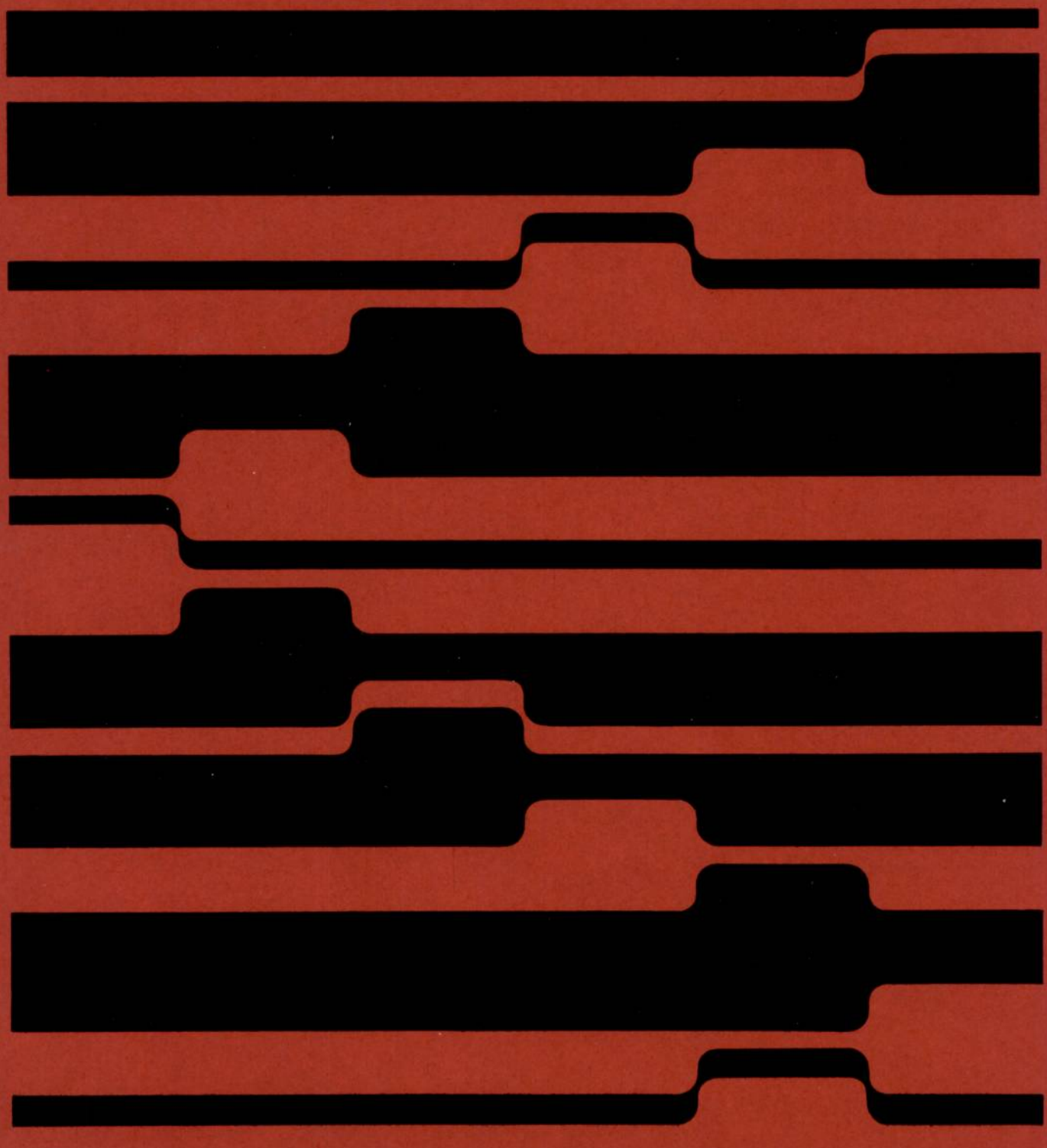
ICL

1904,1905, 1906,1907 Console Operating Manual

1900 Series



ICL International Computers Limited

RECEIVED 19 NOV 1971

User notice

9/6/71

4167

1904/5/6/7 CONSOLE OPERATING (1)

File one copy of this notice with each of the manuals indicated.

CONSOLE TYPEWRITER SWITCH

A console typewriter switch is now fitted as standard on 1904A and 1906A processors and may be fitted as an optional feature on 1904/5/6/7/9 and 1904/5/6/7 E and F processors.

It provides for manual changeover of control, by the operator in the event of console typewriter failure, from an on-line console typewriter to a standby console typewriter. One console typewriter is mounted on the standard desk and the second one on the desk extension.

Operator switches and indicators

All the switches and indicators are mounted on the extension desk trim.

A two position rotary type CHANGEOVER switch is fitted to facilitate the changeover of the signal cables and power from one typewriter to the other, and causes one half of the INDICATOR to light and the other half to extinguish.

The INDICATOR indicates the position of the switch and the state of each typewriter. The top half of the indicator refers to the typewriter mounted on the standard desk, and when lit indicates that typewriter is on-line and no live power is connected to the sockets of the other typewriter. The bottom half of the INDICATOR refers to the typewriter mounted on the extension desk and when lit indicates that typewriter is on-line and no live power is connected to the sockets on the typewriter on the standard desk. THIS INDICATOR IS WHITE.

HOLD (pushbutton/indicator)

This pushbutton/indicator is coloured red and each time it is pushed it changes the HOLD or NO HOLD condition and from lit to unlit or vice-versa. When the switch is illuminated it indicates the HOLD condition, and in this condition both typewriters will be off-line and it will not be possible to input or output messages. Any attempt by Executive to output a message during the HOLD condition will cause the processor to be held up until the HOLD is cleared.

Operating instructions

The following sequence of operations must be carried out by the operator when changing over to a standby console typewriter.

1 Press the (typewriter) HOLD button and see that it is lit.

2 Operate the typewriter CHANGEOVER switch and see that the INDICATOR lights change as previously described.

3 Press the (typewriter) HOLD button and see that the light is extinguished.

Providing the above instructions are adhered to and no attempt is made to use either typewriter as an input device during the changeover then the changeover operation will not affect the operation of the processor.

Note

The HOLD pushbutton must only be used during the sequence of operations described, it must not be left in the HOLD condition longer than is necessary to make the typewriter changeover.

ICL

1904,1905, 1906,1907 Console Operating Manual

1900 Series

Technical Publication 4167

First Edition August 1969

Issued by Technical Publications Service
International Computers Limited
Head Office: ICL House, Putney, London SW15
Produced by ICL Printing Services
at Letchworth, Hertfordshire

Preface

This manual is intended as a guide for operators of the 1904/4A/4E/4F, 1905/5E/5F, 1906 and 1907 central processors.

The manual describes the console typewriter and explains the significance of the various messages which may be input by the operator or output by the processors.

The methods of loading and dumping programs are also explained. It should be noted that loading and operating procedures for the peripheral units are to be found in the *1900 Series Operator's Reference Manual of Peripherals*, which manual should therefore be used in conjunction with this one.

This manual supersedes the first edition of the *1904/5/6/7/9 Console Operating Manual* (T.P. no. 3384) first published in September 1966.

Contents

Chapter 1 Introduction

THE 1900 SERIES

The 1900 Series comprises a single range of processors available in a wide variety of sizes. Operating procedures may be taken as applying equally to all the processors dealt with in this manual unless the contrary is specified.

STANDARD INTERFACE

Most peripherals are connected to the central processor by the ICL 1900 Series standard interface. The standard interface is represented physically by a single multi-pin plug and socket. This forms the sole connection between the peripheral and the processor and is the channel through which control and data impulses are passed to and from the processor.

EXECUTIVE

Executive is a supervisory program held in a protected area of store and thus for all practical purposes may be regarded as a permanent part of the hardware. Each processor is supplied with an Exectuive program specially created to meet the requirements of the particular configuration. The Executive installed in the 1904 and 1905 processors is E4BM, allowing up to four programs to run concurrently in core store. The Executive installed in all the other processors dealt with in this manual is E6BM, allowing up to sixteen programs to run concurrently in core store.

Executive has five main functions:

1 Control of peripheral transfer requests.

2 Control of multiprogramming and subprogramming.

3 Communication with the operator and execution of operator directives.

4 Provision of extracode facilities.

5 Loading and dumping of programs.

Peripheral transfers

Executive supervises the transfer of data between the processor and the peripherals. A peripheral transfer is initiated by a peripheral instruction in the program. Once the transfer has been initiated the control electronics of the peripheral concerned enable the transfer to take place independently of the programs running in the processor.

When a transfer has been terminated Executive ascertains whether or not it has been completed without error. If the transfer has been unsuccessful Executive takes corrective action unless the program contains its own error recovery routines. If further attempts at the transfer are possible, Executive displays a message on the console typewriter either informing the operator that it is making another attempt at the transfer or instructing the operator to reset the input medium and reinitiate the transfer. If the transfer is now completed without error, then as far as the user program is concerned a normal transfer has taken place. When the maximum number of unsuccessful attempts has been made, or when no recovery procedure is possible, Executive displays a message on the console typewriter to this effect. The program that initiated the transfer is suspended awaiting operator action. The operator should then refer to the programmer's instructions. If the program is to be abandoned the operator deletes the program. If the program is to be continued from the point of suspension, the operator inputs the appropriate message as specified in the programmer's instructions.

