503 TECHNICAL MANUAL

VOLUME 2 : PROGRAMMING INFORMATION

PART 2 : 503 LIBRARY OF PROGRAMS

SECTION 1 : RESERVED AREA PROGRAM

PROVISIONAL

Elliott Computing Division, Elliott Bros.(London) Ltd.,
Elstree Way, Borehamwood, Herts.

MAY 1963

503 RESERVED AREA PROGRAM (RAP)

(MK I. ISSUE 3).

(A detailed specification of the program which contains those basic routines which are considered essential. Details of routines which will be included in the Reserved Area if there is room, are given under 'Optional Controls'.)

C.F.DEAL

MAY 1963

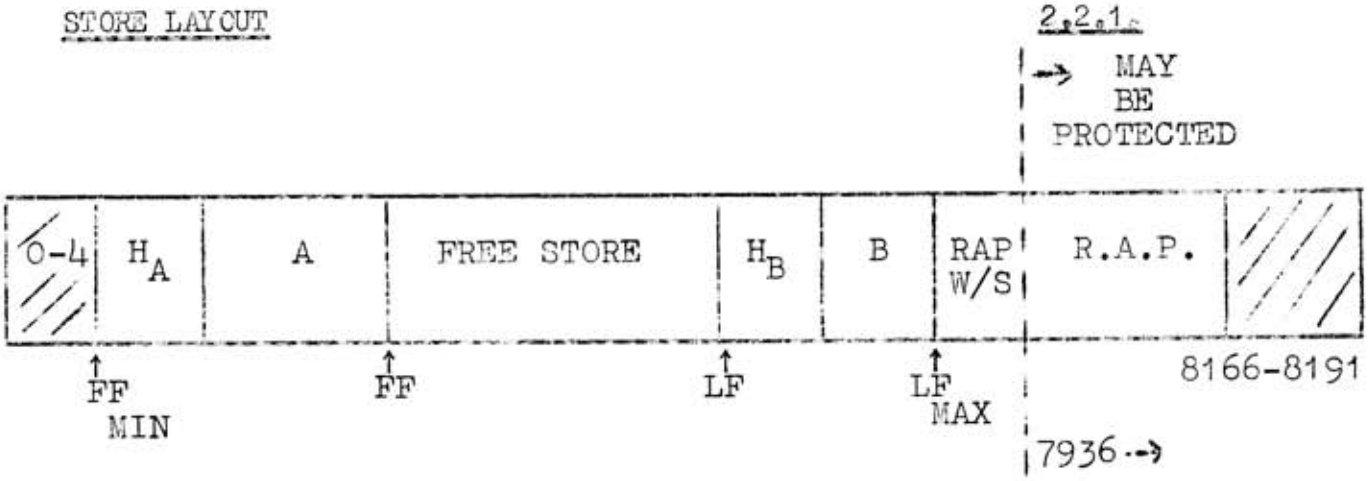
INTRODUCTION 2.2.1.

The Reserved Area Program, in conjunction with the MESSAGE button, provides a means of controlling the 503 computer. Communication, between the operator and the computer, is through a directly coupled, electric typewriter. Pressing the MESSAGE button causes entry to the RAP which then expects a message.

The RAP is read into locations 7936 - 8165 of the computer store by the fixed instructions contained in locations 0-3. The locations 7936 - 8191 form the reserved area of the store and the protection of this area can be removed by depressing the 'NO PROTN' button on the control console (see 1.2.1.)

The RAP contains an Input routine which will read a a relocatable program tape and place the program in the first available space at either end of the available store. Provision is made for several programs to be in the computer store at the same time and communication between such programs is facilitated by means of a system to which each program is linked on being stored. Each program is called by a name, which is held in the program itself, and hence the user does not need to know the stored position of any program. The RAP provides a means of entering the programs so held, and when the running of a program is completed, control of the computer returns to the RAP, where it remains until the next message is received through the typewriter.

