identifier'.

(9) Note on PCP description (2.5.3.)

Page 16

The reference (using an asterisk) to the high-speed character printer should be deleted as this peripheral is no longer included in the 503 catalogue.

(10) Corrections to FORTRAN description (2.4.3.)

The user's attention is drawn to the following amendments which are included in the re-issued sheets distributed with this newsletter:

(i) Spaces, slashes and Hollerith strings in output FORMAT statements are not output unless they are followed by an output element

(ii) The message 'RAPX' at the start of compilation means RAP is corrupt. The store should be cleared and RAP reinput.

(iii) If there is no lineprinter attached to a machine or if the lineprinter is switched off, then the compile time error messages will be output on the typewriter.

N.B. Amendments (ii) and (iii) only apply to FORTRAN issue 2.

(11) Restriction in SAP Mark 1

Issue 3 of SAP Mark 1 contains corrections to errors mentioned in previous newsletters. The following restrictions remains:

The replacement facility is suspended while SAP is reading the identifiers in any basic name (i.e. underlined word) list, and also while it is reading the parameters for SUBR and COMP instructions.

503 COMPUTER MANUAL

PROGRAMMING NEWSLETTER

Tapes which must be used with STAR Issue 2

Phase 1

503 SPAN ISSUE 4
503 SPAN (BASIC) ISSUE 4
503 SAP 2 ISSUE 2
503 SPAN SWAP ISSUE 3
503 PCP ISSUE 4
503 SETCBS ISSUE 1
503 DUMP MK 2 ISSUE 1 (512 buffer)
503 RAP MT ISSUE 1
503 STAR ISSUE 2

Phase 2 – FORTRAN – SAP2

FORTRAN latest issue (March)
503 SSSPAN ISSUE 1
503 STARCH (FOR SAP) ISSUE 1
503 STAR (PHASE 2) ISSUE 2
(+ tapes mentioned in phase 1 above)

Phase 2 – ALGOL

503 ALGOL TAPE 1 ISSUE 1
503 ALGOL TAPE 2 ISSUE 1
MOD 5
503 MOD 10 FOR STAR ISSUE 1
503 STARCH (FOR ALGOL) ISSUE 1
503 STAR TRIGGER ISSUE 1
503 STAR (PHASE 2) FOR ALGOL ISSUE 1
503 DUMP MK 2 ISSUE 1 (64 buffer)

NOTE: The latest issue as indi cated must be used. An approximate date of issue has been stated where a tape has been issued direct by the program writers and not via the program library.

503 COMPUTER MANUAL

PROGRAMMING NEWSLETTER

ELLIOTT-AUTOMATION COMPUTERS LTD

The subjects covered in this newsletter are:-

1. FORTRAN

CONTENTS

1.1 GENERAL

The 503 FORTRAN system consists of two programs, FORTRAN and FORTDR. FORTRAN is the compiler and FORTDR contains the input/output routines and standard functions needed at run time. FORTRAN and FORTDR are both written in SAP2 and operate in conjunction with SPAN and PCP. The compiled object program consists of a set of SPAN program blocks, (chapters), one for the main program and one for each subprogram (but see 1.8). When compilation is finished, the compiler is deleted from the store.

1.2 CONFIGURATION

The 503 FORTRAN system is best suited to a configuration with at least two units of core backing store, at least two magnetic tapes and a lineprinter. If programs are to be read from cards, an Elliott card reader is required. If only one unit of core backing store is used, compilation is slower because magnetic tape has to be used even for small programs (see 1.7). If no lineprinter is available, compile time error messages appear on the typewriter which makes operation of the system much slower. If magnetic tape is not available, the FORTRAN system has to be reloaded from paper tape after each run. 503 FORTRAN cannot be used on a configuration without core backing store.

1.3 FORTRAN TAPES

The six 503 FORTRAN systems tapes are currently

FORTRAN	TAPE 1	(ISSUE 3)
FORTRAN	TAPE 2	(ISSUE 3)
FORTRAN	TAPE 3	(ISSUE 3)
FORTRAN	TAPE 4	(ISSUE 3)
FORTRAN	TAPE 5	(ISSUE 3)
FORTDR		(ISSUE 3)

These tapes are issued by the library and punch marked with legible headings. All other 503 FORTRAN tapes have been issued unofficially by program writers; and are superceded by ISSUE 3 tapes; they should be destroyed.