Multiprogramming and subprogramming

The basic concept of multiprogramming is the ability of Executive to share the time and core store of the processor among a number of programs, thus also making more extensive and practical use of the peripherals available.

A subprogramming facility is available with all multiprogramming Executives whereby parts of a program, called members, can time-share with each other. In many ways subprogramming is similar to multiprogramming, except that whereas programs occupy discrete areas of store protected by hardware lock-outs, subprogramming allows the members of the program to share the program's store but follow their own sequence of instructions. The allocation of processor time among members of a program is controlled by Executive according to the priority given to each by the programmer or operator.

Operator communication

All communications between the operator and the processor take place through a console typewriter via Executive. The typewriter both informs the operator of incidents occurring within the processor, the peripherals and the programs, and is also used by the operator to give instructions to load, activate and delete programs. The instructions to Executive and information from Executive take the form of plain language messages or standard abbreviated forms of these messages. When these messages are typed, they automatically form a permanent record of all communications in the sequence in which they have occurred. Three types of messages are typed:

1 Input messages: A standard set of messages has been devised which is interpreted by Executive and causes it to perform some action (for example, to assign a peripheral to a program), or to display some information required by the operator.

2 Output messages (from Executive): Executive prints out a standard set of messages which may either warn the operator when a peripheral requires attention, or reply to an operator request. Executive also advises the operator of the state of the programs currently being run.

3 Output messages (from a program): A program can print out messages via Executive requesting operator action. These messages are formulated by the individual programmer concerned and do not have a standard format, other than that they must not exceed a text of 40 characters.

Extracodes

The Executive program includes routines, called extracodes, to perform certain 1900 Series program instructions which the hardware is unable to perform. The provision of extracode facilities thus ensures the compatibility of all 1900 Series processors in the performance of 1900 Series program instructions.

Loading and dumping of programs

Executive supervises the loading and dumping of programs. A detailed account of the loading and dumping of programs is given in Chapter 5.

OPERATING SYSTEMS

Many 1900 Series computer installations are likely to be reinforced by the use of one of the ICL 1900 Series operating systems. The most important of these is GEORGE (GEneral ORGanizational Environment).

GEORGE automates many of the operator's functions. Its purpose is to achieve the maximum possible exploitation of hardware by reducing to a minimum the idle machine time during processing. The system improves upon the operating facilities already provided by Executive, thus reducing the need for operator intervention.

GEORGE is able to control the running of a number of programs, either one at a time or simultaneously according to the version in use. GEORGE loads programs, checking their validity (in the same way as it can check all input), and then supervises the runs, controlling such things as the use of peripherals, errors, dumps and restarts, in accordance with the programmer's instructions in the GEORGE input. The operator responds only to specific requests from the system, such as a request for magnetic tapes to be changed.

There are three versions of the system: GEORGE 1, 2 and 3. The facilities available vary from GEORGE 1 with the lowest to GEORGE 3 with the highest degree of sophistication. Any version of GEORGE can be applied to any of the processors dealt with in this manual.

Chapter 2 Operator's controls

INDICATORS AND CONTROLS ON THE CENTRAL PROCESSOR

There is only one indicator visible to the operator on the central processor, and this indicates that the processor is switched on and the real-time clock is operating.

THE CONSOLE TYPEWRITER

The console typewriter (see Figure 1, page 3) provides the means of communication between the operator and the central processor. Instructions to the central processor are entered in plain language on the keyboard, each instruction being typed out on the stationery. Communications to the operator from the processor are also typed out.

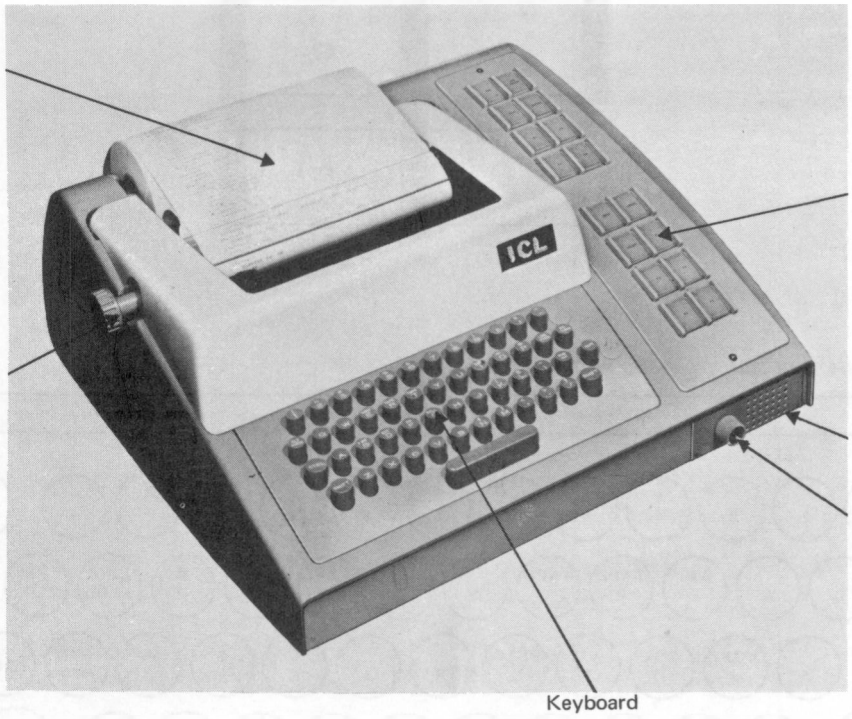


Figure 1 The console typewriter

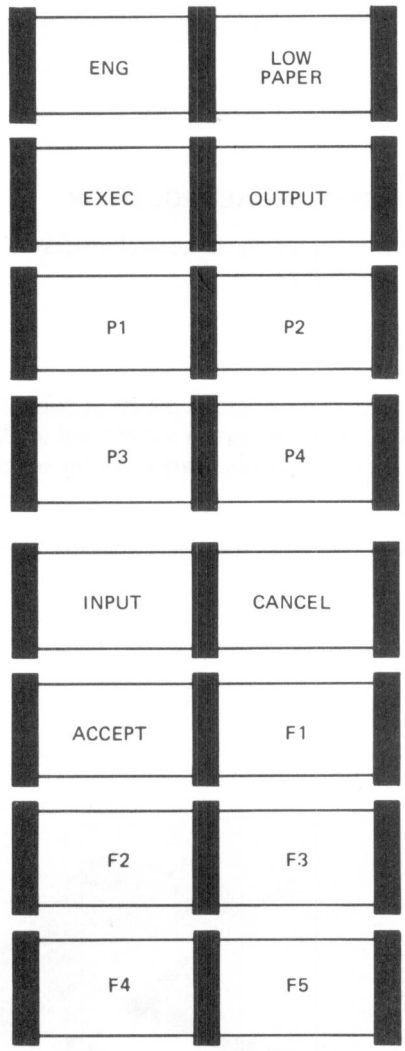


Figure 2 Switch/indicator panel-console typewriter

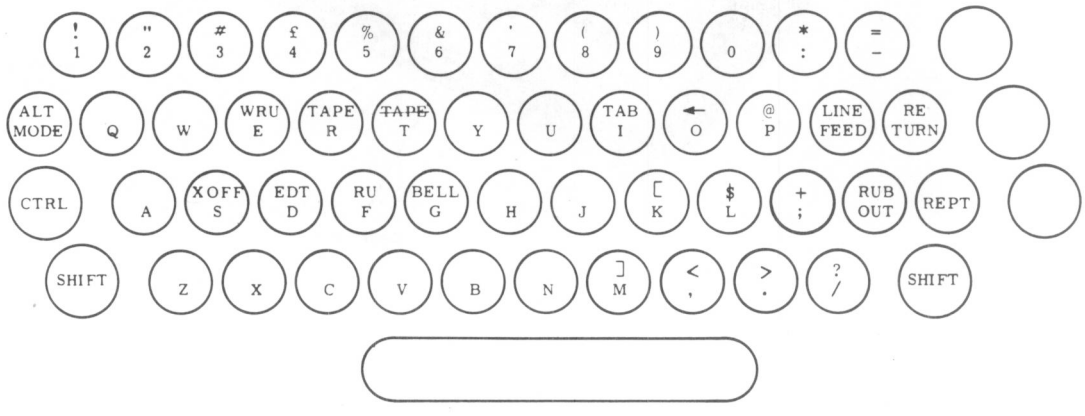


Figure 3 The keyboard of the console typewriter

The maximum speed of printout by the typewriter is 10 characters a second. A mechanical lockout feature exists in the console typewriter which ensures that this is also the maximum rate at which the operator can depress the keys of the typewriter keyboard. Every message, either input or output, occupies a new line, and input messages are automatically indented three spaces so that they may easily be distinguished from the output messages. A continuous roll, 3½ inches (8.89 cm) in diameter and 8½ inches (21.59 cm) wide, of multi-part paper is fitted in the typewriter and provides a permanent record of all messages to or from the central processor. This record is known as the console log.

