

Oxford University Computing Laboratory

Computer Manuals

**ICL 1900 PLAN programming and BASIC software
Unified direct access standards utilities
3rd Edition 1979**

4444

PLAN PROGRAMMING & BASIC SOFTWARE

Unified Direct Access Standards Utilities

3rd Ed. 1979

1900

ERRATA SHEET

TP4444 Unified Direct Access Standard Utilities

Page 39, line -16. Sentence should read:

The program may increase its store size up to 32K words.

Page 49, line -11. Add '2' to end of sentence.

Page 88, line -13. Under heading *Range of values*, insert '106'
after page.

Page 92, line -24. Under heading *Range of values*, insert '106'
after page.

Page 129, line -21. page 20 should read page 2.



PUBLICATION (NOTICE NO.)

4444

UNIFIED DIRECT ACCESS STANDARDS UTILITIES (1)

20/6/79

File one copy of this
notice with each of the
publications indicated

The following will be available on MLT38:

File Organization Subroutines
XPJ1/6M
XPJC/6H
XPJR/1B
XREJ/3C

FILE ORGANIZATION SOFTWARE

The following routines will be issued on the S-RS Subroutine Group for MLT38:

HXPJZCNTRL2/1F
HXPJZCNTRL3/1C
HRAICM/1B
HXPJZIMT/1C
HXPJZPIRMT0/1B
HXPJZPIRMT/1B
HRAIDA/3B
HXPJZPIRDAO/1B
HXPJZWRBKT/1B
HXPJZDARTNS/1B
HXPJZOPF/1B
HRAOPF/4A
HRAIND/1B

With the issue of these routines the errors reported in 1900 Series Housekeeping and Utilities Routines Software Notice items 247/2, 249/1, 250/3, 254/1, 257/1 (parts (i) to (vi)), 260/2, 260/3 and 263/1 are corrected.

The following errors have also been corrected:

- 1 If the message DISPLAY KEYERROR ON DA occurred for a non-serial DA input file, and if output was not to direct access, then HALT KX did not always occur
- 2 A mode-compatible PIR program processing a DA output file would erroneously output DISPLAY P5

XPJ1/6M - Write Identifiers

This new version of the program contains the following changes:

- 1 6208 words of store are required for running (7232 words if initializing EDS 200)
- 2 The line printer output now reports ranges of flawed blocks, with a total number of flawed blocks output at the end

XPJC/1H - DA file allocator

With the issue of XPJC/1H the error reported in 1900 Series Housekeeping and Utility Routines Software Notice item 247/1, is corrected.

XPJR/1B - DA file analysis

With the issue of XPJR/1B the errors reported in 1900 Series Housekeeping and Utility Routines Software Notice item 249/2, are corrected.

The following errors have also been corrected:

- 1 Input of a file in ICL composite file format caused the program to go illegal. A check is now made for input of such a file and an error message output (see below)
- 2 When analysing multi-file area sequential files with a second level overflow area, any file areas holding second level overflow, that followed the file area holding the end of file bucket, were not analysed and subsequently reported on. This caused the Final total to be short by the number of buckets in those file areas.

Changes to the specification

With the issue of XPJR/1B the following changes should be made to the specification of XPJR.

- 1 Page 69, Hardware Requirement

Amend line 1 to read

6528 words of store

- 2 Page 78, line -1

Add the following error message to the FORMAT 1 category of line printer messages

6A IS COMPOSITE FORMAT This message is output if a file to ICL composite formats is supplied for analysis. The run will continue with the next analysis

In addition, the following correction should be made

On page 74, lines 9 to 12 under the heading *Range of values* should read:

Note: If this is not specified, default values of zero will be output in the breakdown analysis lines

XREJ/3C - Create DA file from punched card data

With the issue of XREJ/3C the error reported in 1900 Series Housekeeping and Utility Routines Software Notice item 250/2, is corrected.



PUBLICATION (NOTICE NO.)

4444

UNIFIED DIRECT ACCESS STANDARDS UTILITIES (2)

18/1/79

File one copy of this
notice with each of the
publications indicated

This user notice describes enhancements to XPJW/2B, and lists some corrections to the manual.

XPJW/2B: FILECOPY, DUMP AND RESTORE

XPJW/2B issued in January 1979 on MLT37, incorporates the following enhancements:

- 1 DUMP ALL FILES ON A STORAGE UNIT (FUNCTIONS 3 AND 5) The option is provided of continuing the dump when the message NO FILES DUMPED (message 16 on page 114) is output.

The program outputs a new console message DISPLAY NU immediately before the HALT ZZ associated with the output of the message NO FILES DUMPED. The line printer is released from the program as well.

To continue, the line printer should be reassigned and GO #XPJW 22 input. The program then dumps all the files that have passed the initial checks. Note: This may mean files that should have been excluded are dumped owing to mis-specification in the type 5 parameters

- 2 CHECKING OF PRE-ALLOCATED DIRECT ACCESS FILES (FUNCTIONS 1, 6, 7 AND 8) The program checks that a pre-allocated file used as an output file for function 1 or for a file restore by functions 6, 7 and 8 has the same bucket size, bucket header length, keylength/position, start bucket for second level overflow (as applicable), and type of file organization, as that for the file to be copied (function 1) or the file about to be restored (functions 6, 7 and 8). (Previously only the bucket sizes were checked for equality.)

If any of these checks fail, the following new line printer error message is output and the file ignored:

**** *fn fgn* FILE DETAILS UNEQUAL ERROR *abcde*

where

- a* = 1 indicates bucket sizes unequal
- b* = 2 indicates keylengths/positions are different
- c* = 3 indicates bucket header lengths unequal
- d* = 4 indicates types of file organization are different
- e* = 5 indicates start bucket numbers of second level overflow unequal

For example: if *abcde* is printed as *v2vv5*, both the keylengths/positions and the start bucket numbers of second level overflow are at variance but the bucket sizes, bucket header lengths and types of file organization agree

The line printer error message BUCKET SIZES UNEQUAL (message 3 on page 113) no longer occurs

- 3 REPORTING ZERO LENGTH FILES (FUNCTIONS 1, 2, 3, 4 AND 5) If an attempt to copy a zero length file is made (function 1) or a zero length file is encountered when the user is dumping all the files on a cartridge (functions 3 and 5), the following new warning line printer message is output and the file ignored:
- ** *fn fgn* ZERO LENGTH FILE FOUND
- If an attempt to dump a specified file of zero length is made (functions 2 and 4), the following new line printer error message is output and the program will HALT ZZ:
- **** *fn fgn* ZERO LENGTH FILE FOUND
- 4 The message CANNOT CREATE FILE AREA *x* ERROR *n* (message 4 on page 113) has been enhanced to the format:
- **** *fn fgn* CANNOT CREATE FILE AREA *x* ERROR *n* {SPEC
 ANY }
- where:
- x* is the number of the file area within the file
- SPEC indicates an attempt to create in specified area
- ANY indicates an attempt to create in any area
- n* is a number in the range 0 to 5 and has the following meanings:
- 0 (ANY or SPEC) System Control Area full
 - 1 (SPEC) Part or whole of the area assigned either to a permanent file or temporarily to a scratch file
 - (ANY) No space available
 - 2 Either, for function 1 (ANY or SPEC) the file areas for the input and output file are specified as being on units with the same serial number, or (SPEC) area would create too many fences
 - 3 (ANY or SPEC) Insufficient Executive core store for holding further file area descriptions
 - 4 (SPEC) Invalid area
 - 5 (ANY or SPEC) The first bucket of the first file area to be created does not immediately follow the last bucket on the file
- If this message refers to a file to be dumped or restored, this file is omitted
- 5 ADDITIONAL TAPE WRITE TRANSFER CHECK (FUNCTIONS 2, 3, 4 AND 5) The program outputs a new console message HALT TF if the reply from Executive shows that the number of words written to tape is less than the number expected. The run should be abandoned.
- Also for functions 2 and 3, when no retention period is specified in parameter type 2 for a magnetic dump file that is not to be renamed (parameter type 6 absent), relabelling with #500 takes place. (Previously the mode #500 did not occur, contrary to the *Use of peripherals* section on page 110)

CORRECTIONS TO THE MANUAL

Page v, #XPJR title should read:

...random direct access files

Page 16, line 8, should begin:

buckets in each seek...

Page 17, Figure 1, line 1, should read:

...FILE ALLOCATOR

Page 20, line 4, should read:

*4 Ending cylinder number of...

Page 24, line -7 change DISFILEV,0 to:

DISCFILEV2,0

Page 30, line -1 should read:

...error messages output...

Page 63, line -14 should read:

will continue to the last allocated bucket of the file

Page 76, Figure 13, line -11 should read:

BUCKET

Page 97, line 3 should begin:

defines the dump identifier and the existing #XPJW dump file and the storage unit...

Page 101, line 3 should begin:

defines the dump and the...

Page 108, line 13 should read:

...No extra store for buffer areas is...

Page 123, section headed 'General', line -3 should read:

The skeleton PLAN source segment generated...

Page 123, line -5 should read:

...magnetic tape subfiles

Page 134, column headed *Device*, lines -2 and -1 should read:

CP0 or

MT0 or

Page 140, section headed 'File organization program', line 3 should read:

2 Load a direct access serial, sequential-indexed...

Page 143, section headed 'Output', line 3 should read:

...and scratch tapes will be...

Page 143, lines -8 to -6. The Note belongs with Figure 14.

Page 144, column headed *Parameter field*, line -7 should read:

...tape, etc. (see *General*, page 140)

Page 154, line -18 should read:

Running file copy utility programs

Page 161, message 53, column headed *Action*, insert opposite (b):

The program continues omitting
the block

Page 161, message 53, columns headed *Reason* and *Action*, below (b) insert a new (c) as follows:

(c) A record has been found
whose length is obviously
incorrect

The program continues omitting
the rest of the block. It is
possible that a number of incorrect
records have already been output

Page 164, message 90, column headed *Reason*, line -2 should begin:

the highest...

Page 196, section headed OUT, line 2 should begin:

#XREB...

Page 205, instruction 4, column headed *Console message* should read:

ON #XREB 5

Page 221, line -10 should begin:

Css,d,d,d,.....d, is the identifier...

Chapter 3,title page (page 230) should end:

...macros

© International Computers Limited 1979



PUBLICATION (NOTICE NO.)

4444

UNIFIED DIRECT ACCESS STANDARDS UTILITIES (3)

2/1/80

File one copy of this
notice with each of the
publications indicated

The following programs will be available on MLT 39:

XPJ1/7A
XPJW/2C

XPJ1/7A: WRITE DIRECT ACCESS IDENTIFIERS

This new version of the program contains the following changes:

- 1 6336 words of store are required for running XPJ1 (21440 if initializing EDS200)
- 2 If an error is detected when checkreading a disc at cylinder level and subsequent re-checkreading of a cylinder, block by block, does not flag any errors, then the program will now repeat each block checkread a further two times, in an attempt to detect marginal flaws.

(Note that, although an error is detected when checkreading at cylinder level, this information cannot be used directly to relate to a particular block. The cylinder in error must be re-checkread, block by block, to locate the area to be erased)

- 3 The message:

UNABLE TO LOCATE AND ERASE BLOCK, AFTER CYLINDER READ FAILURE, ON
CYLINDER n

is output to the line printer if, following a cylinder checkread failure, no errors are detected on re-checkreading block by block

XPJW/2C: FILE COPY, DUMP AND RESTORE

With the issue of XPJW/2C, the errors reported in 1900 Series Housekeeping and Utility Routines Software Notice items 259/1, 260/1, 261/3 and 265/1 are corrected.

The following errors are also corrected:

- 1 When attempting a partial restore of a specified file held across two or more reels of a magnetic tape dump file, in certain cases an end of reel sentinel was not recognized, resulting in a run off the end of tape
- 2 For function 2, 3, 4 and 5 the message PART TO COPY OUTSIDE LIMITS was printed instead of ZERO LENGTH FILE FOUND
- 3 For functions 1 and 8 when outputting to a pre-allocated file, the retention period value given in field 6 of Parameter Type 4 was not implemented.
Note to GEORGE 3 and 4 users: A pre-allocated output file for functions 1 and 8 is now opened using mode #300 (versions prior to Mark 1C used #100)



**International
Computers
Limited**

**User
notice**

x1
SL

PUBLICATION (NOTICE NO.)

4444

UNIFIED DIRECT ACCESS STANDARDS UTILITIES (4)

13/8/80

File one copy of this
notice with each of the
publications indicated

The following programs will be available on MLT40:

XPJ1/7B
XREB/3B

XPJ1/7B: Write identifiers

This new version of XPJ1 contains changes to allow an EDS60 to be initialised as a CAFS disc, as follows:

- 1 The flaw index is written to block 64, unless an alternative is specified in optional parameter field 5
- 2 Up to 96 flaw blocks will be used by XPJ1 on initialising the storage unit, as required. No empty flaw blocks will be assigned
- 3 The two character identifier for the storage unit type in parameter field 2 is CF
- 4 The number of cylinders, specified in parameter field 4, is 406
- 5 The block address of the flaw index, specified in the optional parameter field 5, has the range of values 8 to 128, and must be a multiple of 8

XREB/3B: Create DA file from paper tape data

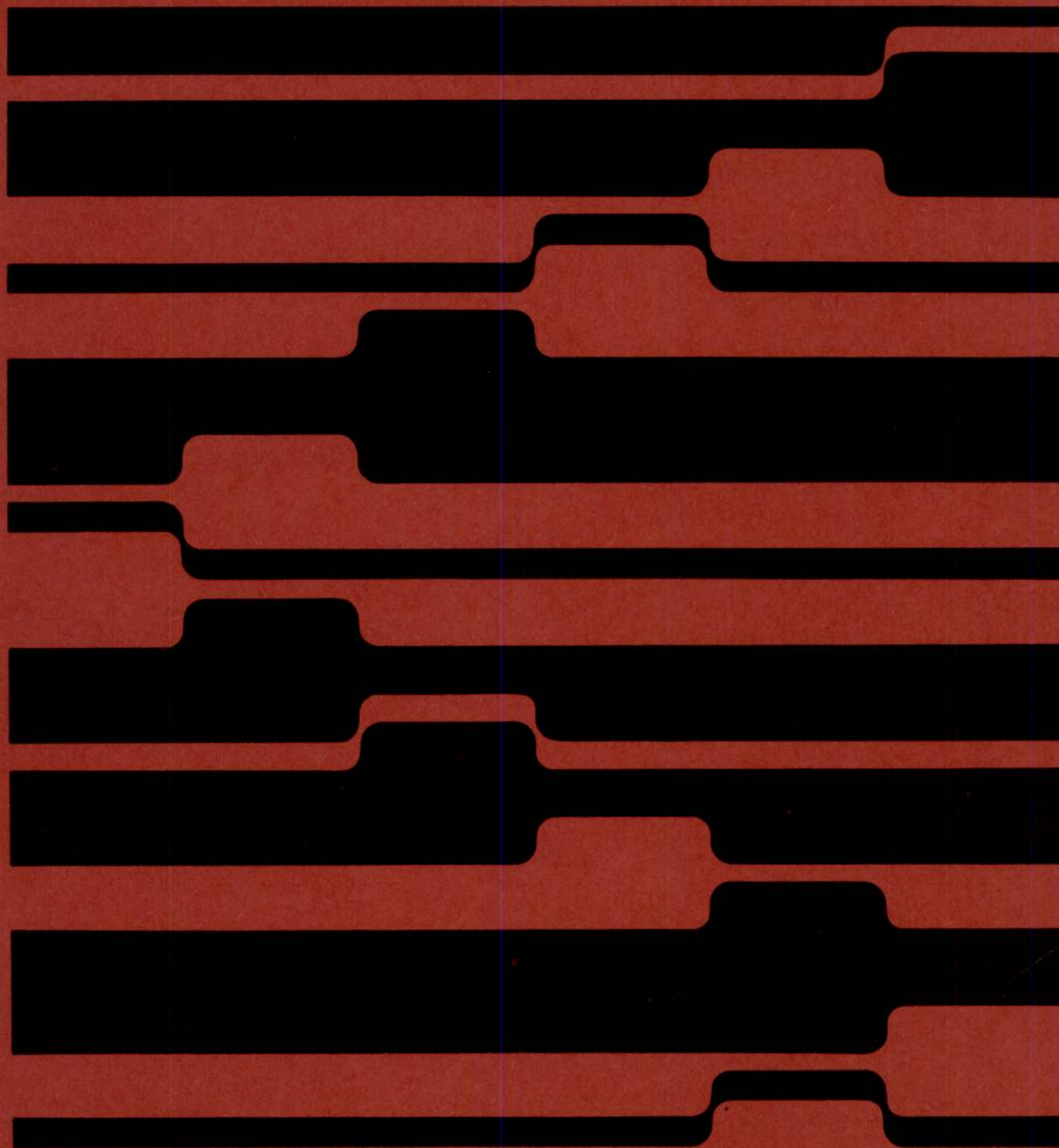
With the issue of XREB/3B, the errors reported in 1900 Series Housekeeping and Utility Routines Software Notice item 250/4 are corrected.

© International Computers Limited 1980



3rd Ed. 1979.

Unified Direct 1900 Series Access Standards Utilities





Unified Direct 1900 Series

Access

Standards

Utilities

ICL endeavours to ensure that the information in this document is correct and fairly stated, but does not accept liability for any error or omission.

The development of ICL products and services is continuous and published information may not be up-to-date. Any particular issue of a product may contain part only of the facilities described in this document or may contain facilities not described here. It is important to check the current position with ICL.

Specifications and statements as to performance in this document are ICL estimates intended for general guidance. They may require adjustment in particular circumstances and are therefore not formal offers or undertakings.

Statements in this document are not part of a contract or program product licence save insofar as they are incorporated into a contract or licence by express reference. Issue of this document does not entitle the recipient to access to or use of the products described, and such access or use may be subject to separate contracts or licences.

Technical Publication 4444

© International Computers Limited 1970

First Edition (TP4232) December 1970

Second Edition (TP4405) May 1975

Third Edition January 1979

ICL will be pleased to receive readers' views on the contents and organisation, etc. of this publication. Please write to

The Registry (Readership Survey)
UK Software and Literature Distribution Centre
International Computers Limited
60 Portman Road
Reading
Berks RG3 1NR

Distributed by
UK Software and Literature Distribution Centre
International Computers Limited
Registered Office: ICL House, Putney, London SW15 1SW
Printed by ICL Printing Services
Works Road, Letchworth, Herts SG6 1JY

Preface

This manual provides details of utility programs which are available to 1900 users who have direct access storage units organized to unified direct access standards (UDAS). Users needing basic direct access standards (BDAS) conversion utilities should retain the first edition of this manual (TP4232). Users needing BDAS utilities should retain the manual *Direct Access Utilities* (Edition 1, TP4190). Details of the ICL 1900 Series direct access sorting programs which conform to the unified direct access standards are described in the manual *Direct Access Sorting* (Edition 3, TP4411).

This manual is intended as a reference document and is therefore likely to be of most use to programmers and operators who require definitive information about UDAS utilities.

This manual supersedes *Unified Direct Access Standards Utilities* (Edition 2, TP4405). Specifications of programs and GEORGE 3 System Macros that have been superseded since that edition are in Appendix 1 of the new edition. Programs have been replaced as follows:

#XPJD by #XPJK

#XPJH by #XPJR

#XPJJ by #XPJK

GEORGE 3 System Macros have been replaced as follows:

DAMAP by DAFAM

DAREPORT by DAFAM

Further information on the 1900 range is provided in the following ICL 1900 Range publications:

Compiling Systems (Edition 2, TP4431)

Direct Access (Edition 2, TP4385)

Operating Systems GEORGE 3 and 4 (Edition 5, TP4345)

Contents

Preface	iii
Chapter 1 Introduction	1
PARAMETER FORMATS	2
GEORGE 3 SYSTEM MACROS	4
S - TRUSTED UTILITIES	5
Chapter 2 Non-trusted and Q-trusted utility programs	
#XMFR Copy a composite direct access file to another direct access file and delete all fragmentation	7
#XPJC Direct access file allocator	15
#XPJK File or cartridge report	31
#XPJL Housekeeping format direct access file print	47
#XPJQ File bucket or hardware block print	61
#XPJR Analysis of serial, sequential and random access data files	69
#XPJW Direct access file copy, dump and restore	81
#XPJX Change data retention period or delete a direct access file	117
File organization software	123
File organization program #XPJZ Generate a segment for a file organization or file copy utility program	125
File organization/file copy utility program	139
#XRAA Direct access composite file structure report	167
#XRAF Off-line print from a direct access file	177
#XREB Create direct access file from paper tape or cards	185
#XRPE Printer report of direct access records	215
Chapter 3 GEORGE 3 System macros	
DAFAM To produce #XPJK reports and /or cartridge reports	231
DAPRINT To print from direct access exofiles, filestore files, or storage units using #XPJL or #XPJQ	235
DAUTIL To run any one of the three UDAS utilities #XPJC, #XPJJ, #XPJX	239

XREMJMAC1	To write to a disc filestore file from data contained in one or more card or paper tape files using #XREB or #XREJ	243
XRPMEMAC1	To print reports of disc or magnetic tape filestore files using #XRPE or #XRPM	247

Chapter 4 S-trusted and systems utility programs

#XPJ1	Write direct access identifiers	251
#XPJ2	Reset system control srea	257
#XPJ7	Multi-storage unit file reset	267

Appendix 1 Specifications of utilities #XPJD, #XPJH and #XPJJ Specifications of GEORGE 3 macros DAMAP and DAREPORT

#XPJD	Map and summary file report for direct access storage units	273
#XPJH	Direct access file analysis or attempt in situ file reorganization	283
#XPJJ	Utilization and storage report	297
DAMAP	To produce an #XPJD storage map of a specified disc storage unit	305
DAREPORT	To produce an #XPJJ file-level report on a specified DA exofile	307
Index		309

Illustrations

Figure 1	Report of the files successfully allocated or deallocated or extended or contracted	17
Figure 2	Line printer error messages output by #XPJC	30
Figure 3	A file report	32
Figure 4	A file table produced by an FA type cartridge report	33
Figure 5	A minimum cartridge report	33
Figure 6	A file table and map produced by an MA type cartridge report for an EDS8	34
Figure 7	A file table and map produced by an M type cartridge report for an EDS60	35
Figure 8	The last page of a map of an EDS60	36
Figure 9	Example of a composite record print and an octal bucket print	49
Figure 10	Example of a full report	71
Figure 11	Example of a short report	72
Figure 12	An example of a summary report for one file area	75
Figure 13	An example of a summary report for four file areas	76
Figure 14	An example of an output report	143
Figure 15	Composite file structure report	172
Figure 16	Map and storage file report for EDS8	274
Figure 17	Map and storage file report for EDS30/60	275
Figure 18	Full report of reorganization	284
Figure 19	Short report of reorganization	285
Figure 20	Summary report of reorganization	285
Figure 21	Full report of analysis	286
Figure 22	Short report of analysis	287
Figure 23	Summary report of analysis	288
Figure 24	File level report	298
Figure 25	Storage unit level report—summary	298
Figure 26	Storage unit level report—allocated areas	298
Figure 27	Storage unit level report—file areas assigned to files starting on other storage units	299
Figure 28	Storage unit level report—unallocated areas	299
Figure 29	Storage unit level report—vacant cells	299

Chapter 1 Introduction

FORMAT OF THE MANUAL

The specifications in this manual are divided into chapters, as follows:

- Chapter 1 Introduction
- Chapter 2 Non-trusted and Q-trusted utility programs
- Chapter 3 GEORGE 3 system macros
- Chapter 4 S-trusted and systems utility programs

Appendix 1, page 273, contains specifications of programs and GEORGE 3 macros that have been superseded since the last edition.

PROGRAM SPECIFICATION FORMATS

The program specifications in this manual are written to a standard format which is described below.

The information in each specification is given under the following headings:

NAME
VERSION
TITLE
HARDWARE REQUIREMENT
DESCRIPTION
General
Input
Output
CONTROL
OPERATING
Trusted status
Executive priority
Use of peripherals
Running under Executive alone
OPERATING INSTRUCTIONS
EXCEPTION CONDITIONS
LINE PRINTER MESSAGES

Further headings or subheadings may be used to suit the needs of a particular utility. The information given under each headings is as follows:

NAME	The four character program name preceded by a hash mark (#)
VERSION	The version number of the utility which the specification describes
TITLE	A brief phrase describing the function of the utility
HARDWARE REQUIREMENT	This gives the store and peripheral requirements of the program. If the requirements of the program vary, or if a peripheral is optional, full details are given

DESCRIPTION	This is divided into three subsections
General	This subsection gives a comprehensive description of the facilities provided by the utility. It does not give details of parameters, input and output data and operating instructions as these follow in the rest of the specification. However, it brings to the programmer's attention any unusual features or error conditions in the program
Input	Information is given here about the nature of all input to the program
Output	This contains information about the output from the program. If there is output on the line printer (except error messages which are given in the subsection <i>Line printer messages</i>) then the format and print positions used by this output is given
CONTROL	This section describes the parameters to be input to the program. For programs #XPJn, these are set out in three columns, under the headings, <i>Parameter field</i> , <i>Range of values</i> and <i>Maximum number of characters</i> , in the order in which they are to be input to the program. Further details are given in the section <i>Parameter formats</i> below
OPERATING	This gives details of information required for the running of the program. It is in four sections
Trusted status	This section is only included if the program has trusted status, and will be either S or Q
Executive priority	This section gives the priority of the program as supplied
Use of peripherals	This section, using three columns, indicates for each peripheral the device number, the type of data for which it is used and details of its assignment and release
Running under Executive alone	This section is in three parts and gives instructions for operating, of exception conditions and error messages when no operating system is in use
OPERATING INSTRUCTIONS	This describes the actions which are required in order to run the program
EXCEPTION CONDITIONS	This section lists the exception conditions which may occur during the running of the program, and suggests the action that should be taken to rectify the condition. The list is in three columns, under the headings <i>Message</i> , <i>Reason</i> and <i>Action</i> . Further details are given in the section <i>Exception conditions</i> below
ERROR MESSAGES	This section lists the error messages output on the line printer. The majority of these messages deal with parameter errors and with the direct access files and storage units and magnetic tape files used by the program

PARAMETER FORMATS

Parameters (where applicable) for all utilities described in this manual may be input on cards or paper tape. The terminology and punching format for the utilities mentioned below is standard. Formats for all other utilities may be found embedded within the individual specifications.

XPJ1	XPJH	XPJR
XPJ2	XPJJ	XPJW
XPJ7	XPJK	XPJX
XPJC	XPJL	XPJZ and file organisation program
XPJD	XPJQ	

Use the following terminology with regard to parameter descriptions:

- 1 (a) **Parameter field.** A parameter field contains the smallest unit of parameter information that can be input to a program. Examples are the name of a file to be opened and the print width of the line printer to be used. A parameter field must be contained within:
 - (i) two commas (except for the first on a parameter card/line)

- (ii) a comma and a parenthesis (or vice versa)
- (iii) two parentheses
- (b) **Parameter group.** A group consists of one or more parameter fields contained within parentheses. Parameter groups are usually used to repeat parameter information of the same type; for example, if several areas of a direct access file are to be printed out, then parameter groups would be used to specify these areas
- (c) **Parameter type.** To perform a particular function, more than one category of parameter information is usually required by the program. For example, to print out parts of a direct access file at least two categories of information are required — details of the file from which the areas are to be printed and details of the areas themselves. Such categories are called parameter types in this manual. The parameter fields comprising a parameter type are usually contained on one card or line, but they may extend over more than one card or line in some circumstances
- (d) **Parameter set.** A parameter set consists of one or more parameter types which is required to perform a function
- (e) **Batch of parameters.** A batch of parameters are all the parameters input to a program during one run. A batch of parameters may perform one or more functions

- 2 The maximum number of columns that may be punched on a card is 76. The maximum number of characters in a paper tape block must not exceed 76 plus the newline character.

With cards, an optional sequence numbering facility is available. Sequence numbers may be punched in columns 77 to 80 inclusive of a parameter card. The sequence number field of the first card of a set determines whether or not sequence checking is to be carried out. If the field is blank the sequence fields of the cards within the set will not be checked. However, if columns 77 to 80 of the first card are not blank, sequence numbers will be checked taking the number of the first card as the start number for the set. When sequence checking is taking place any spaces found in the sequence field will be changed to zeros. If a card is found to be out of sequence a warning message will be output informing the user of the incorrectly sequenced card, but it will still be accepted

- 3 All cards/blocks will have been punched:

- (a) In columns/characters 1 to 4 the program name
- (b) In column/character 5 the parameter type

The parameter type indicator allows easy recognition of the type of data held within the parameter card/block.