STORE LAYOUT



In the above diagram, assume that A and B are the first and second programs to be placed by the RAP and that HA and HB are the locations which contain their respective 'Heads'. The five 'Head' locations contain the following details:-

0, Internal checksum of program
1, Main entry to program. Pointer to previous program
2, Name of program
3, Checkable address points (zero if no check possible)
4, Previous first-free (FF) and last free (LF) location points.

Shortage of space made it necessary that the RAP workspace should be placed immediately outside the reserved area where it is "semi-protected" by the RAP system i.e. the maximum LF pointer points to one location before the RAP workspace, so it will not be overwritten by a program being read into the store. There is, however, no protection against the program itself causing a location of the RAP workspace to be overwritten.

The RAP has available the first free location, last free location and a pointer to the last program placed. To find a named program, the search routine picks up the RAP pointer and steps through the program in the store until either the required program is found or the RAP is reached again, in which case 'NOPROG' is displayed on the output writer and the RAP then awaits the next typewriter message.

CONTROLS

The RAP consists of several routines (or controls); these are known as 'basic controls'. Similar routines may be required for which there is no room in the reserved area; these are known as 'optional controls' and will be input, as normal programs, by the RAP.

A transfer of control occurs on completion of the first word of a message. The first word may be either an identifier or an integer. Further message parameters will be read as required by the program or

routine to which control has been transferred. 2.2.1.

If the first word of a message is an identifier then it is checked against the list of basic controls and, if present, control is transferred to the appropriate section of the RAP. If it is not present in the list then:

1st) The search routine is used to determine whether the program requested is available. (If not available, 'NOPROG' is displayed).

2nd) If possible, a sum check is made to ensure that the program is still correctly stored. (If found to be stored incorrectly, 'ERRSUM' is displayed).

3rd) Unless incorrectly stored, the program is entered via the main entry in the program head. The symbol is output to show that entry has taken place.

If the first message word is an integer then:-

1st) Negative integers or integers greater than the maximum LF location are rejected (see section c of 'Messages').

2nd) If no parameter is specified control is transferred to the first instruction in the location specified. If the integer is followed by the parameter S then control is transferred to the second instruction of the location. In each case the symbol > is output on transfer of control.

BASIC CONTROLS

1) INPUT

On basic machines the Input routine will read a specially prepared relocatable binary tape from whichever tape reader is selected. The form which the binary takes is described in Appendix 1.

On non-basic machines the input medium could be specified by a parameter of the message. This parameter would call say a film handler by the same name as that given to it by the Symbolic Assembly Program.

Unless a message parameter gives indication to the contrary, the program being input is automatically placed in the first available space. Whether it is placed at the upper or lower end of the free store or in a fixed position is a decision which is taken when the binary tape is produced (see Appendix 1 for further details). When a program has been successfully input by the RAP, the program name is output on the same line as the input message.

On entry to the Input routine the Interrupt Permit Register (see 503 specification) is set so that Manual Interrupt is allowed but all Normal Interrupts are not permitted.

2) CONTINUE

Continues a program from the point at which it was left to obey a Manual Interrupt or Error Interrupt. (see Appendix 2).

The last program to be interrupted in either way will be continued from the point of interruption, even if a tape was being input when the interrupt occurred.

3) RESET

Clears the main store, with the exception of the RAP workspace, and sets the FF and LF pointers to their minimum and maximum respectively. The RAP pointer is set so that the RAP points to itself.

4) FREE STORE

Prints the size of the available store on the output writer as a four digit integer with no suppression of leading zeros.

5) LIST

Prints on the output writer a list of all programs in the store in 'chronological' order beginning with the last to be stored. After an extra line feed the size of the available store is printed as in 4).

If a program is found to be stored incorrectly as a result of sum check failure then the program name is followed by an asterisk, (see Appendix 2 for possible action to be taken).