Physical description

The typewriter is situated on a free-standing desk which may also contain a digital clock displaying the time in 24-hour mode. Figure 1 shows in detail the parts of the typewriter that concern the operator:

1 Continuous roll of stationery.

2 Platen handle.

3 Loudspeaker.

4 Volume control.

5 Control and indicator panel.

6 Keyboard.

CONTINUOUS ROLL OF STATIONERY

A continuous 200 feet roll of multi-part stationery can be fitted, and should be replaced when a red line appears on both edges.

PLATEN HANDLE

This handle, when rotated away from the operator, causes paper to be fed through the typewriter. This is normally used when loading the typewriter, or if the operator wishes to tear off the top copy of the stationery containing a record of the input/output messages for a particular processing run. Movement of the platen is automatic during spacing of each line of print.

LOUDSPEAKER AND VOLUME CONTROL

The small loudspeaker emits a pattern of tones to inform the engineer or operator if certain sections of the computer are working incorrectly. The volume control is turned clockwise to increase the volume and anticlockwise to decrease the volume.

CONTROL AND INDICATOR PANEL

Colour and type	*Title*	*Purpose*
Red lamp	ENG	This lamp glows when the engineer is carrying out tests on the central processor. While this lamp is glowing the operator should not carry out any processing runs without the engineer's permission.
Red lamp	LOW PAPER	This lamp glows continuously when there are only ten feet of paper left in the typewriter. The operator need not renew the roll of stationery until a red line appears on each edge of the stationery. If a console logging punch is in use, this lamp flashes when the punch is running out of paper tape.
Blue lamp	EXEC	This lamp glows when the Executive program is running in the central processor (for example, during the servicing of a peripheral).
White lamp	OUTPUT	This lamp glows when an output message from the central processor is being typed.

Colour and type	*Title*	*Purpose*
Yellow lamp Yellow lamp Yellow lamp Yellow lamp	P1 P2 P3 P4	In the 1904 and 1905 these lamps glow in turn when each of the four programs is running in the central processor. In the 1904A/4E/4F/1905E/5F/1906/1907 processors with E6BM Executive these lamps represent, in binary, the number of the program currently running in the central processor. These lamps may be used to check that one program is not monopolizing the central processor. Note: If the GEORGE operating system is in use the lamps have no meaning.
White illuminated press button	INPUT	This button must be pressed before the operator is able to type a message on the keyboard. The lamp continues to glow until either the ACCEPT or CANCEL button is pressed or until the OUTPUT lamp glows.
Red illuminated press button	CANCEL	This lamp glows when the button is pressed; it is pressed if the operator wishes to cancel a message that he has just typed (for example, because of a typing error).
Green illuminated press button	ACCEPT	The lamp glows when the button is pressed; it is pressed after the operator has typed a message on the keyboard and checked it for accuracy.
Yellow illuminated press button	F1	When this button is pressed a single message is read from a specific input device. The message on the input device is typed out and obeyed. The actual device allocated to the F1 button varies according to the installation. Use of this button also allows messages to be input using input media rather than the keyboard. The message on the input medium is typed out and obeyed. Messages input in this way are not interrupted by output messages and reduce the work of the operator as no typing is required.
Yellow illuminated press buttons	F2 F3 F4 F5	These buttons have the same effect as the F1 button if input peripherals are allocated to each button. Only one peripheral can be allocated to each button. Note: The F1/2/3/4/5 buttons have no function if the GEORGE operating system is in use.

KEYBOARD

Figure 3 illustrates the keyboard of the console typewriter. The keyboard consists of 50 keys and a space bar. 42 keys are used for characters and 8 keys control the printing mechanism. The 42 character keys include 26 alphabetic, 10 numeric and 26 other symbols. With the exception of the two SHIFT keys, the 8 keys controlling the printing mechanism do not normally concern the operator as they are used by the engineer during routine testing. When either of the two SHIFT keys is depressed simultaneously with a character key, the character engraved on the upper part of the key is printed. Certain keys have no upper shift character and an interlock is fitted to prevent a SHIFT key being depressed simultaneously with an invalid key. The keyboard prints out only if the INPUT button has first been pressed. When the space bar is depressed, the type-head is spaced once position to the right.

Note: The key labelled ALT MODE is labelled ESCAPE on typewriters incorporating a logging punch.

Reloading the stationery

The LOW PAPER lamp glows red when there are ten feet of paper left in the typewriter. A further warning is given by the appearance of a red line on each edge of the stationery. At the earliest opportunity the operator should carry out the following procedure.

1 Suspend all programs running in the processor, wait for peripheral activity to subside and press the INPUT button. Although output messages are still possible, this preliminary procedure reduces to a minimum the

risk of messages being lost.

2 Tear off the paper containing the typewriter printout.

3 Swing back the typewriter cover.

4 Pull the Paper Release lever forward.

5 Lift up the remainder of the spool of paper and withdraw the spool support from the spool.

6 Insert the spool support into a new roll of paper.

7 Gently push back the spring-loaded arm at the rear of the typewriter and drop in the new spool, ensuring that the paper loader feeds from the bottom of the spool.

8 Feed the paper under the platen (see Figure 4). It is easier if the leading edge of the paper is folded double before being fed through.

9 Rotate the platen handle until about six inches of paper appears.

10 Line up the paper.

11 Return the Paper Release lever to its original position.

12 Close the typewriter cover.

13 Activate the programs which were suspended in the preliminary procedure described above.

THE CONSOLE LOGGING PUNCH

The console logging punch is an optional facility available with all the processors dealt with in this manual except the 1904A. It is basically a small 10 c.p.s. punch unit complete with feed and electrically-driven take-up mechanism and is housed in the right-hand cupboard of the console table and attached to the cupboard door. The punch is connected to the console typewriter control and records every data or control character sent to the typewriter from the keyboard or the central processor. Punching is in eight tracks, even parity.

The punch is invisible to the operator while the cupboard door is shut, but the operator is informed of the paper low condition, the only condition he needs to rectify, by the flashing of the PAPER LOW indicator on the console typewriter (the PAPER LOW indicator is continuously illuminated if the paper in the typewriter is low).

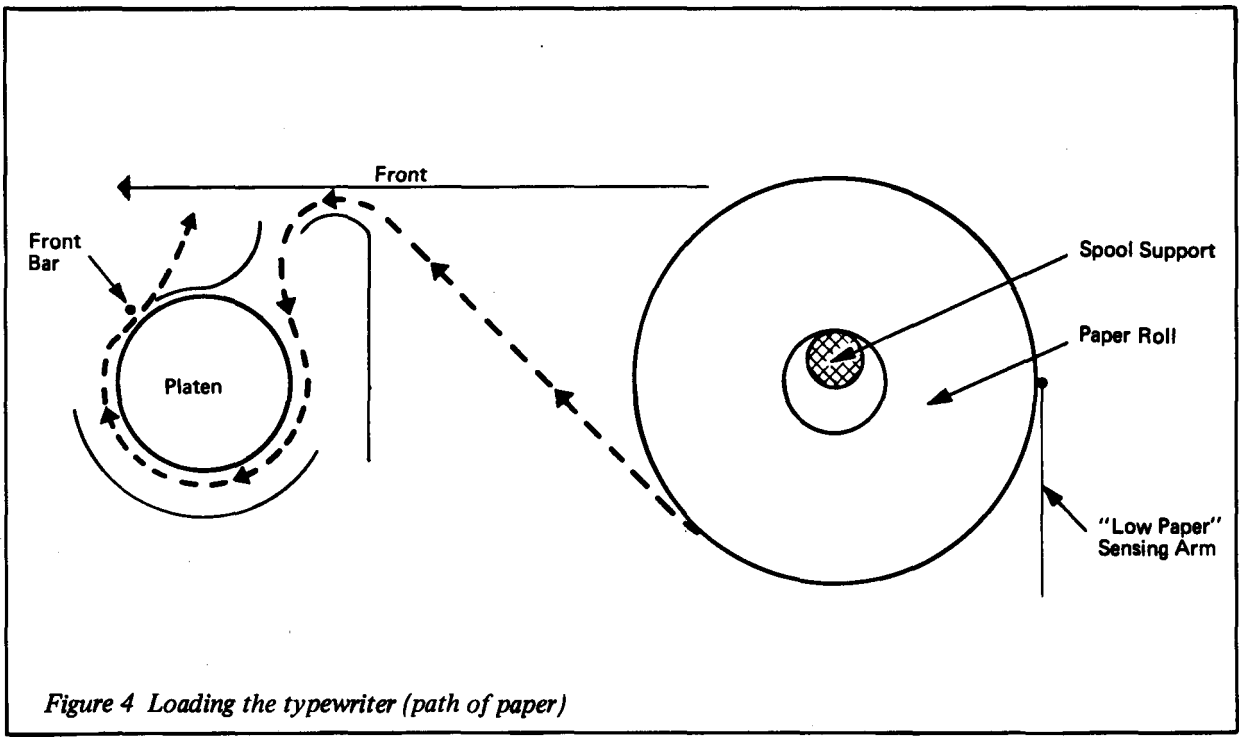


Figure 4 Loading the typewriter (path of paper)

Chapter 3 Operating procedures

SWITCHING ON THE COMPUTER AND LOADING EXECUTIVE

All the processors dealt with in this manual are normally maintained and switched on by a resident engineer. The engineer is also responsible for the loading of Executive.