1.4 GENERAL SYSTEMS TAPES

In order to use the 503 FORTRAN system, the following general system 2 tapes are needed. The issues indicated must be used. They are the most recent issues, and other issues should be destroyed.

SPAN	(ISSUE 4)
SETCBS	(ISSUE 1)
SPAN BASIC	(ISSUE 4)
SAP2	(ISSUE 2)
SPANSWAP	(ISSUE 3)
PCP	(ISSUE 4)
DUMP2	(ISSUE 1)

1.5 OPERATING INSTRUCTIONS

These instructions are for installations with magnetic tape. Otherwise see 1.6.4 overleaf.

Tape issue numbers are not given. Refer to 1.3 and 1.4. RAPMT is assumed to be in store, and a batch tape loaded on Handler 1.

STEP	TAPE IN READER 1	INSTRUCTION TO 503
0		RESET.
1	SPAN	IN.
2		SPAN;5
3	SETCBS	IN.
4		SETCBS. N. *
5		RESET
6	SPAN BASIC	IN.
7	SAP2	IN.
8	SPANSWAP	IN.
9		SPAN;N. *
10	FORTDR	SAP2.
11	FORTRAN TAPE 1	SAP2.
12	FORTRAN TAPE 2	change key 39
13	FORTRAN TAPE 3	change key 39
14	FORTRAN TAPE 4	change key 39
15	FORTRAN TAPE 5	change key 39
16	PCP	SAP2.
17	DUMP2	IN.

STEP	TAPE IN READER 1	INSTRUCTION TO 503
18		DUMP2;5. *
19		DUMP2. BATCH1.
20		RESET.
21		IN;BATCH1.
22	FORTRAN OBJECT TEXT	FORTRAN;n. **

* see 1.6
** see MANUAL 2.4.3 p. 28

1.6 OPERATING INSTRUCTIONS: NOTES

1.6.1 In steps 4 and 9, N is the number of units of core backing store to be used by the FORTRAN system.

$$1 \leq N \leq 8$$

1.6.2 Step 18 should be omitted if the batch tape has previously been prepared. In step 19 the name BATCH1 may be replaced by any name not already used on the batch tape. See MANUAL 2.3.2.46 DUMP2.

1.6.3 For subsequent running of FORTRAN programs when RAPMT is in store, only steps 20-22 are required.

1.6.4 For an installation without magnetic tape the whole of the operating procedure, omitting steps 17-21, must be followed every time a FORTRAN program is to be compiled.

1.7 USE OF MAGNETIC TAPE AT COMPILE TIME

1.7.1 One unit of core backing store.

When only one unit of core backing store (16,384) words is available, translated subprograms are written to handler 8. Before compilation a scratch tape should be available on handler 8, loaded at B.O.T. and set to REMOTE.

The message

H8MAN

is displayed at the start of compilation if handler 8 is not available. Compilation continues when this is rectified.

The message

WRFAIL

is displayed when the system is unable to write correctly to handler 8. Compilation must be restarted after changing the tape, or switching to another tape deck (set to handler 8).

1.7.2 2 or more units of core backing store.

During compilation, the core backing store may become full of the compiled program chapters. In this case the last chapter compiled is written away (banished) to a scratch tape on handler 8, and likewise all subsequent segments. The message

H8MAN

is displayed at this point, if handler 8 is not available. Compilation continues when it becomes available.

1.8 SEGMENTATION OF FORTRAN PROGRAMS

The maximum length of a compiled chapter of a FORTRAN program is 1 520 words. (3 040 instructions). For many programs this is insufficient for a main program or a subprogram. 503 FORTRAN provides a method of splitting any main or sub-program between a number of chapters, up to a maximum of 100 chapters for the complete program.

A new chapter is normally started for each main or subprogram, and terminated at the end of the main or subprogram. However, the occurrence in the program of the significant comment

C SEGMENT

has the immediate effect of terminating the current chapter and starting a new one. A jump from the old chapter to the new is automatically compiled. The significant comment must have the form

```
1 TAPE
            C tab SEGMENT
       or   C 5 spaces SEGMENT
2 CARDS
            C 5 blank columns SEGMENT.
```

A segment comment may occur anywhere, but it should be born in mind that a transfer between chapters (segments) takes longer than a transfer within a chapter. Hence it would be inadvisable to place a segment comment inside a DO loop and in any case to have a large number of GOTO's which 'cross' a segment comment.