If it is permissible for a parameter type to extend over more than one card/block, details are given under the description of the parameter type

- 4 Column/character 6 of the first parameter card/block of a parameter set will be punched with the function identifier. As each program may be used to process direct access files or units in several different ways the function identifier informs the program of the manner of processing
- 5 Each parameter field must be followed immediately by a terminating comma or by a parenthesis. Spaces must not be left between parameter fields
- 6 A parameter field may not extend over more than one card/block
- 7 Where a value need not be specified for a parameter field, the terminating comma must be present
- 8 Where a parameter field is omitted, the terminating comma must also be omitted
- 9 A comma must not be used for any purpose other than for a terminating character. Parentheses may not be used for any purpose other than for enclosing a group of parameter fields. Commas and parentheses are valid when held within record keys
- 10 Many of the parameter types in the specifications contain parameter groups (a description of parameter types and groups is given in 1 above). The two rules given below must be followed when specifying parameter groups:
 - (a) Each parameter field in a group (except the last) must be terminated by a comma. Even if the parameter itself is not specified the comma must still be present. If, for example, the optional parameter fields B, C and D are left out of the parameter group A to F, the group would be specified as follows:

(A , , , E,F)

- (b) If the last parameter field(s) of a group is omitted, the close parenthesis is "brought forward" to replace the terminating comma of the last parameter specified. If, for example, the optional parameter fields E and F are left out of the group A to F, the group would be specified as follows:

(A,B,C,D)

Both the rules apply to one parameter group so that if, for example, optional parameter fields B, C, E and F are left out of the parameter group A to F, the group would be specified as follows:

(A, , , D)

- 11 A batch of parameter cards/blocks must be followed by a card/block with 4 asterisks (****) punched in the first 4 columns/character positions. This batch terminator causes the run to be terminated
- 12 With most of the utility programs, a L`OAD` or F`IND` message may be punched on the same card/block as the batch terminator. Such a message will cause the program, in an error free run, to output the message on the console and delete itself. The message may start anywhere after column/character 4 on the batch terminator card/block, but it must start with the characters LO or FI and must be wholly contained on the card/block. If more than a 40 character message is input only the first 40 characters are output on the console.

If no message is input, the program halts HH in an error free run. If an error occurs during the run the program halts ZZ whether or not a L`OAD` or F`IND` message was input

EXCEPTION CONDITIONS

Details of individual exception conditions are given in the specifications. However, the following general points should be noted with regard to exception conditions.

- 1 As the unified direct access standards utility programs can only be used with the unified direct access standards Executives, any attempt to use these programs with any other Executives will result in the following message being output to the console typewriter:

#name; HALTED:— EXECUTIVE NOT U.D.A.S.

Some of the utilities do not output this standard message. With such a utility, the actual message output is included in the exception conditions
- 2 The format of the message output when a storage unit is not on-line varies with the Executive being used and can take one of the following two forms:

#name; LDC *nnnnnn

or

#name; LOAD STORAGE UNIT *nnnnnn

where *nnnnnn is, in both cases, the serial number of the storage unit.

In this manual, the LDC version is used

When running unified direct access standards utilities care should be taken to ensure that there are never two files having the same name and generation number on-line at the same time. The exception to this rule is when unit serial numbers are specified in the parameters in which case the rule will only apply to files having the same name and generation number on the same storage unit.

GEORGE 3 SYSTEM MACROS

Chapter 3 describes five GEORGE 3 System Macros which are available for running some of the unified direct access standards utility programs under GEORGE 3. The macros are:

- 1 DAFAM which can be used to produce #XPJK file reports and/or catridge reports
- 2 DAPRINT which can be used to run #XPJL or #XPJQ on direct access exofiles, filestore files or storage units
- 3 DAUTIL which can be used to run #XPJC, #XPJJ or #XPJX
- 4 XREBJMAC1 which can be used to run programs #XREB and #XREJ

5 XRPMEMAC1 which can be used to run printer report programs #XRPE and #XRPM

The parameters associated with these macros follow the conventions described in *Operating Systems GEORGE 3 and 4*.

S-TRUSTED UTILITIES

The information in this section applies only to the S-trusted utilities. The specifications of these utilities are given in Chapter 4.

The MS message

If required, parameters can be input to the S-trusted utilities through the console typewriter. This is done by using the MS console message. The message is available on Executives that include range compatible trusted facilities.

An S-trusted utility should be entered at entry point 22 if it is required to input parameters through the console typewriter. Once entered, Executive will output the following message when the utility is ready to accept a parameter line:

#xxxx; DISPLAY:- MS?

where xxxx is the name of the utility.

The MS message should then be typed in. It has the following format:

MS #xxxx = *parameter line*

where

xxxx is the name of the S-trusted utility in use.

parameter line is the parameter line input.

Successful input of the parameter line is indicated by the Executive message O.K. being output on the consol. If the input of the parameter line is not successful, one of the following error messages will be output:

<i>Message</i>	<i>Reason</i>
ERROR B	The parameter line typed in is too long
ERROR M	The program is not ready for an MS parameter

The SD message

The SD message is an Executive message provided in all unified direct access standards Executives. Its purpose is to assist in the processing of storage units not conforming to unified direct access standards, namely corrupted units and virgin units. The message is used to inform Executive of the characteristics of such a unit so that its System file may be used by the S-trusted utilities.

The SD message has the following format:

SD w x y z

where

w is the device number of the storage unit.

x is the storage unit serial number.

y is the number of cylinders.

z is the number of blocks per cylinder.

Executive outputs the console message O.K. if the SD message is accepted. If the SD message is not accepted, one of the following error messages will be output:

<i>Message</i>	<i>Reason</i>
ERROR B	Either the shape specified by y and z is not valid for this device, or the serial number specified by x belongs to another on-line unit

Message

ERROR D

ERROR J

ERROR P

Reason

The System file of the specified unit has already been opened for writing

The control header in the specified unit already conforms to unified direct access standards

The unit is inoperable

Definition of block address

The *block address* referred to in the specifications of the S-trusted utilities is defined as follows:

Block address = System file bucket number minus one.

Chapter 2 Non-trusted and Q-trusted utility programs

NAME

#XMFR

VERSION

Mark 7A

TITLE

Copy a composite direct access file to another direct access file and delete all fragmentation

HARDWARE REQUIREMENT

5120 words of store minimum. (See *Operating instruction 1*, page 10)

1 card reader or 1 paper tape reader

2 direct access files

Optionally, 1 line printer at end of run

DESCRIPTION**General**

The program copies a composite direct access file to another direct access file and reorganizes the output file so that no fragmentation of the directories or data subfiles appears.

The composite files and/or subfiles created by certain items of ICL software are marked as 'not to be reorganized' and #XMFR will take the appropriate action with such files and/or subfiles.

If a parity error occurs while reading from the input file, the program provides an option whereby the subfile in error may be omitted from the reorganized output file. A line printer report of omitted subfiles (if any) is given at the end of the run.

Input

Input to the program consists of the direct access file to be copied and parameters on paper tape or cards as described in the section *Control*, page 8.

The input file must be in composite file format and must conform to ICL composite file standards. A binary output file produced by the direct access consolidator (#XPCCK) should not be reorganized by #XMFR. (See #XPEC in *Compiling Systems*.)

Output

The output file need not have bucket headers written to it prior to the #XMFR run. The output file will contain a copy of the data on the input file. Directories will be moved to the beginning of the file and all fragmentation of directories and data subfiles will be removed, except where the subfiles are marked as 'not to be reorganized'. The Space subfile of the reorganized file will be output at the beginning of the subfile directory.

Notes:

- 1 If the top level directory subfile on the input file contains only data subfile descriptions and is not fragmented, then the top level directory subfile written to the output file will occupy at least as many buckets as on input. In this case some buckets of the directory subfile may contain only bucket headers
- 2 When opening the output file, #XMFR will check that the number of blocks per bucket and the size of the bucket headers are the same as those in the input file

#XMFR

- 3 If the output file is of insufficient length to contain the reorganized data, an attempt will be made to extend it by one sixteenth of its length, on the same cartridge. If this is unsuccessful the file will be extended by the number of blocks that are available on the cartridge
- 4 Unless the program reaches a successful conclusion (see *Operating instruction 6*, page 10) the status of the output file is always indeterminate

Listing

If the parity error option (described later in section *Parity errors*, page 9) is requested and parity errors occur, a line printer will be allocated at the end of the run. It will list the subfiles omitted because of parity errors.

CONTROL

The operation of the program #XMFR is governed by control statements and associated parameters which may be punched on 8 track paper tape or on cards. There are five statements, the *IN, *OUT and *END statements being mandatory and the others optional. The *END statement must be last but the other statements may appear in any order. Each statement must appear on a new line of paper tape or a new card.

*IN statement

This statement specifies the name of the input file. The format is as follows:

*IN(*fn*(*fgn*))

where

fn is the name of the input file and consists of up to twelve characters. Leading spaces are ignored and if the name is less than twelve characters the name is space filled.

fgn is the file generation number of the input file. It should be in the range 0 to 4096 (4096 should be specified to open any file of this name). It may be omitted, in which case the enclosing brackets should also be omitted, and the file of the specified name with the highest generation number is opened.

*OUT statement

This statement specifies the name of the file to be output. The user should note that any renaming will not take place until the program has come to a successful conclusion. The statement format is as follows:

*OUT(*fn*(*fgn*)=*nfn*(*nfgn*))

where

fn is the name of the output file and consists of up to twelve characters. Leading spaces are ignored and if the name is less than twelve characters the name is space filled.

fgn is the file generation number of the output file. It should be in the range 0 to 4096 (4096 should be specified to open any file of this name). It may be omitted, in which case the enclosing brackets should also be omitted, and the file of the specified name with the highest generation number is opened

nfn is the new name to be given to the output file. It may be omitted in which case the preceding = should also be omitted, and the file name and generation number will remain unchanged.

nfgn is the new file generation number in the range 0 to 4095 and may be omitted in which case a file generation number of zero will be given.

*VAR statement

This statement is optional and enables the size of subfile description records to be fixed at the word length specified. The absence of this statement causes all subfile descriptions to be transcribed to the output file in full and the space subfile descriptions to be ten words long. Additional words are zero filled. The format is as follows:

*VAR(*nn*)

where

nn specifies the size of subfile description records and is in the range 9 to 25 inclusive.

***DEL statement**

This statement is optional, and enables another program to be loaded into store at the conclusion of #XMFR. The format is as follows:

***DEL (*name,file*)**

where

name is the four character program name.

file is the last four characters of the program file where the program is to be found. It may be omitted in which case the preceding comma should also be omitted.

***END statement**

This statement marks the end of the parameters and is compulsory. The format is as follows:

***END**

PARITY ERRORS

The setting of Switch 0 provides an option whereby, if a parity error is reported on reading from the input file, the reorganization will continue omitting the subfile in which the error occurred.

If Switch 0 is ON, #XMFR will omit from the output file the complete subfile if a parity error occurs on reading a data subfile or the subfile and any lower levels of subfile if the error occurs on reading a directory subfile. #XMFR cannot, however, continue if a parity error occurs on reading the top level directory. At the end of a run, if any parity error has been reported, a line printer report in the following format is given of the subfiles that have been omitted:

PARITY ERRORS OCCURRED DURING #XMFR AS FOLLOWS:-

PARITY ERROR IN <i>subfilename</i>	}	for top-level data subfile
<i>subfilename</i> REMOVED FROM OUTPUT FILE		
PARITY ERROR IN <i>subfilename 3</i>	}	for level 3 data subfile
IN <i>subfilename 2</i>		
IN <i>subfilename 1</i>		
<i>subfilename 3</i> REMOVED FROM OUTPUT FILE		
PARITY ERROR IN <i>subfilename</i>	}	for directory subfile
<i>subfilename</i> REMOVED FROM OUTPUT FILE. WITH ITS LOWER LEVELS		

If during the analysis of the input file to produce this report a further parity error occurs the line printer message:

PARITY ERROR HAS OCCURRED WHILST GIVING INFORMATION ABOUT THIS ERROR – ABANDONED

is output and the current report is abandoned. There is a limit of ten on the number of parity errors that #XMFR can handle in this way. If any further parity error is reported the run will be abandoned and no line printer report will be produced.

If Switch 0 is OFF, #XMFR will immediately terminate the run if a parity error is reported.

Regardless of the setting of Switch 0, #XMFR will immediately terminate the run if a parity error occurs on writing to the output file.

#XMFR

OPERATING

Executive priority

The program as supplied has a priority of 70.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
TR0 or CR0	Parameters	The reader is assigned at the start of the run and is released when the parameters have been read and checked
LP0	Parity error diagnostic messages (optional)	If Switch 0 is set, the printer is assigned towards the end of the run and released when the messages have been printed
DA0	Input file (must contain a composite file structure)	This file is opened after the parameters have been checked and retained until the end of the run
DA1	Output file (must have the same number of blocks per bucket, and the same size bucket headers as the input file)	This file is opened after the parameters have been checked and retained until the end of the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XMFR (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>). The message HALTED:- LD will only be output if the loading operation was successful The time taken to complete a reorganization using #XMFR may be reduced if the program is loaded into more than 5120 words of store	HALTED:- LD
2 Load the direct access files	
3 Load the parameters in the appropriate reader	
4 If the parity error option is required:	ON #XMFR 0
5 To activate the program,	
(a) For parameters on paper tape input:	GO #XMFR 20
(b) For parameters on cards input:	GO #XMFR 21
6 At the end of a successful run, the program will display the messages:	DISPLAY:- BUCKETS USED <i>mmmm</i> DISPLAY:- BUCKETS SPARE <i>nnnn</i> where <i>mmmm</i> is the number of buckets occupied by the reorganized data and <i>nnnn</i> is the number of buckets that remain unoccupied on the output file

Narrative

The program will delete itself with the message:
unless another program has been requested in which
case the program will delete itself with the message:

Console message

DELETED:- OK

DELETED:- FI *#name #file*

CONSOLE MESSAGES

Message

DISPLAY:- NO MORE PARITY ERRORS CAN BE
DEALT WITH

DISPLAY:- OUTPUT FILE NOW *nnnn* BUCKETS

DISPLAY:- OUTPUT FILE RENAMED *file name*

Meaning

The parity error list is full. The run will continue
but if another parity error occurs the run will
be abandoned

This message is produced when the output file
is extended (see Note 3, page 8)

If the rename option is selected in *OUT
parameter, this message is produced at the end of
the run when the file has been renamed

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XMFR during execution and loading. Prior to entry into #XMFR it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:-LD reference must be made to the section *Exception conditions* in FIND A BINARY PROGRAM in *Compiling systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	Card reader is not available	Make a card reader available, then input: GO #XMFR
2	HALTED:- DUPLICATE PARAMETER – FIX AND RESTART	Parameter error	Either (a) Abandon the run or (b) Substitute correct parameters, then input: GO #XMFR
3	HALTED:- INCORRECT PARAMETER – FIX AND RESTART	Parameter error	Either (a) Abandon the run or (b) Substitute correct parameters, then input: GO #XMFR
4	HALTED:- LP	Line printer is not available	Make a line printer available, then input: GO #XMFR
5	HALTED:- MC	Non-specific system error	Abandon the run. If the error persists report for investigation

#XMFR

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
6	HALTED:- NO IN PARAMETER	Parameters do not specify an input file	Either (a) Abandon the run or (b) Substitute correct parameters, then input: GO #XMFR
7	HALTED:- NO OUT PARAMETER	Parameters do not specify an output file	Either (a) Abandon the run or (b) Substitute correct parameters, then input: GO #XMFR
8	HALTED:- TR	Tape reader is not available	Make a tape available, then input: GO #XMFR
9	HALTED:- TRANSFER FAILURE	Card or paper tape reader error	Replace parameters, then input: GO #XMFR
10	HALTED:- EE	An error has occurred. This is preceded by one of the following DISPLAY messages 11 to 20	
11	DISPLAY:- INCOMPATIBLE BUCKET HEADERS	The input and output files have different length bucket headers	Abandon the run
12	DISPLAY:- INCOMPATIBLE BUCKET SIZES	The input and output files have a different number of blocks per bucket	Abandon the run
13	DISPLAY:- INPUT FILE CANNOT BE COPIED: <i>x</i>	If <i>x</i> = 1 the input file is marked as 'not to be reorganized'. If <i>x</i> = 2 the input file is not in composite file format	Abandon the run
14	DISPLAY:- <i>filename</i> CANNOT BE OPENED: <i>y</i>	<i>filename</i> is the name of the file in question. If <i>y</i> = 1, the file is not in the system. If <i>y</i> = 2, there is a failure of the integrity code check. If <i>y</i> = 3, there is insufficient store for holding further file descriptions, or an attempt has been made to open more than the maximum permitted number of files. If <i>y</i> = 4, the purge date is not exceeded	Make the file available Abandon the run Delete another program using direct access, then input: GO #XMFR Abandon the run

<i>Message</i>	<i>Reason</i>	<i>Action</i>
15 DISPLAY:- INPUT FILE INVALID FORMAT: z DISPLAY:- IN BUCKET n	n is the bucket number which contains the error. z = 1 if a subfilename does not start with an alphabetic character. z = 2 if there is no directory description record at the start of a directory. z = 3 if a subfile description record is of incorrect length. z = 4 if a record referred to is not present in the bucket. z = 5 if the record size is greater than the bucket size	Abandon the run
16 DISPLAY:- LOGICAL BUCKET NUMBER OUT OF RANGE DISPLAY:- BUCKET m DISPLAY:- REFERENCED BY BUCKET n	m is the bucket number which is out of range. n is the number of the bucket which contained a pointer to m	Abandon the run
17 DISPLAY:- OUTPUT FILE CANNOT BE EXTENDED DISPLAY:- TO MORE THAN BUCKET n	There is insufficient space on the cartridge containing the output file for the file to be extended any further	Abandon the run
18 DISPLAY:- DISC ERROR n	n = 1 for flaw area full. n = 2 for unclearable parity error on output file	Abandon the run
19 DISPLAY:- PARITY FAILURE	Only if switch 0 not set; the parity failure occurred when reading from the input file	Abandon the run
20 DISPLAY:- PARITY FAILURE DISPLAY:- IN TOP LEVEL DIRECTORY – RUN ABANDONED	Irrespective of the state of switch 0, the run cannot continue as it has been impossible to read the top level directory	
21 HALTED:- xx	An error has been detected when loading #XMFR by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

NAME

#XPJC

VERSION

Mark 1G

TITLE

Direct access file allocator

HARDWARE REQUIREMENT

3200 words of store

1 card reader or 1 paper tape reader

1 line printer

1 or more direct access storage units

DESCRIPTION**General**

This program can perform four functions:

Function 1 Create a new direct access file

Function 2 Delete a direct access file whose retention periods have been exceeded

Function 3 Create additional file areas

Function 4 Delete file areas from a file whose data retention period has been exceeded

With functions 1 and 3 the program will write bucket headers, if requested.

Although the program allows for a large number of file areas in a file, the actual number of file areas which may be allocated (using functions 1 or 3) is dependent on Executive.

Up to four types of parameters are associated with each of the four functions of #XPJC. The first parameter field within a parameter type has a unique number which identifies the type, as follows:

<i>Parameter type</i>	<i>Description</i>
1	This parameter type describes the file upon which action will be taken
2	This parameter type contains information concerning the type of file being created
3	This parameter type actions the writing of bucket headers and describes the information to be written in the bucket headers of the file areas being created
4	This parameter type describes the file areas being created

If several parameter types are necessary as input to a single function they must appear in the order given below.

FUNCTION 1: CREATING A NEW FILE

The creation of a file in full ICL direct access housekeeping format (that is bucket headers being written to the file) can be achieved by the input of parameter types 1, 2, 3 and 4.

A file of zero length is first created and then the requested file areas are created by extending the file into a given area or extending the file by a number of buckets. The bucket number recorded in the file index cell as end of file bucket will always be set as 1 and the data retention period will be set to zero.

#XPJC

The omission of type 2 parameters will result in a serial file being created. The field index and parameters cell will be set up to indicate two word bucket headers, variable length records and a maximum record length equal to the bucket size (in words) less two.

Type 3 parameters may only be used in conjunction with types 2 and 4 and, if present will result in bucket headers being written to the file, using the write n words mode where n takes the value 2 or 3 (see section *Reading and Writing Data*, Chapter 14 in *Direct Access* for details of the contents of the buckets after writing). If auxiliary headers are specified they will be allocated in home and first level overflow buckets only. The number of home bucket in each seek area is specified as a percentage known as the seek area packing density. #XPJC will subtract from the number of buckets in a seek area the number of index buckets within the seek area before calculating the number of home buckets from the percentage specified (the calculation is rounded down except for a zero result where it is rounded up to 1). The remaining buckets in the seek area will be assigned as first level overflow buckets.

If type 3 parameters are omitted the original contents of the allocated areas will be unchanged.

With type 4 parameters the creation of file areas may be specified in two ways:

- 1 File areas to be created in specified areas. This method is mandatory if a file is allocated with a second level overflow area
- 2 A number of buckets may be specified to be allocated anywhere using up to a maximum number of file areas. That is #XPJC will attempt to extend by one contiguous area equal in size to the buckets specified, if this is not possible it will extend by the amount available. It will then repeat the extension process on the remaining balance until such time as this is zero or the maximum number of file areas have been attained, whichever is the earlier. It should be noted that the final allocation of buckets may be more than that specified due to Executive rounding (see section *Extending and contracting a file*, Chapter 14 in *Direct Access*). This method must not be used if a file is allocated with a second level overflow area

Within any one set of parameters only one method of file area allocation can be used.

For type 4 parameters, if more than one file area is to be created, either the group within the type specifying the file area to be created must be repeated, or the parameter type must be repeated as a whole. The file areas described are generally created immediately, so that if for any reason a file area description is found to be invalid, the descriptions preceding it have been processed, and the file areas created. If, however, a second level overflow area is specified type 4 parameters are validated as usual but are held in store until the size of the second level overflow area can be validated. If the number of buckets contained in the seek areas reserved for the second level overflow area exceeds 511 or if the asterisk terminating the function is omitted the file will not be allocated.

If type 4 parameters are omitted, a zero length file will be created.

FUNCTION 2: DELETING A FILE

The deletion of a file can be achieved by the input of type 1 parameters.

The file is first contracted back to zero length and then deleted, subject to its retention periods being exceeded.

FUNCTION 3: CREATING ADDITIONAL FILE AREAS

The extension of a file in full ICL direct access housekeeping format can be achieved by the input of parameter types 1, 3 and 4.

If type 3 parameters are present bucket headers will be written to the additional file areas. If this type is present, however, it should be identical to the one used in the original creation of the file.

With type 4 parameters, if more than one file area is to be created, either the group within the type specifying the file area to be created must be repeated, or the parameter type must be repeated as a whole. The file areas described are created immediately, so that if for any reason a file area description is found to be invalid, the descriptions preceding it have been processed, and the file areas created.

If file areas are added to a file which has a second overflow area, the new file areas will be considered to be part of this overflow area. The size of the resultant overflow area will not be checked. A second overflow area cannot be created by using this function.

FUNCTION 4: DELETING A FILE AREA

Deletion of file areas is achieved by the input of type 1 parameters. The data retention period of the file must have been exceeded. File areas will only be deleted from the end of the file. Starting with the final file area, the file is successively contracted by the length of each file area until the count of file areas to be deleted is exhausted. The bucket number recorded in the file index cell as the end of file bucket will be unchanged.

Input

Input to the program consists of one or more storage units conforming to the 1900 Series unified direct access standards and a set of parameters for each function to be executed. The parameters are described in the section *Control*, below.

Output

The output from the program will consist of one or more storage units with the system control area(s) updated to show the new state of the files on these units.

Line printer output

A report of the files successfully allocated or deallocated or extended or contracted is output on the line printer, using print positions 1 to 96. The layout of the report is shown in Figure 1 below.

#XPJC	MKnn	dd/mm/yy	hh/mm/ss	FILE ALLOCATION			PAGEyyyy
FILE IDENTITY	FGN	FILE RET	DATA RET	TOTAL BKTS	2ND LEVEL ST BKT	FUNCTION	
xxxxxxxxxxxx	xxxx	xxxx	xxxx	xxxxxxx	xxxxxxx	ALLOCATED	
xxxxxxxxxxxx	xxxx					DEALLOCATED	
xxxxxxxxxxxx	xxxx	xxxx	xxxx	xxxxxxx	xxxxxxx	EXTENDED	
xxxxxxxxxxxx	xxxx	xxxx	xxxx	xxxxxxx	xxxxxxx	CONTRACTED	

Figure 1 Report of the files successfully allocated or deallocated or extended or contracted

where

dd/mm/yy is the date on which the report is printed.

hh/mm/ss is the time at which the report is printed.

xxxxx represents information relevant to each file.

Error messages are also output to the line printer, details are given in the section *Line printer error messages*, page 29.

CONTROL

This section describes the parameters associated with the four functions of #XPJC described in the section *General*. Further details of parameter formats are given in the section *Parameter formats*, page

Function 1: create a new file

PARAMETER TYPE 1

	Parameter field	Range of values	Maximum number of characters
1	Program name (XPJC) and parameter type	XPJC1	5 (fixed)
2	Function identifier	1	1 (fixed)

#XPJC

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
3	Unit serial number	000001 to 777777 (an octal number)	6 (fixed)
4	Name of the file to be created	Up to 12 alphanumeric, space (V) or minus (–) characters, with the first character alphabetic	12
5	Generation number of the new file	0 to 4095	4
6	De-allocation retention period of the new file	0 to 4095 (days)	4
7	Integrity code for the new file	1, 2 or 3	1 (fixed)
8	Bucket size	1, 2, 4 or 8 (blocks)	1 (fixed)
9	Flaw indicator	M where M indicates own monitoring of flaws. Note: If this parameter is omitted, flaws will be monitored by Executive	1 (fixed)

PARAMETER TYPE 2

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJC) and parameter type	XPJC2	5 (fixed)
2	Record length indicator for the new file	F or V where F = fixed length V = variable length Note: If this parameter is not specified, V is assumed	1 (fixed)
3	File organization code for the new file	Q, R or S where Q = sequential file R = random file S = serial file	1 (fixed)
4	Maximum size of record in the new file	2 to 1022 (words) Note: If this parameter is not specified, the maximum size will be set to the length of the bucket, less the length of the bucket header, less the length of the auxiliary header (if present)	4
5	Key length for the new file	0 to 64 (characters)	2
6	Number of characters in record (including word 0 of the record) preceding first character of key	0 to 4087	4

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
7 Number of seek areas in the second level overflow area of the new file	0 to 99 see note below	2
8 Bucket header indicator	3 or 2 Note: If this parameter is omitted, 2 is assumed	1 (fixed)

Note: The number of buckets that may be contained within a second level overflow area must not exceed 511 (see section *Function 1: Creating a new file* on page 15).

PARAMETER TYPE 3

If this parameter type is included bucket headers will be written to the file.

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJC) and parameter type	XPJC3	5 (fixed)
2 Seek area packing density for the new file (that is, percentage of home buckets per seek area)	0 to 99 Note: A value of 0 represents no first level overflow buckets (that is, 100% home buckets)	2
3 Size of auxiliary header for the new file	0 to 999 (words per bucket)	3
4 Index bucket indicator for the new file	I Note: If index buckets are not required, this parameter field and parameter fields 5 to 7 should be omitted	1 (fixed)
5 Size of level 1 index	0 to 99 (buckets)	2
6 Size of each level 2 index	0 to 99 (buckets)	2
7 Size of each level 3 index	0 to 99 (buckets)	2

PARAMETER TYPE 4

This parameter type has two formats.

Format 1

File area descriptions specifying unique areas. Mandatory if second level overflow required

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJC) and parameter type	XPJC4	5 (fixed)
2 Type of file area(s) being created	S Note: If more than one type 4 parameter line is input, this parameter field may be omitted in the second and subsequent lines. If omitted, the preceding comma must also be omitted	1 (fixed)

#XPJC

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*3	Starting cylinder number of the file area	0 to 4095	4
*4	Ending cylinder of the file area	0 to 4095	4
*5	Starting block number (a multiple of 8) within the cylinder	0 to 8388600	7
*6	Ending block number (a multiple of 8 minus 1) within the cylinder	7 to 8388607	7
*7	Serial number of the unit on which file area(s) is to be created	000001 to 777777 (an octal number) Note: If this parameter is omitted, the file area(s) will be created on the last previously specified unit serial number for this file (that is, from a previous file area description or from parameter type 1)	6 (fixed)

*The group of parameter fields 3 to 7 must be enclosed in parentheses and may be repeated. If parameter field 7 is omitted the close parenthesis replaces the terminating comma of parameter field 6. The parameter group defining one file area must be contained within one card or line, although several file area descriptions may appear on one card or line.

Format 2

File areas to be allocated anywhere and second level overflow not required

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJC) and parameter type	XPJC4	5 (fixed)
2	Type of file area(s) being created	A Note: If more than one type 4 parameter line is input, this parameter field may be omitted in the second and subsequent lines. If omitted, the preceding comma must also be omitted	1 (fixed)
*3	Maximum number of file areas that can be created by this allocation	1 to 9999	4
*4	Number of buckets to be allocated	1 to 8388607 Note: More buckets than requested may be allocated by Executive due to rounding	7
*5	Serial number of the unit on which file area(s) is to be created	000001 to 777777 (an octal number) Note: If this parameter is omitted, the file area(s) will be created on the last previously specified unit serial number for this file (that is, from a previous file area description or from parameter type 1)	6 (fixed)

*The group of parameter fields 3 to 5 must be enclosed in parentheses and may be repeated. If parameter field 5 is omitted the close parenthesis replaces the terminating comma of parameter field 4. Parameter fields 3 to 5 must be contained within one card or line, although several groups may appear on one card or line.

Each parameter field must be followed by a comma except when followed by a parenthesis. The complete set of parameters for this function must be terminated by an asterisk (*).

Function 2: delete a file

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJC) and parameter type	XPJC1	5 (fixed)
2	Function identifier	2	1 (fixed)
3	Serial number of the unit containing the file index and parameters entry of the file to be deleted	000001 to 777777 (an octal number)	6 (fixed)
4	Name of the file to be deleted	Up to 12 alphanumeric, space (V) or minus (—) characters, with the first character alphabetic	12
5	Generation number of the file to be deleted	0 to 4095	4

Each parameter field must be followed by a comma; the complete set of parameters for this function must be terminated by an asterisk (*).