OPERATOR'S INSTRUCTIONS

In order to run any program the operator must be supplied by the programmer with clear operating instructions prepared especially for each program.

These instructions should be arranged in a sequence which is governed by:

1 The requirements of the particular computer in use.

2 The requirements of the program running in the processor.

3 The input/output requirements of the program.

Only the programmer can know the precise actions which the program demands of the operator at various stages. He must therefore clearly set down:

1 Normal actions required by the operator, such as feeding paper tape, adjusting printing paper, setting program switches and manual typing instructions.

2 Recovery procedures to be taken when foreseeable errors occur.

3 Actions to be taken when unexpected events occur during program running.

The operator has the task of carrying out the programmer's instructions exactly as they are received and as efficiently as possible. He is also expected to correct any mistake which he had made so that the program run is as valid as possible.

METHODS OF INPUTTING MESSAGES

Instructions to Executive from the operator are given by messages of a standard format which may be input from the console typewriter or from the paper tape reader(s) or card reader(s) via the F button facility.

Inputting a message via the F button facility

If input messages are held on cards or paper tape, the operator inputs a message by pressing the appropriate F button (see *Control and indicator panel*, page 5).

Inputting a message via the console typewriter

CONSTITUENTS OF AN INPUT MESSAGE AND CHARACTER SEQUENCE

The basic items comprising an input message, and the sequence to be adhered to, are as follows:

1 The key word. This is always present and must never be preceded by any other character.

2 The member number. Where there are several members to a program, some messages require an indication of the appropriate member on which to act. The indication is given in the form of a numeric character which immediately precedes the #of the program-name. This number should be omitted for a single-member program. The value of this numeric character should be 1, 2, or 3 for the appropriate member.

The character 0 (for member 0) is optional and usually omitted.

3 A symbol # to indicate, when a program name is mentioned in a message, that the following four characters represent the program name.

4 A four-character program name following without space after the # symbol.

5 A series of numbers, separated by one or more non-numeric characters (usually spaces), representing information such as peripheral unit numbers or program entry points. These numbers are assumed to be decimal, unless immediately preceded by the symbol * in which case they are interpreted as octal numbers.

TYPING PROCEDURE

If messages are to be input via the console typewriter, the following procedure is adopted:

1 Press the INPUT button. This causes an entry to Executive.

2 If Executive can accept an input message a new line and three spaces are output and the INPUT button lights up informing the operator that the typewriter is in input mode.

3 When the INPUT button is illuminated the required message may be typed.
Note: The typed message must observe prescribed format rules and must be one of the standard messages described in *Input messages available,* page 11 .

4 When a message has been typed, it should be checked for accuracy.

5 If an error is discovered, the CANCEL button should be pressed. This causes the INPUT light to go out and the message CANCEL to be typed out. The message typed is ignored and the whole procedure (as described in 1 to 4 above) must then be repeated.

6 When the operator is satisfied that the message is correct, the ACCEPT button is pressed.

7 The INPUT light goes out and the message is input to Executive for interpretation.
Note: The pressing of the INPUT button while a message is being output has no effect until the output message is finished. The INPUT button is then illuminated and the operator may type the input message, unless a further message is to be output. In this case the input button is immediately extinguished, CANCEL is typed out and the message output. The operator must then press the INPUT button again before inputting a message.

INTERPRETATION OF MESSAGES BY EXECUTIVE

Acceptable messages

When the procedure described in *Inputting a message via the console typewriter,* page 9, has been carried out, if Executive is satisfied that the message is acceptable, and no other reply is called for, it types out OK on the same line as the message. The instructions contained in the message are then carried out.

Note: Executive acknowledges messages from devices other than the console typewriter by printing on the typewriter the message read from the input device.

Unacceptable messages

If the message is not acceptable to Executive for any reason, regardless of the input device concerned the appropriate error indication is printed out immediately after the message and thereafter the message is ignored. The table in Appendix A gives the reasons for the rejection of messages. The original message should then be examined and another attempt made to input it.

If the typewriter is required by Executive for output while an input message is being typed, Executive may take priority. The input message is cancelled and CANCEL typed out, and the message being typed is ignored. The operator must then carry out again the procedure described in *Inputting a message via the console typewriter,* page 9.

Chapter 4 Console typewriter messages

ARRANGEMENT OF MESSAGES ON THE CONSOLE LOG

On the console log all input messages are automatically indented three spaces. This facility enables an observer to distinguish between input and output messages. It should be noted, however, that messages input on paper tape and cards via the F button facility are not indented. Executive output messages other than replies are indented one space. Output messages from a program are not indented.

INPUT MESSAGES

Representation of input messages

In the description of input messages in this section the following conventions are used:

1 Standard words, or parts of standard words, that are actually typed on the console log are given in upper case in the examples and text (for example, GI, ON).

2 Variables such as program names and unit numbers are given in lower case italics in the examples and text (for example, *#program-name* represents #FRED, #AB10 or any other four-character program name).

3 Any optional item in a message (for example, *storage-location*) or any item which improves the comprehensibility of a message (for example, the complete word GIVE where only the first two characters GI are essential) is enclosed in special parentheses, for example: ⟨*storage-location*⟩ or GI⟨VE⟩.

Example

A typical input message as a sentence reads:

DUMP THE PROGRAM NAMED JACK ON UNIT 6

The message format is:

DUMP #JACK ON 6

The message format can be further abbreviated by typing only the first two characters of the key word and omitting the word ON:

DU #JACK 6

The unit number is an optional item in the message. The format used in describing the message is therefore:

DU⟨MP⟩ *#program-name* ⟨ON⟩ ⟨*unit-number*⟩

Input messages available

The input messages available on the 1904/4A/4E/4F, 1905/5E/5F, 1906 and 1907 central processors are listed below in alphabetical order.

AL⟨TER⟩ *#program-name location* ⟨TO⟩ *new-contents-of-location*

CH⟨ANGE⟩ ⟨OPERATOR'S⟩ *unit-number* ⟨MODE⟩ *mode-number*

DE⟨LETE⟩ *#program-name*

DU⟨MP⟩ *#program-name* ⟨ON⟩ ⟨*unit-number*⟩

FI⟨ND⟩ *#program-name* ⟨*#program-name*⟩ ⟨*unit-number*⟩ ⟨*unit-number*⟩ ⟨*store-allocation*⟩

GI⟨VE⟩ *#program-name* ⟨OPERATOR'S⟩ *unit-number* ⟨AS PROGRAM⟩ *programmer's-number* ⟨MODE⟩ ⟨*mode-number*⟩

GO⟨*member*⟩*#program-name*

GO *#program-name*⟨AT⟩ *store-location*

LO⟨AD⟩ *#program-name* ⟨ON⟩ *unit-number*⟨*unit-number*⟩⟨*unit-number*⟩⟨*store-allocation*⟩

MS *#program-name = message*

OF⟨F⟩ *#program-name bit-number*⟨*bit-number*⟩⟨*bit-number*⟩⟨*bit-number*⟩

ON *#program-name bit-number*⟨*bit-number*⟩⟨*bit-number*⟩⟨*bit-number*⟩

OU⟨TPUT⟩ *#program-name* ⟨WORD⟩ *location* ⟨ON CONSOLE⟩

OU⟨TPUT⟩ *#program-name* ⟨START⟩ *start-address* ⟨AREA⟩*length-of-area-from start-address*⟨ON UNIT⟩ *unit-number*

PR⟨INT PRIORITY LIST⟩

RE⟨VISE PRIORITY⟩ *member#program-name new-priority*

RI⟨GHT⟩ *unit-number*

SP⟨ACE⟩

SU⟨SPEND⟩⟨*member*⟩*#program-name*

TA⟨KE⟩ *#program-name unit-number*

WR⟨ONG⟩ *unit-number*

Description of input messages

The following paragraphs describe the various messages, and give examples of their uses.

AL⟨TER⟩ *#program-name location*⟨TO⟩*new-contents-of-location*

This instruction is used to alter the contents of a specific storage location. The message is obeyed without suspension of the program running in store. An address outside the object program's allocated store in the ALTER instruction causes an error message to be output on the console typewriter.

Example AL #JACK 3742 *35631123

AL is the key word, JACK is the program name, 3742 is the address of the word to be altered and *35631123 is the new content of the word.

CH⟨ANGE⟩⟨OPERATOR'S⟩*unit-number*⟨MODE⟩*mode-number*

This instruction is used to change the mode of operation of a peripheral to that indicated. The peripheral retains the new mode until Executive is next reloaded into the processor or until another CHANGE message is input. If a GIVE message containing a new mode of operation is now entered, the new mode applies only until the file is closed; the deck then reverts to its original state prior to the GIVE message.

Note: This message is applicable only to 1971, 1972 and 1973 magnetic tape units.

Example CH 6 *24

CH is the key word, 6 is the unit number of the tape deck and *24 is an octal number, supplied by the programmer to the operator, which gives details of the new recording or reading mode.