6) CANCEL

Effectively removes a program or programs from the store by backdating the RAP pointer and the FF and/or LF pointers.

Where a program is named by a message parameter then that program and the programs input after it (i.e. named before it in the List) will be removed.

Before the backdating process is begun a check is made to ensure that the named program is present (to guard against possible spelling mistakes). 'NOPROG' is displayed if the program is absent.

If no parameter is read then only the last program is removed.

OPTIONAL CONTROLS

1) Comment

Copies to the output writer the message which follows until full-stop. Any of the characters on the typewriter may be used in a comment. No action is taken.

2) Read

Reads a number of identifiers and integers to specified locations of the store so that they may be used by a main program.

Precautions

When the MESSAGE button is depressed, control is transferred to the 'precaution' section which tests whether the RAP was being used when the interrupt occurred:-

If NO: Outputs '?' on a new line on the output writer to indicate that Manual Interrupt is accepted and then the RAP awaits a message.

If YES: Tests whether the Input routine was being used.

YES: Manual Interrupt is accepted. This is necessary to prevent the wrong program tape being input.

NO: Completes the operation being done until about to leave the RAP, then accepts Manual Interrupt.

NB: There are three subroutines that a program in the main store may make use of :-

1) Read word (integer or identifier) from typewriter.

2) Print alphanumeric word on output writer.

3) Search main store for named program.

Each of the above subroutines has a special entry for use by programs held in the main store by which a marker is set which holds up the acceptance of Manual Interrupt until exit is about to be made from the subroutine. Any other entry is unprotected.

Messages

When a message is being typed any of the upper case characters may be used with the exception of =, but only the letters may be used from the lower case. These restrictions enable messages to be typed more easily and hence minimise the time consumed.

The uses to which the characters are restricted are as follows:-

a) + and - are used to precede integers (+ is optional)

b) A letter (upper or lower case) must be the first character of an identifier. The rest of the identifier may be numerals and/or upper and lower case letters. (N.B. 'ALGOL' is treated as distinct from 'ALGOL').

c) ; is used as an end of word symbol.

d) '.' denotes the end of the message (and also the end of the last word of that message).

e) All other characters, apart from 'space' which is ignorable, are treated as non-acceptable. On detection of a non-acceptable character, the RAP will output | (non-escapable) followed by capital X. This will form the symbol ✱ The RAP then expects the whole message to be repeated on a new line. This symbol is used to denote a general message error detected by the RAP.

Cancellation of Messages.

If the user wishes to cancel a message before '.' is typed he may type | (non-escapable and non-acceptable). The RAP then recognizes this as non-acceptable and proceeds as in e) above. The complete message must then be re-started.

Release of Manual Interrupt Inhibition

The inhibition on Manual Interrupt is released either on exit from the RAP or on entry to the Input routine.

For all other basic controls the inhibition is retained.

ERROR INDICATIONS

Error Interrupt

When the machine detects an error condition an ERROR INTERRUPT takes place automatically and any further interrupts are inhibited.

Control is immediately transferred to the basic Error Interrupt routine which is held in the RAP. This will output 'ERRINT' followed by an indication of which error digits are present (see Appendix 2).

The main store is then searched to determine whether a more comprehensive error diagnostic program is present. If so, then the inhibition on interrupts is released before transfer to this program. If not, then the RAP outputs 'NOPROG' and awaits a message.

Error Words

When certain 'error' conditions are detected by the RAP the appropriate indications are displayed on the output writer. Each word is output on a new line and the RAP then awaits the next message from the operator.

The 'error' indications output by the RAP are as follows:

ERRSUM : a) A sum check error has been detected on attempting entry to a program.

b) A tape sum check error has been detected while a program is being input.

NOROOM : There is not enough space in the available store for the program being input.

NOPROG : Named program is not available.

UNCHEK : Program is not sum-checkable. This is not strictly an error but serves as an indication that a program will not be protected on entry. It is output on the same line as the program name only on input.