1.9 ERROR NUMBERS

For issue 3, the following compile time error message should be added to the list on p19 ff. of the FORTRAN specification.

Error No.	DESCRIPTION OF ERROR
58	FORMAT label used previously as a non-FORMAT label outside a READ or WRITE statement.

1.10 PROGRAM TITLES 2.4.3 2.1.4 (page 2)

The text of 2.1.4 should be replaced by

2.1.4 The main program must start with a comment line, which indicates the six letter name of the program. The format must be,

```
1 TAPE
    newline C tab PROGNAME
or  newline C 5 spaces PROGNAME

2 CARDS
    C 5 blank columns PROGNAME
```

The first six alphanumeric characters of this comment are taken as the program name. The first character must be a letter and the next five either digits or letters. Layout characters (space tab etc.) may not occur in the first six characters. The remainder of the comment is ignored.

2. INTERN

2.1 The following change to the description of INTERN should be made

page 5 (issue 1)
(b) Block END TEST

Add:

"An extra block should be written following the block containing the special item indicating the end of inserted information. The contents and size of this block are irrelevant."

2.2 The present tape of INTERN, (issue 1) contains a fault which will cause an incorrect exit to be made at the end of a call of INTERN. A corrected version, (issue 2) will be issued shortly.

3. AUTALG

A number of errors have been discovered in AUTALG (issue 1). These will be corrected in issue 2. Until then, the following restrictions should be observed to ensure correct translation.

3.1 A TITLE should be terminated by blank cr lf

3.2 The words LINE EXIT STOP WAIT TELEPRINTER should be terminated by cr lf not sp cr lf

3.3 A SET list should not contain , cr lf

3.4 The construction

```
JUMP IF A=SINB
```

should be avoided.

4. FEAT

The following amendment should be made

Appendix 2 page 3 paragraph 1.

Add:

"After <u>begin</u> is output on punch 1, SUBSCR OFLO can be caused by error types 10 and 5."

5. SAP

An address of the form

LINK CP - i

where i is an unsigned integer, will be incorrectly assembled when a program containing it is assembled to paper tape by SAP1. Otherwise it is assembled correctly. The effect may be achieved in all cases by another construction.

```
e.g.             30  LINKCP-1
may be written   67  <-1>
                 30  LINKCP
```

6. PCP

The following amendment should be made:

Appendix 1 (issue 1)

The box, Tape Reader / F. function should read

1	prepare
2	provide
3	provide and prepare

7. SPAN

Ammend as follows:

page 8 (issue 3)

line 24	for "20 18 : 30 0 "
	read"20 18 / 30 0 "
line 30	for "10 18 : 20 0 "
	read"10 18 / 20 0 "

8 DUMP2

Amend as follows:

TAPES

The tape provided is in symbolic assembly code (i. e. mnemonic form) for input by SAP mark 1. It is recommended that a binary tape be produced for normal use. This is done using SAP as usual, but with key 35 depressed, so that no sum check is made when DUMP2 is entered.

9. SAP2 2. 4. 1.

All references in the description of SAP2 to IMP, DIOR, DSP, EXEC should be deleted. The description is being rewritten to correct this and to take account of other changes. In cases of uncertainty, refer to SAP2 Appendix 1. which was written more recently.

503 COMPUTER MANUAL
PROGRAMMING NEWSLETTER
ENGLISH ELECTRIC COMPUTERS LTD.

This Newsletter is issued with 503 computer manual amendment no. 2/0033.

The subjects covered in this Newsletter are:-

P12

1. ALGOL 3

The new ALGOL compiler for non-basic configurations is now published and a full description of the new system is given in the new section of the 503 Manual (2.1.5).

2. ALGOL 1 EXTENSIONS

As mentioned in the amendment list, previous issues of the "MOD" tapes should be destroyed. The amendment list gives the current issues. A new description of the extensions to the Algol 1 compiler will be published shortly as Appendix 3 to section 2.1.3 of the 503 Manual.