DE⟨LETE⟩ *#program-name*

This message causes Executive to delete the program from the processor. A DELETED message is produced (see

MESSAGES OUTPUT DURING A PROGRAM RUN, page 19). All programs in higher numbered store locations are moved to fill in the space originally occupied by the deleted program. In certain circumstances, the deletion cannot take place while a peripheral allocated to one of these programs is active. If there is a significant delay, a BUSY message is output (see *MESSAGES OUTPUT ON PERIPHERAL OCCURRENCES*, page 21).

Example DE #JACK

DE is the key word and JACK the program name

DU⟨MP⟩ *#program-name* ⟨ON⟩ ⟨*unit-number*⟩

The message has two different meanings depending on whether or not a unit number is specified.

If no peripheral is specified the message causes Executive to search all the currently unassigned tape decks until it locates a scratch tape. It then labels this tape PROGRAM∇*name* and writes the program area in binary format on to it.

The operator may specify either a card or paper tape punch in the message, but not a line printer because the data to be dumped is not in a recognisable printing format. The message causes the program area to be output in binary format to the peripheral whose unit number is specified.

Notes:

1 Before giving a DUMP instruction the operator must first ensure that all peripheral activity has ceased. If the transfers to or from the peripherals are allowed to continue incorrect data may be dumped.

2 This directive must not be used to dump a program on to a specified magnetic tape, that is, the unit number must not refer to a tape deck.

3 For further information on the DUMP directive see *DUMPING THE PROGRAM*, page 24.

Examples DU #JACK

DU is the key word and JACK the program name.

DU #JACK 6

DU is the key word, JACK the program name and 6 the unit number of a card or paper tape punch.

FI⟨ND⟩ *#program-name*⟨*#program-name*⟩⟨*unit-number*⟩⟨*unit-number*⟩⟨*store-allocation*⟩

This message is used to load programs held in magnetic file storage (for example, magnetic tape, E.D.S. or F.D.S.). The first name specified is the name of the program to be loaded. For magnetic tape, if no other name is specified, Executive searches for a file of this name and loads the program contained in it. If a second program name is specified, Executive searches instead for a file of this name and loads from it the first-named program. For other forms of magnetic file storage, Executive automatically loads a search program to carry out the same procedure.

The message can be extended to allow the operator to specify up to three further parameters. These parameters are numbers which may represent peripherals to be assigned to the program and/or the size of store to be allocated to it. Executive assumes that a specified number of 63 or less denotes a peripheral and 64 or more a store allocation. Zero is not allowed as a parameter and direct access peripherals may not be allocated in this way. Only one device of any peripheral type may be allocated, as peripherals are always assigned as programmer's unit zero. Only one field may define store allocation. If a store allocation is specified, it overrides the store request held in the program's request slip. The number of words of store to be allocated may be typed as either a decimal or an octal number. An octal number must be preceded by an asterisk (*).

Note: for further information on the FIND directive see *LOADING THE PROGRAM*, page 23; also see the 1900 Series manuals *Operating Instructions for Standard Software* and *1900 Series Library Specifications*, Part 4, Utility Programs.

GI⟨VE⟩ *#program-name* ⟨OPERATOR'S⟩*unit-number*⟨AS PROGRAM⟩ *programmer's-number*⟨MODE⟩⟨*mode-number*⟩

The allocation of peripherals to a program is often dealt with by Executive when the program is first loaded. A programmer, however, may require that one particular peripheral of a group must be assigned to his program at a particular moment (for example, if a system has two paper tape readers, one of which has been set up to read in non-standard format), and this message enables the operator to specify to Executive which peripheral to assign.

Example GI #JACK 4 0

GI is the key word, JACK is the program name, 4 is the unit number of the peripheral to be allocated, and 0 is the programmer's symbolic number for the peripheral. When the programmer's number is omitted, 0 is assumed.

A second form of this message is available for use with magnetic tape decks, when it is required to change the parity checking mode, gap length or packing density of the tape.

Example GI #JACK 6 0 *24

GI is the key word, JACK is the program name, 6 is the unit number of the magnetic tape deck, 0 is the programmer's symbolic number for the tape deck and *24 is an octal number, supplied by the programmer to the operator, which gives details of the recording or reading mode. The new mode applies only until the file is closed; the deck then reverts to the original state prior to the GIVE message.

GO⟨*member*⟩*#program-name*

If a program or program member is suspended, this message causes the program to restart from the point at which it stopped.

Examples GO #JACK

GO is the key word and JACK the program name.

GO 1#JACK

GO is the key word, 1 is the program member, and JACK is the program name.

GO *#program-name*⟨AT⟩ *store-location*

This message is normally used to activate a program initially. In this case the store location is usually in the range 20 to 29.

If a program stops and cannot be continued, this message causes program member 0 to restart at the storage location specified (for example, to commence an appropriate error recovery procedure).

Notes:

1 This type of GO directive can be used to enter only program member 0 directly. If the stop occurs during the running of another program member, it is suspended until restarted by program member 0.

2 This form of message should not be used when a FIX message has requested the operator's attention on a peripheral and no action has yet been taken.

Example GO #JACK 20

GO is the key word, JACK is the program name, and 20 is the specific location (usually in the range 20 to 29) where the program is to be started.

LO⟨AD⟩ *#program-name* ⟨ON⟩ *unit-number*⟨*unit-number*⟩⟨*unit-number*⟩⟨*store-allocation*⟩

This message causes the program to be read into store from the peripheral whose unit number is specified.

The message may be extended to allow the operator to specify up to three further parameters. These parameters are numbers which may represent peripherals to be assigned to the program and/or the size of store to be allocated to it. Executive assumes that a specified number of 63 or less denotes a peripheral and 64 or more a store allocation. Only one parameter may refer to store allocation and only one device of any peripheral type may be assigned in this way. If a store allocation is specified, it overrides the store request held in the program's request slip. The number of words of store to be allocated may be typed as either a decimal or an octal number. An octal number must be preceded by an asterisk (*). Zero is permitted as the first parameter from an F button message, when the program is to be loaded from the peripheral on which the message is input. Subsequent parameters may not be zero.

Note: This message is not used by an operator to load a program directly from magnetic file storage. The FIND message is used for this purpose.

Example LO #JACK 4 *3600

LO is the key word, JACK is the name of the program to be loaded, 4 is the unit number of the input peripheral and *3600 is the number of words of store required (in octal format).

MS *#program-name = message*

The MS message is a facility available for inputting messages to trusted programs. The operating instructions inform the operator when to type in the message. The message must start after the = sign, and must not exceed 38 characters in length. Spaces are not ignored. The message must not contain either of the symbols * or ↑. If the format of the message following the = sign is unacceptable, Executive types out ERROR M on the console log (see *CONSOLE INPUT ERRORS,* page 25).

OF⟨F⟩ *#program-name bit-number⟨bit-number⟩⟨bit-number⟩⟨bit-number⟩*

ON *#program-name bit-number⟨bit-number⟩⟨bit-number⟩⟨bit-number⟩*

These two messages set a 0 or 1 (respectively) in one or more of the 24 bit positions of word 30, which is reserved for this purpose. Each bit acts as a switch enabling the operator to affect the running of the program: for example, the program might normally print a complete table of results, but if an ON message is typed in at the beginning of the program and the program tests word 30, only a summary of results would be printed. An ON or OFF message is obeyed without the suspension of the program running in store.

Example ON #JACK 9 10 21

ON is the key word, JACK is the program name, and bit numbers 9, 10 and 21 of word 30 are set to 1.

OU⟨TPUT⟩ *#program-name*⟨WORD⟩*location*⟨ON CONSOLE⟩

This message causes the contents of the word specified to be output on the console typewriter. If it is suspected that a program is looping, the operator may output word 8, which contains the address of the next instruction in the program, at frequent intervals to establish whether or not a program returns repeatedly to the same instruction.

Example OU #JACK 4764

OU is the key word, JACK is the program name and 4764 is the address of the word to be output.

OU⟨TPUT⟩ *#program-name*⟨START⟩*start-address*⟨AREA⟩*length-of-area-from-start-address*⟨ON UNIT⟩*unit-number*

This message outputs an area of store to a line printer, card punch or paper tape punch, commencing from the location specified. The message suspends the program until the required area of store has been output; the program is then restarted without operator intervention. A series of zero words to be output on the line printer is represented by a single blank line.

Example OU #JACK 9647 500 7

OU is the key word, JACK is the program name, 9647 is the start address, 500 is the number of words to be output and 7 is the unit number of the peripheral.

PR⟨INT PRIORITY LIST⟩

This message causes Executive to type out on the console typewriter in priority sequence the priority of each program member in the machine. The first numeral typed out is the program member number, which is 0, 1, 2, or 3, or 7 if it is being processed by Executive (for example, during a load or dump). This numeral is followed by the four-character name of the program, which is followed in turn by two characters which denote the priority of the program member. The greater this priority number, the higher it is placed in the priority list. The priority number of program member 0 is followed by the unit numbers of all peripherals currently assigned to that program.

Notes:

1 In the event of two programs having the same priority, Executive usually assigns the higher priority to the program most recently loaded.