Function 3: create additional file areas

PARAMETER TYPE 1

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJC) and parameter type	XPJC1	5 (fixed)
2	Function identifier	3	1 (fixed)
3	Serial number of the unit containing the file index and parameters entry of the file requiring the additional file area(s)	000001 to 777777 (an octal number)	6 (fixed)
4	Name of the file requiring additional file area(s)	Up to 12 alphanumeric, space (V) or minus (—) characters, with the first character alphabetic	12
5	Generation number of the file specified in parameter field 4	0 to 4095	4

PARAMETER TYPE 3

If this parameter type is present bucket headers will be written to the additional file areas. If present it should be identical to the one used in the original creation of the file.

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJC) and parameter type	XPJC3	5 (fixed)

#XPJC

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
2	Seek area packing density in the additional file areas (that is, percentage of home buckets per seek area)	0 to 99 where 0 = 100%	2
3	Size of auxiliary header in the additional file areas	0 to 999 (words per bucket)	3
4	Index bucket indicator	I Note: If index buckets are not required, this parameter field and parameter fields 5 to 7 should be omitted	1 (fixed)
5	Size of level 1 index	0 to 99 (buckets)	2
6	Size of each level 2 index	0 to 99 (buckets)	2
7	Size of each level 3 index	0 to 99 (buckets)	2

PARAMETER TYPE 4

This parameter type has two formats

Format 1

File area descriptions specifying unique areas

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJC) and parameter type	XPJC4	5 (fixed)
2	Types of file area(s) being created	S Note: If more than one type 4 parameter line is input, this parameter field may be omitted in the second and subsequent lines. If omitted, the preceding comma must also be omitted	1 (fixed)
*3	Starting cylinder number of the file area	0 to 4095	4
*4	Ending cylinder number of the file area	0 to 4095	4
*5	Starting block number (a multiple of 8) of the file area	0 to 8388600	7
*6	Ending block number (a multiple of 8 minus 1) of the file area	7 to 8388607	7
*7	Serial number the unit on which the file area(s) is to be created	000001 to 777777 (an octal number) Note: If this parameter is omitted, the file area(s) will be created on the last previously specified unit serial number for this file (that is, from a previous file area description or from parameter type 1)	6 (fixed)

*The group of parameter fields 3 to 7 must be enclosed in parentheses and may be repeated. If parameter field 7 is omitted the close parenthesis replaces the terminating comma of parameter field 6. The parameter group defining one file area must be contained within one card or line, although several file area descriptions may appear on one card or line.

Format 2

File areas to be allocated anywhere

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJC) and parameter type	XPJC4	5 (fixed)
2 Type of file area(s) being created	A Note: If more than one type 4 parameter line is input, this parameter field may be omitted in the second and subsequent lines. If omitted, the preceding comma must also be omitted	1 (fixed)
*3 Maximum number of file areas that can be created by this allocation	1 to 9999	4
*4 Number of buckets to be allocated	1 to 8388607 Note: Executive may allocate more buckets than requested due to rounding	7
*5 Serial number of the unit on which file area(s) is to be created	000001 to 777777 (an octal number) Note: If this parameter is omitted, the file area(s) will be created on the last previously specified unit serial number for this file (that is, from a previous file area description or from parameter type 1)	6 (fixed)

*The group of parameters fields 3 to 5 must be enclosed in parentheses and may be repeated. If parameter field 5 is omitted the close parenthesis replaces the terminating comma of parameter field 4. Parameter fields 3 to 5 must be contained within one card or line, although several groups may appear on one card or line.

Each parameter field must be followed by a comma except when followed by a parenthesis. The complete set of parameters for this function must be terminated by an asterisk (*).

Function 4: delete file areas

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJC) and parameter type	XPJC1	5 (fixed)
2 Function identifier	4	1 (fixed)
3 Serial number of the unit containing the file index and parameters entry of the file from which the file area is to be deleted	000001 to 777777 (an octal number)	6 (fixed)
4 Name of the file from which the file area is to be deleted	Up to 12 alphanumeric, space (V) or (—) characters, with the first character alphabetic	12

#XPJC

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
5	Generation number of the file specified in parameter 4	0 to 4095	4
6	Number of file areas to be deleted from the file	1 to 9999	4

Each parameter field must be followed by a comma; the complete set of parameters for this function must be terminated by an asterisk (*).

End of parameters

A card or line with four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a LOad or FInd message. Details are given in the section *Parameter formats*, page 2.

Examples

1 FUNCTION 1 (with parameter types 1 and 4 only: file area descriptions specifying areas)

A serial file called DISCFILE is to be created, with generation number 0, deallocation retention period 30 days, integrity code 3, a bucket size of 2 blocks and 2 word bucket headers. The file index and parameter entry for file DISCFILE is to be held on unit 200000 and the file is to have four file areas. Bucket headers are not to be written to the file.

File areas 1 and 2 are to be created on unit 200000, occupying cylinders 20 to 50, blocks 0 to 39 and cylinders 80 to 88, blocks 0 to 79 respectively. File areas 3 and 4 are to be created on unit 300000, occupying cylinders 10 to 24, blocks 0 to 39 and cylinder 60, blocks 0 to 79 respectively.

The parameters required to create this file would take the following form (if input on cards):

card 1 XPJC1,1,200000,DISCFILE,0,30,3,2,
card 2 XPJC4,S(20,50,0,39)(80,88,0,79)
card 3 XPJC4(10,24,0,39,300000)(60,60,0,79)*
card 4 ****

2 FUNCTION 1 (with parameter types 1 and 4: file areas to be created anywhere)

A serial file called DISCFILEV2 is to be created, with generation number 0, deallocation retention period 20 days, integrity code 2, a bucket size of 4 blocks and 2 word bucket headers. The file index and parameters entry for DISCFILEV2 is to be held on unit 777001 and the file is to have up to 12 unspecified file areas, with up to six on unit 777001 and up to six on unit 777002. The allocation of buckets to the file is to be such that 160 buckets are to be contained in one file area, 240 buckets in up to 3 file areas, 350 buckets in up to 3 file areas, 80 buckets in up to 2 file areas and 40 buckets in one file area. Bucket headers are not to be written to the file.

Note: In certain cases Executive will round up the number of buckets allocated. For details see section *Extending and contracting files*, Chapter 14 of *Direct Access*.

The parameters required to create this file would be as follows (if input on cards):

card 1 XPJC1,1,777001,DISFILEV,0,20,2,4,
card 2 XPJC4,A(1,160)(3,240)(2,350)
card 3 XPJC4(3,350,777002)(2,80)(1,40)*
card 4 ****

3 FUNCTION 1 (with parameter types 1, 2 and 4 only)

A serial file call PAYROLLV3 is to be created having the same basic details as DISCFILE in example 1. In addition, it is to have variable length records, serial organization, a maximum record size of 100 words and

a key length of 15 characters starting at the 6th character from the beginning of a record. Two word bucket headers are assumed. Bucket headers are not to be written to the file.

The parameters required to perform this function would take the following form (if input on paper tape):

XPJC1,1,200000,PAYROLLV3,0,30,3,2,↓

XPJC2,V,S,100,15,5,0,↓

XPJC4,S(20,50,0,39)(80,88,0,79)↓

XPJC4(10,24,0,39,300000)(60,60,0,79)*↓

where ↓ indicates a new line character

4 FUNCTION 1 (with parameter types 1, 2, 3 and 4)

A file called ACCOUNTVPR3 is to be created having the same details as PAYROLLV3 in example 3, except that the file is to have generation number 40 and deallocation retention period 0, own monitoring of flaws and is to be sequential. In addition, the seek area packing density is to be 75% and the size of the auxiliary header six words. Index tables, having two buckets at each level of indexing, are required. Five seek areas are to be allocated as a second level overflow area. Bucket headers will be written to the file.

The parameters required to perform this function would take the following form (if input on cards):

card 1 XPJC1,1,200000,ACCOUNTVPR3,40,0,3,2,M,

card 2 XPJC2,V,Q,100,15,6,5,2,

card 3 XPJC3,75,6,1,2,2,2,

card 4 XPJC4,S(20,50,0,39)(80,88,0,79)

card 5 XPJC4(10,24,0,39,300000)(60,60,0,79)*

5 FUNCTION 2

The two files specified in examples 1 and 4, DISCFIL (with generation number 0 and an expired retention period) and ACCOUNTVPR3 (generation number 40) are to be deleted.

The parameters required to perform this function would take the following form (if input on paper tape):

XPJC1,2,200000,DISCFIL,0,*↓

XPJC1,2,200000,ACCOUNTVPR3,40,*↓

where ↓ is a newline character

6 FUNCTION 3 (with parameter types 1 and 4 only)

The file DISCFIL specified in example 1 is to have an additional file area created on unit 200000, occupying cylinders 50 to 53, blocks 0 to 79. Bucket headers are not to be written to the additional file area.

The parameters required to perform this function would take the following form (if input on cards):

card 1 XPJC1,3,200000,DISCFIL,0,

card 2 XPJC4,S(50,53,0,79,200000)*

7 FUNCTION 3 (with parameter types 1, 3 and 4)

The file ACCOUNTVPR3 specified in example 4 is to have an additional file area created on unit 300000, occupying cylinders 61 to 81, blocks 0 to 79. This additional file area is to have the same details as the existing file areas, and bucket headers are to be written to it.

The parameters required to perform this function would take the following form (if input on paper tape):

XPJC1,3,200000,ACCOUNTVPR3,40,↓

XPJC3,75,6,1,2,2,2,↓

XPJC4,S(61,81,0,79,300000)*↓

#XPJC

where ↓ is a newline character

8 FUNCTION 4

The file ACCOUNTVPR3 specified in example 4 is to have all the file areas on unit 300000 deleted (including the area created in example 7).

The parameters required to perform this function would take the following form (if input on cards):

card 1 XPJC1,4,200000,ACCOUNTVPR3,40,5,*

9 CREATING A PROGRAM LIBRARY FILE (FUNCTION 1)

When #XPJC is used to create a program library file, parameter types 1 and 4 of function 1 are always required, the file name must be of the form PROGRAMVname (where name consists of four alphanumeric characters of the user's choice with the first character a letter).

The following parameters would create the serial program library file named PROGRAMVXYZ4 with generation number 69, deallocation retention period 5 days, integrity code 3, one block buckets, two word bucket headers with one file area occupying cylinders 20 to 40, blocks 0 to 79 of unit 550000. The file index and parameters entry is to be held in the system control area of unit 550000. Bucket headers are not to be written to the file.

The parameters would take the following form (if input on paper tape):

XPJC1,1,550000,PROGRAMVXYZ4,69,5,3,1,↓

XPJC4,S(20,40,0,79,550000)*↓

where ↓ indicates a newline character

10 BATCHING FUNCTIONS AND LOADING FOLLOWING PROGRAM

The allocational and deletions described in examples 3, 5 and 9 above are to be done in one run of #XPJC and it is required that the program #XPJK be called at the end of the run.

The parameters would take the following form (if input on paper tape):

XPJC1,1,200000,PAYROLLV3,0,30,3,2,↓

XPJC2,V,S,100,15,5,0,↓

XPJC4,S(20,50,0,39)(80,88,0,79)↓

XPJC4(10,24,0,39,300000)(60,60,0,79)*↓

XPJC1,2,200000,DISCFILE,0,*↓

XPJC1,2,200000,ACCOUNTVPR3,40,*↓

XPJC1,1,550000,PROGRAMVXYZ4,69,5,3,1,↓

XPJC4,S(20,40,0,79,550000)*↓

***FIV#XPJKV#DISC↓

where ↓ indicates a newline character

OPERATING

Executive priority

The program as supplied has a priority of 50.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0	Parameters	The reader is assigned as the beginning and is released as soon as the terminator parameter has been read
LP0	Report and error messages	The line printer is assigned at the beginning of the run and is used throughout the run
DA1	Units on which new files or file areas are to be created or deleted	Used throughout the run
DA3	Work file	Used when needed and released when no longer required

Running under Executive alone**OPERATING INSTRUCTIONS**

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJC (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required storage units	
3 Load the parameters in the appropriate reader	
4 To activate the program:	
(a) For parameters on paper tape, input:	GO #XPJC 20
(b) For parameters on cards, input:	GO #XPJC 21
5 When the allocation or deallocation is complete the program will output one of the following messages:	
(a) If a delete message was correctly input, and a successful run had occurred:	DELETED:- <i>user message</i>
(b) If the run was error free and no delete message was input:	HALTED:- HH
(c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:	DISPLAY:- DE HALTED:- HH
(d) If any error had occurred during the run (except the one described in (c) above), whether or not a delete message was input: (see also section <i>Line printer error messages</i> , page 29)	HALTED:- ZZ
6 For a further run, return to Operating Instruction 2	

#XPJC

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJC during execution and loading. Prior to entry into #XPJC it must have been loaded correctly and this is signalled by HALTED:— LD. When a loading exception condition occurs before HALTED:—LD reference must be made to the section *Exception conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:— CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJC or (b) Abandon the run
2	DISPLAY:— DE	The end parameter delete message did not start with the characters LO or FI	The run continues
3	HALTED:— LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJC or (b) Abandon the run
4	DISPLAY:— PE	Transfer failure on the line printer	No action. The run continues
5	DISPLAY:— PL	A paper low condition has been detected	Load more paper and re-allocate the printer
6	HALTED:— RF	A card or paper tape block has been misread	To reread the parameter line reload the card/block and input: GO #XPJC
7	HALTED:— TR	There is no tape reader available	Either (a) Make a tape reader available and input: GO #XPJC or (b) Abandon the run
8	HALTED:— xx	An error has been detected when loading #XPJC by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when executive attempts to open the storage unit it will output a storage unit not on-line message, for example #XPJC; LDC * nnnnnn (the actual format of the message varies according to which executive is being used). The user then has two options to take:

- 1 Put the requested store unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJC 27 to abandon the current function and go to the next input

LINE PRINTER ERROR MESSAGES

If any of the line printer messages listed in this section are printed, the run will terminate with the console message **HALTED:— ZZ.**

The error messages are numbered from one. These numbers do not appear in the actual printout but are included as a guide to indicate with which function each error may occur, as follows:

Errors that may occur with all functions: 1, 4, 12, 19, 21 and 23

Errors that may occur with function 1: 2, 3, 5, 8, 9, 14, 15, 16, 18, 20 and 22

Errors that may occur with function 2: 6, 7, 10, 11 and 13.

Errors that may occur with function 3: 2, 5, 6, 7, 9, 10, 14, 15, 16, 18, 20 and 22.

Errors that may occur with function 4: 7, 10 and 17.

Note: With the exception of errors 3, 4, 16, 17, 19, 21 and 23 the rest of the errors result from Executive replies on unsuccessful attempts to open, create, delete, contract and extend files. For further information see *Direct Access* (Chapter 14).

#XPJC

#XPJC	MKxx	dd/mm/yy	hh/mm/ss	FILE ALLOCATOR				PAGEyyyy
		FILE IDENTITY	FGN	FILE RET	DATA RET	TOTAL BKTS	2ND LEVEL ST BKT	FUNCTION
1 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		SYSTEMS CONTROL AREA FULL			NO ACTION
2 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	PAR TYPE 4, NUMBER x, DESCRIPTION xx WHOLE (OR PART) OF AREA ALREADY IN USE	XXXX	XXXXXX XXXXXXXX	ALLOCATED
3 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		TOO MANY 2ND LEVEL OVERFLOW SEEK AREAS			NO ACTION
4 ***		XXXXXXXXXX XXXXXXXXXX (where ↑ indicates the error)	XXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXX		PARAMETER ERROR ↑			NO ACTION
5 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	PAR TYPE 4, NUMBER x, DESCRIPTION xx NO BLOCKS AVAILABLE ON *XXXXXX	XXXX	XXXXXX XXXXXXXX	EXTENDED
6 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		FILE OPENED BY ANOTHER PROGRAM			NO ACTION
#XPJC	MKxx	dd/mm/yy	hh/mm/ss	FILE ALLOCATOR				PAGEyyyy
		FILE IDENTITY	FGN	FILE RET	DATA RET	TOTAL BKTS	2ND LEVEL ST BKT	FUNCTION
7 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		FILE NOT IN SYSTEM			NO ACTION
8 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		DUPLICATE ON *XXXXXX			NO ACTION
9 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	PAR TYPE 4, NUMBER x, DESCRIPTION xx SPECIFIED NUMBER OF FILE AREAS TOO SMALL	XXXX	XXXXXX XXXXXXXX	ALLOCATED
10 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		INTEGRITY CODE FAILURE			NO ACTION
11 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	DATA RETENTION PERIOD NOT EXCEEDED			NO ACTION
12 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		INSUFFICIENT EXECUTIVE CORE			NO ACTION
13 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	FILE RETENTION PERIOD NOT EXCEEDED			NO ACTION
#XPJC	MKxx	dd/mm/yy	hh/mm/ss	FILE ALLOCATOR				PAGEyyyy
		FILE IDENTITY	FGN	FILE RET	DATA RET	TOTAL BKTS	2ND LEVEL ST BKT	FUNCTION
14 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	PARITY ERROR IN BUCKET XXXXXXXX	XXXX	XXXXXX XXXXXXXX	EXTENDED
15 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		INSUFFICIENT PROGRAM CORE			NO ACTION
16 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	SEEK AREA(S) TOO BIG FOR 2 WD BUCKET HDRS	XXXX	XXXXXX XXXXXXXX	ALLOCATED
17 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX		TOO MANY FILE AREAS BEING DELETED			NO ACTION
18 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	FLAW AREA FULL	XXXX	XXXXXX XXXXXXXX	ALLOCATED
19 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	SEQUENCE ERROR, PARAMETER XPJCx, SEQ XXXX	XXXX	XXXXXX XXXXXXXX	EXTENDED
20 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	PAR TYPE 4, NUMBER x, DESCRIPTION xx INVALID FILE AREA STRUCTURE	XXXX	XXXXXX XXXXXXXX	ALLOCATED
21 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	ABANDON ENTRY POINT USED	XXXX	XXXXXX XXXXXXXX	EXTENDED
22 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX	TOO MANY FENCES BEING CREATED	XXXX	XXXXXX XXXXXXXX	ALLOCATED
2-3 ***		XXXXXXXXXX XXXXXXXXXX	XXXX XXXX	XXXX		XXXX	XXXXXX XXXXXXXX	NO ACTION

Note: Message 23 only occurs when an asterisk or **** terminator is missing.

Figure 2 Line printer error message output by #XPJC

NAME

#XPJK

VERSION

Mark 1A

TITLE

File or cartridge report.

HARDWARE REQUIREMENT

10880 words of store

Optionally 1 card reader or 1 paper tape reader

1 line printer

1 or more cartridges

DESCRIPTION**General**

Program #XPJK reports on individual files on a cartridge (*file report*) or on the state of a cartridge as a whole (*cartridge report*). There is no limit to the number of reports which may be obtained in any one run.

FILE REPORT

For each file specified, the program prints the details of the file as held in the file index cell, followed by a table giving details of all the file areas allocated to the file. The program validates this information and reports on any errors found (see Figure 3, page 32).

CARTRIDGE REPORT

The following options are available:

<i>Report types</i>	<i>Contents</i>
F	A table of all the files on the cartridge in alphabetical order. (Except if a filename or file generation number is changed by another program accessing a file during the processing of this table, then that file will be out of order in the table.) The table gives details of the files as held in the file index cells
FA	As for report type F, plus, for each file, a list of all its file areas in order of allocation
M	As for report type F, plus a map of the cartridge
MA	As for report type FA, plus a map of the cartridge

The following information is output after each of the above report types and is by default the minimum report (see Figure 4, page 33).

- 1 A table of file areas allocated to files on other cartridges
- 2 A table of unallocated areas on the cartridge
- 3 General information about the cartridge

Reports may be obtained for any cartridge conforming to unified direct access standards (see *Direct access*) subject to the following limits:

Maximum number of blocks per cylinder	1000
Maximum number of cylinders	1000

#XPJK

The program will attempt to increase its store in the following situations:

- 1 If there are more than 1090 files on a cartridge the program attempts to extend its store in steps of 256 words to cater for the residual files. Failure to extend results in these residual files not being in alphabetic order
- 2 When mapping a cartridge larger than an EDS8 the program attempts to run at maximum efficiency by extending its store to the following approximate values:

EDS30	—	13 Kwords
EDS60	—	21 Kwords
EDS200	—	32 Kwords

The only result of a failure to extend store is reduced efficiency.

The user may restrict the extending of the program's store size by specifying a maximum store value in the parameters (in the absence of this parameter a default maximum store size of 32K words is used)

The program validates the information being reported on and lists any errors found. For full validation of the system control area of a cartridge a map must be requested.

Input

Input to #XPJK consists of:

- 1 One or more cartridges on which reports are to be made
- 2 Parameters to the program
 - (a) These may be on cards or paper tape. Each parameter line is processed as it is read, and hence there is no limit to the number of files and/or cartridges which may be reported on. The reader will be released before the reports specified in the parameter line immediately preceding the terminator are printed
 - (b) When the program is being run under GEORGE 3 or GEORGE 4, entry point 2 may be used to read parameters with the GEORGE Command Issuer (see section *Examples of running under GEORGE 3 or GEORGE 4*, page 40)
 - (c) A type MA cartridge report can be produced by ALTERing word 0 of the program to the cartridge serial number and using entry point 3 (see section *Operating Instructions*, page 40)

Output

Output is always to the line printer. If no switches are set, the program requests a printer with at least 120 print positions and then uses the full print width of the printer allocated wherever appropriate.

Alternatively switch 0, 1 or 2 can be set to request a printer with at least 120, 132 or 160 print positions respectively and then use no more than 120, 132 or 160 print positions even if more print positions are available. (See *Operating Instruction 3*, page 40.)

EXAMPLES

#XPJK MK1A																		10AUG78		12/41/30		REPORT ON SPECIFIED FILES										CSN 721101				PAGE 1			
FILE NAME		GEN NO		VERS NO		INT CDE		DATE WRITTEN		DATA RETN		FILE RETN		OWN MON		FIL ORG		EOB TAG		ST LEV		2ND O/F		BLK BKT		WAS MDR		REC LTH		KEY POSN		KEY LTH		NO F/A					
SUBGROUPS2CB		66		0		3		1AUG78		0		0		N		SR		889		0		1		2		126		V		0		0		4					
F/A NO		ON CSN		ST CYL		END ENH		ST ENH		NO BKT		F/A NO		ON CSN		ST CYL		END ENH		ST ENH		NO BKT		F/A NO		ON CSN		ST CYL		END ENH		ST ENH		NO BKT					
1		721101		5		6F		0		295		392		2		721101		7		7		160		295		136		3		721101		8		0		79		80	
4		721101		8		8		80		150		80																											
TOTAL BKTs =																																							

Figure 3 A file report

[illegible]

#XPJX MK1A	10AUG78	12/43/15	FILE AREA TABLES AND GENERAL INFORMATION	CSN 716405	PAGE 2									
FILE AREAS ALLOCATED TO FILES ON OTHER CARTRIDGES														
HOME CSN	ST CYL	END RLK	NO BLKS	HOME CSN	ST CYL	END RLK	NO BLKS	HOME CSN	ST CYL	END RLK	NO BLKS			
721101	402	405F	0 23	96	721101	398	401F	0 23	96	721101	394	397F	0 23	96
721101	386	389F	0 23	96	721101	368	371F	0 23	96	721101	372	385F	0 23	136
721101	33	35F	0 23	72										
TOTAL BLKS = A88														
UNALLOCATED AREAS														
ST CYL	END BLK	NO BLKS	ST CYL	END BLK	NO BLKS	ST CYL	END BLK	NO BLKS						
36	106F	0 23	1704	330	351F	0 23	528							
TOTAL BLKS = 2232														
CYLINDERS ON CARTRIDGE 406									BLOCKS PER CYLINDER 24			BLOCKS ON CARTRIDGE 9744		
FILE INDEX CELLS 16 18									START OF OVERLAY AREA 24					
FILE AREA CELLS 96 112									END OF USED OVERLAY AREA 24					
FLAW BLOCKS 0 7									END OF RESERVE OVERLAY AREA 266					
SCRATCH INHIBIT CLEAR									FVCHANGEABLE					

4444(1.79)

#XPJK

#XPJK MK1A		8AUG78	11/43/04	FILE/FILE AREA TABLE										CSN 716005										PAGE 1	
FILE CODE	FILE NAME	GEN NO	VERB NO	INT CDE	DATE WRITTEN	DATA RETN	FILE RETN	OWN MON	FIL ORG	EOF TAG	ST LEV	2ND O/P	BLK BKT	WDS HAN	REC LTH	KEY POSN	KFY LTH	ST CYL	END BLK	ST BLK	END BLK	WDS	BUFS		
SS	ICLSCAFILE	0	0	3	70EC73	n	0	Y	80	1	0	1	2	0	V	0	0	0	10F	0	23	964			
AA	ICLAWDEFAULT	10	3	3	1840V77	n	0	Y	SR	1	0	1	2	126	V	0	0	190	393F	0	23	96			
AB	ICLKGEOMCHRO	1	1	3	238VP76	n	0	N	SR	1	0	1	2	126	V	0	0	178	184F	0	23	168			
AC	ICLKGEOMCHRO	2	0	3	2640V76	n	0	N	SR	1	0	1	2	126	V	0	0	319	325F	0	23	168			
AD	INPUT FILE	4	0	3	31MAR77	n	0	Y	SR	1	0	8	2	1022	V	0	0	330	331F	0	23	64			
AE	PROGRAM XALE	1	0	3	2240V77	0	0	N	SR	256	0	1	2	126	V	0	0	240	249F	0	23	960			
																		258	261F	0	23	96			
																		364	367F	0	23	96			
																		TOTAL				432			
AF	PROGRAM XAPH	1	0	3	104AV77	n	0	N	SR	21	0	1	2	126	V	0	0	292	295F	0	23	96			
AG	PROGRAM XPSA	1	0	3	78VP77	n	0	N	SR	393	0	1	2	126	V	0	0	215	231F	0	23	468			
AH	PROGRAM XJGA	1	0	3	13JAN75	n	0	N	SR	17	0	1	2	126	V	0	0	305	314F	0	23	960			
AI	PROGRAM XPJ2	99	0	3	208VP77	n	0	N	SR	30	0	1	2	126	V	0	0	126	329F	0	23	96			
AJ	PROGRAM XPL2	0	0	3	1940V76	n	0	N	SR	120	0	1	2	126	V	0	0	185	194F	0	23	960			
AK	PROGRAM XPS5	0	0	3	340V76	n	0	N	SR	36	0	1	2	126	V	0	0	195	198F	0	23	96			
AL	SUBGROUPS2F7	0	0	3	30AUG77	n	0	N	SR	557	0	1	2	126	V	0	0	153	161F	0	23	916			
																		162	165F	0	23	96			
																		166	169F	0	23	96			
																		170	173F	0	23	96			
																		174	177F	0	23	96			
																		TOTAL				400			

#XPJK MK1A		8AUG78	11/43/04	MAP OF CARTRIDGE										CSN 716005										PAGE 4		
BLK		CYLINDER																								BLK
000		002	004	006	008	010	012	014	016	018	020	022	024	026	028	030	032	034						000		
001		003	005	007	009	011	013	015	017	019	021	023	025	027	029	031	033	035						008		
000	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	000	
008	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	008	
016	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	016	
036		038	040	042	044	046	048	050	052	054	056	058	060	062	064	066	068	070						000		
037		039	041	043	045	047	049	051	053	055	057	059	061	063	065	067	069	071						008		
000	IAQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	000		
008	AQ	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	008		
016	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	016		
072		074	076	078	080	082	084	086	088	090	092	094	096	098	100	102	104	106						000		
073		075	077	079	081	083	085	087	089	091	093	095	097	099	101	103	105	107						008		
000	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	000		
008	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	008		
016	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	AQ	016		
108		110	112	114	116	118	120	122	124	126	128	130	132	134	136	138	140	142						000		
109		111	113	115	117	119	121	123	125	127	129	131	133	135	137	139	141	143						008		
000	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	000		
008	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	008		
016	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	AP	016		
324		326	328	330	332	334	336	338	340	342	344	346	348	350	352	354	356	358						000		
325		327	329	331	333	335	337	339	341	343	345	347	349	351	353	355	357	359						008		
000	AC	AI	AI	AI	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	000		
008	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	008		
016	AC	AI	AI	AI	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	AD	016		
360		362	364	366	368	370	372	374	376	378	380	382	384	386	388	390	392	394						000		
361		363	365	367	369	371	373	375	377	379	381	383	385	387	389	391	393	395						008		
000	IAH	AM	AM	AM	AE	AE	AE	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	000		
008	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	008		
016	AM	AM	AM	AM	AE	AE	AE	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	016		
396		398	400	402	404																			000		
397		399	401	403	405																			008		
000	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	000		
008	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	008		
016	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	SS	016		

UNALLOCATED AREAS ARE LEFT BLANK

I INDICATES THE START OF A FILE AREA

SS REPRESENTS "DITTO"