'OPEN' SUBROUTINES

The following RAP subroutines may be used by a program in the main store. Each has a special external entry for this purpose in order that they are protected in case of Manual Interrupt (see 'precautions'). All have a common link.

1) Read Word from Typewriter

When exit is made from this subroutine the word, which has just been read, is held in location 7914. The overflow register is set if the word is an identifier and clear if it is an integer. The character on which exit is made is held in location 7913. When an unacceptable character is typed, ?? is displayed on the output writer. The complete word must then be repeated.

2) Print alphanumeric word on output writer

This subroutine is entered with the word to be printed packed in the accumulator (see Appendix 1 for form of word). The subroutine has two entry points; one for printing the word on a new line and the other for printing it on the same line.

3) Search for named program

This subroutine is entered with the name of the required program held in the accumulator. On exit from the subroutine the address of the first location of the program head is in the accumulator unless the program is not available, in which case the content of the accumulator is zero. This subroutine may also be used to effect an entry other than the standard entry held in the program head.

CONTENTS OF RESERVED AREA

ROUTINE OR BLOCK	LENGTH
Read character from typewriter.	$21\frac{1}{2}$ locations
Read word from typewriter.	$25\frac{1}{2}$ "
Print word on output writer.	$12\frac{1}{2}$ "
Search Main store.	21 "
Continue.	$6\frac{1}{2}$ "
Input.	18 "
Form Internal Check Sum.	7 "
Manual Interrupt.	$24\frac{1}{2}$ "
Transfer Control.	$5\frac{1}{2}$ "
Reset.	$6\frac{1}{2}$ "
Error Interrupt.	10 "
Error word print.	5 "
Cancel.	16 "
Free Store.	$3\frac{1}{2}$ "
List.	$9\frac{1}{2}$ "
Integer print.	$5\frac{1}{2}$ "
Constants.	26 "
t o t a l:	224 locations

This leaves six spare locations.

The space required immediately outside the Reserved Area for the R/P workspace is 26 locations.

28 locations are also reserved for use by Error Interrupt, Manual Interrupt and five control Interrupts.

In the following theoretical example the control messages are underlined. All other words or symbols (apart from comments) have been output by the computer.

?
COMMENT The operator will generally ask for a list of programs in the store and remove those not required.

LIST
CHECK
SAP *
PRINT
5649
CANCEL; SAP.
COMMENT CHECK and SAP have been removed.
IN. ALGOL UNCHEK.
IN. DRS
COMMENT. The program (OBJECT) for translation by ALGOL is now placed under the reader.

ALGOL. OBJECT FREE STORE 4495 - 6600

END
LIST.
OBJECT
DRS
ALGOL
PRINT

2105
OBJECT.
?
COMMENT. The operator has pressed the message button in order to place the correct data tape under the reader.

CONT.

END OF PROGRAM

For a complete list of control messages see Appendix 2.

RELOCATABLE BINARY

This is a special form of binary tape which is designed to be read in under the Standard Input routine. Programs prepared in this form may be placed in the free store either from the first free location upwards (i.e. FF +) or up to the last free location (i.e. LF -). A Translator exists which will convert programs from their TI form to this binary form. Sum-checked binary versions of existing 803 Library programs may also be converted with the Translator. These, however, are fixed programs and are dealt with more fully later in this Appendix.

The decision as to whether a program is placed FF + or LF - is taken when the program is coded since this is determined by the presence of 'action-words' (i.e. instruction-pairs which are immediately obeyed) which precede and succeed the program proper.