2 If there are no programs currently held, the reply NIL is typed out.

Example PR

OJACK99 6 26 28 29

OALPH90 7 24 26

1JACK50

PR is the key word and 0 and 1 are the program member numbers. The fact that there is only one such number in the case of ALPH means that ALPH is a single-member program. JACK and ALPH are the names of the two programs running in the processor. 99, 90 and 50 are the various priorities of the program members (the number is assigned by the programmer but can be changed by a number of methods). 6, 26, 28 and 29, and 7, 24 and 26 are the unit numbers of the peripherals currently assigned to #JACK and #ALPH respectively.

RE⟨VISE PRIORITY⟩ *member#program-name new-priority*

This message revises the priority of the program member and changes its position in the priority list according to its new priority. The new priority is always in decimal format.

Note: In response to a REVISE input message, Executive on a 1904 or 1905 prints out the new priority list. On all other processors dealt with in this manual, Executive prints out and confirms the priority of the member of the program referred to in the REVISE message.

Example RE 1 #JACK 93

RE is the key word, 1#JACK is the name of the program member whose priority is to be changed and 93 is the new priority of the program member.

RI⟨GHT⟩ *unit-number*

This message returns the peripheral whose unit number is specified to Executive's list of peripherals. This message is used only after a WRONG message.

Note: This message is not applicable to direct access devices.

Example RI 26

RI is the key word and 26 the unit number of the peripheral to be returned to Executive's list of peripherals.

SP⟨ACE⟩

This message causes Executive to print out on the console typewriter a list containing the amount of core store (in words) and the type and numbers of peripherals available.

Examples

1 For 1904 and 1905 processors,

```
SP
CORE    12288
TR       2
TP       1
LP       1
CR       2
CP       1
MT       4
```

2 For 1904A/4E/4F, 1905E/5F, 1906 and 1907 processors,

```
SP
CORE    12288
TR       2  4
TP       7
LP      11
CR      14 16
CP      19
MT      22 24 26 27
```

Note: In Example 1 the number of available peripherals of each type is printed out. In Example 2 the unit numbers of the available peripherals of each type are printed out.

SU⟨SPEND⟩⟨*member*⟩*#program-name*

This message causes the program or program member to be suspended until the operator restarts it with a GO message.

Example SU 1#JACK

SU is the key word, 1 is the program member to be suspended and JACK is the program name.

Note: This message may also be used to suspend a program that is under Executive control. The format is:

SU 7*#program-name*

TA⟨KE⟩ *#program-name unit-number*

This message causes the specified peripheral to be removed from the program and makes the peripheral available for general use.

Example TA #JACK 6

TA is the key word, JACK is the name of the program and 6 is the unit number of the peripheral to be

removed from #JACK.

WR⟨ONG⟩ *unit-number*

This message removes from Executive's list of peripherals the peripheral whose unit number is specified, and thus prevents subsequent allocation of the peripheral to a program. This message enables the operator to inform Executive that a peripheral is out of action (for example, when it is under the engineer's control for servicing).

Note: This message is not applicable to direct access devices.

Example WR 26

WR is the key word and 26 the unit number of the peripheral to be removed from Executive's list of peripherals.

OUTPUT MESSAGES

Representation of output messages

In the description of output messages in this section the following conventions are used:

1 Standard words that actually appear on the console log are given in upper case in the examples and text (for example, BUSY, HALTED).

2 Variables such as program names and file names are given in lower case italics (for example, *file-name* represents WORKINGFILEX or any other 12-character file name.

In addition, variables such as program member numbers and unit numbers are represented by single characters as follows:

s represents a program member number.

q represents a unit number

a represents any alphabetic character.

n represents any numeric character.

3 Any optional item such as a message after the word HALTED is enclosed in special parentheses (for example, HALTED :- ⟨*message*⟩).

Description of output messages

The output messages available for the processors dealt with in this manual have been divided for convenience into four categories:

1 Messages output in reply to input messages.

2 Messages output on program loading.

3 Messages output during a program run.

4 Messages output on peripheral occurrences.

MESSAGES OUTPUT IN REPLY TO INPUT MESSAGES

All input messages are acknowledged by an Executive output message which indicates whether the input message is acceptable or not.

OK is typed out on the same line as the input message if the input message has correct format and can be obeyed, except with AL, ON, OF, OU, PR, RE and SP, which type a direct reply.

ERROR is typed out on the same line as the input message if the input message has incorrect format or cannot be obeyed. ERROR is followed by an alphabetic character which indicates the type of error made (see *CONSOLE INPUT ERRORS*, Appendix A).

MESSAGES OUTPUT ON PROGRAM LOADING

The following messages may be output after a program has been loaded.

CORE

CORE *nnnn*

The amount of core store allocated to the program may be output as illustrated above on a 1904 or 1905 processor.

FAULT

LO *#program-name n* ⟨*n*⟩⟨*n*⟩⟨*nnnnn*⟩ FAULT *a*

FI *#program-name* ⟨*#program-name*⟩⟨*n*⟩⟨*n*⟩⟨*nnnnn*⟩ FAULT *a*

If an input message requesting the loading of a program is unacceptable to Executive, the word FAULT is output on the same line as the input message followed by an alphabetic character indicating the reason for the fault (see *PROGRAM LOADING ERRORS*, Appendix B).

HALTED:- LD

s#program-name; HALTED :- LD

This message may be output when program loading is complete. The operator usually activates the program with a GO input message specified in the programmer's instructions.

loading allocation list

CR 0 5

MT 0 26

This list contains one entry for each peripheral allocated in response to the program's request block. Each entry consists of an abbreviation of the device type, the programmer's number and the operator's number.

priority list

0*program-name*50

1*program-name*30

2*program-name*20

This list gives the priority assigned to each member of the program.

program name extension

a#program-name; ⟨*program-name-extension*⟩

a is the letter P unless the program has trusted status. The program name extension is not present for all programs and is sometimes used as an accounting code.

MESSAGES OUTPUT DURING A PROGRAM RUN

The following messages from Executive are output to inform the operator of the state of a particular program currently held in the processor.

ATTACH WPR

s#program-name; UNIT *q* :- ATTACH WPR

The program has requested allocation of the specified tape with a write permit ring and the only unallocated tapes of this name on engaged decks have no write permit ring fitted. The operator should not carry out this instruction unless satisfied that the required tape has been incorrectly loaded.

CLOCKED

s#program-name; CLOCKED :- *nn*

This message indicates millions of processor clock pulses, rounded down to the lowest million, that have been counted during the running of the specified program.

CLOSED

CLOSED: UNIT *q* :- *file-name reel-number generation-number tape-serial-number* REPEATS

This message indicates that the current program has completed its operation on the specified tape deck and that the tape is no longer assigned to that program. The deck concerned is still engaged and unless it is a scratch tape it can be opened by the appropriate instruction from any program.

CORE

s#program-name; CORE *nnnnn*

The program may output the amount of core store allocated to it. This is also output when the amount of core store allocated to a program has changed.

DELETED

#program-name; DELETED :- ⟨message⟩

This message is output as a result of a program instruction for the deletion of the specified program. This message is also output in response to a DELETE input message.

DISPLAY

s#program-name; DISPLAY :- *message*

This message is output as a result of a program instruction to report an event taking place in the specified program.

HALTED

s#program-name; HALTED :- ⟨*message*⟩

This message is output as a result of a program instruction requesting the suspension of the specified program or program member. This message may also be output when program loading is complete.

HALTED :- AWAITING DECK

s#program-name; HALTED :- AWAITING DECK

The specified program cannot attempt dynamic allocation of a magnetic tape because no unallocated deck is engaged.

ILLEGAL

s#program-name; ILLEGAL :- *a nnnn*

This message reports a programming error. *a* is the type of error (see *PROGRAM RUNNING ERRORS*, Appendix C) and *nnnn* is the address of the rejected instruction. On the following line the contents of the address are printed out as an instruction and as an octal number.

LOAD

s#program-name; LOAD *file-name reel-number*

This message is a request from the program to load the specified tape.

REMOVE WPR

s#program-name; UNIT *q* :- REMOVE WPR

The program has requested allocation of a specified tape without a write permit ring and the only unallocated tapes of this name have a write permit ring fitted.

UNLOAD; UNIT *q* :- *file-name reel-number generation-number tape-serial-number n* REPEATS

This message is similar to CLOSED (see above) except that the deck has been disengaged, usually so that the tape can be removed to store.

0*#program-name; #old-program-name*

When a program changes its name the new name is printed out followed by the old name.

MESSAGES OUTPUT ON PERIPHERAL OCCURRENCES

The following messages are output by Executive to report on peripheral occurrences. *s*, the member-number, is usually 0 but when certain Executive actions such as loading and dumping are in progress it is 7.

BUSY

s#program-name; UNIT *q* :- BUSY

The message informs the operator that a requested action, particularly the deletion of a program, cannot take place immediately due to the activity of the specified peripheral.

ERR.

s#program-name ; UNIT *q* :- ERR.

Following a transfer failure on magnetic tape the maximum number of attempts has been made to transfer the data.

FAIL

s#program-name ; UNIT *q* :- FAIL

Following a parity failure the maximum number of attempts has been made to transfer the data.