== REPRESENTS FILE AREAS ALLOCATED TO FILES ON OTHER CARTRIDGES

Figure 6 A file table and map produced by an MA type cartridge report for an ED5

#XPJK MK1A		9AUG78		14/10/58		FILE TABLE		CSN 721101												PAGE 23		
FILE CODE	FILE NAME	GEN NO	VERS NO	INT CDE	DATE WRITTEN	DATA RETN	FILE RETN	OWN MON	FIL ORG	EOB TAG	ST LEV	2ND O/P	BLK BKT	WDS REC	REC LTH	KEY POSN	KEY LTH	NO P/A	TOTAL SKTS			
D2	TABLE	0	11	3	18APR78	0	0	N	SR	1			1	126	V	0	0	1	80			
D3	TESTFILE	0	1	3	18APR78	0	0	N	SR	1			2	0	V	0	0	1	1036			
D4	WASTE-1	0	0	3	18APR78	4095	0	N	SR	1			2	1022	V	0	0	1M	12			
D5	WORK1	0	1	3	18APR78	0	0	N	SR	1			2	126	V	0	0	1	296			
D6	WORK2	0	0	3	18APR78	0	0	N	SR	1			2	126	V	0	0	1	296			
D7	WORKFILE04	0	1	3	18APR78	1	0	N	SR	7			2	126	V	0	0	1	296			
D8	WUAB=SOURCE	0	2	3	18APR78	0	0	N	SR	1			2	126	V	0	0	1	216			
D9	WUBA=SOURCE	0	3	3	18APR78	0	0	N	SR	1			2	126	V	0	0	4	312			
EA	WUCA=SOURCE	0	2	3	18APR78	0	0	N	SR	1			2	126	V	0	0	1	296			
ER	WUCH=SERIAL	0	0	3	5JUN78	0	0	N	SR	23			2	126	V	0	0	4	248			
EC	WUDA=SERIAL	0	1	2	30MAY78	0	0	N	SR	210			3	12	V	4	4	1	296			
ED	WUDA=SERIALB	0	0	2	30MAY78	0	0	N	BQ	101			2	12	V	4	4	1	444			
EE	WUDA=SERIALC	0	0	2	30MAY78	0	0	N	SO	51			2	12	V	4	4	1	296			
EF	WUDA=SERIALD	0	1	2	30MAY78	0	0	N	SO	26		121	8	3	12	V	4	4	2	180		
EG	WUDA=SERIAL	0	1	3	18APR78	0	0	N	SR	1			2	126	V	0	0	1	296			
EH	WUDB=RANDOM	0	1	1	18APR78	0	0	N	RN	1			3	12	V	4	4	1	200			
EI	WUDB=RANDOM	1	0	1	31MAY78	0	0	N	SO	43			3	12	V	4	4	1	320			
EJ	WUDB=RANDOM	2	3	1	31MAY78	0	0	N	RN	1		71	8	3	12	V	4	4	1	80		
EK	WUDB=RANDOM	3	6	1	7AUG78	0	0	N	RN	1		121	4	3	12	V	4	4	1	140		
EL	WUDB=RANDOM	4	2	1	31MAY78	0	0	N	RN	1			3	12	V	4	4	1	200			
EM	WUDB=RANDOM	22	0	3	18APR78	0	0	N	SR	43			2	126	V	0	0	1	128			
EN	WUDB=RANSEF	2	0	3	18APR78	0	0	N	SR	1			2	126	V	0	0	1	80			
EP	WUDI=FILE	0	0	2	29JUN78	0	0	N	SR	8			2	126	V	0	0	1	72			
EQ	WUDI=SERIAL	0	1	3	18APR78	0	0	N	SR	1			2	1022	V	0	0	1	37			
ER	WUWA=SOURCE	0	1	3	18APR78	0	0	N	SR	1			2	126	V	0	0	1	296			
ES	WUIA=SOURCE	0	2	3	18APR78	0	0	N	SR	1			2	126	V	0	0	1	296			

#XPJK MK1A		9AUG78		14/10/58		MAP OF CARTRIDGE		CSN 721101												PAGE 33		
BLK		CYLINDER																				BLK
		180	182	184	186	188	190	192	194	196	198	200	202	204	206	208	210	212	214			
		181	183	185	187	189	191	193	195	197	199	201	203	205	207	209	211	213	215			
000	AU AU AU AU AU AU AU						1A7C2ICR1ER1C2		ICJ1BD			ICL1A2:DR DR					1D5:06	1EVI:CC:DX:DX			000	
008							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			008	
016							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			016	
024							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			024	
032							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			032	
040							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			040	
048							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			048	
056							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			056	
064							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			064	
072							A7 C2 CA ER C2		CJ BD			CL A2 DR DR					05 D6	FV CC DX DX			072	
080							1A7:01		ICA:ER:CI		105	1DQ	DR DR	183			05 D6	FV CC DX DX			080	
088							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			088	
096							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			096	
104							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			104	
112							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			112	
120							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			120	
128							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			128	
136							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			136	
144							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			144	
152							A7 B1		CA ER CI		05	00	DR DR	03			05 D6	FV CC DX DX			152	
160							AU:06:CC:03:0F:00		1C3		104	1B3	DR DR	03			05 D6	FV CC DX DX			160	
168							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			168	
176							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			176	
184							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			184	
192							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			192	
200							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			200	
208							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			208	
216							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			216	
224							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			224	
232							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			232	
240							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			240	
248							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			248	
256							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			256	
264							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			264	
272							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			272	
280							AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			280	
288	AU AU AU AU AU AU AU						AU:06:CC:03:0F:00		C3		04	03	DR DR	03			05 D6	FV CC DX DX			288	

Figure 7 A file table and map produced by an M type cartridge report for an EDS60

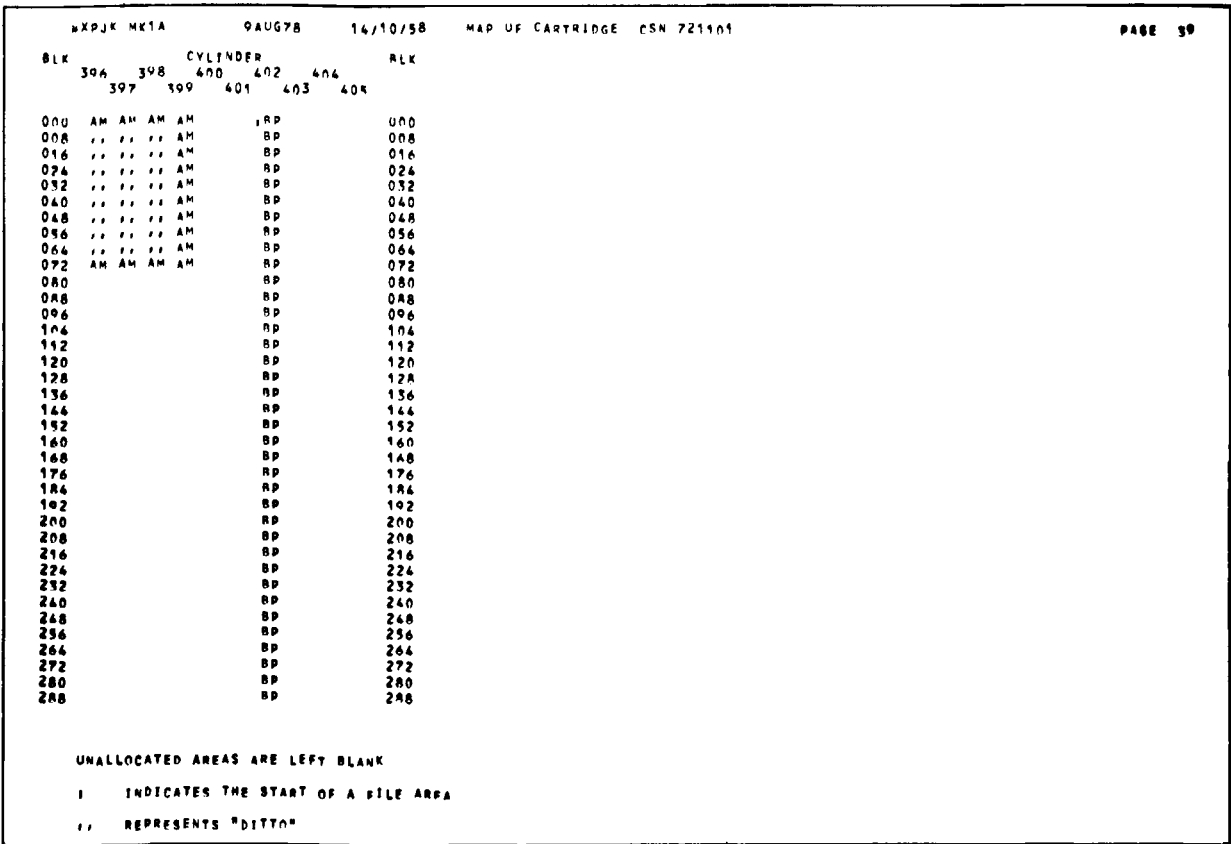


Figure 8 The last page of a map of an EDS60

FILE DETAILS

The following headings are used when reporting the contents of the file index cells of files:

Heading	Data under the heading
FILE CODE	A two character code by which the file is represented on the map, their being 1088 codes available. When all the codes are exhausted they are repeated. This heading is present only if a map was requested
FILE NAME	Name of the file. In a cartridge report, the first 1090 files will always be ordered alphabetically. Further files are ordered provided sufficient store can be obtained
GEN NO	Generation number
VERS NO	Version number
INT CDE	Integrity code (1, 2 or 3)
DATE WRITTEN	The date on which the file was last opened for writing
DATA RETN	Data retention period
FILE RETN	Deallocation file retention period
OWN MON	Own monitoring of flaws? (Y or N)
FIL ORG	File organisation SR – serial, SQ – sequential, RN – random
EOF TAG	The end of file bucket number recorded in the file index cell

Heading	Data under the heading
ST 2ND LEV O/F	The bucket number of the start bucket of the second level overflow area
BLK BKT	Number of blocks per bucket (1, 2, 4 or 8)
BKT HDR	Bucket header size in words (2 or 3)
WDS REC	Maximum number of words per record
REC LTH	Fixed or variable length records (F or V)
KEY POSN	The number of characters preceding the key in the record
KEY LTH	Key length in characters
NO F/A	The number of file areas allocated to the file. In F and M type cartridge reports a suffix M to this field indicates that at least one file area is on another cartridge (that is, it is a multicartridge file).
	This heading is not present in an MA type cartridge report

For all these headings, the value found in the file index cell is printed whether it is valid or not. Any invalid values are reported in an error line.

FILE AREA DETAILS

The following headings are used when reporting the information associated with the various types of file area cell:

Heading	Data under the heading
F/A NO	The number of the file area in a file. This heading is only present in a file report
ON CSN	The serial number of the cartridge holding the file area
HOME CSN	The serial number of the cartridge holding the file index cell of the file to which the file area is allocated
ST CYL	The number of the first cylinder used by the file area
END CYL	The number of the last cylinder used by the file area. A suffix F to this field indicates that the file area uses the full cylinder
ST BLK	The number of the first block used in each cylinder
END BLK	The number of the last block used in each cylinder
NO BLKS	The number of blocks in the file area
NO BKTS	The number of buckets in the file area of a file
TOTAL BLKS	The total number of blocks in the file areas of a file area chain
TOTAL BKTS	The total number of buckets in a file

In a file report, values are printed under all these headings whether they are valid or not. Any invalid values are reported in an error line.

In a cartridge report, an error found in the chain of file areas allocated to a file on the cartridge causes the whole chain to be considered corrupt. The file area description is filled with asterisks and an error line is printed. In the chain of unallocated areas and the chain of file areas allocated to files on other cartridges, all file areas up to the one in error are described, the fields describing the one in error are filled with asterisks and then an error line is printed.

Fields in the two tables given in section *General Information*, page 31 will be filled with asterisks if they are invalid or cannot be calculated. An error line is printed identifying any problems.

MAP

A map of a cartridge shows file areas allocated to files on the cartridge, file areas allocated to files on other

#XPJK

cartridges (code ==) and unallocated areas. Unallocated areas are left blank and pages of a map which would be totally blank are not printed. The start of each allocated area is indicated by a colon, the edges of the area are defined by its file code and the middle, if there is one, is filled with the code , ,. In error situations, the map shows any undefined areas (code --) and any areas with multiple references (code ??).

CONTROL

This section describes the parameters associated with the two types of reports described in the section *Description*, page 31. Further details of the formats of parameters are described in the section *Parameter Formats*, page 2.

File report parameters

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJK) and parameter type	XPJK1	5 (fixed)
*2	File name of the file to be reported on	Up to 12 alphanumeric, space (V) or minus (–) characters, with the first character alphabetic	12
*3	Generation number of the file to be reported on	–1, 0 to 4095 Note: If this is not specified, a generation number of zero is assumed. If this is specified as –1 the files with the highest generation number will be reported on	4
*4	Serial number of the cartridge holding the file to be reported on	1 to 777777 (an octal number) Note: If this is omitted, Executive will search for the file with the specified file name and generation number	6

* The group of parameter fields 2 to 4 must be enclosed in parentheses and may be repeated if more than one file is to be reported on. A group must be wholly contained on one line. The maximum number of characters in a parameter line is 76 (not including the newline character on paper tape).

If all the files to be reported on cannot be specified on the first line then further parameter lines must be used.

Each parameter field must be followed by a comma except when followed by a parenthesis.

Sequence numbers are not checked.

Cartridge report parameters

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJK) and parameter type	XPJK2	5 (fixed)
2	Maximum number of words of store (as a multiple of 1024) that the program may occupy	11 to 32 Note: If this is omitted a value of 32 is assumed	2 (fixed)
*3	Serial number of the cartridge to be reported on	1 to 777777 (an octal number)	6

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*4 Report type	One of F, FA, M or MA (as described in section <i>Cartridge Report</i> , page)	2
	Note: If this is omitted the minimum report is given	

*The group of parameter fields 3 and 4 must be enclosed in parentheses and may be repeated if more than one cartridge is to be reported on. A group must be wholly contained on one line. The maximum number of characters in a parameter line is 76 (not including the newline character on paper tape). If all the cartridges to be reported on cannot be specified on the first line then further parameter lines must be used.

Each parameter field must be followed by a comma except when followed by a parenthesis.

Sequence numbers are not checked.

End of parameters

A card or line with four asterisks (****) in the first four character positions must follow a batch of parameters. These asterisks may be followed by a LOad or FInd message – details are given in the section *Parameter Formats*, page 2.

Examples

- 1 FILE REPORT Reports are required on files PROGRAMUTIL with generation number 1, PAYROLLDATA with generation number 13, INPUT-FILE with generation number 123 and DATABASE400 with generation number 0. PAYROLLDATA and INPUT-FILE are held on cartridge 000152 and DATABASE400 on cartridge 654001.

The parameters required to perform these reports would take the following form (if input on cards):

```
card 1 XPJK1 (PROGRAMUTIL,1) (PAYROLLDATA,13,152) (INPUT-FILE,123,152)
card 2 XPJK1 (DATABASE400,,654001)
card 3 ****
```

- 2 CARTRIDGE REPORT

- (a) For cartridge 000154 a table of all the files, without a list of file areas allocated to each, is required. For cartridge 654006 a map is required with a list of the file areas allocated to each file in the file table. For cartridge 000153 a minimum report is required. The program may measure its store size up to 32K words.

The parameters required to perform these reports would take the following form (if input on paper tape):

```
XPJK2(154,F) (654006,MA) (153)↓
****↓
```

where ↓ represents a newline character

- (b) A map of an ED60 007546 is required with the associated file table not giving lists of allocated file areas. Also it is required to ensure the program does not increase its store size beyond 15 Kwords. #XPJW is to be loaded after the termination of the program

The parameters required to perform this report would take the following form (if input on cards):

```
card 1 XPJK2,15(7546,M)
card 2 ****FIV#XPJW#UTIL
```

OPERATING

Trusted status

The program has Q-trusted status.

#XPJK

Executive priority

The program as supplied has a priority of 50.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0 (optional)	Parameters	The reader is assigned as required and is used to queue the parameters. It is released before the reports specified in the parameter line, immediately preceding the terminator of four asterisks, are done
LP0	Output of reports	The line printer is used throughout the run
DA0	File being reported on	Used while reporting on specified files
DA1	System file of the cartridge being reported on	Used while reporting on cartridges

Running under Executive alone

OPERATING INSTRUCTIONS

Narrative

Console message

- 1 Load the program #XPJK (see FIND A BINARY PROGRAM in *Compiling Systems*)

HALTED:— LD

The message HALTED:—LD will only be output if the loading operation was successful

- 2 Load the required cartridge(s)
- 3 If the executive in use interprets property codes then the program will use the full width of the line printer provided. If the executive in use does not interpret property codes then the program uses 120 print positions

Provided the executive in use checks property codes, a specific print width can be specified as follows:

(a) For a print width of 120 print positions	ON #XPJK0
(b) For a print width of 132 print positions	ON #XPJK1
(c) For a print width of 160 print positions	ON #XPJK2

Note: If switches are set with an executive that does not check property codes, #XPJK assumes the requested print width whatever the width of the line printer provided. Where the line printer width obtained is the lesser, then information will be lost. For example a column of *unallocated area* information.

- 4 To activate the program:

either

(a) If parameters are on paper tape, input:	GO #XPJK20
or	
(b) If parameters are cards, input:	GO #XPJK21
or	
(c) For an MA type cartridge report input the cartridge serial number <i>x</i> of the cartridge to be reported on	AL #XPJK0* <i>x</i>
Then activate the program with	GO #XPJK23

*Narrative**Console message*

5	To abandon the current report, input:	GO #XPJK27
6	At the end of the run the program will output one of the following:	
	(a) If no delete message was input in an error free run:	HALTED:—HH
	(b) If a delete message was correctly input and the run was error free:	DELETED:— <i>user message</i>
	(c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:	DISPLAY:—DE HALTED:—HH
	(d) If any error occurred during the run whether or not a delete message was input: (see section <i>Line printer messages</i> , page 43)	HALTED:—ZZ
7	For a further run return to operating instruction 2	

Examples of running under GEORGE 3 or GEORGE 4

- 1 A report is required on the file FILEM with generation number 7 on cartridge 720610. Also for cartridge 072016, a table of all the files, with a list of file areas allocated to each file, is required.


```

LOAD :LIB.PROGRAM XPJK
CREATE !
ASSIGN *LP0,!
LISTFILE !,*LP
ERASE !
OL *CR0
(((
ENTER 1
XPJK1(FILEM,7,720610)
XPJK2(72016,FA)
****
))))

```
- 2 Maps are required for cartridge 020010 and 172032, both without a list of file areas allocated to each file in the file tables. Also a report is required on file SUBROUTINES with generation number 4095 on cartridge 000014.


```

LOAD :LIB.PROGRAM XPJK
CREATE !
ASSIGN *LP0,!
LISTFILE !,*LP
ERASE !
ENTER 2,PARAM(XPJK2(20010,M)(172032,M),XPJK1(SUBROUTINES,4095,14),****)

```

Note: When using entry point 2, the maximum length of each parameter is 76 characters.

Exception conditions

The following is a list of exception conditions reported by #XPJK during execution and loading. Prior to entry into #XPJK it must have been loaded correctly and this is signalled by HALTED:—LD. When a loading exception condition occurs before HALTED:—LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

<i>Messages</i>	<i>Reason</i>	<i>Action</i>
1 HALTED:—CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJK or (b) Abandon the run

#XPJK

	<i>Messages</i>	<i>Reason</i>	<i>Action</i>
2	DISPLAY:—DE	The end parameter message did not start with the characters LO or FI	The run continues
3	HALTED:—LP— NEEDS MIN <i>n</i> PP	The program requires a line printer with at least <i>n</i> print positions	Either (a) Make such a line printer available and input: GO #XPJK or (b) Abandon the run
4	DISPLAY: MISUSE OF ENTRY 7 — RUN ABORTED HALTED:—ZZ	The program cannot recover from the incorrect use of entry point 7	
5	HALTED:—MX	A chain in the system control area has reached a length of 7000 cells	If a loop is suspected then abandon the report by inputting: GO #XPJK27 Otherwise continue by inputting: GO #XPJK The program will halt MX again after a further 7000 cells
6	DISPLAY:—PE	Transfer failure on the line printer	No action. The run continues
7	DISPLAY:—PL	Paper low condition has been detected	Load more paper and re-engage the printer
8	HALTED:—RF	A card or tape block has been misread	Either (a) Reread the parameter line by reloading the card/block and inputting: GO #XPJK or (b) Abandon the run
9	HALTED:—ST	There is insufficient store available for a map	Either (a) When more store is available, input: GO #XPJK or (b) Abandon the report by inputting: GO #XPJK27 or (c) Abandon the run

<i>Messages</i>	<i>Reason</i>	<i>Action</i>
10 HALTED:--TR	There is no tape reader available	Either (a) Make a tape reader available and input: GO #XPJK or (b) Abandon the run
11 HALTED:--xx	An error has been detected when loading #XPJK by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when executive attempts to open the storage unit it will output a storage unit not on-line message, for example :

#XPJK; LDC*nnnnnn (the actual format of the message varies according to which executive is being used)

The user then has two options to take:

- 1 Put the requested storage unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJK 27 to abandon the current report

Line printer messages

If any of the messages given below are output on the line printer, the message HALTED:--ZZ will be output on the console at the end of the run.

Unless otherwise stated, the program abandons the current report and proceeds with the next one.

<i>Message</i>	<i>Action</i>
1 <i>parameter line</i> ↑↑	The line has a parameter error. One or more upward arrows indicate the offending character(s). Any correctly specified reports up to the error are done before proceeding to the next parameter line
2 **** <i>fn fgn</i> NOT IN SYSTEM **** <i>fn fgn</i> INTEGRITY CODE FAIL **** <i>fn fgn</i> INSUFFICIENT EXEC STORE }	Executive replies to an attempt to open a file
3 **** <i>fn fgn</i> ABANDON ENTRY POINT USED	Entry point 7 has been used to abandon the current file report
4 **** ABANDON ENTRY POINT USED	Entry point 7 has been used to abandon the current cartridge report
5 **** PARITY ERROR: BLOCK <i>n</i>	A parity error has occurred on reading block <i>n</i>
6 **** UNACCEPTABLE CARTRIDGE SIZE	The cartridge has an unacceptable number of blocks per cylinder or an unacceptable number of cylinders
7 -- REPRESENTS UNREFERENCED AREAS	The cartridge has areas that are not referenced by the system control area. The report is not abandoned

<i>Message</i>	<i>Action</i>
8 ?? REPRESENTS AREAS WITH MULTIPLE REFERENCES	The cartridge has areas that are referenced more than once by the system control area. The report is not abandoned
9 **** MAP IS UNRELIABLE – ERRORS FOUND IN SCA	Errors have been found in the chain of file index cells and/or the chains of file area cells. The report is not abandoned
10 **** ERROR <i>list of error codes</i> (as defined below)	Errors have been found in the system control area. The report is not abandoned
Possible error codes:	
01 Invalid file name	
02 No integrity code	
03 Date written is invalid or today's date is wrong	
04 Invalid number of blocks per bucket. The program assumes eight blocks per bucket in order to carry out further checks	
05 The maximum number of words per record is either less than two or greater than the bucket size minus the bucket header size. The program assumes a value of <i>bucket size – bucket header size</i> in order to carry out further checks	
06 The key length is greater than 64 characters. The program assumes a value of zero in order to carry out further checks	
07 The key does not fit into the maximum record size	
08 The number of file areas allocated to the file does not agree with the value in the file index cell	
09 The bucket number of the start of second level overflow is greater than the last bucket number in the file	
10 There is an invalid field in a file area cell belonging to the chain of file areas allocated to a file on this cartridge. The whole file area chain is considered corrupt	
11 A file area cell in the chain of file areas allocated to a file on this cartridge starts in the red tape of a block or runs off the end of a block. The whole file area chain is considered corrupt	
12 A pointer to the next file area cell in the chain of file areas allocated to a file on this cartridge is out of range. The whole file area chain is considered corrupt	
13 There are more than 512 file area cells in the chain of file areas allocated to a file on this cartridge. The whole chain is considered corrupt	
19 In a minimum report, one or more chains of file areas allocated to files on this cartridge have errors 11, 12 or 13	
21 The next file index cell starts in the red tape of a block or runs off the end of a block. The program acts as if the end of the file index chain has been reached	
22 The pointer to the next file index cell is out of range. The program acts as if the end of the file index chain has been reached	
24 The file index chain is in a loop. The program acts as if the end of the chain has been reached	
25 The used file index cells are not close packed. The program acts as if the end of the file index chain has been reached	
26 There are no files (except the system file) on the cartridge. A minimum report is given instead of F, FA, M or MA type reports	

- 27 There is no ICLSCAFILE on the cartridge
- 28 Files with the same file name and generation number have been found on the cartridge during a cartridge report. The file table output immediately preceding this message should be scanned for all such occurrences
- 30 There is an invalid field in the file area cell belonging to the chain of file areas allocated to files on other cartridges. The program acts as if the end of the chain has been reached
- 31 The file area cell in the chain of file areas allocated to files on other cartridges starts in the red tape of a block or runs off the end of a block. The program acts as if the end of the chain has been reached
- 32 The pointer to the file area cell in the chain of file areas allocated to files on other cartridges is out of range. The program acts as if the end of the chain has been reached
- 40 There is an invalid field in the file area cell belonging to the chain of unallocated areas. The program acts as if the end of the chain has been reached
- 41 The file area cell in the chain of unallocated areas starts in the red tape of a block or runs off the end of a block. The program acts as if the end of the chain has been reached
- 42 The pointer to the file area cell in the chain of unallocated areas is out of range. The program acts as if the end of the chain has been reached
- 51 A cell in the chain of unused file area cells starts in the red tape of a block or runs off the end of a block
- 52 A pointer to the next cell in the chain of unused file area cells is out of range
- 60 A flaw index cell is invalid
- 61 A flaw index cell starts in the red tape of a block or runs off the end of a block
- 62 A pointer to a flaw index cell is out of range
- 72 The start of the overlay area is out of range
- 80 The end of the used overlay area comes before the start of the overlay area
- 82 The end of the used overlay area is out of range
- 90 The end of the reserved overlay area comes before the end of the used overlay area
- 92 The end of the reserved overlay area is out of range

NAME

#XPJL

VERSION

Mark 1D

TITLE

Housekeeping format direct access file print

HARDWARE REQUIREMENT

5376 words of store

1 paper tape reader or 1 card reader

1 line printer

1 or more direct access storage units

DESCRIPTION**General**

The program prints out data from all or part of a file, which is formatted to ICL direct access housekeeping formats, either bucket by bucket (physically) or record by record (logically), starting at a specified bucket number or record key. Any number of the functions given below can be performed on one file during a run. Three options are available for the format of the printout. Each word of data can be printed as eight octal characters, as four alphanumeric characters and a signed decimal integer, or in all three forms together.

For a file not formatted to ICL direct access housekeeping formats, the program #XPJQ should be used. (See #XPJQ, page 61.)

The following functions are available:

Record print (logical)

- 1 Print records for *x* home buckets starting from a specified bucket number
- 2 Print *x* records starting from a specified record key
- 3 Print records starting from a specified bucket number until the record containing a specified key is encountered
- 4 Print records starting from a specified record key until the record containing another specified key is encountered

Printing will cease if an end of file bucket, that is a bucket with bit 13 set in the first word of the bucket header, is encountered before the end condition is met. Words 2 to 17 of this bucket will be printed out.

Bucket print (physical).

- 1 Print *x* buckets starting from a specified bucket number
- 2 Print *x* buckets starting from the first bucket of a specified file area

Note: The data output for a bucket is from the words defined as used by the bucket header, plus auxiliary header data where applicable.

Printing will stop if the last bucket allocated to a file is encountered before the count of buckets has expired.

The format of the print layout for each function is given in the section *Output*, page 48.

#XPJL

Input

Input to the program consists of the direct access data files to be printed out; parameters to the program may be input from cards or paper tape. The data files must conform to the 1900 Series Unified Direct Access Housekeeping formats. Parameters are described in the section *Control*, page 49.

Output

The data file is printed out on a line printer using print positions 1 to 96, 1 to 120 or 1 to 160 (see the section *Control*, page 49). The format of the print layout is as follows:

#XPJL MK *n* *dd/mm/yy hh/mm/ss* HOUSEKEEPING FILE PRINT PAGE *yyy*

FILE *xxxxxxxxxxxx* GEN NO. *xxxx* STORAGE UNIT *xxxxxx*

type BUCKET NO. *xxxxxxx*

RECORD KEY *key* (*present only if records are being printed out*)

WORD

xxx *x..x* *x..x* *x..x* *x..x* *x..x* *x..x* *x....* *....*

xxx *x..x* *x..x* *x..x* *x..x* *x..x* *x..x* *x....* *....*

xxx *x..x* *x..x* *x..x* *x..x* *x..x* *x..x* *x....* *....*

AUXILIARY HEADER (*present only if buckets are being printed out*)

WORD

x *x..x* *x..x* *x..x* *x..x* *x..x* *x..x* *x...* *....*

x *x..x* *x..x* *x..x* *x..x* *x..x* *x..x* *x...* *....*

x *x..x* *x..x* *x..x* *x..x* *x..x* *x..x* *x...* *....*

where

dd/mm/yy is the date on which the data file is printed out.

hh/mm/ss is the time at which the data file is printed out.

xx... represents information relevant to the file. The number of characters per word printed will vary with the type of print format specified.

key is the record key and may be up to 64 characters in length.

type is one of the following:

END OF FILE

INDEX

HOME

HOME (EXT. CHAIN)

1ST LEVEL O'FLOW

2ND LEVEL O'FLOW

The word TAG will be printed against the record key when a tag has been processed to find a record. In a random file this record may be encountered other than by processing its tag. When this occurs the line TAGGED RECORD will precede the record key line.

The number of words printed per line varies according to the print format and the number of print positions used, as follows:

<i>Print format</i>	<i>Number of words per line</i>		
	<i>96 print positions</i>	<i>120 print positions</i>	<i>160 print positions</i>
Octal	8	10	13
Character/decimal	5	6	9
Composite	3	4	5

The two examples given in Figure 9 below show the layout of a composite record print and an octal bucket print respectively, using a 96 print position printer.