2.1 Algol 1 MOD 11 (code statements)

The facility provided by this extension to the compiler is as described in the new ALGOL 3 section of the 503 Manual (2.1.5, 4.7.1).

3. DUMP2

Issue 2 corrects parity errors which existed in certain Issue 1 tapes and also corrects the error detailed below under "SPECIAL OPTIONS" It makes it possible to write STAR batches to magnetic tape using the program STCOPY (see P12 8.2). The description of DUMP2 in the 503 Manual (2.2.3.46) should be amended.

TAPES - Add:-

"The binary version should be produced with key 35 depressed to prevent a sumcheck being formed."

STORE USED Delete N.B. sentence and insert:-

"locations 4 and 8176 are preserved on dumping. If an interrupt occurs during running of a program it is necessary to preserve and restore these locations."

METHOD OF USE. Paragraph 2, replace first sentence by:-
"For a Total Dump, DUMP2 may be input at any stage but must not be placed below location 100."

SPECIAL OPTIONS. Page 10, line 12 - replace BUF6 reference in example by BUF6 [; <+32>];

4. FORTRAN

Item 1.5 of Newsletter P11 should be amended as follows:-

Step 9 (Key 34 depressed) span; N.

NOTE The batch tape must have been previously prepared to contain program batches, see 503 Manual 2.2.3.46, page 4.

4.1 FORTDR

A new tape is issued (Issue 4) to correct an error found in certain Issue 3 tapes.

5. SAP

SAP 1 Issue 3 contains an error which affects SAC programs which are output as SAC relocatable binary tapes and which are longer than 4096 locations, excluding dataspace. Assembly into store is not affected.

The error will occur if after the 4096th wholeword in the body of the program (an instruction - pair would constitute a wholeword) a reference to a blockname or global label, A, occurs and A is in a block not so far met.

Binary tapes produced from such programs are incorrect. It is possible that the error will be indicated by the message NOPROG, displayed during input.

6. MTALG

6.1 Issue 3 corrects certain errors in Issue 2, allows the use of MTALG with ALGOL 1 MOD 8 or MOD 9 in store, and extends to 8 the number of handlers usable.

6.2 The description in the 503 Manual (2.2.3.43) should be amended.

GLOBAL DECLARATIONS paragraph. Delete up to the words ......"where P is the handler number, $1 \leq P < 4$." and insert:-

"GLOBAL DECLARATIONS

The following variables are declared globally in the procedure package:-

<u>integer</u> curr, curr1, curr2, curr3, curr4, curr5, curr6, curr7, curr8,
max, max 1, max2, max 3, max4, max5, max6, max7, max8,
errs, errs1, errs2, errs3, errs4, errs5, errs6, errs7, errs8,
param, param1, param2, param3, param4, param5, param6, param7, param8,

<u>boolean</u> parity, wpermit, shortblock, longblock, endtape, begtape, allowS, allowL, chsum;

The variables curr, max, errs and param are used as workspace by the procedures. They may also be used by the main program, but their values are lost when a procedure is called.

In the following description, the notation currP, maxP, errsP and paramP is used for the variables curr1, curr2, ...curr8, max1, ... max8, etc. where P is the handler number, $1 \leq P \leq 8$".

SUMMARY OF PROCEDURES. Delete this paragraph and insert:-

"SUMMARY OF PROCEDURES

The tape has the following form:-

```
comment MTALG;
begin integer  curr,curr1,curr2,curr3,curr4,curr5,curr6,curr7,curr8,
max,max1,max2,max3,max4,max5,max6,max7,max8,
errs,errs1,errs2,errs3,errs4,errs5,errs6,errs7,errs8,
param,param1,param2,param3,param4,param5,param6,param7,param8;
boolean  parity,wpermit,shortblock,longblock,endtape,begtape,chsum,
allowS,allowL;
procedure            display
procedure            test
procedure            delay
procedure            conclude
procedure            control
procedure            rewind
procedure            select
procedure            tmout
procedure            dispeot
procedure            search
procedure            readwd
procedure            writewd
boolean procedure    wready
boolean procedure    rready
comment              initial settings;
Ⓗ
precompile ;
Ⓗ
```

The main program must include one extra end to go with the begin at the beginning of the procedure package."