FAIL*n*

The interpretation of this message depends on whether magnetic tape or E.D.S. is in use:

s#program-name; UNIT *q* :- FAIL*n*

This message indicates a transfer failure on the specified magnetic tape deck. The reason for the failure is indicated by the value of *n:*

n = 0 The maximum number of attempts has been made on the block.

n = 1 Executive has probably lost its place on the tape (due to block splitting).

n = 2 The program has attempted to read a blank tape.

n = 3 A major failure has occurred due to one of the following:

1 An attempt to read back beyond the beginning of the tape.

2 An attempt to write with the write permit ring absent (applicable only to tape decks with non-standard interface, for example, the 1974 magnetic tape system).

3 The unit being put off-line after being assigned to the program.

s#program-name; UNIT *q* :- FAIL*n* ⟨BN **nnnnnn*⟩⟨SYSFAIL⟩

This message indicates a transfer failure on the specified E.D.S. transport. It is not displayed by Executives earlier than Mark IV/2.

BN **nnnnnn* is the address in octal format of a block on the E.D.S. file. The significance of the address is determined by the value of the number *n* after the word FAIL. No address is output if the value of *n* is 3.

n = 0 The address output is that of a block which has been read unsuccessfully and no replacement block has been found in the flaw area.

n = 1 The address output is that of a replacement block found by Executive in the flaw area but from which it has made six unsuccessful attempts to read.

n = 2 The address output is that of a block which Executive has unsuccessfully attempted to erase.

n = 3 A failure in the hardware has occurred. This message is output if the E.D.S. transport unexpectedly becomes inoperable while transfers are pending or in progress, or if the control unit rejects control codes and qualifiers when it should accept them.

The word SYSFAIL may follow any of the above FAIL messages. It indicates that the transfer that has failed was a systems transfer carried out by Executive to either the systems control area or the auxiliary control area's flaw index. The first case is potentially the more serious as the control areas of all the discs on-line at the time may be affected. In this case the value of the program member number *s* is 7. When a SYSFAIL message is output the appropriate E.D.S. transport is automatically disconnected and the program concerned suspended. The operator should draw the attention of the engineer to the circumstances.

FIX

s#program-name; UNIT *q* :- FIX

This message requests the operator's attention on the specified peripheral (for example, if it has been accidentally disengaged).

FLAW

s#program-name; UNIT *q* :- FLAW BN **nnnnnn*

This message indicates that the E.D.S. block whose address has been output has been erased by Executive and a replacement block created in the flaw area. This message is not displayed by Executives earlier than Mark IV/2.

FREE

s#program-name; UNIT *q* :- FREE

This message is output as a result of a program instruction requesting the release of the specified peripheral from the program's allocation. It is also output if a magnetic tape deck assigned to the program becomes operable after previously going inoperable.

R&FX

s#program-name; UNIT *q* :- R&FX

This message requests that the operator fix the specified peripheral and reposition the medium on it.

USED AS

s#program-name; UNIT *q* :- USED AS *n* ⟨MODE**n*⟩

This message is output as a result of an instruction requesting the allocation of the specified peripheral as the program's unit *n* of the appropriate type. When 1971, 1972 or 1973 magnetic tape decks are allocated the message is extended with MODE**n* specifying the mode of the tape deck.

Chapter 5 Loading and Dumping programs

LOADING THE PROGRAM

Generally the loading of a program is operator-initiated though under certain circumstances it may be initiated by the program.

Program-initiated loading

There are two occasions when program-initiated loading may take place:

1 During the running of a program, more program may be read in from a peripheral.

2 When a program deletes itself by means of a 160/2 (DELTY) program instruction and types a LOAD message on the console typewriter, Executive obeys the message output by the program as if it has been typed by the operator, thus enabling one program to initiate the loading of another (see *ACTION OF THE FIND DIRECTIVE WITH MAGNETIC TAPE*, page 23).

Operator-initiated loading

There are two operator directives which can initiate program loading:

1 LO⟨AD⟩ (see LOAD message, page 20).

This directive causes Executive to load a program from the specified device, which is normally a paper tape reader or card reader.

2 FI⟨ND⟩ (see FIND message, page 13).

An installation usually has its library of programs stored on magnetic tape or direct access devices. If this is the case the normal method of loading a program is to use the directive FIND, quoting the name of the program and whatever other parameters are required.

ACTION OF THE FIND DIRECTIVE WITH MAGNETIC TAPE

In response to FI#JACK Executive searches all the currently unassigned tape decks for a tape with the file name PROGRAM ∇ JACK and on finding it loads from it the program #JACK, which is the first program on tape. Further action is then determined by the program loaded. If Executive does not find a file called PROGRAM ∇ JACK, it instructs the operator to load one with the message LOAD PROGRAM JACK and deletes #JACK.

In response to FI #JACK #FRED (where #FRED is a search program such as #TAPE) Executive searches all the currently unassigned tape decks for a tape with the file name PROGRAM ∇ FRED. If no such tape is found the message LOAD PROGRAM FRED is output instructing the operator to load the required tape and #FRED is deleted. If the tape is found Executive loads the first program from it, which must be #FRED, and passes to it the name #JACK and whatever parameters were specified in the FIND directive. #FRED then searches the file PROGRAM∇ FRED, locates #JACK on it and outputs the message 0#FRED; DELETED:- LO #JACK *n* (where *n* is the unit number of the tape deck containing #JACK) followed by the parameters specified in the FIND directive. Executive then interprets the message LO #JACK *n* as if it has been typed in by the operator and loads #JACK into store from the specified tape deck. Depending on the entry block of #JACK the program then either proceeds without operator intervention or the message O#JACK; HALTED :- LD is output. The operator then activates #JACK with a GO message specified in the programmer's instructions.

ACTION OF THE FIND DIRECTIVE WITH DIRECT ACCESS DEVICES

In response to any FIND directive Executive automatically loads a search program, which must previously have been set up on the particular device holding the file, to search for the required program on a particular type of direct access device (for example, E.D.S. or F.D.S.). In response to FI #JACK, the search program locates a file called PROGRAM ∇ JACK and loads from it the program #JACK, which is the only program held on that file. In response to FI #JACK #FRED, the search program locates a file called PROGRAM FRED and loads the program #JACK from the subfile called PROGRAM ∇ JACK.

The action taken if either the file or subfile PROGRAM ∇ JACK is not found depends on the search program concerned: for example, some search programs initiate a search on media other than that for which they are responsible.

Program loading errors

Appendix B gives a full list of the error messages which may be displayed during the loading of a program.

DUMPING THE PROGRAM

The dumping of a program is the process of preserving on magnetic tape, cards or paper tape a binary representation of the state of a program and the peripherals allocated to it at the time, if any.

The main purpose of a program dump is to provide a permanent binary representation of a program after compilation. Dumping the program at intervals during a long processing run also enables restart points to be established and thus obviates the need to return to the beginning of a program when a serious error is discovered.

The dumping of a program can be initiated by the program or the operator.

Program-initiated dumping

The programmer ensures that all peripheral activity relating to his program has ceased before the appropriate dump instruction is issued in the program. This instruction causes all members of the program to be suspended and the complete program area to be output on the peripheral specified in the program. On completion of the dumping the program continues with the instruction following the dump instruction without operator intervention.

Operator-initiated dumping

The operator may initiate the dumping of a program by inputting a DUMP message (see page). Before doing so the operator should suspend all members of the program and ensure that all peripheral activity has ceased, otherwise incorrect data may be output.

If a particular device is specified by the operator it may only be a card or paper tape punch and must not be a device currently assigned to another program.

If no device is specified by the operator Executive locates an unassigned scratch tape on which to dump the program. It labels this tape PROGRAM ∇ *name* and dumps the program on to it in binary format. If Executive is unable to locate a suitable tape a message is output on the console typewriter instructing the operator to load a scratch tape.

If the operator is required to restart the program from the dump point, he does so by inputting the appropriate message on the console typewriter: LO *#name* if the dump is on cards or paper tape or FI *#name* if the dump is on magnetic tape.

Appendix A Console input errors

ERROR	
A	Program or member not in machine.
B	Incorrect data after message. Recognised message not permitted on machine. Entrusted program is already under care. Qualifier of CH is too high.
C	No such message. Trusted program already has a program under control. No 160/7 order.
D	Program already in Executive action.
E	Program name already in use. Invalid name.
F	Impermissible message from program on deletion. Impermissible message from trusted program.
G	Maximum number of programs already in store.
H	Not a load, output or dump device.
I	–
J	Peripheral not available.
K	Insufficient core available for loading.
L	Peripheral busy.
M	Program already has a peripheral of this type and unit number allocated to it. Message too long. Datum or limit outside reservations.
N	Output address beyond program's limit. Message area outside reservations.