[illegible]

Figure 9 Example of a composite record print and an octal bucket print

CONTROL

This section describes the sets of parameters associated with the six functions described in the section *General*, page . Further details of parameter formats are given in the section *Parameter formats*, page .

For all functions: parameter type 1

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJL) and parameter type	XPJL1	5 (fixed)
2	Print width	0, 1 or 2 where 0 = 96 print positions 1 = 120 print positions 2 = 160 print positions	1 (fixed)

#XPJL

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
3	Print format	O, S or C where O = octal S = character/decimal C = composite	1 (fixed)
4	Name of file	Up to 12 alphanumeric space (V) or minus (–) characters, with the first character alphabetic	12
5	Generation number of file	–1, 0 to 4095 Note: If this is not specified, it is taken as 0	4
6	Serial number of unit on which the file is held	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)

For record print: parameter type 2

There are four functions for a record print all having type number 2.

FUNCTION 1: PRINT HOME BUCKETS STARTING FROM A SPECIFIED BUCKET NUMBER

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJL) and parameter type	XPJL2	5 (fixed)
2	Function code	1	1 (fixed)
*3	Start bucket number	1 to 8388607	7
*4	Number of home buckets to be printed	1 to 8388607 Note: If this is omitted, printing will continue until an end of file bucket is encountered	7

*The group of parameter fields 3 and 4 must be enclosed in parentheses and may be repeated. If these parameter fields extend over more than one card or line, parameter field 1 (program name and parameter type) must be repeated, but parameter field 2 (function code) may be omitted.

FUNCTION 2: PRINT RECORDS STARTING FROM A SPECIFIED KEY

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJL) and parameter type	XPJL2	5 (fixed)
2	Function code	2	1 (fixed)

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*3 Record key at which printing is to start	Up to 64 alphanumeric characters	64
*4 Bucket number containing start record key	1 to 8388607 Note: This is optional and need not be specified	7
*5 Number of records to be printed	1 to 8388607 Note: If this is omitted, printing will continue until an end of file bucket is encountered	7

*The group of parameter fields 3 to 5 must be enclosed in parentheses and may be repeated. If these parameter fields extend over more than one card or line, parameter field 1 (program name and parameter type) must be repeated, but parameter field 2 (function code) may be omitted. Details of the parameter format for record keys are given at the end of this section.

FUNCTION 3: PRINT RECORDS STARTING FROM A SPECIFIED BUCKET TO A SPECIFIED KEY

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJL) and parameter type	XPJL2	5 (fixed)
2 Function code	3	1 (fixed)
*3 Bucket number at which printing is to start	1 to 8388607	7
*4 Record key at which printing is to stop	Up to 64 alphanumeric characters	64

*The group of parameter field 3 and 4 must be enclosed by parentheses and may be repeated. If these parameter fields extend over more than one card or line, parameter field 1 (program name and parameter type) must be repeated, but parameter field 2 (function code) may be omitted. Details of the parameter format for record keys are given at the end of this section.

FUNCTION 4: PRINT RECORDS FROM A SPECIFIED KEY TO A SPECIFIED KEY

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJL) and parameter type	XPJL2	5 (fixed)
2 Function code	4	1 (fixed)
*3 Record key at which printing is to start	Up to 64 alphanumeric characters	64

#XPJL

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*4	Bucket number containing start record key	1 to 8388607 Note: This is optional and need not be specified	7
*5	Record key at which printing is to stop	Up to 64 alphanumeric characters	64

*The group of parameter fields 3 to 5 must be enclosed in parentheses and may be repeated. If these parameter fields extend over more than one card or line, parameter field 1 (program name and parameter type) must be repeated, but parameter field 2 (function code) may be omitted. Details of the parameter format for record keys are given at the end of this section.

For bucket print: parameter type 3

There are two functions for a bucket print each having the type number 3.

FUNCTION 1: PRINT BUCKETS FROM A SPECIFIED BUCKET NUMBER

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJL) and parameter type	XPJL3	5 (fixed)
2	Function code	1	1 (fixed)
*3	Start bucket number	1 to 8388607	7
*4	Number of buckets to be printed	1 to 8388607 Note: If this is omitted, printing will continue until the last allocated bucket of a file	7

*The group of parameter fields 3 and 4 must be enclosed in parentheses and may be repeated. If these parameter fields extend over more than one card or line, parameter field 1 (program name and parameter type) must be repeated, but parameter field 2 (function code) may be omitted.

FUNCTION 2: PRINT BUCKETS STARTING FROM THE FIRST BUCKET OF A SPECIFIED FILE AREA

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJL) and parameter type	XPJL3	5 (fixed)
2	Function code	2	1 (fixed)
*3	Start file area	1 to 32767	5
*4	Number of buckets to be printed	1 to 8388607 Note: If this is omitted, printing will continue to the last allocated bucket of a file	7

*The group of parameter fields 3 and 4 must be enclosed in parentheses and may be repeated. If these parameter fields extend over more than one card or line, parameter field 1 (program name and parameter type) must be repeated, but parameter field 2 (function code) may be omitted.

RECORD KEYS PARAMETER FORMAT

Each parameter field must be terminated by a comma except when followed by a parenthesis. The fields for one parameter type must be contained on one card or line except when a record key is used (functions 2, 3 and 4) and the physical length of the key is such that it cannot be contained on one card or line. In this case, a comma must be the last character on the line and a comma must immediately follow parameter field 1 (XPJL2) on the continuation card or line.

END OF PARAMETERS

A card or line with four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a L_Oad or F_Ind message, Further details are given in the section *Parameter formats*, page 2.

Examples

1 RECORD PRINT: FUNCTION 1

It is required to print 100 home buckets, starting at bucket 500, from the file DAFILE1 in octal format on a 120 print position printer. The generation number of the file is 100, and the file index and parameters entry is held on unit 000632.

The parameters to perform this function would take the following form (if input on cards):

card 1 XPJL1,1,O,DAFILE1,100,000632,

card 2 XPJL2,1(500,100)

2 RECORD PRINT: FUNCTION 2

It is required to print two lots of 50 records from the file DAFILE2 (highest generation number on-line) in composite format on a 160 print position printer. The first lot is to be printed from the record with key number 1072483248ADLJCDT785025947183TUWA1283384. The second lot is to be printed from the record with key number 1138547021BCWRTCX834015875932YWGH8521976 which is held in bucket 186.

The parameters required to perform this function would take the following form (if input on paper tape):

XPJL1,2,C,DAFILE2,-1,↓

XPJL2,2(1072483248ADLJCDT785025947183TUWA1283384,,50)↓

XPJL2(1138547021BCWRTCX834015875932YWGH8521976,↓

XPJL2,186,50)↓

where ↓ indicates a newline character

3 RECORD PRINT: FUNCTION 3

It is required to print out the following records in character/decimal format from the file DAFILE3:

(a) From bucket 230 to the record with the key number 8237AZ2B3HW

(b) From bucket 780 to the record with the key number 8239CT8L4YD

The generation number of DAFILE3 is 5 and it is held on unit 003272. The line printer to be used has 96 print positions.

The parameters to perform this function would take the following form (if input on cards):

card 1 XPJL1,0,S,DAFILE3,5,003272,

card 2 XPJL2,3(230,8237AZ2B3HW)

card 3 XPJL2,3(780,8239CT8L4YD)

#XPJL

4 RECORD PRINT: FUNCTION 4

It is required to print out the following records in composite format from the file DAFILE4:

- (a) From the record with key number 56D7485HWIO48973AD3C2 which is contained in bucket number 234 to the record with key number 56D7486ABN8765MJ4A6
- (b) From the record with key number 56D7486CPEF72345QR4G2 to the record with key number 56D7487LUVA82537LP5B7

The generation number of the file DAFILE4 is 15 and it is held on unit 001372. The line printer to be used has 160 print positions.

The parameters to perform this function would take the following form (if input on paper tape):

XPJL1,2,C,DAFILE4,15,001372,↓

XPJL2,4(56D7485HWIO48973AD3C2,234,↓

XPJL2,56D7486ABN8765MJ4A6)↓

XPJL2(56D7486CPEF72345QR4G2,,56D7487LUVA82537LP5B7)↓

where ↓ indicates a newline character

5 BUCKET PRINT: FUNCTION 1

It is required to print in octal format the 50 buckets starting at bucket 150 and the 10 buckets starting at bucket 360 of the file DAFILE5. The generation number of DAFILE5 is 20 and it is held on unit 330671. The line printer to be used has 120 print positions.

The parameters to perform this function would take the following form (if input on cards):

card 1 XPJL1,1,O,DAFILE5,20,330671

card 2 XPJL3,1(150,50)(360,10)

6 BUCKET PRINT: FUNCTION 2

It is required to print out the following in composite format from the file DAFILE6:

- (a) The first 120 buckets of file area 2
- (b) The buckets from the beginning of file area 5 to the end of the file

The generation number of DAFILE6 is 0 and it is held on unit 062173. The line printer to be used has 96 print positions.

The parameters required to perform these functions would take the following form (if input on cards):

card 1 XPJL1,0,C,DAFILE6,,062173,

card 2 XPJL3,2(2,120)(5)

7 FINDING NEXT PROGRAM AND BATCHING FUNCTIONS

If the parameters given in examples 1, 3 and 6 are to be input as a batch and it is then required to find #XPJC to delete the file areas specified in example 6, the parameters would take the following form (if input on paper tape):

XPJL1,1,0,DAFILE1,100,000632,↓

XPJL2,1,(500,100)↓

XPJL1,0,S,DAFILE3,5,003272,↓

XPJL2,3(230,8237AZ2B3HW)↓

XPJL2,3(780,8239CT8L4YD)↓

XPJL1,0,C,DAFILE6,,062173,↓

XPJL3,2(2,120)(5)↓

****FI#XPJC#DISC↓

OPERATING**Executive priority**

The program as supplied has a priority of 50.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0	Parameters	The reader is assigned at the beginning of the run and released when the batch terminator has been read
LP0	Output and error messages	The printer is used throughout the run
DA1	Input	The unit is used throughout the run

Running under Executive alone**OPERATING INSTRUCTIONS**

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJL (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:— LD will only be output if the loading operation was successful	HALTED:— LD
2 Load the required storage unit(s)	
3 Load the parameters in the appropriate reader	
4 To activate the program:	
(a) For parameters on paper tape, input:	GO #XPJL 20
(b) For parameters on cards, input:	GO #XPJL 21
5 When the run is complete the program will output one of the following:	
(a) If a delete message was correctly input and a successful run has occurred:	DELETED:- <i>user message</i>
(b) If no delete message was input in an error free run:	HALTED:- HH
(c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:	DISPLAY:- DE HALTED:- HH

#XPJL

Narrative

Console message

- (d) If any error has occurred during the run (except the error described in (c) above), whether or not a delete message was input:

HALTED:- ZZ

(Note: see the section
Line printer messages.)

- 6 For a further run return to operating instruction 2

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJL during execution and loading. **Prior to entry into #XPJL it must have been loaded correctly and this is signalled by HALTED:- LD.** When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions in FIND A BINARY PROGRAM* in *Compiling Systems*.

<i>Message</i>	<i>Reason</i>	<i>Action</i>
1 HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJL or (b) Abandon the run
2 DISPLAY:- DE	The end parameter delete message did not start with the characters LO or FI	The run continues
3 HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJL or (b) Abandon the run
4 HALTED:- NEEDS LP OF MINIMUM SIZE xxx	A line printer with xxx print positions is required	Either (a) Make the appropriate line printer available and input: GO #XPJL or (b) Abandon the run
5 DISPLAY:- PE	Transfer failure on the line printer	No action. The run continues
6 DISPLAY:- PL	A paper low condition has been detected.	Load more paper and re-allocate the printer

<i>Message</i>	<i>Reason</i>	<i>Action</i>
7 HALTED:- RF	A card or paper tape block has been misread	To reread the parameter line reload the card/block and input: GO #XPJL
8 HALTED:- TR	There is no tape reader available.	Either (a) Make a tape reader available and input: GO #XPJL or (b) Abandon the run
9 HALTED:- xx	An error has been detected when loading #XPJL by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in <i>FIND A BINARY PROGRAM</i> in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when executive attempts to open the storage unit it will output a storage unit not on-line message, for example #XPJL; LDC * *nnnnnn* (the actual format of the message varies according to which executive is being used). The user then has two options to take:

- 1 Put the requested storage unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJL 27 to abandon the present set of parameters and go to the next

LINE PRINTER MESSAGES

If any of the messages given below (except message 19) are output on the line printer, the message HALTED:- ZZ will be output on the console at the end of the run.

<i>Message</i>	<i>Reason and/or action</i>
1 ALLOCATED PRINTER = xxx P.P	The number of print positions required in the second or subsequent parameter type 1 (XPJL1) card or line is greater than the number which was allocated for the first parameter type 1 card or line. The offending parameter type 1 card or line will be printed out preceding this message and flagged as a PARAMETER ERROR (see below). The program will then skip to the next parameter type 1 card or line and the run will continue
2 **** PARAMETER ERROR xxxxxxxxxxxxxxxxxxxxxxxxxxxxx ↑	The upward pointing arrow indicates the offending parameter field. This can occur if the length of a specified record key is not equal to the key length recorded in the file index cell for the file to be printed. The program skips to the next parameter type 1 or the batch terminator card or line
3 PRINT WIDTH NOT GIVEN	If a parameter error occurs before a correct print width indicator is found on the first parameter card or line, normal parameter reporting cannot be followed. The program will store the erring card or line and search the following card or lines for a parameter type 1 having a correct print width indicator. If one is found, the printer will be allocated and all the previous type 1 parameters and the first card or line will be reported

#XPJL

<i>Message</i>	<i>Reason and/or action</i>
	as parameter errors. The correct type 1 parameter will then be processed normally as the first card or line.
	If either the batch terminator (****) is encountered, or more than 20 incorrect type 1 parameters occur, the message PRINT WIDTH NOT GIVEN will be output on any line printer and the program will terminate the run with the message HALTED:- ZZ.
4 ** SEQUENCE ERROR PARAMETER XPJLx SEQ xxxx	The card is not in sequence (see the section <i>Parameter formats</i> , page 2). It will, however, be processed normally.
Messages 5 to 8 to begin with the name and generation number of the file to which the rest of the message refers.	
5 xxxxxxxxxxxx xxx ABANDON ENTRY POINT USED	GO #XPJL 27 has been input on the console
6 xxxxxxxxxxxx xxx INTEGRITY CODE FAILURE	The file specified in the message is ignored. The run continues normally
7 xxxxxxxxxxxx xxx INSUFFICIENT EXECUTIVE CORE	The file specified in the message is ignored. The run continues normally
8 xxxxxxxxxxxx xxx NOT IN SYSTEM	The file specified in the message is ignored. The run continues normally
Messages 9 to 13 may be output during a bucket print.	
9 BUCKET NO. OUT OF RANGE	The run continues with the next group of parameter fields
10 FILE AREA NUMBER xxxxx NOT IN FILE	The file does not contain the number of file areas specified as the start file area in the parameters. The run continues with the next group of parameter fields
11 INVALID BUCKET HEADER	The bucket is printed and the run continues with the next bucket
12 PARITY ERROR	The run continues with the next bucket
13 UNRECOGNISABLE REPLY	This message is output after an attempt has been made to read a bucket. The run continues with the next bucket
Messages 14 to 28 may be output during a record print.	
14 BUCKET NO. OUT OF RANGE	The run continues with the next group of parameter fields
15 END OF FILE BUCKET	The 16 words of end of file information are printed out and then the run continues with the next group of parameter fields
16 INDEX ENTRY NOT IN FILE	This occurs when a specified record key is greater than the highest record key recorded in the indexes. The run continues with the next group of parameter fields
17 INVALID BUCKET HEADER	The whole of the bucket is printed and the run continues with the next group of parameter fields
18 INVALID RECORD LENGTH	The whole of the bucket is printed and the run continues with the next group of parameter fields

<i>Message</i>	<i>Reason and/or action</i>
19 MAX. REC. SIZE = <i>xxxx</i>	This is printed after each record which is greater than the maximum file record size. It is only a warning message and the run continues normally with the next record. The message HALTED:- ZZ will not be output at the end of the run
20 NO 2ND LEVEL O/F FOR EXT. CHAIN	The bucket with the extension chain pointer is printed out. The run continues with the next group of parameter fields
21 PARITY ERROR	The run continues with the next group of parameter fields
22 RECORD NOT FOUND	The run continues with the next group of parameter fields
23 SPECIFIED BUCKET IN 2ND LEVEL O/F	The run continues with the next group of parameter fields
24 TAG IN SERIAL FILE	The bucket containing the tag is printed out. The run continues with the next group of parameter fields
25 TAG RECORD KEY <i>nnn</i> POINTS TO A TAG BUCKET NO. <i>xxxxxxx</i>	The bucket number refers to the bucket pointed to by the initial tag. The run continues with the next group of parameter fields
26 TAG RECORD KEY <i>nnnnnnnn</i> RECORD MISSING BUCKET NO. <i>xxxxxxx</i>	The bucket number refers to the bucket which should have contained the record. The run continues with the next group of parameter fields
27 TOO MANY LEVELS OF INDEX	The run continues with the next group of parameter fields
28 UNRECOGNISABLE REPLY	This message is output after an attempt has been made to read a bucket. The run continues with the next group of parameter fields

NAME

#XPJQ

VERSION

Mark 1C

TITLE

File bucket or hardware block print

HARDWARE REQUIREMENT

3136 words of store

1 paper tape reader or card reader

1 line printer

1 or more direct access storage units

DESCRIPTION**General**

The program will print specified buckets or part buckets (for example, bucket headers) of a named file organized to any data format, or selected blocks of a storage unit specified by hardware addresses. Four options are available for the format of the printout. Each word of data can be printed as eight octal characters, as four alphanumeric characters, as a signed decimal integer, or in all three forms together.

Input

Input to the program consists of direct access storage units organized to ICL 1900 Series unified direct access standards from which selected buckets or blocks are to be printed; parameters to the program can be input on cards or paper tape. Files organized to any data formats may be processed by #XPJQ. The parameters are described in the section *Control*, page 62).

Output

The selected buckets or blocks are printed out on the line printer using print positions 1 to 96, 1 to 120 or 1 to 160 (see the section *Control*, page 62).

For both options the first line of the printout has the following format:

#XPJQ MK*n* *dd/mm/yy* *hh/mm/ss* FILE OR HARDWARE BLOCK PRINT PAGE *yyy*

where

dd/mm/yy is the date on which the blocks or buckets are printed out.

hh/mm/ss is the time on which the blocks or buckets are printed out.

yyy is the page number.

BUCKET PRINT FORMAT

The format for subsequent lines in a bucket print is as follows:

#XPJQ

FILE xxxxxxxxxxxx GEN NO. xxxx STORAGE UNIT xxxxxx

BUCKET xxxxxxxx

WORD

xxx x..x x..x x..x x..x x..x x..x x... x...

xxx x..x x..x x..x x..x x..x x..x x... x...

xxx x..x x..x x..x x..x x..x x..x x... x...

where

xx... is information relevant to the file. The number of characters per word printed will vary with the print option used.

HARDWARE BLOCK PRINT FORMAT

The format for subsequent lines in a hardware block print is as follows:

STORAGE UNIT xxxxxx

CYLINDER xxxx

BLOCK xxxxxxxx

WORD

xxx x..x x..x x..x x..x x..x x..x x... x....

xxx x..x x..x x..x x..x x..x x..x x... x....

xxx x..x x..x x..x x..x x..x x..x x... x....

where

xx... is information relevant to the storage unit. The number of characters per word printed will vary with the print option used.

The number of words printed per line varies according to the print format and the number of print positions used, as follows:

Print format	Number of words per line		
	96 print positions	120 print positions	160 print positions
octal	8	10	13
character	14	18	24
decimal	8	11	15
composite	3	4	5

CONTROL

This section describes the two parameter types that may be input to #XPJQ. Further details of parameter formats are given in the section *Parameter formats*, page 2.

PARAMETER TYPE 1: BUCKET PRINT

	Parameter field	Range of values	Maximum number of characters
1	Program name (XPJQ) and parameter type	XPJQ1	5 (fixed)

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
2 Print width	0, 1 or 2 where 0 = 96 print positions 1 = 120 print positions 2 = 160 print positions	1 (fixed)
3 Print format	O, A, D or C where O = octal A = character D = decimal C = composite	1 (fixed)
4 Name of file from which buckets have to be printed	Up to 12 alphanumeric space (V) or minus (–) characters, with the first character alphabet	12
5 Generation number of file	–1, 0 to 4095 Note: If this is omitted it will be taken as zero	4
6 Serial number of the unit from which buckets are to be printed	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)
*7 Number of the bucket from which printing is to start	1 to 8388607 Note: If this is not specified, printing will start from the beginning of the file	7
*8 Number of buckets to be printed	1 to 8388607 Note: If this is not specified, printing will continue to the end of the file	7
*9 Number of words, starting at the first word, to be printed from each bucket	1 to 1024 Note: If this is omitted, whole buckets will be printed	4

*The group of parameter fields 7 to 9 must be enclosed in parentheses. These parameter fields may be repeated on the first card or line if more than one area is to be printed. A second card or line may be given if these parameter fields extend over more than one card or line. The second card or line may omit parameter fields 2 to 6 if the print width and format are to remain the same. If a print out of the whole file is required, parameter fields 7 to 9 may be omitted, including the parentheses; in this case, the last parameter field on the card or line is terminated by a space.

The print width (parameter field 2) and print format (parameter field 3) must be specified on the first card or line received by the program. These fields are optional in all the following type 1 or type 2 parameters. If they are not specified, the previously defined values will be taken.

Each parameter field must be terminated by a comma except when followed by a parenthesis.

#XPJQ

PARAMETER TYPE 2: BLOCK PRINT

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJQ) and parameter type	XPJQ2	5 (fixed)
2	Print width	0, 1 or 2 where 0 = 96 print positions 1 = 120 print positions 2 = 160 print positions	1 (fixed)
3	Print format	O, A, D or C where O = octal A = character D = decimal C = composite	1 (fixed)
4	Serial number of the unit from which blocks are to be printed	000001 to 777777	6 (fixed)
*5	Number of cylinder from which printing is to start	0 to 9999	4
*6	Number of block within cylinder from which printing is to start	0 to 8388607	7
*7	Number of blocks to be printed	1 to 8388607	7

*The group of parameter fields 5 to 7 must be enclosed in parentheses. These parameter fields may be repeated on the first card or line if more than one area is to be printed out. A second card or line may be given if these parameter fields extend over more than one card or line. The second card or line may omit parameter fields 2 to 4 if the print width and format are to remain the same.

The print width (parameter field 2) and print format (parameter field 3) must be specified on the first card or line received by the program. These fields are optional in all the following type 1 or type 2 parameters. If they are not specified, the previously defined values will be taken.

Each parameter field must be terminated by a comma except when followed by a parenthesis.

END OF PARAMETERS

A card or line containing four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a L_Oad or F_Ind message. Further details are given in the section *Parameter formats*, page 2.

Examples

A number of buckets are to be printed in character format from file FILEA (generation number 0) which is held on the unit with serial number 100777. The line printer to be used has 120 print positions. The areas to be printed are as follows: the first 30 words from each of the first 20 buckets of the file, 10 buckets starting at bucket number 80, the first 20 words from each of 5 buckets starting at bucket number 161, the first 100 words from each of 10 buckets starting at bucket number 180 and 40 buckets starting at bucket number 250.

One area is to be printed out in character format from FILEB, with an assumed generation number 0. This area is 20 buckets in length starting at bucket number 4.

The whole of FILEC, with an assumed generation number 0, is to be printed out in composite format.

In addition a number of blocks are to be printed out from the unit with serial number 100607, also on a line printer using 120 print positions. Three areas are to be printed in octal format. These areas are as follows: 160

blocks starting at block number 0 of cylinder 10, 200 blocks starting at block number 40 of cylinder 32 and 80 blocks starting at block number 0 of cylinder 150.

The batch of parameters to perform these functions would take the following form (if input on paper tape):

```
XPJQ1,1,A,FILEA,0,100777(1,20,30)(80,10)↓
XPJQ1(161,5,20)(180,10,100)(250,40)↓
XPJQ1,,,FILEB(4,20)↓
XPJQ1,,C,FILEC↓
XPJQ2,,O,100607(10,0,160)(32,40,200)(105,0,80)↓
****↓
```

where ↓ indicates a newline character.

OPERATING

Executive priority

The program as supplied has a priority of 80.

Trusted status

The program has Q-trusted status.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0	Parameters	The reader is assigned at the beginning of the run and released when the batch terminator has been read
LP0	Output and error reports	The line printer is used throughout the run
DA0	Input	The unit is used throughout the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJQ (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required storage unit(s)	
3 Load the parameters in the appropriate reader	
4 To activate the program:	
(a) If parameters are on paper tape, input:	GO #XPJQ 20
(b) If parameters are on cards, input:	GO #XPJQ 21
5 When the run is complete the program will output one of the following:	
(a) If a delete message was correctly input and a successful run has occurred:	DELETED:- user message

#XPJQ

<i>Narrative</i>	<i>Console message</i>
(b) If no delete message was input in an error free run:	HALTED:- HH
(c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:	DISPLAY:- DE HALTED:- HH
(d) If any error has occurred during the run (except the error described in (c) above), whether or not a delete message was input:	HALTED:- ZZ
(Note: see the section <i>Line printer messages</i> .)	
6 For a further run, return to operating instruction 2	

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJQ during execution and loading. Prior to entry into #XPJQ it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

<i>Message</i>	<i>Reason</i>	<i>Action</i>
1 HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJQ or (b) Abandon the run
2 DISPLAY:- DE	The end parameter delete message did not start with the characters LO or FI	The run continues
3 HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJQ or (b) Abandon the run
4 HALTED:- NEEDS LP OF MINIMUM SIZE xxx	A line printer with xxx print positions is required	Either (a) Make the specified size line printer available and input: GO #XPJQ or (b) Abandon the run
5 DISPLAY:- PE	Transfer failure on the line printer	No action. The run continues

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
6	DISPLAY:- PL	Paper low condition has been detected	Load more paper and re-engage the printer
7	HALTED:- RF	A card or paper tape block has been misread	To reread the parameter line, reload the card/block and input: GO #XPJQ
8	HALTED:- TR	There is no tape reader available	Either (a) Make a tape reader available and input: GO #XPJQ or (b) Abandon the run
9	HALTED:- xx	An error has been detected when loading #XPJQ by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in <i>FIND A BINARY PROGRAM</i> in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when executive attempts to open the storage unit it will output a storage unit not on-line message, for example #XPJQ; LDC * nnnnnn (the actual format of the message varies according to which executive is being used). The user then has two options to take:

- 1 Put the requested storage unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJK 27 to abandon the function and continue as normal

LINE PRINTER MESSAGES

If any of the messages given below are output on the line printer, the message HALTED:- ZZ is output on the console at the end of the run.

With messages 1 to 4, each message begins with the name and generation number of the file to which the rest of the message refers.

	<i>Message</i>	<i>Reason and/or action</i>
1	xxxxxxxxxxxx xxxx ABANDON ENTRY POINT USED	The console message GO #XPJQ 27 has been input
2	xxxxxxxxxxxx xxxx INSUFFICIENT EXECUTIVE CORE	Only the parameters for the file specified are ignored. The run continues normally
3	xxxxxxxxxxxx xxxx INTEGRITY CODE FAILURE	Only the parameters for the file specified are ignored. The run continues normally
4	xxxxxxxxxxxx xxxx NOT IN SYSTEM	Only the parameters for the file specified are ignored. The run continues normally
5	ALLOCATED PRINTER = xxx P.P.	This message is output when the printer size specified in the second or subsequent parameter card or line is larger than the printer already allocated to the program, as determined by Property Codes. The rejected parameter card or line is printed out preceding this message and flagged as a PARAMETER ERROR.

#XPJQ

<i>Message</i>			<i>Reason and/or action</i>
			The program continues with the next parameter card or line
6	BUCKET BLOCK xxxxxxxx	OUT OF RANGE	The program continues with the next group of parameters
7	BUCKET BLOCK xxxxxxxx	PARITY ERROR	The program continues to read the next bucket or block
8	BUCKET BLOCK xxxxxxxx	UNRECOGNISABLE REPLY	This message is given after an attempt has been made to read a bucket or block. The program continues to read the next block
9	**** PARAMETER ERROR xxxxxxxxxxxxxxxxxxxxx ↑		The arrow indicates the offending parameter. The program continues with the next group or the next set of parameters as appropriate
10	PRINT WIDTH NOT GIVEN		If a parameter error occurs before a correct print width indicator is found on the first parameter card or line normal parameter error reporting cannot be followed. The program will store the erring card or line and search the following card or lines until a correct print width indicator is found. If one is found, the printer will be allocated and all the previous cards or lines will be reported as parameter errors, and the accepted card or line will be processed normally as the first card or line. If more than 20 erring cards or lines or the batch terminator (****) is encountered, the program will output the message PRINT WIDTH NOT GIVEN on any line printer and terminate the run with the console message HALTED:- ZZ
11	** SEQUENCE ERROR, PARAMETER XPJQx SEQ xxxx		The card is out of sequence (see Chapter 1) but will be processed normally

NAME

#XPJR

VERSION

Mark 1A and later versions

TITLE

Analysis of serial, sequential and random direct access data files.

HARDWARE REQUIREMENT

6400 words of store

1 card reader or paper tape reader

1 line printer

1 or more direct access storage units

DESCRIPTION

General

This program analyses a complete file or part of a file (that is one or more complete file areas) conforming to ICL 1900 Series Direct Access Housekeeping Standards and outputs to the line printer a report on the contents of buckets in the specified area. The file may be serial, sequential or random.