7. PCP

The references to ALGOL MK2 in the description of PCP should be deleted from sections 2.5.3.1.1 and 2.5.3.2.2 of the 503 Manual.

8. STAR

8.1 STAR/PHASE2/SAP

The limits for this program are 27.45.

Thus to assemble it type SAP2. 27.45.

8.2 ST COPY

This is a new program to enable STAR batches on magnetic tape to be copied. Details are given in Appendix I to section 2.5.5 of the 503 Manual.

8.3 STRAP

As this program has now been replaced by RAPMT (see 503 Manual, 2.2.1, Appendix I) the references to STRAP on page 1 of section 2.5.4 of the Manual should be altered accordingly.

9. SPAN

Amend Manual (2.5.2, Appendix 3, page 2) as follows:-

after line 14 insert:-

"SPAN ERROR 9	Core backing store not available (e.g. because of parity failure)"

after line 19 insert:-

"SETOER	incorrect handler number in call of set origin (H, ORIGIN) (allowed range is $1 \leq H \leq 8$)

The following errors are not detected specifically, but will eventually cause one or other of the above error messages.

1. incorrect value of ORIGIN in set origin (H, ORIGIN)
   (allowed range is 0≤ORIGIN≤8191)

2. incorrect handler number in banish [HANDLER]
   (allowed range is 0≤HANDLER≤8)".

10. SOURCE CODE TAPES

The tapes listed in the amendment list can now be obtained on application to the Library.

11. LPRALG

A new issue of this tape is now published (Issue 2), which allows the use of LPRALG with ALGOL 1 MOD8 or MOD9 in store and to add the procedure call punch (n) where n = 4. The description of LPRALG in the 503 Manual (2.2.3.24) should be amended.

METHOD OF USE - Add:-

"punch (M)

A call of punch(M) where M = 4 sets the lineprinter as the current output device."

P13
(JULY, 1969)

503 COMPUTER MANUAL
PROGRAMMING NEWSLETTER

INTERNATIONAL COMPUTERS LIMITED

This newsletter is issued with 503 computer manual amendment No. 2/0034.

The subjects covered in this newsletter are:

1. ALGOL 1

1.1 ALGOL 1 ISSUE 2

This issue of the compiler replaces Issue 1 for basic machines. Known errors in Issue 1 have been removed. Issue 2, however may not be used in conjunction with any of the MOD tapes, and cannot be placed on Core-Backing store using Tape 3. Details of the restrictions in Issue 2 and the Issue 1 errors which have been removed are to be found in Appendix 4 to section 2.1.3 of the Manual. In addition, the following amendments should be made to section 2.1.3:-

Page 20

(i) 'grouping (E)' has been modified so that the permitted range of E is $1 \leq E \leq 12$. In addition, outside this range, the presumed setting is "prefix (££1??)."

(ii) 'advance (E)' has a permitted range of $1 \leq E \leq 3$.

Page 23

(i) Elliott orders are not permitted in read and print statements.

(ii) If a label is used within an elliott order, the label must be placed in the current block. Failure to observe this will result in Error No. 51.

(iii) There are two extensions to the type of identifier which may be used in an elliott order:-

(a) The name of a type procedure - provided the elliott order is within the body of the procedure.

(b) String Identifiers. The identifier refers to the first location occupied by the string.

1.2 ALGOL 1 EXTENSIONS

A description of the extensions or MOD tapes to the ALGOL 1 compiler is now to be found in Appendix 3 to section 2.1.3 of the Manual.

1.2.1 ALGOL 1 MOD 12

ALGOL 1 MOD 12 is an extension to ALGOL 1 Issue 1 for installations without backing store. It enables Boolean expressions to stand in print statements, and introduces two procedures, which affect character handling and cause a return to the presumed settings for read and print statements respectively. It also modifies the continuation values used after certain errors have been detected in the standard procedures "sqrt", "exp" and "ln".

1.2.2 INSERT

This program enables users to make new ALGOL 1 systems tapes so as to include the extensions or MODS.

2. ALGOL 3

2.1 T.I. TAPES AND DOCUMENTATION

The T.I. versions of A3C, A3D and A3L are now available on request together with documentation. This consists of the:

OPERATING INSTRUCTIONS for producing new versions of the systems programs on paper tape (Appendix 3 section 2.1.5) or core-backing store (Appendix 4 section 2.1.5).