Appendix B Program loading errors

FAULT	
A	Not a binary block. Invalid length.
B	Block destination beyond program's limit.
C	Block checksum fail.
D	Not a request slip when one was expected.
E	A request slip when not expected. More than one supplementary request block.
F	Block count fail.
G	Program member already in. No program member 0.
H	–
I	–
J	Peripheral not available.
K	Insufficient core for loading.
L	Name on request slip not name in message.
M	–
N	–

Appendix C Program running errors

ILLEGAL

A	Extracode not recognised. The program has attempted to attain a higher trusted status. Program has insufficient trusted status to perform action requested.
B	Address outside program reservations. Address greater than 2 for a 167 program instruction.
C	Peripheral not allocated. Peripheral not fully allocated. Non-existent line (to communications multiplexor). Trusted program has no program under control. Extracode from wrong level.
D	No request slip when one was expected. No request slip in machine when a 167 program instruction was issued relating to it. Pocket select needed (Universal Document Transport or cheque sorter). Program under control already in Executive action. Invalid order for this device.
E	Invalid name. Trusted program trying to create an extra request slip. Read or write needed (U.D.T. or cheque sorter).
F	Invalid order. Pocket select out of range (cheque sorter). Order from non-zero member of multi-trusted program. Attempt to activate a program member which is already active. Transfer too large for device. Bucket size invalid.
G	No program under control. Member not in machine. Member already in machine. No member 0. File of this unit number already open. Not subprogramming level.
H	One SHOT of invalid order.
I	Write data invalid.
J	Peripheral not available.
K	Zero transfer. Zero core requested. Relative datum zero.
L	Invalid mode. A 165 program instruction with operand invalid. A 165 program instruction with accumulator invalid. File protected.
M	Invalid mode. Contract on zero file. All attempt to suspend on a peripheral in direct response mode. Illegal operation on magnetic tape. Unit number in use.

Appendix D A sample console log

```
14/30/00
   FI #XPLG #TAPE  OK
 P#TAPE;
MT 0  31
0TAPE40
CORE    512
 0#TAPE; DELETED :- LO #XPLG 31 OK
 P#XPLG;
MT 0  31
0XPLG93
CORE  11200
 0#XPLG; HALTED :- LD
  GO #XP    CANCEL
 7#SYMM; UNIT 33 :- FAIL
 7#SYMM; HALTED :-
14/31/00
   WR 33  OK
   GO #XPLG 21  OK
 0#XPLG; UNIT 15 :- USED AS  0
 0#XPLG; UNIT 12 :- USED AS  0
UNLOAD; UNIT 26 :- TESTFILE0000    0 :- *00004107  6 REPEATS
   PR
7SYMM50
0XPLG93  12  15
0TEST40  18
0SAND01   6  17
   DE #SYMM  OK
 0#SYMM; CLOCKED :- 12
 0#SYMM; DELETED :-
14/32/00
 0#XPLG; LOAD SCRATCH TAPE
   SP
CORE   9664
TR     1
TP     2
LP     1
CR     1
CP     2
MT     4
 0#XPLG; UNIT 29 :- USED AS  1
   LO #DERB  4  OK
 7#DERB; UNIT  4 :- R&FX
 P#DERB;72495
TR 0   4
LP 0  16
0DERB01
CORE   2336
14/33/00
 0#DERB; HALTED :-
 0#XPLG; LOAD SCRATCH TAPE
 0#SAND; UNIT  6 :- FREE
```

```
 0#SAND; UNIT 17 :- FREE
 0#SAND; CLOCKED :- 14
 0#SAND; DELETED :-
   GI #XPLG 33    ERROR J
   RI 33  OK
   GI #XPLG 33  OK
   GO #XPLG  OK
14/34/00
 0#XPLG; UNIT 33 :- USED AS  2
AL #DERB   666  *01010301
 020 0 1 *0301 *01010301
AL #DERB   889  *04030261
 100 0 3 *0261 *04030261
AL #DERB   990  *00200132
 040 0 0 *0132 *00200132
   GO #DERB 20
 0#DERB; UNIT  6 :- USED AS  0
 0#TEST; CLOCKED :- 14
 0#TEST; DELETED :-
14/35/00
 0#XPLG; DISPLAY :- END OF #ARCH
 0#XPLG; DISPLAY :- COMPILED #ARCH        000567
 0#XPLG; UNIT 15 :- FREE
 0#XPLG; UNIT 12 :- FREE
CLOSED; UNIT 29 :- SCRATCH TAPE *00077011
CLOSED; UNIT 33 :- PROGRAM TAPE0000    0 :- *00077117
 0#XPLG; HALTED :- END OF BATCH
   DE #XPLG  OK
 0#XPLG; DELETED :-
CLOSED; UNIT 31 :- PROGRAM TAPE0000  328 :- *00014245
14/36/00
   FI #ARCH #TAPE  OK
 P#TAPE;
MT 0  33
0TAPE40
CORE    512
 0#TAPE; DELETED :- LO #ARCH 33  OK
 P#ARCH;76009
MT 0  33
0ARCH01
CORE   2656
 0#ARCH; HALTED :- LD
   DU #ARCH  7  OK
 7#DERB; ILLEGAL :- C 2008
 157 0 0 *1542 *06741542
   OU #DERB 0 2500 14  OK
14/37/00
14/38/00
 7#ARCH; UNIT  7 :- FREE
   GO #ARCH 20  OK
 0#ARCH; UNIT  9 :- USED AS  0
 7#DERB; UNIT 14 :- FREE
14/39/00
   DE #DERB  OK
 0#DERB; CLOCKED :- 10
 0#DERB; DELETED :-
14/40/00
```

EXPLANATION OF EVENTS

14/30/00

At the moment there are three programs running in the processor and the operator now loads a fourth. #XPLG is a PLAN compiler held on a library tape. The source program it is to compile is #ARCH which is held on cards already loaded into the card reader unit number 12. The deck holding #XPLG is put on-line and the message FI #XPLG #TAPE is typed on the console typewriter. #TAPE is a search program which is conventionally held as the first program on all library tapes. Executive searches all the unallocated tape decks on-line at the moment for a deck called PROGRAM ∇ TAPE and loads from it the first program #TAPE. This program then searches all the programs held on that deck for #XPLG. On finding it #TAPE deletes itself and outputs on the console log LO #XPLG followed by the unit number of the tape deck, which in this case is 31. Executive then obeys this message as if it had been typed by the operator. When #XPLG has been loaded and is ready to continue the message O#XPLG; HALTED:- LD is output. The operator now attempts to input the message GO #XPLG 21 to activate the program but Executive interrupts the message to report a failure in a transfer initiated by #SYMM on the tape deck unit number 33.

14/31/00

The operator puts deck 33 off-line with a WR message and then activates #XPLG. #XPLG reports that it is using the line printer unit number 15 and the card reader unit number 12. An UNLOAD message is output by #TEST requiring the operator to unload the tape TESTFILE and remove it to store. The operator checks on the number of programs held in the processor by inputting a PR message (#SYMM is placed at the head of the list because it is currently under Executive control) and on finding that four programs are running in the processor, which is the maximum on a processor with E4BM Executive, he deletes #SYMM, which has had a transfer failure, to enable another program to be loaded.

14/32/00

A message is output from #XPLG requiring a scratch tape to be put on-line. Before loading the new program the operator checks on the core store and peripherals available. #XPLG reports that the scratch tape on deck 29 has been assigned to it as its own unit 1 (unit 0 being the library tape from which #XPLG was loaded), and the operator loads the new program #DERB on the tape reader unit number 4. During the loading of this program a message is output requiring the operator's attention on the tape reader. The trouble is cleared and the program is loaded satisfactorily. The number 72495 output after the program name is an accounting code which will be used by a log analysis program for charging purposes.

14/33/00

#DERB reports that it has been loaded satisfactorily and #XPLG outputs a message requiring a second scratch tape. #SAND reports the release of the peripherals allocated to it and deletes itself. The operator tries to assign to #XPLG the scratch tape used by the deleted #SYMM, but forgets that he has put it off-line with a WR message. ERROR J is output to inform the operator that the peripheral is not available. The operator then inputs a RI message, assigns the deck to #XPLG and activates the program.

14/34/00

#XPLG reports that it is using deck 33 as its own unit 2. Before activating #DERB the operator inputs three AL messages using the F button facility (this is why the messages are not indented). The operator then activates #DERB, which reports that it is using the tape reader unit number 6. #TEST deletes itself after a satisfactory run.

14/35/00

#XPLG reports that it has compiled #ARCH and dumped it on unit 33 headed by #TAPE (this is why the deck is referred to as PROGRAM TAPE instead of PROGRAM ARCH). The operator then deletes #XPLG. The deck holding #XPLG is automatically closed so that it can not be accessed by another program.

14/36/00

The operator loads #ARCH into store in the same way as #XPLG previously. The programmer has asked for a binary dump of his program, so before activating #ARCH the operator dumps the program on the tape punch unit number 7. #DERB then goes illegal on a 157 (PERI) instruction and the contents of the illegal instruction are printed out on the following line as an instruction and as an octal number. The operator then causes the whole program area to be output on a line printer, as required in the programmer's instructions.

14/37/00

14/38/00

#ARCH reports the completion of the dump and the operator activates the program. The data pack for #ARCH has been set up on the card reader unit number 9 and #ARCH reports that this peripheral has been assigned to it. #DERB reports that the program area has been output and the line printer has been released.

14/39/00

The operator deletes #DERB.

Amendment record

Should any additions or amendments to this Manual be necessary, they will be issued in the form of Amendment Lists.
When such changes have been incorporated, an appropriate entry should be made in the table below and the pink instruction sheet(s) should be filed behind this page.

List number	Item number
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to
	to