The program analyses only those buckets containing information that belongs to the data file. Buckets beyond the end of the data file are counted to give a total number of buckets not belonging to the data file. The definition of the end of the data file varies according to the type of file:

- 1 The end of data in a serial file is indicated by the end of file bucket, which is counted as part of the file
- 2 The end of data in a sequential file is taken as the end of the seek area containing the end of file bucket. All buckets up to and including the end of file bucket plus all the first level overflow buckets in the seek area containing the end of file bucket are counted as part of the file
- 3 The end of data in a random file is indicated by the start of second level overflow or, if no second level overflow is present, the end of the area allocated to the file

The analysis proceeds one file area at a time, at one of the three levels of reporting that are available. These are full, short and summaries only:

- 1 **FULL REPORT** This report prints:
 - (a) The details of every bucket belonging to the data file
 - (b) A summary of every file area analysed
 - (c) A final summary of all the file areas analysed

See Figure 10, page 71, for example
- 2 **SHORT REPORT** This report prints:
 - (a) The details of home buckets, belonging to the data file, that have tags, overflow records, extension chains or contain less than a specified number of words available
 - (b) The end of file bucket and first level overflow buckets belonging to the data file

#XPJR

- (c) A summary of every file area analysed
- (d) A final summary of all the file areas analysed

See Figure 11, page 72, for example

3 SUMMARIES ONLY REPORT This report prints:

- (a) A summary of every file area analysed
- (b) A final summary of all the file areas analysed

See Figure 12, page 75, for example

Error messages are invariant across the levels of reporting.

A report may be used as a means of determining whether a file needs to be reorganized. The File Organization Software described in Chapter 2 of this manual should be used for reorganization.

The size of the program's buffer area is initially 1024 words. This may be increased by specifying the amount of core store that the program may extend into, thereby improving the efficiency of the program.

Input

Input to the program consists of:

- 1 Direct access data files conforming to ICL Direct Access Housekeeping Format, held on storage units conforming to ICL Unified Direct Access Standards
- 2 Parameters input from cards or paper tape (see *Control*, page 73)

Output

The analyses of files are output on the line printer using print positions 1 to 96. The possible forms that the report can take are:

- 1 Full report (see Figure 10, page 71, for example)
- 2 Short report (see Figure 11, page 72, for example)
- 3 Summaries only report which consists of both a *summary report* and a *final summary* (see Figures 12 and 13, pages 75 and 76, for examples)

If the file area being analysed is beyond the file area containing the end of file bucket then no detailed analysis is given and the summaries only report is output.

Error messages are also output to the line printer (see section *Line printer messages*, page 78).

SUMMARY REPORT

In the summary report various different sets of totals are printed these being dependent on certain qualifications that are determined during run time. Each set of totals printed is a combination of the following descriptive total lines:

<i>Total type</i>	<i>Descriptive total printed</i>
1	NUMBER OF RECORDS = n
2	NUMBER OF BUCKETS UP TO END OF FILE = n For a serial file this is the count of all buckets up to and including the end of file bucket. For a sequential file this is the count of all buckets up to and including the end of file bucket plus any first level overflow buckets in the seek area containing the end of file bucket. For a random file this is the count of all buckets up to the start of second level overflow area
3	NUMBER OF SPARE BUCKETS = n For a sequential file any spare home buckets in the seek area containing the end of file bucket are included in this count
4	NUMBER OF BUCKETS IN 2ND LEV O/F AREA = n
5	TOTAL NUMBER OF BUCKETS = n

#XPJR MK 1A 11/08/78 11/53/11 DATA FILE ANALYSIS PAGE 1					
FILE PROGRAM OVPM GEN NO. 0 STORAGE UNIT 721101 ALLOCATED AS SEQUENTIAL FULL REPORT					
BUCKET	TYPE	HOMERECORDS	O'FLOW RECORDS	TAGS	WORDS AVAIL.
1	INDEX				124
2	INDEX				86
3	INDEX				126
4	HOMER	3	0	0	66
5	HOMER	3	0	0	66
6	HOMER	3	0	0	66
7	HOMER	3	0	0	66
8	HOMER	3	0	0	66
9	HOMER	3	0	0	66
10	HOMER	3	0	0	66
11	HOMER	3	0	0	66
12	HOMER	3	0	0	66
13	**KEY SEQUENCE ERROR:KEY 0026				
13	**KEY SEQUENCE ERROR:KEY 0027				
13	HOMER	1	0	0	66
14	HOMER	3	0	0	66
15	HOMER	3	0	0	66
16	HOMER	3	0	0	66
17	HOMER	3	0	0	66
18	HOMER	3	0	0	66
19	HOMER	3	0	0	66
20	HOMER	3	0	0	66
21	HOMER	3	0	0	66
22	HOMER	3	0	0	66
23	HOMER	0	0	62	2
241	EXTENSION	6	0	0	6
242	EXTENSION	6	0	0	6
243	EXTENSION	6	0	0	6
244	EXTENSION	6	0	0	6
245	EXTENSION	6	0	0	6
246	EXTENSION	6	0	1	44
24	END OF FILE				126
65	1ST LEV O/F	0	0	0	124
66	1ST LEV O/F	0	0	0	124
67	1ST LEV O/F	0	0	0	126
68	1ST LEV O/F	0	0	0	126
69	1ST LEV O/F	0	0	0	126
70	1ST LEV O/F	0	3	0	66
71	1ST LEV O/F	0	6	0	6
72	1ST LEV O/F	0	6	0	6
73	1ST LEV O/F	0	6	0	6
74	1ST LEV O/F	0	6	0	6
75	1ST LEV O/F	0	6	0	6
76	1ST LEV O/F	0	6	0	6
77	1ST LEV O/F	0	6	0	6
78	1ST LEV O/F	0	6	0	6
79	1ST LEV O/F	0	6	0	6
80	1ST LEV O/F	0	6	0	6

#XPJR MK 1A 11/08/78 11/53/11 DATA FILE ANALYSIS PAGE 2					
FILE AREA 1 SUMMARY AND FINAL SUMMARY					
BUCKET TYPE	NUMBER UNUSED UP TO EOF	NUMBER USED UP TO EOF	NUMBER WITH OVERFLOW		NUMBER WITH < 0 WORDS SPARE
			1ST LEVEL	2ND LEVEL	
INDEX	1	2			
HOMER	0	20	2	1	0
1ST LEV O/F	5	11			
EXTENSION		6			
NUMBER OF RECORDS = 152					
NUMBER OF BUCKETS UP TO END OF FILE = 40					
NUMBER OF SPARE BUCKETS = 200					
NUMBER OF BUCKETS IN 2ND LEV O/F AREA = 80					
TOTAL NUMBER OF BUCKETS = 320					

Figure 10 An example of a full report

#XPJR

#XPJR MK 1A 11/08/78		12/26/81	DATA FILE ANALYSIS		PAGE 1
FILE	PROGRAM	OVPM	GEN NO.	0	STORAGE UNIT 721101 ALLOCATED AS SEQUENTIAL
SHORT REPORT					
BUCKET	TYPE	HOME RECORDS	O'FLOW RECORDS	TAGS	WORDS AVAIL.
13	**KEY SEQUENCE ERROR KEY 0026				
13	**KEY SEQUENCE ERROR KEY 0027				
23	HOME	0	0	62	2
241	EXTENSION	6	0	0	6
242	EXTENSION	6	0	0	6
243	EXTENSION	6	0	0	6
244	EXTENSION	6	0	0	6
245	EXTENSION	6	0	0	6
246	EXTENSION	6	0	1	44
24	END OF FILE	0	0	0	126
65	1ST LEV O/F	0	0	0	126
66	1ST LEV O/F	0	0	0	126
67	1ST LEV O/F	0	0	0	126
68	1ST LEV O/F	0	0	0	126
69	1ST LEV O/F	0	0	0	126
70	1ST LEV O/F	0	3	0	66
71	1ST LEV O/F	0	6	0	6
72	1ST LEV O/F	0	6	0	6
73	1ST LEV O/F	0	6	0	6
74	1ST LEV O/F	0	6	0	6
75	1ST LEV O/F	0	6	0	6
76	1ST LEV O/F	0	6	0	6
77	1ST LEV O/F	0	6	0	6
78	1ST LEV O/F	0	6	0	6
79	1ST LEV O/F	0	6	0	6
80	1ST LEV O/F	0	6	0	6
FILE AREA 1 SUMMARY AND FINAL SUMMARY					
BUCKET TYPE	NUMBER UNUSED UP TO EOF	NUMBER USED UP TO EOF	NUMBER WITH OVERFLOW 1ST LEVEL 2ND LEVEL		NUMBER WITH < 0 WORDS SPARE
INDEX	1	2	2	1	0
HOME	0	20			
1ST LEV O/F	5	11			
EXTENSION		6			
NUMBER OF RECORDS = 152					
NUMBER OF BUCKETS UP TO END OF FILE = 40					
NUMBER OF SPARE BUCKETS = 200					
NUMBER OF BUCKETS IN 2ND LEV O/F AREA = 80					
TOTAL NUMBER OF BUCKETS = 320					

Figure 11 An example of a short report

Serial files

For serial files there are three possible sets of totals:

- 1 File area being analysed has not reached the end of file bucket.
Totals printed from the list above are total types 1 and 5
- 2 File area being analysed includes the end of file bucket but has some spare buckets between the end of file bucket and end of file area.
Totals printed from the list above are total types 1, 2, 3 and 5
- 3 File area being analysed is beyond the file area containing the end of file bucket.
Totals printed from the list above are total types 3 and 5

Sequential files

For sequential files there are seven possible sets of totals. In addition to these, there is first of all a breakdown analysis showing the number of buckets used and unused within index/home/first level overflow buckets, the number of home buckets using first level overflow/second level overflow, and the number of buckets used in second overflow area. This breakdown analysis is only printed for the first four types of sequential file summary and not the last three.

- 1 File area being analysed does not contain second level overflow area and has not reached end of file bucket.
Totals printed from the list above are total types 1 and 5

- 2 File area being analysed does not contain second level overflow area but includes the end of file bucket and also has some spare buckets between end of file bucket and end of file area.
Totals printed from the list above are total types 1, 2, 3 and 5
- 3 File area being analysed has the end of file bucket as the last home bucket of a seek area which is immediately followed by second level overflow area.
Totals printed from the list above are total types 1, 2, 4 and 5
- 4 File area being analysed contains the end of file bucket followed by some spare buckets followed by second level overflow area.
Totals printed from the list above are total types 1, 2, 3, 4 and 5
- 5 File area being analysed does not contain second level overflow area and is beyond the file area containing the end of file bucket.
Totals printed from the list above are total types 3 and 5
- 6 File area being analysed contains second level overflow buckets only.
Totals printed from the list above are total types 4 and 5
- 7 File area being analysed is beyond the file area containing the end of file bucket and has some spare buckets prior to the start of second level overflow area.
Total printed from the list above are total types 3, 4 and 5

Random files

For random files there are three possible sets of totals. In addition to these, there is first of all a breakdown analysis showing the number of buckets used and unused within home buckets and the number of buckets used in second level overflow area. This breakdown analysis is only printed for the first two types of random file summary and not the last one.

- 1 File area being analysed does not contain second level overflow area.
Totals printed from the list above are total types 1 and 5
- 2 File area being analysed contains both home and second level overflow areas.
Totals printed from the list above are total types 1, 2, 4 and 5
- 3 File area being analysed contains second level overflow area only.
Totals printed from the list above are total types 4 and 5

The breakdown analysis when applicable to sequential/random file summaries may also print the total number of home buckets with less than *nnnn* words spare if requested. The value *nnnn* is supplied on the parameter card and is described in section *Control*, below.

Final summary

When more than one file area is being analysed for a file the final summary is a total of all the previous summary totals for the file and is printed as a separate summary. Otherwise, when only one file area is being analysed, the final summary is not shown separately and the summary print heading is changed to:

FILE AREA *xxxx* SUMMARY AND FINAL SUMMARY

CONTROL

This section describes the one parameter type that may be input to #XPJR. For further details of parameter formats see section *Parameter formats*, page 2..

#XPJR

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJR) and parameter	XPJR1	5 (fixed)
2	Form of analysis required	F, A, or S F = full A = short S = summaries	1 (fixed)
3	Summary information required if any home buckets have less than the number of words specified in this parameter spare	1 to 1022 Note: If this is not specified, information regarding spare words in home buckets will not be printed in any summaries	4
4	Program store size	7 to 32 (in multiples of 1024 words) Note: This is optional and may be omitted	2
*5	Name of the file to be analysed	Up to 12 alphanumeric, space (V) or minus (—) characters, with the first character alphabetic	12
*6	Generation number of the file	—1, 0 to 4095 Note: If this is not specified, a generation number of zero is assumed	4
*7	Serial number of the unit holding the file	000001 to 777777 (in octal characters) Note: This field is optional and need not be specified	6 (fixed)
*8	Number of the first file area to be analysed	1 to 511 Note: If this is not specified, the value 1 is assumed	3
*9	Number of file areas to be analysed	1 to 511 Note: If this is omitted, all file areas following the start file area will be analysed	3

*The group of parameter fields 5 to 9 must be enclosed in parentheses. Each parameter field must be terminated by a comma except when followed by a parenthesis.

End of parameters

A card or line containing four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a L_Oad or F_Ind message. Further details are given in the section *Parameter formats*, page 2.

Example

A short analysis, including information on home buckets that have less than 35 words spare is to be given of file areas 2, 3 and 4 of file INVOICEDTLS, generation number 4, held on storage unit 720127. Then a full analysis of the file PAYROLLDATAV, generation number 6. No extra store is required.

The parameters to perform this function would take the following form (if input on cards):

card 1 XPJR1,A,35(INVOICEDETLS,4,720127,2,3)

card 2 XPJR1,F,,(PAYROLldata,6,,)

card 3 ****

```

#XPJR MK 1A 11/08/78      12/20/01      DATA FILE ANALYSIS      PAGE 2

FILE PROGRAM OVPM  GEN NO.  0  STORAGE UNIT 721101  ALLOCATED AS SEQUENTIAL
SUMMARIES ONLY REPORT

13 **KEY SEQUENCE ERROR:KEY 0026
13 **KEY SEQUENCE ERROR:KEY 0027

FILE AREA 1 SUMMARY AND FINAL SUMMARY
BUCKET TYPE      NUMBER UNUSED      NUMBER USED      NUMBER WITH OVERFLOW      NUMBER WITH < 0
                  UP TO EOF          UP TO EOF          1ST LEVFL 2ND LEVEL      WORDS SPARE

INDEX            1                    2
HOME             0                    20           2           1           0
1ST LEV O/F      5                    11
EXTENSION        6                    6

NUMBER OF RECORDS =      152
NUMBER OF BUCKETS UP TO END OF FILE =      40
NUMBER OF SPARE BUCKETS =      200
NUMBER OF BUCKETS IN 2ND LEV O/F AREA =      80
TOTAL NUMBER OF BUCKETS =      120

```

Figure 12 An example of a summary report for one file area

OPERATING

Executive priority

The program as supplied has a priority of 60.

Use of peripherals

Device	Use	Assignment
TR0 or CR0	Parameters	The reader is assigned at the beginning of the run and released when the terminator parameter has been read
LP0	Report and error messages	The printer is used throughout the run if a report is being output; otherwise assigned when an error occurs
DA0	Input medium	Used throughout the run

#XPJR

#XPJR MK 1A 11/08/78 12/20/01	DATA FILE ANALYSIS	PAGE 3
FILE WUCH=SERIAL GPN NO. 0 STORAGE UNIT 721101 ALLOCATED AS SERIAL SUMMARIES ONLY REPORT		
FILE AREA 1 SUMMARY NUMBER OF RECORDS = 32 TOTAL NUMBER OF BUCKETS = 8		
#XPJR MK 1A 11/08/78 12/20/01	DATA FILE ANALYSIS	PAGE 5
BUCKET		
FILE AREA 2 SUMMARY NUMBER OF RECORDS = 54 NUMBER OF BUCKETS UP TO END OF FILE = 15 NUMBER OF SPARE BUCKETS = 0 TOTAL NUMBER OF BUCKETS = 80		
#XPJR MK 1A 11/08/78 12/20/01	DATA FILE ANALYSIS	PAGE 7
FILE AREA 3 SUMMARY NUMBER OF RECORDS = 0 NUMBER OF BUCKETS UP TO END OF FILE = 0 NUMBER OF SPARE BUCKETS = 3 TOTAL NUMBER OF BUCKETS = 80		
#XPJR MK 1A 11/08/78 12/20/01	DATA FILE ANALYSIS	PAGE 9
BI		
FILE AREA 4 SUMMARY NUMBER OF RECORDS = 0 NUMBER OF BUCKETS UP TO END OF FILE = 0 NUMBER OF SPARE BUCKETS = 3 TOTAL NUMBER OF BUCKETS = 83		
FINAL SUMMARY NUMBER OF RECORDS = 86 NUMBER OF BUCKETS UP TO END OF FILE = 22 NUMBER OF SPARE BUCKETS = 3 TOTAL NUMBER OF BUCKETS = 105		

Figure 13 An example of a summary report for four file areas

Running under Executive alone

OPERATING INSTRUCTIONS

Narrative	Console message
1 Load the program #XPJR (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>)	HALTED:— LD
The messages HALTED:— LD will only be output if the loading operation was successful	
2 Load the required cartridge(s)	
3 Load the parameters in the appropriate reader	
4 To activate the program, either:	
(a) If parameters are on paper tape, input:	GO #XPJR 20
or	
(b) If parameters are on cards, input:	GO #XPJR 21
5 When the run is complete, the program will output one of the following messages:	
(a) If a delete message was correctly input, and a successful run has occurred:	DELETED:— user message
(b) If the run was error free and no delete message was input:	HALTED:— HH

*Narrative**Console message*

- (c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:

DISPLAY:- DE

HALTED:- HH

- (d) If any error had occurred during the run (except the one described in (c) above), whether or not a delete message was input:

HALTED:- ZZ

Note: If this message is output at the end of the run, one or more error messages will have been output on the line printer (see *Line printer messages*, page 78)

- 6 For a further run, return to Operating Instruction 2

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJR during execution and loading. Prior to entry into #XPJR it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJR or (b) Abandon the run
2	DISPLAY:- DE	The end parameter delete message did not start with the character LO or FI	The run continues
3	HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJR or (b) Abandon the run
4	DISPLAY:- PE	Transfer failure on the line printer	No action. The run continues
5	DISPLAY:- PL	Paper low condition detected on the line printer	Load more paper and re-engage the printer
6	HALTED:- RF	A read failure has occurred when reading parameters	Reset the last parameter read and input: GO #XPJR
7	HALTED:- TR	There is no tape reader available	Either

#XPJR

<i>Message</i>	<i>Reason</i>	<i>Action</i>
		(a) Make a tape reader available and input: GO #XPJR
		or
		(b) Abandon the run
8 HALTED:- xx	An error has been detected when loading #XPJR by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when executive attempts to open the storage unit it will output a storage unit not on-line message, for example

#XPJR; LDC * nnnnnn (the actual format of the message varies according to which executive is being used)

The user then has two options to take:

- 1 Put the requested storage unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJR 27 to abandon the current analysis

LINE PRINTER MESSAGES

If any of the error messages given below are output on the line printer, the message HALTED:- ZZ will be output on the console at the end of the run.

The formats of these messages fall into four categories.

Format 1

****fn fgn message*

where

fn is the name of the file.

fgn is the file generation number.

message is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
1 FILE AREA <i>nnnn</i> NOT IN FILE	This message will also occur if the file being analysed is zero length. The value of <i>nnnn</i> is supplied during run time and is the number of the file area in question. The run will continue with the next analysis
2 FILE NOT IN SYSTEM	The run will continue with the next analysis
3 INSUFFICIENT EXECUTIVE CORE	The run will continue with the next analysis
4 INTEGRITY CODE FAILURE	The run will continue with the next analysis
5 PARAMETER ERROR xxxxxxxxxxxxxxxx ↑	Where the arrow indicates the offending parameter. The run will continue with the next analysis
6 ABANDON ENTRY POINT USED	This message is output if entry point 7 is used during the running of the program

Format 2*lbn message* :KEY *key*

where

lbn is the logical bucket number of the current bucket.*key* is the key of the record in error.*message* is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
7 **TAG IN WRONG TYPE OF BUCKET	The run continues by processing the next record
8 **OVERFLOW RECORD IN WRONG TYPE OF BUCKET	The run continues by processing the next record
9 **CORRUPT RECORD OR TAG	The key is not printed when the word count of the record/tag is invalid. The run continues by processing the next bucket
10 **KEY SEQUENCE ERROR	The run continues by processing the next record

Format 3*lbn message* :BUCKET *bkt* :KEY *key*

where

lbn is the logical bucket number of the current bucket*bkt* is the logical bucket number of the bucket pointed to.*key* is the key of the record in error*message* is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
11 **TAG POINTS TO A TAG	The run continues by processing the next record
12 **TAG POINTS TO A HOME RECORD	The run continues by processing the next record
13 **TAG NOT FOUND	The run continues by processing the next record
14 **OVERFLOW RECORD POINTS TO OVERFLOW RECORD	The run continues by processing the next record
15 **OVERFLOW RECORD POINTS TO HOME RECORD	The run continues by processing the next record
16 **OVERFLOW RECORD NOT FOUND	The run continues by processing the next record
17 **ERROR <i>x</i> FOLLOWING UP TAG	When reading the bucket, pointed to by the tag or overflow record, if an invalid bucket read occurs then <i>x</i> is set with one of the following values
18 **ERROR <i>x</i> FOLLOWING UP OVERFLOW RECORD	
	<i>x</i> = 1 Parity error
	<i>x</i> = 2 Unrecognisable reply from executive
	<i>x</i> = 3 Bucket number out of range or zero
	<i>x</i> = 4 Bucket read has an invalid bucket header
	<i>x</i> = 5 Bucket read is of wrong type viz tag points to an Index bucket
	The run continues with the next record

#XPJR

Format 4

lbn message

where

lbn is the logical bucket number of the current bucket.

message is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
19 **PARITY ERROR	The run continues by processing the next bucket
20 **UNRECOGNISABLE REPLY	The run continues by processing the next bucket
21 **INVALID BUCKET HEADER	The run continues by processing the next bucket
22 **BUCKET NUMBER OUT OF RANGE	If the bucket number is not zero then the analysis is abandoned and the run continues with the next analysis otherwise the run continues by processing the next bucket
23 **INVALID BUCKET TYPE: xxxx	The bucket type is not allowed in this context, for example, in a sequential file an Index bucket is found after Home buckets in the same seek area. The value of xxxx can be :INDEX :HOME :1ST LEV O/F :2ND LEV O/F depending on the type of bucket read

NAME

#XPJW

VERSION

Mark 2A

TITLE

Direct access file copy, dump and restore

HARDWARE REQUIREMENT

6336 words of store (minimum)

1 card reader or 1 paper tape reader

1 line printer

1 or more magnetic tape decks (optional)

1 or more direct access storage units

DESCRIPTION**General**

The program can be used to copy all or part of one direct access file to another, or to create security dumps of a number of direct access files or part files, or to restore files or part files from #XPJW created security dumps. Files organized to any data formats may be manipulated by #XPJW which processes at allocated bucket level. Note that #XPJW is a data security program, therefore files of zero length will be neither copied nor dumped.

The copy facility can be used to copy all the allocated or just specified buckets from one direct access file to another; the bucket size and bucket header length of both files must be identical as the output data will be in the same form as the input data. Creation of the output file may be requested, but it should be noted that for sequential and random files #XPJW will only attempt creation of file areas to the same co-ordinates as the input file (see further under *The creation and extension of direct access files*, page 83).

The dump facility enables security dumps to be made of all the allocated or just specified buckets of a number of specified direct access files, or all the allocated buckets for all the files described in the system control area of a specified unit. Dumps can be made to a direct access file or a named magnetic tape; successive dumps may be made to a single file. If the dump is to magnetic tape, multi-reel dumps may be produced and scratch tapes will be used for the second and subsequent reels. Creation and/or extension of direct access dump files may be requested. The format of the dump is unusable except as input to #XPJW.

The restore facility enables information for all or specified files held in a particular #XPJW created dump to be restored. Creation of the file to which the restore is to be made may be requested, but it should be noted that for sequential and random files #XPJW will only attempt creation of file areas to the same co-ordinates as the original dumped file (see further under *The creation and extension of direct access files*, page 83).

The eight functions undertaken by the program are:

FUNCTION 1: COPY FROM ONE DIRECT ACCESS FILE TO ANOTHER

Copy all the allocated buckets or just specified buckets of a direct access file to all or part of another. A pre-allocated output file may be extended or the output file may be created. The output file may be renamed. Care should be taken that a pre-allocated output file has the same bucket size, bucket header length (also where applicable key length and position, levels of index and start bucket number of second level overflow) as the input file. A value may be given for the end of file bucket number (EOFBN) recorded in the file index cell of the output file. If this option is not used and the first bucket to be read from the input file has the same number as the

#XPJW

first bucket to be written to on the output file, then if the part of the file to be copied contains the end of file bucket, the EOFBN recorded in the file index cell of the output file will be set to that of the input file, otherwise it will remain unchanged.

FUNCTION 2: DUMP SPECIFIED FILES

Dump all the allocated buckets of specified files and/or specified buckets of files to a direct access file or named magnetic tape. A pre-allocated direct access dump file may be extended or the dump file may be created. The dump file may be renamed. All dumped files will have information from their file index cells and file area cells in the system control area recorded for use when restoring.

FUNCTION 3: DUMP ALL THE FILES ON A STORAGE UNIT

Dump all the allocated buckets for all the files described in the system control area of a specified storage unit to a direct access file or named magnetic tape. Files to be excluded from the dump may be specified. A pre-allocated direct access dump file may be extended or the dump file may be created. The dump file may be renamed. File areas on the specified storage unit which are extensions of files originating on other storage units will not be dumped. However, files that extend on to other storage units from the specified storage unit will be dumped in their entirety. All dumped files will have information from their file index cells and file area cells in the system control area recorded for use when restoring.

FUNCTION 4: DUMP SPECIFIED FILES FOLLOWING PREVIOUS #XPJW DUMPS

Dump all the allocated buckets of specified files and/or specified buckets of files to follow previous #XPJW dumps to the specified direct access or magnetic tape dump file. The direct access dump file may be extended. All dumped files will have information from their file index cells and file area cells in the system control area recorded for use when restoring.

FUNCTION 5: DUMP ALL THE FILES ON A STORAGE UNIT FOLLOWING PREVIOUS #XPJW DUMPS

Dump all the allocated buckets for all the files described in the system control area of a specified storage unit to follow previous #XPJW dumps to the specified direct access or magnetic tape dump file. Files to be excluded from the dump may be specified. The direct access dump file may be extended. File areas on the specified storage unit which are extensions of files originating on other storage units will not be dumped. However, files that extend on to other storage units from the specified storage unit will be dumped in their entirety. All dumped files will have information from their file index cells and file area cells in the system control area recorded for use when restoring.

FUNCTION 6: RESTORE SPECIFIED FILES

Restore specified direct access files and/or part files from a specified dump on a specified #XPJW created dump file. Files and file areas to which dumped files are to be restored may be created. Care should be taken that a pre-allocated file to be restored to has the same bucket size, bucket header length (also where applicable, key length and position, levels of index and start bucket number of second level overflow) as the dumped file. No files may be renamed. A dumped file may be restored to a storage unit different from the unit from which the file was dumped. The end of file bucket number (EOFBN) recorded in the file index cell of a restored file will be set to that of the dumped file provided that the part of the file to be restored contains the end of file bucket. Otherwise the EOFBN of the restored file will be unchanged.

FUNCTION 7: RESTORE ALL FILES FROM A DUMP

Restore all direct access files from a specified dump on a specified #XPJW created dump file. Files to be excluded from the restore may be specified. Files and file areas to which dumped files are to be restored may be created. Care should be taken that a pre-allocated file to be restored to has the same bucket size, bucket header length (also where applicable key length and position, levels of index and start bucket number of second level overflow) as the dumped file. No files may be renamed. Dumped files may be restored to a storage unit different from the unit from which the files were dumped. The end of file bucket number (EOFBN) recorded in the file index cell of a restored file will be set to that of the dumped file provided that the part of the file to be restored contains the end of file bucket. Otherwise the EOFBN of the restored file will be unchanged.

FUNCTION 8: RESTORE SPECIFIED FILES TO ONE OR MORE DIFFERENT FILES

Restore specified direct access files and/or part files from a specified dump on a specified #XPJW created dump file, to one or more different files which may be renamed. This function can also be used to restore information to different buckets of the file from which the dump was made. Files and file areas to which dumped files are to be restored may be created. Care should be taken that a pre-allocated file to be restored to has the same bucket size, bucket header length (also where applicable, key length and position, levels of index and start bucket number of second level overflow) as the dumped file. A value may be given for the end of file bucket number (EOFBN) recorded in the file index cell of the restored file. If this option is not used and the first bucket to be read from the dumped file has the same number as the first bucket to be written to on the restored file, then if the part of the file to be restored contains the end of file bucket, the EOFBN of the restored file will be set to that of the dumped file. Otherwise the EOFBN of the restored file will be unchanged.

Tape block count checking

During a restore from a magnetic tape dump file, the number of blocks read from the tape is checked against the block count stored at dump time in an end of subfile sentinel and, if applicable, an end of reel sentinel. This block count check can only be made if the complete dump of a file or part file is to be restored. However, for a partial restore, checks are made on the format of the dump tape, which will result in an error being flagged if control blocks or sentinels are not encountered as expected.