The above appendices are issued but the following are available only on request.

* 503 A3C Program Sheets and Reference Tables
* 503 A3D Program Sheets and Reference Tables
* 503 A3L Program Sheets and Reference Tables
  503 ALGOLB and ALGOLM Flow Charts
+ 503 and 803L Compiler Notes

*

The reference tables give a list of relative addresses and all references to them by other locations in that section of the compiler with a note of the instruction.

e.g.

```
31, 102* 15, 109 (30) 3, 109 (30)
32, 102* ......
```

This extract shows that location 31 of block 102 is referred to in two places i.e. locations 3 and 15 of block 109. In each case the instruction is 30.

There are in fact two sets of tables for each section of the compiler, one for references by group 4 instructions and the other for references by other than group 4 instructions.

+ The notes have proved helpful in 803/503 compiler development work, and are already being made available to 803 users. The notes are not in the usual manual format (parts are in manuscript) and it should be emphasised that they are basically a rough guide to the way in which the compiler works. The detailed information might be incorrect for any one compiler.

Users who will require copies of any of the documentation available on request are asked to write to the address below.

The documentation will then be forwarded as soon as possible.

ISSUE TEAM
Software Issue Branch,
I.C.L.
30/31 Friar Street,
Reading,
Berks.,
RG1 1JP.

2.2 ALGOL 3 DESCRIPTION AMENDMENTS

The following amendments should be made to the description in section 2.1.5 of the MANUAL:-

CHAPTER 3

Page 52 - line 7 should read:
"integer parameter and must satisfy"
Page 57 - In the example at the foot of the page the first x in line L should be an n.

CHAPTER 4

Page 70 - 'Address' column
Delete 2nd paragraph and replace by the following:-

"The address of the representative location of the array. This location holds the address of the first element of the array in the least significant 17 bits. It holds additional information required by A3D between bits 20 and 39 e.g. 00 A/26 0 will set the first element of the main store array A to zero, whatever ranges the subscript(s) of A have (see 4.7.2.1)"

<u>Page 85</u> - Paragraph 2, last line.

Delete "applied" and insert "applies".

<u>Page 87</u> - line 2

Delete "and" insert "of".

<u>Page 88</u> - Paragraph 1.

Add the following at the end of the paragraph:

"Texts already on the library tape"

"To change a text on magnetic tape, a new text with the same name is written up. As the dictionary is updated, the original text can no longer be accessed. However, if the contents of an auxiliary text are changed, the new code will be referred to only when that particular text is called.

<u>Example</u>

Texts written up in the usual way:-

a, b, c, d ;
text of a Ⓗ
text of b Ⓗ
text of c Ⓗ
text of d Ⓗ Ⓗ

Contents of one auxiliary text changed:-

b ; new text of b Ⓗ Ⓗ

If <u>library</u> d ; is included in a program, the effect will be to call down the original texts a, b, c and d. The new text of b will only be called down by the statement '<u>library</u> b ; ' "

CHAPTER 5

Page 101

Insert error number 51 i.e.

51 Error in a call of procedure "elliott"

CHAPTER 7

Page 180 - "Error Messages"

Delete the "Meaning and effect" comment for 'GMT error MAN H N' and substitute the following:-

"The required handler 'N' is in manual. The program continues when this is corrected".

Also add the following message:-

Message	Meaning and effect
GMT error MAN H 'N' NO WPR	The required handler N has no write permit ring. No continuation is possible.

2.3 A3D ISSUE 2

This new issue corrects an error in the arcsin, arccos routines.

2.4 A3C ISSUE 2

A new A3C tape is distributed with this newsletter, correcting an error in Issue 1, affecting '<u>for</u>' statements. If the control variable was a name parameter or an array element stored in core-backing store, the code compiled for the statement following <u>do</u> might in certain circumstances, be incorrect leading to undefined errors.

2.5 ALGOLB ISSUE 2

A new issue of ALGOLB and ALGOLB (NO LINEPRINTER) is distributed with this newsletter. When creating the system with the new issue, information about each stage is stored on core-backing store and not in main store. This means that once the system has been set up on core-backing store, the executive may be input from paper tape, provided that it occupies the same position in main store as the executive occupied when the system was created. This facility already exists with ALGOLM.