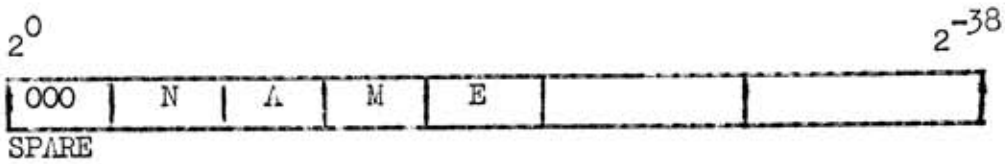
The Standard Input routine simply reads a word, modifies it as necessary, and detects whether it is to be placed in the store or obeyed immediately. The action-words are used to perform the housekeeping functions appropriate to the particular type of program being input. These functions are described in detail later in this Appendix.

The first non-blank character of all tapes of this binary form has the value +80. This character may not be produced on a flexowriter and is therefore used to enable the Standard Input routine to reject tapes of the wrong form.

The program being read in contains the following information:

1) Total space required by the program (including workspace). This is stored temporarily in the first location of the program head.

2) The program name or identifier. This must begin with a letter and is packed as 6-bit characters in the following form:



i.e. The first 3 bits are always left spare and the name is then stored so that any spare bits are at the least significant end of the locationd.

Although the name must begin with a letter, the remaining characters may be taken from these detailed below.

Character	Tape Value	Packed Value
A - Z	33 - 58	1 - 26
a - z	97 - 122	27 - 52
0 - 9	16 - 25	53 - 62

N.B. Only the first six characters of any name or identifier. are preserved, the remainder being ignored on input.

3) Main entry to the program in instruction form.

4) The checkable area pointers (where possible) in order that an internal sum check may be made on repeated entry to the program finally placed. These take the form:

00 (size of checkable area less one) : 00 (1st address of this area),

5) The negative check sum of the tape words.

Binary Representation

Six 8-bit tape characters are used to specify each word. Of these 48 bits only 42 are actually mixed into the accumulator, the parity bit from each of the six characters having been removed. Since the machine word length is 39 bits, we have three spare bits on the 'tape word' which are used to determine how each word is modified by the basic address (.e. first location of program) and whether it should be stored or obeyed.

These three bits are known as the 'control' and are situated at the most significant end of each word. The first two bits indicate which of the four possible modifications is required and the third bit whether the word is to be obeyed or stored (0 for obey, 1 for store).

Action word controls

Value		
0	(00 0) R:R	Add basic address to both halves and obey.
2	(01 0) R:a	Add basic address to 1st half and obey.
4	(10 0) a:R	Add basic address to 2nd half and obey.
6	(11 0) a;a	Obey word as read.

Normal word controls

Value

1 (00 1) R:R Add basic address to halves and store.

3 (01 1) R:a Add basic address to 1st half and store.

5 (10 1) a:R Add basic address to 2nd half and store.

7 (11 1) a:a Place word as read.

Functions performed by Action Words

'Pre' action words These come before the program proper.

1) Clear location which holds the tape check sum.

2) Test whether one free location exists. If not, output 'NOROOM'. If so, set storage count of Standard Input routine to value of first free pointer. (The space (N) required by the program now printed with control 7 is thus automatically placed in the first free location.)

3) Form the basic address modifier a:R. Use N to test whether free store large enough. Output 'NOROOM' if not. Otherwise, reset storage count so that N is overwritten. N is also held in the program head (for later use by the 'post' action words).

4) Form the basic address modifiers R:a and R:R.

'Post' Action words

1) Perform tape sum-check. If incorrect, output 'ERRSUM' and await next message, clear Binary flag.

2) Form internal check-sum (if possible) and exchange with N which is held in first location of program head. Use N to adjust the first free or last free pointer.

3) Adjust the RAP pointer and set the previous RAP pointer in the second location of the program head.

4) Store details of the previous FF and LF pointers in the fifth location of the head.

5) Output name of program on same line as input message.

6) If no sum check is possible, output 'UNCHECK' on the same line as the input message.

7) Go to await next message if Input flag set. Otherwise obey last action words to restore pointers for the 're-read' facility.

FIXED PROGRAMS

The use of 'fixed' programs is not recommended since they have no head and, therefore, no protection apart from a tape sum-check on input. Here the user must revert entirely to the 803 system of program management. Instead of being able to call a program by name he must know the program's position and enter it only by messages of the form "+N."