Note: Tape dump files output from versions of #XPJW prior to version 2A will not contain block counts in sentinels so no block count check can be made.

The creation and extension of direct access files

In the dump functions (2, 3, 4 and 5) the dump file will be extended automatically onto the storage unit holding the last file area in existence and, if there is no space on that unit, the extensions will be made onto a unit specified in the parameters. Optionally an extension size may be specified, if this is omitted the dump file will be extended for each file to be dumped. Where there are a large number of files or file areas to be dumped it is better to pre-allocate the complete dump file using #XPJC rather than allow excessive extension. Not only will progressively more time and store be used on reporting and processing such extensions but Executive's file handling limit may be reached.

All file extensions will be reported on the line printer (see *Output*, page 85).

For file creation or file area creation (all functions) a parameter type 7 (see *Control*, page 86) must be given. The absence of this parameter implies no creation of files or file areas. Copying or restoring data to a sequential or random file which is to be created by the program will only be undertaken if file areas to the same co-ordinates as the input or dumped file can be created. However a serial file will be created and extended, if space allows by the necessary number of blocks, should the correct areas not be available, see further below.

Copying or restoring data to a pre-allocated sequential or random file when additional file areas must be created by the program will not be undertaken if the last bucket in existence on the file and the first bucket in the first file area to be created are not logically consecutive. However a serial file will be extended if space allows by the necessary number of blocks.

All file creations will be reported on the line printer (see *Output*, page 85).

The method of file creation for each function is as follows.

FUNCTIONS 1 AND 8

For function 8 as the details of the dumped file's file index cell and file area cells have been recorded this file can be regarded as if it were the input file for the copy function (function 1).

If a storage unit is not specified for the output file and the required file is not on-line, file creation will not be attempted. If a storage unit is specified and the required file is not present on that unit, #XPJW will first attempt to create a file of zero length on that unit using fields as follows:

<i>Field</i>	<i>Contents</i>
File name	As specified in the parameters

#XPJW

<i>Field</i>	<i>Contents</i>
Generation number	As specified in the parameters
Version number	Zero
Integrity code	As input file
Data retention period	As specified in parameters
Deallocation/file retention period	Zero
EOFBN	One (probably updated at end of run)

Other fields will be copied from the input file.

The program will then attempt to create a file area, on that unit, by extending the file into a given area using the same area co-ordinates as the corresponding file area of the input file. If this is not successful it will then attempt the same extension on the storage unit specified for extensions (if given). Assuming this to be successful further file areas will be created on the extension storage unit. Failure will lead to the creation of a random or sequential file being abandoned. However, for a serial file extension of the file by the necessary number of buckets will then be attempted on the unit holding the last file area in existence, or for a zero length file the unit holding the file index cell. If this is unsuccessful the same extension will be attempted on the storage unit specified for extensions. Assuming this to be successful further extensions will be restricted to that unit only.

FUNCTIONS 2, 3, 4 AND 5

For all creation and extension purposes the dump file will be regarded as a serial file even if pre-allocated otherwise.

If a storage unit is not specified for the dump file and the file is not on-line file creation will not be attempted and the function will be abandoned. If a storage unit is specified and the file is not present on that unit #XPJW (functions 2 and 3 only) will attempt to create a file of zero length on that unit using fields as follows: .

<i>Field</i>	<i>Contents</i>
File name	As specified in the parameters
Generation number	As specified in the parameters
Integrity code	3
Data retention period	As specified in the parameters
File type	Serial
Bucket size	8
EOFBN	1
Other fields	Zero

Assuming this to be successful extensions of the file by a number of buckets will then be attempted on that unit. If specified the extension size will be used otherwise an extension will take place for each file to be dumped. If unsuccessful the same extensions will be attempted on the storage unit specified for extensions (if given). Assuming this to be successful further extensions will be restricted to that unit only.

FUNCTIONS 6 AND 7

The following applies whether a file is being restored to the same storage unit as at dump time or to a specified unit. If the required file is not on-line #XPJW will first attempt to create a file of zero length on the relevant unit using the information stored at dump time. It will then attempt to create file areas by extending the file into a given area, using the co-ordinates stored at dump time for the file area. Failure will lead to the restoration of a random or sequential file being abandoned. However for a serial file extension of the file by the necessary number of buckets will then be attempted.

Input

Input to the program for functions 1 to 5 consists of direct access storage units organized to ICL direct access standards plus for functions 4 and 5 a direct access or magnetic tape dump file previously created by #XPJW.

Input to the program for functions 6 to 8 consists of the direct access or magnetic tape dump file previously created by #XPJW. Parameters to the program can be input from cards or paper tape.

The internal data formats of files to be copied or dumped may be to any standards.

Output

Output from the program consists of a direct access file (function 1) a direct access or magnetic tape dump file (functions 2 to 5) and restored direct access files (functions 6 to 8).

Magnetic tape dump files will be in 1900 composite file format and will have a maximum block size of 1024 words, or the size specified in the parameters (see *Control*, page 86). Each dump is held in a composite subfile, whose level 1 start of subfile sentinel has the subfile name *aaaaXPJW0000* (*aaaa* is the specified dump identifier) and which contains level 0 subfiles for each file or part file dumped with subfile names equal to the name of the dumped file. Files are dumped in the same order as that reported on the line printer list of dumped files.

When a direct access file is restored the version number is increased by one from its original value.

LINE PRINTER OUTPUT

Of the three facilities available (file copy, dump and restore) only the dump facility is reported on. A report of the dumped files is output on a line printer using print positions 1 to 96. The report has the following layout:

#XPJW MKn dd/mm/yy hh/mm/ss FILE COPY,DUMP AND RESTORE PAGE yyy

JOB a

DUMP — name SER.NO. xxxxxx FILE xxxxxxxxxxxx GEN.NO. xxxx RETN. xxxx

SER.NO. FILE GEN.NO. START BKT NO. OF BKTS

```

.      .      .      .      .
.      .      .      .      .
.      .      .      .      .

```

MAXIMUM BLOCK SIZE WORDS (*this is present for dumps on magnetic tape only*)

LAST BUCKET USED n (*this is present for dumps on direct access storage units only*)

where

dd/mm/yy is the date on which the listing is output.

hh/mm/ss is the time at which the listing is output.

name is the name of the dump.

xxxx is information relevant to the dump file.

. . . . is information relevant to each dumped file.

n is the number of last bucket used on the dump file.

Note: The retention period is omitted from listings of functions 4 and 5.

Additional line printer output may be output at the time of creating or extending a file on a direct access storage unit, and has the following format:

```

SER.NO.  FILE  GEN.NO.
CREATED  .      .      .

```


#XPJW

where

... is information relevant to the file that has been created.

	SER.NO.	FILE	GEN.NO.	START CYL	END CYL	START BLK	END BLK	BUCKETS
EXTENDED	.	.	.	.	.	.	.	.

where

... is information relevant to the file that has been extended.

CONTROL

This section describes the types of parameters associated with the eight functions described in the section *General*, page 81. Further details of the parameter formats are given in the section *Parameter formats*, page 2.

For every function the first parameter line must be of type 1 or 2.

Function 1: copy from one direct access file to another

A minimum of two parameter types must be input to the program. Type 1 defines the input file and type 4 defines the output file. Types 6 and 7 are optional parameter types. Type 6 may be input to rename the output file. Type 7 may be input to create the output file and file areas.

PARAMETER TYPE 1: DIRECT ACCESS INPUT FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW1	5 (fixed)
2 Function code	1	1 (fixed)
3 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
*4 Name of the file to be copied from	See Note 5 on page 106	12
*5 Generation number of the file to be copied from	–1, 0 to 4095 Note: If this is not specified it will be taken as 0	4
*6 Serial number of the storage unit holding the file index cell of the file to be copied from	000001 to 777777 (an octal number) Note: This is optional and need not be specified	6 (fixed)
*7 Number of buckets to be copied	0 to 8288607 Note: If this is specified as 0, all of the file from the bucket specified in parameter 8 to the last allocated bucket for the file will be copied	7
*8 Number of the first bucket to be copied	0 to 8388607 Note: If this is specified as 0, copying will start at the beginning of the file	7

*The group of parameter fields 4 to 8 must be enclosed in parentheses.

PARAMETER TYPE 4: DIRECT ACCESS OUTPUT FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW4	5 (fixed)
2 Serial number of unit to which extensions will be made	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)
*3 Name of the file to be copied to	See Note 5 on page 106	12
*4 Generation number of the file to be copied to	–1, 0 to 4095	4

#XPJW

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*5 Serial number of the storage unit holding the file index cell of the file	000001 to 777777 (an octal number) Note: If the file index of the file is to be created, this specifies the unit to hold the cell. Otherwise this need not be specified	6 (fixed)
*6 Retention period to be given to the file	0 to 4095 (in days)	4
*7 Number of the first bucket to be copied to	0 to 8388607 Note: A value of zero implies start at the beginning of the output file	7
*8 Value to be given to the EOFBN of the file	1 to 8388607 Note: This is optional and may be omitted	7

*The group of parameter fields 3 to 8 must be enclosed in parentheses.

PARAMETER TYPE 6 (OPTIONAL): RENAME OUTPUT FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW6	5 (fixed)
*2 New name to be given to the output file	See Note 5 on page	12
*3 New generation number to be given to the output file	0 to 4095	4

*The group of parameter fields 2 and 3 must be enclosed in parentheses.

PARAMETER TYPE 7 (OPTIONAL): CREATE THE OUTPUT FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW7	5 (fixed)
2 Create the output file and/or its file areas	C	1 (fixed)

Each parameter field must be terminated by a comma (or a space for parameter field 2 of type 7) except when followed by a parenthesis.

Function 2: dump specified files

A minimum of two parameter types must be input to the program. Type 1 (for direct access) or type 2 (for magnetic tape) defines the dump file and dump identifier and type 5 defines the files to be dumped. Types 6 and 7 are optional parameter types. Type 6 may be input to rename the dump file. Type 7 may be input to create and/or extend the dump file on a direct access storage unit.

PARAMETER TYPE 1: FOR DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW1	5 (fixed)
2 Function code	2	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
5 Serial number of unit to which extensions will be made	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)
*6 Name of the dump file	See Note 5 on page 106	12
*7 Generation number of the dump file	-1, 0 to 4095	4
*8 Serial number of the storage unit holding the file index cell of the dump file	000001 to 777777 (an octal number) Note: If the file index cell of the file is to be created, this specifies the unit to hold the cell. Otherwise, this need not be specified	6 (fixed)
*9 Retention period to be given to the file	0 to 4095 (in days)	4

*The group of parameter fields 6 to 9 must be enclosed in parentheses.

PARAMETER TYPE 2: FOR MAGNETIC TAPE DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW2	5 (fixed)
2 Function code	2	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2

#XPJW

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*5 Filename of the named tape to be used as the dump file (Scratch tapes will be used for any continuation reels)	See Note 5 on page 106	12
*6 Generation number of the named tape to be used for the dump file	0 to 8388607 Note: If this is not specified, it is taken as 0 (see Note 1 on page 105)	7
*7 Serial number of the named tape to be used for the dump file	1 to 37777777 (an octal number) Note: This is optional and need not be specified	8
*8 Retention period to be given to the dump file	0 to 4095 (in days) Note: If this is not specified the retention period will remain unchanged	4
*9 Maximum block size required for the dump file	1, 2, 3 or 4 (as a multiple of 1024) Note: If this is omitted, it is taken as 1	1

*The group of parameter fields 5 to 9 must be enclosed in parentheses.

PARAMETER TYPE 5: DIRECT ACCESS FILES TO BE DUMPED

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW5	5 (fixed)
*2 Name of the file to be dumped	See Note 5 on page 106	12
*3 Generation number of the file to be dumped	–1, 0 to 4095	4
*4 Serial number of the storage unit holding the file index cell of the file to be dumped	000001 to 777777 (an octal number) Note: This is optional and need not be specified	6 (fixed)
*5 Number of buckets to be dumped	0 to 8388607 Note: If this is not specified or specified as zero, all of the file from the bucket specified in parameter field 6 to the last allocated bucket for the file will be dumped	7
*6 Number of the first bucket to be dumped	0 to 8388607 Note: If this is omitted or specified as zero, dumping will begin from the first bucket of the file	7

*The group of parameter fields 2 to 6 must be enclosed in parentheses and may be repeated for each file to be dumped. If all the files to be dumped cannot be specified on the first card or line then more parameter cards or lines with the same type number (5) can be specified. A group of parameter fields must be wholly contained on one card or line.

PARAMETER TYPE 6 (OPTIONAL): RENAME DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW6	5 (fixed)
*2 New name to be given to the dump file	See Note 5 on page	12
*3 New generation number to be given to the dump file	Either 0 to 4095 for direct access dump files or 0 to 8388607 for magnetic tape dump files	4 7

*The group of parameter fields 2 and 3 must be enclosed in parentheses.

PARAMETER TYPE 7 (OPTIONAL): CREATE AND/OR EXTEND DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW7	5 (fixed)
2 Create the dump file	C Note: This is optional and may be omitted	1 (fixed)
3 Number of buckets by which the dump file is to be extended	1 to 8388607 Note: If this is omitted, where extension is necessary the program will extend the dump file for each file to be dumped. This may lead to Executive's file handling limit being exceeded	7

Each parameter field must be terminated by a comma except when followed by a parenthesis.

#XPJW

Function 3: dump all the files on a storage unit

At least one parameter type must be input to the program. Type 1 (for direct access) or type 2 (for magnetic tape) defines the dump file and dump identifier and the storage unit from which the dump is to be made. Types 5, 6 and 7 are optional parameter types. Type 5 may be input if files are to be excluded from the dump. Type 6 may be input to rename the dump file. Type 7 may be input to create and/or extend the dump file on a direct access storage unit.

PARAMETER TYPE 1: FOR DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW1	5 (fixed)
2 Function code	3	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
5 Serial number of unit to which extensions will be made	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)
*6 Name of the dump file	See Note 5 on page	12
*7 Generation number of the dump file	-1, 0 to 4095	4
*8 Serial number of the storage unit holding the file index cell of the dump file	000001 to 777777 (an octal number) Note: If the file index cell of the file is to be created, this specifies the unit to hold the cell. Otherwise, this need not be specified	6 (fixed)
*9 Retention period to be given to the dump file	0 to 4095 (in days)	4
10 Serial number of the storage unit whose files are to be dumped	000001 to 777777 (an octal number)	6 (fixed)

*The group of parameter fields 6 to 9 must be enclosed in parentheses.

PARAMETER TYPE 2: FOR MAGNETIC TAPE DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW2	5 (fixed)
2 Function code	3	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*5 Filename of the named tape to be used as the dump file. (Scratch tapes will be used for any continuation reels)	See Note 5 on page 106	12
*6 Generation number of the named tape to be used for the dump file	0 to 8388607 Note: If this is not specified it is taken as 0. (See Note 1 on page 105)	7
*7 Serial number of the named tape to be used for the dump file	1 to 37777777 (an octal number) Note: This is optional and need not be specified	8
*8 Retention period to be given to the dump file	0 to 4095 (in days) Note: If this is not specified the retention period will remain unchanged	4
*9 Maximum block size required for the dump file	1, 2, 3 or 4 (as a multiple of 1024) Note: If this is omitted, it is taken as 1	1
10 Serial number of the storage unit whose files are to be dumped	000001 to 777777 (an octal number)	6 (fixed)

*The group of parameter fields 5 to 9 must be enclosed in parentheses.

PARAMETER TYPE 5 (OPTIONAL): DIRECT ACCESS FILES TO BE EXCLUDED FROM THE DUMP

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW5	5 (fixed)
*2 Name of the file to be excluded from the dump	See Note 5 on page 106	12
*3 Generation number of the file to be excluded from the dump	0 to 4095	4

*The group of parameter fields 2 and 3 must be enclosed in parentheses and may be repeated for each file to be excluded from the dump. If all the files to be excluded cannot be specified on the first card or line then more parameter cards or lines with the same type number (5) can be specified. A group of parameter fields must be wholly contained on one card or line.

PARAMETER TYPE 6 (OPTIONAL): RENAME DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW6	5 (fixed)
*2 New name to be given to the dump file	See Note 5 on page 106	12

#XPJW

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*3 New generation number to be given to the dump file	Either 0 to 4095 for a direct access dump file or 0 to 8388607 for a magnetic tape dump file	4 7

*The group of parameter fields 2 and 3 must be enclosed in parentheses.

PARAMETER TYPE 7 (OPTIONAL): CREATE AND/OR EXTEND DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW7	5 (fixed)
2 Create the dump file	C Note: This is optional and may be omitted	1 (fixed)
3 Number of buckets by which the dump file is to be extended	1 to 8388607 Note: If this is omitted, where extension is necessary the program will extend the dump file for each file to be dumped. This may lead to Executives file handling limit being exceeded	7

Each parameter field must be terminated by a comma (or a space for parameter field 10 of types 1 and 2) except when followed by a parenthesis.

Function 4: dump specified files following previous #XPJW dumps

A minimum of two parameter types must be input to the program. Type 1 (for direct access) or type 2 (for magnetic tape) defines the dump identifier and the existing #XPJW dump file and type 5 defines the files to be dumped. Type 7 is an optional parameter type and may be input to extend the dump file on a direct access storage unit.

PARAMETER TYPE 1: FOR DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW1	5 (fixed)
2 Function code	4	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
5 Serial number of the unit to which extensions will be made	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)
*6 Name of the existing dump file	See Note 5 on page 106	12
*7 Generation number of the existing dump file	–1,0 to 4905	4
*8 Serial number of the storage unit holding the file index cell of the existing dump file	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)

*The group of parameter fields 6 to 8 must be enclosed in parentheses.

PARAMETER TYPE 2: FOR MAGNETIC TAPE DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW2	5 (fixed)
2 Function code	4	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
*5 Name of the existing dump file	See Note 5 on page 106	12
*6 Generation number of the existing dump file	0 to 8388607 Note: If this is not specified, it is taken as 0	7
*7 Serial number of the first magnetic tape holding the existing dump file	1 to 37777777 (an octal number) Note: This is optional and need not be specified	8

#XPJW

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*8 Maximum block size required for the dump file	1, 2, 3 or 4 (as a multiple of 1024) Note: If this is omitted it is taken as 1	1

*The group of parameter fields 5 to 8 must be enclosed in parentheses.

PARAMETER TYPE 5: DIRECT ACCESS FILES TO BE DUMPED

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW5	5 (fixed)
*2 Name of the file to be dumped	See Note 5 on page 106	12
*3 Generation number of the file to be dumped	–1,0 to 4095	4
*4 Serial number of the storage unit holding the file index cell of the file to be dumped	000001 to 777777 (an octal number) Note: This is optional and need not be specified	6 (fixed)
*5 Number of buckets to be dumped	0 to 8388607 Note: If this is not specified or specified as zero, all of the file from the bucket specified in parameter field 6 to the last allocated bucket for the file will be dumped	7
*6 Number of the first bucket to be dumped	0 to 8388607 Note: If this is omitted or specified as zero, dumping will begin from the first bucket of the file	7

*The group of parameter fields 2 to 6 must be enclosed in parentheses and may be repeated for each file to be dumped. If all the files to be dumped cannot be specified on the first card or line then more parameter cards or lines with the same type number (5) can be specified. A group of parameter fields must be wholly contained on one card or line.

PARAMETER TYPE 7 (OPTIONAL): EXTEND THE DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW7	5 (fixed)
2 Number of buckets by which the dump file is to be extended	1 to 8388607 Note: If this is omitted, where extension is necessary the program will extend the dump file for each file to be dumped. This may lead to Executives file handling limit being exceeded	7

Each parameter field must be terminated by a comma except when followed by a parenthesis.

Function 5: dump all the files on a storage unit following previous #XPJW dumps

At least one parameter type must be input to the program. Type 1 (for direct access) or type 2 (for magnetic tape) defines the existing dump file and the storage unit from which the dump is to be made. Types 5 and 7 are optional parameter types. Type 5 may be input if files are to be excluded from the dump. Type 7 may be input to extend the dump file on a direct access storage unit.

PARAMETER TYPE 1: FOR DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW1	5 (fixed)
2 Function code	5	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
5 Serial number of the unit to which extensions will be made	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)
*6 Name of the existing dump file	See Note 5 on page 106	12
*7 Generation number of the existing dump file	-1,0 to 4095	4
*8 Serial number of the storage unit holding the file index cell of the existing dump file	000001 to 777777 Note: This is optional and may be omitted	6 (fixed)
9 Serial number of the storage unit whose files are to be dumped	000001 to 777777 (an octal number)	6 (fixed)

*The group of parameter fields 6 to 8 must be enclosed in parentheses.

PARAMETER TYPE 2: FOR MAGNETIC TAPE DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW2	5 (fixed)
2 Function code	5	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
*5 Name of the existing dump file	See Note on page 106	12
*6 Generation number of the existing dump file	0 to 8388607	7

#XPJW

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
*7	Serial number of the first magnetic tape holding the existing dump file	1 to 37777777 (an octal number) Note: This is optional and need not be specified	8
*8	Maximum block size required for the file	1, 2, 3 or 4 (as a multiple of 1024) Note: If this is omitted it is taken as 1	1
9	Serial number of the storage unit whose files are to be dumped	000001 to 777777 (an octal number)	6 (fixed)

*The group of parameter fields 5 to 8 must be enclosed in parentheses.

PARAMETER TYPE 5 (OPTIONAL): DIRECT ACCESS FILES TO BE EXCLUDED FROM THE DUMP

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJW) and parameter type	XPJW5	5 (fixed)
*2	Name of the file to be excluded from the dump	See Note 5 on page 106	12
*3	Generation number of the file to be excluded from the dump	0 to 4095	4

*The group of parameter fields 2 and 3 must be enclosed in parentheses and may be repeated for each file to be excluded from the dump. If all the files to be excluded cannot be specified on the first card or line then more parameter cards or lines with the same type number (5) can be specified. A group of parameter fields must be wholly contained on one card or line.

PARAMETER TYPE 7 (OPTIONAL): EXTEND DIRECT ACCESS DUMP FILE

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJW) and parameter type	XPJW7	5 (fixed)
2	Number of buckets by which the dump file is to be extended	1 to 8388607 Note: If this is omitted, where extension is necessary the program will extend the dump file for each file to be dumped. This may lead to Executive's file handling limit being exceeded	7

Each parameter field must be terminated by a comma (or a space for parameter field 9 of types 1 and 2) except when followed by a parenthesis.

Function 6: restore specified files

A minimum of two parameter types must be input to the program. Type 1 (for direct access) or type 2 (for magnetic tape) defines the dump and the dump file (#XPJW created) from which the restore is to be made, and type 5 defines the files to be restored. Type 7 is an optional parameter type and may be input to create the files and file areas to which the restore is to be made.

PARAMETER TYPE 1: FOR DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW1	5 (fixed)
2 Function code	6	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
*5 Name of the dump file	See Note 5 on page 106	12
*6 Generation number of the dump file	-1,0 to 4095	4
*7 Serial number of the storage unit holding the file index cell of the dump file	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)

*The group of parameter fields 5 to 7 must be enclosed in parentheses.

PARAMETER TYPE 2: FOR MAGNETIC TAPE DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW2	5 (fixed)
2 Function code	6	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
*5 Name of the dump file	See Note 5 on page 106	12
*6 Generation number of the dump file	0 to 8388607	7
*7 Serial number of the first magnetic tape holding the dump file	1 to 37777777 Note: This is optional and may be omitted	8

*The group of parameter fields 5 to 7 must be enclosed in parentheses.

#XPJW

PARAMETER TYPE 5: DIRECT ACCESS FILES TO BE RESTORED

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJW) and parameter type	XPJW5	5 (fixed)
*2	Name of the file to be restored	See Note 5 on page 106	12
*3	Generation number of the file to be restored	0 to 4095	4
*4	Serial number of the storage unit to which the restore is to be made, if different to the unit from which the dump was made	000001 to 777777 (an octal number) Note: If this is not specified, the restore will be made to the unit from which the dump was made	6 (fixed)
*5	Number of buckets to be restored from the dump	0 to 8388607 Note: If this is specified as zero, restoration will take place up to the last bucket dumped	7
*6	Number of the first bucket to be restored	0 to 8388607 Note: If this is specified as zero, restoration will begin from the first bucket dumped	7

*The group of parameter fields 2 to 6 must be enclosed in parentheses and may be repeated for each file to be restored. If all the files to be restored cannot be specified on the first card or line then more parameter cards or lines with the same type number (5) can be specified. A group of parameter fields must be wholly contained on one card or line.

PARAMETER TYPE 7(OPTIONAL): CREATE DIRECT ACCESS FILES FOR RESTORE

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJW) and parameter type	XPJW7	5 (fixed)
2	Create the files and/or file areas to which the restore is to be made	C	1 (fixed)

Each parameter field must be terminated by a comma except when followed by a parenthesis.

Function 7: restore all files from a dump

At least one parameter type must be input to the program. Type 1 (for direct access) or type 2 (for magnetic tape) defines the dumped and the dump file (#XPJW created) from which the restore is to be made. Types 5 and 7 are optional parameter types. Type 5 may be input to exclude files from the restore. Type 7 may be input to create the files and the file areas to which the restore is to be made.

PARAMETER TYPE 1: FOR DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW1	5 (fixed)
2 Function code	7	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
*5 Name of the dump file	See Note 5 on page 106	12
*6 Generation number of the dump file	–1,0 to 4095	4
*7 Serial number of the storage unit holding the file index cell of the dump file	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)
8 Serial number of the storage unit to which the restore is to be made, if different to the unit from which the dump was made	000001 to 777777 (an octal number) Note: If this is omitted, the restore will be made to the unit from which the dump was made	6 (fixed)

*The group of parameter fields 5 to 7 must be enclosed in parentheses.

PARAMETER TYPE 2: FOR MAGNETIC TAPE DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW2	5 (fixed)
2 Function code	7	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as multiple of 1024) Note: This is optional and may be omitted	2
*5 Name of the dump file	See Note 5 on page 106	12
*6 Generation number of the dump file	0 to 8388607	7
*7 Serial number of the first magnetic tape holding the dump file	1 to 37777777 (an octal number) Note: This is optional and may be omitted	8

#XPJW

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
8	Serial number of the storage unit to which the restore is to be made, if different to the unit from which the dump was made	000001 to 777777 (an octal number) Note: If this is omitted, the restore will be made to the unit from which the dump was made	6 (fixed)

*The group of parameter fields 5 to 7 must be enclosed in parentheses.

PARAMETER TYPE 5 (OPTIONAL): FILES TO BE EXCLUDED FROM THE RESTORE

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJW) and parameter type	XPJW5	5 (fixed)
*2	Name of the file to be excluded from the restore	See Note 5 on page 106	12
*3	Generation number of the file to be excluded from the restore	0 to 4095	4

*The group of parameter fields 2 and 3 must be enclosed in parentheses and may be repeated for each file to be excluded from the restore. If all the files to be excluded cannot be specified on the first card or line then more parameter cards or lines with the same type number (5) can be specified. A group of parameter fields must be wholly contained on one card or line.

PARAMETER TYPE 7 (OPTIONAL): CREATE DIRECT ACCESS FILES FOR RESTORE

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJW) and parameter type	XPJW7	5 (fixed)
2	Create the files and/or file areas to which the restore is to be made	C	1 (fixed)

Each parameter field must be terminated by a comma (or a space for parameter field 8 of types 1 and 2) except when followed by a parenthesis.

Function 8: restore specified files to one or more different files

A minimum of three parameter types must be input to the program. Type 1 (for direct access) or type 2 (for magnetic tape) defines the dump and the dump file (#XPJW created) from which the restore is to be made. Type 3 defines a file that is to be restored, and type 4 defines the file to which the restore is to be made. Types 6 and 7 are optional parameter types. Type 6 may be input to rename the file to which the restore is to be made. Type 7 may be input to create the file and/or file areas to which the restore is to be made. Types 3 and 4 and optionally types 6 and 7 must be repeated for each dumped file which is to be restored to a different file.

PARAMETER TYPE 1: FOR DIRECT ACCESS DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW1	5 (fixed)
2 Function code	8	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
*5 Name of the dump file	See Note 5 on page 106	12
*6 Generation number of the dump file	-1,0 to 4095	4
*7 Serial number of the storage unit holding the file index cell of the dump file	000001 to 777777 (an octal number) Note: This is optional and may be omitted	6 (fixed)

*The group of parameter fields 5 to 7 must be enclosed in parentheses.

PARAMETER TYPE 2: FOR MAGNETIC TAPE DUMP FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW2	5 (fixed)
2 Function code	8	1 (fixed)
3 Dump identifier	4 characters to identify the dump	4 (fixed)
4 Number of words of program store required to include increased buffer area (see Note 6 on page 106)	7 to 32 (as a multiple of 1024) Note: This is optional and may be omitted	2
*5 Name of the dump file	See Note 5 on page 106	12
*6 Generation number of the dump file	0 to 8388607	7
*7 Serial number of the first magnetic tape of the dump file	1 to 37777777 (an octal number) Note: This is optional and may be omitted	8

*The group of parameter fields 5 to 7 must be enclosed in parentheses.

#XPJW

PARAMETER TYPE 3: DIRECT ACCESS FILES TO BE RESTORED

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW3	5 (fixed)
*2 Name of the file that is to be restored	See Note 5 on page 106	12
*3 Generation number of the file to be restored	0 to 4095	4
*4 Number of buckets to be restored from the dump	0 to 8388607 Note: If this is specified as zero, restoration will take place up to the last bucket dumped	7
*5 Number of the first bucket to be restored	0 to 8388607 Note: If this is specified as zero, restoration will begin from the first bucket dumped	7

*The group of parameter fields 2 to 5 must be enclosed in parentheses.