3. ALGOL 1 PLOTTER PACKAGE

A new tape is issued (Issue 3) together with a revised description. The new issue has been developed in conjunction with the G.P.O. Alterations and improvements include a new character set, replacement of many "elliott" orders by ALGOL statements and the facility for altering the size of the centered characters. The following is a summary of alterations and improvements on Issue 2:-

1. Where possible elliott orders have been replaced by ALGOL statements.

2. The method of altering the dynamic routines has been replaced by a new and improved method.

3. The global variable "page" may now take the values 1,3,5, & 7. When page = 3,5 or 7 this has the effect of rotating the plotter through 90^o, 180^o or 270^o in an anti-clockwise direction.

4. The global variable "censize" has been introduced which enables the users program to alter the size of the centered characters. This variable's standard setting is 10 which is set by the plotter package. The produced character then occupies a square 10 x 10 plotter steps.

5. A new character set has been introduced with a new and improved method of plotting these characters. The procedure 'plotchar' has been introduced which outputs all characters, from procedures 'concharacter' and 'output'.

6. The lineprinter routines have been removed since ALGOL1/MOD3 must be in store when using this plotter package.

7. Lower case letters have been omitted and are now output as upper case.

8. If the plotter is in manual, the error message 'PL man' is displayed on the typewriter. When the manual button is released the plotter automatically continues.

3.1 AIDS FOR ALTERING THE PLOTTER PACKAGE CHARACTER SET.

Appendices 2 and 3 to the revised description give details (and listings) of routines to convert co-ordinates to "elliott" orders and plot co-ordinates.

3.2 EDIT FOR LOWER CASE CHARACTERS

Appendix 1 refers to this edit (issued as a tape EDITPLOT) which will produce a version of the new Plotter Package for users who require lower-case cursive script.

4. ALGOL 3 PLOTTER PACKAGE

This is a version of the revised Plotter Package (Issue3) modified for use with ALGOL 3. A revised version of the program CHAROUT has also been produced - CHAROUT (PLOTTER VERSION). These tapes are described in Appendix 2 to section 2.1.5 of the Manual.

5. MTFEAT

This is a version of FEAT designed to use magnetic tape instead of core-backing store. A few minor improvements and alterations to the translator have been made but the basic function remains unaltered.

A new description is now available; the tape is supplied on request.

5.1 FEAT STANDARD PROCEDURES 1 and 1c

A new issue of each of these tapes (Issue 2) has been produced to correct certain errors in Issue 1 and extend the facilities available. The tapes are supplied on request.

5.2 MTSTOR

This magnetic tape program is required by MTFEAT. It can, however, be used as a general purpose program to write and read blocks to and from magnetic tape on handler 4. The description is issued with this newsletter; the tape is supplied on request.

6. CP

This new program consists of a number of general purpose routines. It includes the facility to dump main store on backing store, retrieve it, list the limits of each program and the space occupied.

7. EDIT 8

The program description (section 2.2.3.27 of the MANUAL) should be amended as follows:-

Notes

Delete 3 and insert

"3. In treating FL, IB and DL commands, space is allowed for 30 non-ignorable characters in the edit string. If the edit string of an FL,IB or DL command exceeds 30 characters, the effect of the command is undefined."

8. PCPP

A new issue of this tape has been produced (Issue 2) correcting certain errors in Issue 1. This is available from the address on page 5 by request.

9. INTERN

A new issue of this tape has been produced (Issue 2) correcting certain errors in Issue 1. This is available from the address on page 5 by request.

10. PCP

The description (section 2.5.3 of the MANUAL) should be amended as follows:

> BSUPPLY macro produces 14 bits in the accumulator. Bits 1-13 specify the address of the codeword of the buffer supplied.
>
> Bit 14 is 1 if PCP supplied the buffer (and on output will automatically return it.)
>
> Bit 14 is 0 if buffer supplied by the users program.

11. SOFTWARE DEVELOPMENT

The issue of programs and documentation described in this newsletter marks the end of new software development for the 503. However, the 4100/503 LIBRARY SECTION will still be available to answer queries.