When a program of this type is input, the RAP pointer, LF and FF pointers are reset as for a clear main store in order that the RAP system be protected.

The Translator will convert programs from their sum-checked binary form, automatically placing the necessary action words before and after the program proper. Where the coded program is in several blocks, action words will be placed to set the storage count to the required value.

OPTIONAL PLACING OF PROGRAMS

The Translator may also be used to prepare programs (from their TI form) which are relocatable but whose position in the store is determined by the first parameter of the input message. These programs have no head and the RAP pointers will again be reset as for a clear main store.

To place a program of this type from location N onwards the user must type 'IN;N.' and enter by means of the instruction 'N.'

SUMMARY OF MESSAGES

The following summary is intended to provide the user with a quick reference table of all available control messages and their effects:-

MESSAGE.	EFFECT.
IN.	Causes the computer to read a specially prepared, relocatable binary program tape.
IN;NAME.	The program being input replaces the program called NAME. This should only be used to restore a corrupt program.
IN;N.	Reads a special binary tape whose position in the store is optional (see Appendix 1). The integer N specifies where the program is to be placed.
NAME.	Transfers control to the program called NAME
N.	Transfer to first instruction in location N.
N;S.	Transfer to second instruction in location N.
CONT.	Continues a program from the point at which it was left to obey a Manual Interrupt.
CONT;ERRINT.	Continues a program from the point at which it was left to obey an Error Interrupt.
LIST.	Prints out in 'chronological' order the names of the programs in the store beginning with the last to be stored. If a program is incorrectly stored its name is followed by an asterisk. The size of the available store is printed after each list.
CANCEL.	Removes last program stored.
CANCEL;NAME.	Beginning with the last program stored, removes all programs as far back as and including the named program by changing the RAP pointers.
RESET.	Clears the main store and resets the RAP pointers.
FREE STORE.	Prints the size of the available store on the output writer.

The error indications which call for corrective action by the operator are as follows:-

CONDITION	ACTION
1) ERRSUM	a) During input of program tape: Read tape in again. No updating of the RAP pointers takes place until the check on the tape sum has been passed. b) On attempting entry to a program: see section 3)
2) NOROOM	Cancel a program or programs until the available store is large enough.
3) * after a listed program name. (i.e. corrupt program)	There are three possible actions which may be taken: a) Where store space is of no account: Read the program to the next available space as usual. Although there will then be two versions of the program in the store the corrupt version will never be found by the search routine since the last program placed is always the first to be examined. b) If the programs input after the now corrupt program (i.e. above it in the list) are no longer required: Type CANCEL; 'NAME'. . The program may then be read in again. c) If the programs above the corrupt program in the list are required: Type IN; 'NAME'. having placed the program tape under the reader. The corrupt program and those 'above' it are temporarily cancelled and then restored. on successful input of the program tape. The new list of programs is automatically output after input of this type. If "ERRSUM" is displayed as a result of tape-suncheck failure then those programs cancelled temporarily are now lost. The user is no worse off than if he had taken action b). He must now read in the necessary tapes again.

N.B. Action c must be used with great care. If a program requiring more store space than that which it replaces has been input by mistake then this will become apparent during the output of the new list of programs. The operator must type 'RESET'. and read in all the necessary programs again.

SIGNIFICANCE OF ERROR DIGITS 2.2.1.

When an error interrupt occurs the number which follows the word 'ERRINT' has the significance shown below:

1. Floating point overflow. The instruction causing overflow is completed before error interrupt takes place.

2. Should not occur.

3. Parity error in the main store. The instruction or autonomous transfer during which the error occurs is completed before error interrupt takes place.

4. Attempted use of a peripheral device which is not available.

5. Attempted impermissible reference to the reserved area. The instruction containing the impermissible reference is not obeyed.