PARAMETER TYPE 4: DIRECT ACCESS FILES TO WHICH RESTORE IS TO BE MADE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJW) and parameter type	XPJW4	5 (fixed)
2 Serial number of unit to which extensions will be made	000001 to 777777 Note: This is optional and may be omitted	6 (fixed)
*3 Name of the file to which the restore is to be made	See Note 5 on page 106	12
*4 Generation number of the file to which restore is to be made	–1,0 to 4095	4
*5 Serial number of the storage unit holding the file index cell of the file	000001 to 777777 Note: If the file index cell of the file is to be created, this specifies the unit to hold the cell. Otherwise, this need not be specified	6 (fixed)
*6 Retention period to be given to the file	0 to 4095 (in days)	4
*7 Number of the bucket at which the restore will begin	0 to 8388607	7
*8 Value to be given to the EOFBN of the file	1 to 8388607 Note: This is optional and may be omitted	7

*The group of parameter fields 3 to 8 must be enclosed in parentheses.

PARAMETER TYPE 6 (OPTIONAL): RENAME RESTORE FILE

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJW) and parameter type	XPJW6	5 (fixed)
*2	New name to be given to the file to which the restore is to be made	See Note 5 on page 106	12
*3	New generation number to be given to the file to which the restore is to be made	0 to 4095	4

*The group of parameter fields 2 and 3 must be enclosed in parentheses.

PARAMETER TYPE 7 (OPTIONAL): CREATE DIRECT ACCESS FILE TO WHICH RESTORE IS TO BE MADE

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJW) and parameter type	XPJW7	5 (fixed)
2	Create the file and/or file areas to which the restore is to be made	C	1 (fixed)

Each parameter must be terminated by a comma (or a space for parameter field 2 of type 7) except when followed by a parenthesis.

End of parameters

A card or line with four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a LOad or FInd message. Further details are given in the section *Parameter formats*, page 2.

Notes:

- 1 If the generation number is specified as -1, the highest generation number of the specified file on-line will be used. If a generation number is not specified it will be taken as a generation number of zero whether it is an input or output file. A generation number of -1 cannot be specified with parameter types 3 and 6 or with parameter type 5 for functions 3, 5, 6 and 7
- 2 The dump identifier is a 4-character name, such as FRED, which serves to identify a particular dump on a multi-dump file
- 3 The program handles end of file tags by reading or updating the value held in the file index cell of the file, and not by reading or updating a bit setting in a bucket header
- 4 When a dump is made to a direct access device, the number of dump file buckets needed can be calculated using the following formula:

$$\begin{array}{rclclcl}
 \text{Number of} & & \text{Number of} & & \text{Number of} & & \text{Number of blocks} \\
 \text{dump file} & & \text{files to} & & \text{file areas} & & \text{in file to be} \\
 \text{buckets} & & \text{be dumped} & & \text{to be dumped} & & \text{dumped} \\
 \text{needed} & = & \frac{\text{5 x number}}{\text{of blocks/}} & + & \frac{\text{15 x number}}{\text{of blocks/}} & + & \frac{\text{Number of blocks/}}{\text{bucket of dump}} \\
 & & \text{bucket of} & & \text{bucket of} & & \text{file} \\
 & & \text{dump file} & & \text{dump file} & & \\
 & & \text{A} & & \text{B} & & \text{C}
 \end{array}$$

Terms A, B and C are each rounded up to the next integer. Term C is repeated for each file in the dump. The dump file will be extended if there are insufficient buckets

#XPJW

- 5 A number of checks are carried out before processing is allowed to commence.

A check is made on each parameter that is within the stated range and maximum size.

The program carried out checks on each direct access file, excluding the dump file, before processing commences. The generation number is regarded as part of the file name. A check is made to ensure that a file is not specified more than once for a function (this does not apply to function 1). For function 8 this check is performed on files defined by parameter type 3 only. Before creating a file a check is made to ensure that the file name consists of up to 12 alphanumeric, space or minus characters, and that the first character is alphabetic. This check is not performed when files are created in functions 6 and 7. A file which is not created by the program may have a file name consisting of any character other than a comma or close parenthesis.

When dumping or restoring a check is made to ensure that each file is on-line and can be opened. For functions 3 and 5 the program will open the systems file of the unit to be dumped to ascertain the files to be read. The user should ensure that Executive or another program is not writing to this system file while #XPJW is reading it.

On restoring, a check is made to ensure that the specified file is in the dump and that the specified section of a file lies within the allocated file boundaries

- 6 To improve the efficiency of data transfers the user can request a larger store area. This additional store will be used as buffer areas. If this facility is not used, transfers are effected by using a single buffered method with two buffers of 1024 words each. If parameters are input to increase the program store and, for magnetic tape, to increase maximum block size, a double buffered method may be used. For transfers between one direct access storage and another a buffer area of at least 4096 words should be requested for double buffered transfers to be effected.

If the user wishes to specify an amount of additional store that may be used as extra buffer space, the store size should be calculated as shown below. Then either the program should be loaded into the store required or the size should be specified in the type 1 parameter.

The store size may be calculated with the following formulae:

- (a) Store size needed if dump medium is magnetic tape is $1024(5+lm+n)$ words

where

l = buffer size for magnetic tape buffers expressed as fraction of 1024

$m = 1$ if single buffering on magnetic tape or if $l \geq 3$

$m = 2$ if double buffering

n = buffer size for direct access buffer expressed as a fraction of 1024

- (b) Store needed if dump medium is direct access is $1024(5+pq)$ words

where

$p = 2$ if single buffering

$p = 4$ if double buffering

q = buffer size expressed as a fraction of 1024

When calculating store size for #XPJW, an allowance additional to that calculated from the formula given above may be required according to the number of files involved, as given below

- (a) FUNCTION 1 No additional allowance is necessary
- (b) FUNCTIONS 2, 3, 4, 5 AND 6 For every file, above 12 files, to be dumped or restored allow 10 extra words
- (c) FUNCTION 7 For every file above 11 files to be excluded from the restore allow 10 extra words
- (d) FUNCTION 8 For every file, above 5 files, to be restored allow 24 extra words

In each case the allowance should be rounded up to the next multiple of 1024.

When restoring from a magnetic tape dump the program will ensure that it has enough storage for the block size used. The program will continue with the run provided that the program can obtain the minimum store requirement even though this may be less than the specified amount

Summary of parameter types used for each function

<i>Function</i>	<i>Parameter type</i>	<i>Contents</i>	<i>Necessity</i>
1	1	Input file	Compulsory
	4	Output file	Compulsory
	6	Rename	Optional
	7	Create output file	Optional
2	1 or 2	Dump file	Compulsory
	5	Specified files	Compulsory
	6	Rename	Optional
	7	Create dump file	Optional
3	1 or 2	Dump file	Compulsory
	5	Files to be excluded	Optional
	6	Rename	Optional
	7	Create dump file	Optional
4	1 or 2	Dump file	Compulsory
	5	Specified files	Compulsory
	7	Extend dump file	Optional
5	1 or 2	Dump file	Compulsory
	5	Files to be excluded	Optional
	7	Extend dump file	Optional
6	1 or 2	Dump file	Compulsory
	5	Specified files	Compulsory
	7	Create files	Optional
7	1 or 2	Dump file	Compulsory
	5	Files to be excluded	Optional
	7	Create files	Optional
8	1 or 2	Dump file	Compulsory
	3	Dumped file	Compulsory
	4	Output file	Compulsory
	6	Rename	Optional
	7	Create files	Optional

Examples

FUNCTION 1

- 1 Fifteen buckets starting at bucket number 44 on the file INPUT FILE, with generation number 4, on the storage unit 717001 are to be copied to the file OUTPUT FILE, with generation number 0 starting at bucket number 400. The file will be renamed FILEA and have a retention period of 20 days. If necessary, the file will be extended on to 721313 provided that the storage unit holding the last file area of the output file is full. No extra store for buffer areas is required.

The parameters required to achieve this would be as follows:

XPJW1,1(INPUT FILE,4,717001,15,44)

XPJW4,721313(OUTPUT FILE,,,20,400)

XPJW6(FILEA)

#XPJW

- 2 A file, **SECONDFILE**, with generation number 0 is to be created before all allocated buckets for **FIRSTFILE** with generation number 0 are copied to it. The store size for the program is to be extended to 10,240 words to speed up data transfer.

XPJW1,1,10(FIRSTFILE,,,0,0)

XPJW4(SECONDFILE,0,71501,2,0)

XPJW7,C

FUNCTION 2

Five files are to be dumped to the magnetic tape **SCRATCH TAPE** to be renamed **DUMPFIL** with generation number 6 and a retention period of 70 days. The program will check that the tape serial number is 100233. The dump, which will have the identity **PAUL** will consist of the whole of **FILEA** (generation number 0), the whole of **FILEB** (generation number 0), 10 buckets starting at the second bucket of **FILEC** (generation number 2) whose **FIP** cell is on 717007, the first 4 buckets of **FILED** (generation number 0) and the rest of the file starting at bucket 100 of **FILEE** (generation number 0). No extra store for buffer is required. The parameters required to achieve this are as follows:

XPJW2,2,PAUL(SCRATCH TAPE,,100233,70)

XPJW5(FILEA)(FILEB)(FILEC,2,717007,10,2)

XPJW5(FILED,,,4)(FILEE,,,,100)

XPJW6(DUMPFIL,6)

FUNCTION 3

- 1 With the exception of the files **TED**, **FRED**, both with generation number 0, **RED**, with generation number 3 and **BLACK**, with generation number 3920, all the files on the storage unit 724100 are to be dumped to the magnetic tape **MTDUMP**, with generation number 2, and the dump will have the identity **JUNE**. The maximum block size will be 4096 words. No extra store for buffer areas is required. The parameters required to achieve this are as follows:

XPJW2,3,JUNE(MTDUMP,2,,,4)724100,

XPJW5(TED)(FRED)(RED,3)

XPJW5(BLACK,3920)

- 2 All the files on the storage unit 717001 will be dumped to the disc file **DISCDUMP** (generation number 0) on unit 777665. The dump identifier is **CAKE**. The file **DISCDUMP** is to be created by the program and it will be extended onto 777666 if no free space exists on 777665. The file will have no retention period. The dump file will be extended by at least 482 buckets to accommodate the files to be dumped. No extra store for buffer areas is required. The parameters required to achieve this are as follows:

XPJW1,3,CAKE,777666(DISCDUMP,,777665)717001

XPJW7,C,482

FUNCTION 4

A dump with the identity **APPE** is to be added to the dump file produced by the example for function 2. This dump will consist of 30 buckets starting at bucket 15 of **UNCLE** (generation number 0) on unit 721461. The maximum block size will be 2048 words. The program will extend itself to occupy 20,480 words of store.

XPJW2,4,APPE,20(DUMPFIL,6,100233,2)

XPJW5(UNCLE,,721461,30,15)

FUNCTION 5

A dump with the identity **SUDU** is to be added to the dump file produced by the second example for function 3. Files are to be dumped from storage unit 717011. The files to be excluded from the operation are **FIRST** (generation number 2), **SECOND** (generation number 10), **THIRD** (generation number 22), **FOURTH** (generation number 0), **FIFTH** (generation number 0) and **SIXTH** (generation number 1). A storage unit has not been specified for extensions. No extra store for buffer areas is required.

XPJW1,5,SUDU(DISC DUMP)717011,
 XPJW5(FIRST,2)(SECOND,10)
 XPJW5(THIRD,22)(FOURTH)(FIFTH)(SIXTH,1)

FUNCTION 6

Six files are to be restored from the dump OAKE on the disc file HDUMP. All the dumped buckets of ONE (generation number 2) and TWO (generation number 0) will be reinstated. File THREE (generation number 22) will have all of its dumped buckets restored to unit 724131. Only the first 5 buckets of SIX (generation number 1), the rest of the file starting at bucket 30 of FIVE (generation number 10) and two buckets starting at bucket 2 of FOUR (generation number 0) will be reinstated. Files and/or file areas are not to be created by the program. No extra store for buffer areas is required.

XPJW1,6,OAKE(HDUMP)
 XPJW5(ONE,2)(TWO)(THREE,22,724131)
 XPJW5(SIX,1,,5)(FIVE,10,,,30)
 XPJW5(FOUR,,,2,2)

FUNCTION 7

- 1 All the files in the dump PAUL on the magnetic tape DUMPFIL (generation number 6) are to be restored.

XPJW2,7,PAUL(DUMPFIL,6)

- 2 With the exception of the files FOUR (generation number 0), THREE (generation number 22) and THREE (generation number 21), all the files on the dump OAKE on the magnetic tape HDUMP are to be restored to the unit 772320. The program is allowed to create files and/or file areas.

XPJW2,7,OAKE(HDUMP)772320,
 XPJW5(FOUR)(THREE,22)(THREE,21)
 XPJW7,C,

FUNCTION 8

A dump, having the identifier IDEN on the disc file REPEAT which has a generation number of 0, is to be read. Starting at the first bucket dumped, all the dumped buckets of INA are to be copied to OUTA starting at bucket 1. OUTA is to be renamed RENA with a generation number of 0. Starting at bucket 20, all the dumped buckets of INB (generation number 3) are to be copied to OUTB (generation number 1), on storage unit 717007, starting at bucket 9. OUTB, which is to have file areas created for it, will have a retention period of 200 days. Starting at the first bucket dumped, all the dumped buckets of INC (generation number 0) are to be copied to OUTC (generation number 0), starting at bucket 3. OUTC will have a zero retention period and its EOFBN will be set to 70. 350 buckets are to be copied from the first bucket dumped of IND (generation number 0) to OUTD (generation number 3) starting at bucket 1. OUTD will be renamed REND with a zero generation number and storage unit 727230 may be used for extensions.

XPJW1,8,IDEN(REPEAT)
 XPJW3(INA)
 XPJW4(OUTA)
 XPJW6(RENA)
 XPJW3(INB,3,,20)
 XPJW4(OUTB,1,717007,200,9)
 XPJW7,C,
 XPJW3(INC)
 XPJW4(OUTC,,,,3,70)

#XPJW

XPJW3(IND,,350)

XPJW4,727230(OUTD,3)

XPJW6(REND)

OPERATING

Trusted status

The program has Q-trusted status.

Executive priority

The program as supplied has a priority of 70.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
TR0 or CR0	Parameters	The reader is assigned at the beginning of the run and is released when the batch terminator has been read
LP0	Output of dump and file extension reports and error messages	The line printer is assigned when a summary listing for a function is to be given or an error message is to be given. It is released at the end of the run
DA0, MT0	Input	The storage unit or magnetic tape is used throughout the run
DA1, MT1	Output	The storage unit or magnetic tape is used throughout the run. For functions 2 and 3 the named magnetic tape is opened with mode #300 and renamed by mode #500. Scratch tapes are opened for continuation reels, using mode #400
DA2	Systems file when dumping a complete unit, or scratch file	The unit is used within a job. The scratch file is used whilst creating file areas

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJW (see FIND A BINARY PROGRAM in <i>Compiling systems</i>) The message HALTED:— LD will only be output if the loading operation was successful	HALTED:— LD
2 Put the required input/output media on-line	
3 Load the parameters in the appropriate reader	

	<i>Narrative</i>	<i>Console message</i>
4	To activate the program: either	
	(a) If parameters are on paper tape, input:	GO #XPJW 20
	or	
	(b) If parameters are on cards, input:	GO #XPJW 21
5	At the end of the run, the program will output one of the following:	
	(a) If a delete message was correctly input, and a successful run has occurred:	DELETED:- <i>user message</i>
	(b) If the run was error free and no delete message was input:	HALTED:- HH
	(c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:	DISPLAY:- DE HALTED:- HH
	(d) If any error had occurred in the run (except the console typewriter exception conditions described below): (see also section <i>line printer messages</i> , page 112)	HALTED:- ZZ
6	For a further run, return to operating instruction 2	

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJW during execution and loading. Prior to entry into #XPJW it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	There is no card reader available	Make a card reader available and input: GO #XPJW
2	DISPLAY:- DE	The end parameter delete message did not start with the characters LO or FI	The run continues
3	HALTED:- LP	There is no line printer available	Make line printer available and input: GO #XPJW
4	HALTED:- NEEDS TAPE*nnnnnnnn	The check on a magnetic tape serial number has failed, where nnnnnnnn is the number specified in the parameters	Either (a) If the correct magnetic tape is available input: GO #XPJW or (b) Abandon the run by inputting: GO #XPJW 27

#XPJW

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
5	DISPLAY:- PE	Transfer failure on line printer	No action. The run continues
6	DISPLAY:- PL	A paper low condition has been detected	Load more paper and re-engage the printer
7	HALTED:- RF	A card or paper tape block has been misread	To reread the parameter line, reload the card/block and input: GO #XPJW
8	HALTED:- STxxxx	The program requires a minimum of xxxx more words of store	Either (a) When more core store is available input: GO #XPJW or (b) To abandon this function, input: GO #XPJW 27
9	HALTED:- TR	There is no paper tape reader available	Make a tape reader available and input: GO #XPJW
10	HALTED:- xx	An error has been detected when loading #XPJW by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when executive attempts to open the storage unit it will output a storage unit not on-line message, for example:

#XPJW; LDC * *nnnnnn* (the actual format of the message varies according to which executive is being used).

The user then has two options to take:

- 1 Put the requested storage unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJW 27 to abandon this function

LINE PRINTER MESSAGES

If any of the line printer messages listed in this subsection are printed, the run will terminate with the console message HALTED:- ZZ.

The first line printed has the following format:

#XPJW MKnn dd/mm/yy hh/mm/ss FILE COPY, DUMP AND RESTORE PAGEyyy

where

dd/mm/yy is the date on which the program is run.

hh/mm/ss is the time at which the program is run.

The second line to be printed will indicate the function to which the listed messages refer and it will have the following format:

JOB *a*

where *a* is the job number within the batch.

The format of the line printer messages is as follows:

**** *fn fgn message*

where

fn is the file name.

fgn is the file generation number.

(*fn fgn* are omitted from the line printer messages marked *).

message is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
1 ABANDON ENTRY POINT USED	This message is output if entry point 7 is used during the running of the program
2 BLOCK COUNT ERROR	A block count error has been encountered when restoring the file from a tape dump file. The resultant direct access file will be corrupt. The program will continue and try to restore any further files from the dump
3 BUCKET SIZES UNEQUAL	This message may be output for functions 1, 6, 7 and 8. This file is ignored
4 CANNOT CREATE FILE AREA <i>x</i> ERROR <i>n</i>	where <i>x</i> is the number of the file area within the file <i>n</i> is a number in the range 0 to 5 and will have the following meaning: 0 System Control Area full 1 Part or whole of the area assigned 2 For function 1, the file areas for the input and output file are specified as being on units with the same serial number 3 Executive cannot handle any more files 4 Invalid area 5 The first bucket of the first file area to be created does not immediately follow the last bucket already on the file If this message refers to a file to be dumped or restored this file will be omitted; otherwise the function will be abandoned
5 CANNOT CREATE FILE ERROR <i>n</i>	where <i>n</i> is either 0 or 3 and will have the following meaning: 0 Systems control area full 3 Executive cannot handle any more files If this message refers to a file to be dumped or restored, this file will be omitted; otherwise, the function will be abandoned
6 DUMP NOT FOUND - <i>xxxx</i>	The program cannot find the dump <i>xxxx</i> on the dump file

<i>Message</i>	<i>Reason and/or action</i>
7 DUPLICATE DUMP NAME -xxxx	A dump name xxxx already exists on the dump file. This message is output for functions 4 and 5 only
8 DUPLICATE FILE NAME	A file other than a dump file, of this name and generation number has been specified more than once for this function
9 FLAW AREA FULL xxxxxx	where xxxxxx is the serial number of the relevant storage unit. If this message refers to a file to be dumped or restored, this file will be omitted; otherwise the function will be abandoned
10 FORMAT ERROR ON nnnnnnnn	Either (a) During a restore from or when appending to, a tape dump file, a tape mark is encountered followed by an unrecognisable qualifier block or another tape mark. nnnnnnnn is the serial number of the tape concerned or (b) During a restore of only part of a file from a tape dump file a data block has been encountered when a sentinel or control block is expected. Alternatively a similar condition has been encountered when restoring a complete file or part file from a tape file created by versions of #XPJW previous to version 2A. This is likely to be due to a block count error on the dump file. The rest of the restore will be abandoned and a list of any files successfully restored will be output on the line printer
11 INSUFFICIENT EXECUTIVE CORE	If this file is to be dumped or restored, it will be omitted; otherwise, the function will be abandoned
12 INTEGRITY CODE FAILURE	If this file is to be dumped or restored, it will be omitted; otherwise, the function will be abandoned
13 INVALID CHARACTER	The file will not be created because the file name contains characters other than alphanumeric, space (V) or minus (-) characters
14 NEEDS xxxx WORDS OF STORE	The additional store is required for the list of files or the increased buffer area
15 NEEDS xxxxxxxx MORE BUCKETS	The file needs to be extended by at least xxxxxxxx buckets. If the file is a dump file xxxxxxxx is either the number of buckets required for the file currently being dumped or the number of buckets requested by parameter (type 7)
16 NO FILES DUMPED	Either (a) All the files to be dumped have failed the initial checks or a file to be excluded was not found or

<i>Message</i>	<i>Reason and/or action</i>
	(b) There are no files on the unit to be dumped or none remain after excluding files from a dump
	or
	(c) A file to be excluded from a dump occurs more than once on a unit
17 NO FILES RESTORED	Either
	(a) All files to be restored have failed the initial checks
	or
	(b) There are no files on the unit to be restored
	or
	(c) If files are excluded from the restore, no other files remain
18 NOT IN DUMP	Either the file does not appear in the dump or the part of it that is required is not in the dump. The file will be ignored
19 NOT ON LINE	If this file is to be dumped or restored, it will be omitted; otherwise the function will be abandoned
20 NOT ON UNIT xxxxxx	This file does not exist on unit xxxxxx. If the file was to be excluded from a dump, the function will be abandoned
*21 PARAMETER ERROR xxxxxxxxxxxxxxxxx ↑	where ↑ indicates the offending parameter
22 PARITY ERROR ON nnnnnnnn	A parity error has occurred on magnetic tape serial number nnnnnnnn. The function is abandoned
23 PARITY ERROR ON BKT xxxxxxx	where xxxxxx may be the number of the bucket concerned. However, the parity error may be in a bucket after xxxxxx, since the program can read or write more than one bucket at a time. This file will be ignored
24 PARTIAL RESTORE —NO BLOCK COUNT CHECK Note: This message has only two preceding asterisks (**)	This message is output to the line printer when only part of the file dump is to be restored from a tape dump file, thus precluding block count checks. The program continues
25 PART TO COPY OUTSIDE LIMITS	Either the start bucket number or the number of buckets to be actioned for a function is outside the physical limits of the file. The file will be ignored
26 PURGE DATE UNEXPIRED	If this file is to be dumped or restored, it will be omitted; otherwise, the function will be abandoned
27 RESTORED	This message is output for all files which have been successfully restored (except by function 7) if a fault is encountered on the dump file

#XPJW

	<i>Message</i>	<i>Reason and/or action</i>
28	SEQUENCE ERROR, PARAMETER XPJWt SEQxxxx Note: This message has only two preceding asterisks (**)	where <i>t</i> is the parameter type xxxx is the contents of columns 77 to 80.
29	UNACCEPTABLE DUMP FILE	A dump file on a storage unit is not in a format acceptable to the program

NAME

#XPJX

VERSION

Mark 1B

TITLE

Change data retention period or delete a direct access file

HARDWARE REQUIREMENT

- 1088 words of store
- 1 paper tape reader or 1 card reader
- 1 line printer
- 1 or more direct access storage units

DESCRIPTION

General

- The program has two functions
- 1 To change the data retention period of a direct access file
 - 2 To delete a direct access file regardless of any retention period

Input

Input to the program consists of one or more direct access storage units; parameters to the program can be input from cards or paper tape. Files on the storage units must conform to the ICL 1900 Series Unified Direct Access Standards. The parameters are described in the section *Control*, page 118.

Output

Output consists of one or more direct access storage units, with the system control area(s) updated to show the new state of files on the unit(s). A report on the specified storage unit(s) is output on the line printer using print positions 1 to 96. The format of the print layout is as follows:

#XPJX	MKn	dd/mm/yy	hh/mm/ss	RETENTION PERIOD CHANGE/FILE DELETE	PAGEyyy
UNIT SERIAL NO.	FILE IDENTITY	FGN	DATA RET	FUNCTION	
xxxxxx	xxxxxxxxxxxx	xxxx	xxxx	a . . . a	

where

dd/mm/yy is the date on which the report is printed.

hh/mm/ss is the time at which the report is printed.

yyy represents the page number.

xxx . . . x represents information relevant to each file.

a . . . a is either RETENTION PERIOD CHANGED or DEALLOCATED.

Note: If a file is deallocated the DATA RET column will be blank.

#XPJX

CONTROL

This section describes the parameters that are to be input to the program #XPJX for the two functions. Further details of parameter formats are given in section *Parameter formats*, page 2.

Function 1: change data retention period of a file

PARAMETER TYPE 1

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJX) and parameter type	XPJX1	5 (fixed)
2	Function identifier	1	1 (fixed)
3	Serial number of the unit on which the file is held	000001 to 777777 (an octal number)	6 (fixed)
4	Name of the file	Up to 12 alphanumeric, space (V) or minus (–) characters, with the first character alphabetic	12
5	Generation number of the file	0 to 4095	4
6	New data retention period	0 to 4095	4

Function 2: delete a file

PARAMETER TYPE 1

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJX) and parameter type	XPJX1	5 (fixed)
2	Function identifier	2	1 (fixed)
3	Serial number of unit on which the file is held	000001 to 777777 (an octal number)	6 (fixed)
4	Name of file to be deleted	Up to 12 alphanumeric, space (V) or minus (–) characters, with the first character alphabetic	12
5	Generation number of file to be deleted	0 to 4095	4

Each parameter field must be terminated by a comma. The complete set of parameters for a function must be terminated by an asterisk (*).

End of parameters

A card or line containing four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a L0ad or FInd message. Further details are given in the section *Parameter formats*, page 2.

Example

A file, OLDFILE, with generation number 7, is held on storage unit 700043. The retention period is to be changed to 365. A file, DATAFILE, with generation number 3, is held on storage unit 700044. The retention period is to be changed to 52. Also on storage unit 700044 is a file DATBFILE with generation number 2, which is to be deleted.

The parameters required to perform these functions would take the following form (if input on paper tape);

XPJX1,1,700043,OLDFILE,7,365,*↓

XPJX1,1,700044,DATAFILE,3,52,*↓

XPJX1,2,700044,DATBFILE,2,*↓

****↓

where ↓ indicates a newline character.

OPERATING

Trusted status

The program has Q-trusted status.

Executive priority

The program as supplied has a priority of 50.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0	Parameters	The reader is assigned at the beginning of the program and released when the batch terminator has been read
LP0	Report and error messages	The line printer is used throughout the run
DA0	Input file	The storage unit is used throughout the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJX (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required storage units	
3 Load the parameters in the appropriate reader	
4 To activate the program:	
(a) If parameters are on paper tape, input:	GO #XPJX 20
(b) If parameters are on cards, input:	GO #XPJX 21
5 When the batch terminator is read the program will output one of the following:	
(a) If a delete message was correctly input, and a successful run has occurred:	DELETED:- user message
(b) If no delete message was input in an error free run:	HALTED:- HH

#XPJX

<i>Narrative</i>	<i>Console message</i>
(c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:	DISPLAY:- DE HALTED:- HH
(d) If any error occurred during the run (except the error described in (c) above), whether or not a delete message was input: (Note: See the section <i>Line printer messages</i> , page 121)	HALTED:- ZZ

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJX during execution and loading. Prior to entry into #XPJX it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

<i>Message</i>	<i>Reason</i>	<i>Action</i>
1 HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJX or (b) Abandon the run
2 DISPLAY:- DE	The end parameter delete message did not start with the characters LO or FI	The run continues
3 HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJX or (b) Abandon the run
4 DISPLAY:- PE	Transfer failure on the line printer	No action. The run continues
5 DISPLAY:- PL	A paper low condition has been detected	Load more paper and re-engage the printer
6 HALTED:- RF	A card or paper tape block has been misread	To reread the parameter line reload the card/block and input: GO #XPJX
7 HALTED:- TR	There is no tape reader available	Either (a) Make a tape reader available and input: GO #XPJX

<i>Message</i>	<i>Reason</i>	<i>Action</i>
		or
		(b) Abandon the run
8 HALTED:- xx	An error has been detected when loading #XPJX by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when executive attempts to open the storage unit it will output a storage unit not on-line message, for example:

#XPJX; LDC *nnnnnn (the actual format of the message varies according to which executive is being used)

The user then has two options to take:

- 1 Put the requested storage unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJX 27 to abandon the current function

LINE PRINTER MESSAGES

If any of the error messages given below are output on the line printer, the message HALTED:- ZZ will be output on the console at the end of the run. Each message has the following format:

**** usn fn fgn message**

where

usn is the unit serial number.

fn is the name of the file.

fgn is the file generation number.

message is one of the following:

- 1 FILE NOT IN SYSTEM
- 2 FILE OPENED BY ANOTHER PROGRAM
- 3 INSUFFICIENT EXECUTIVE CORE
- 4 INTEGRITY CODE FAILURE
- 5 PARAMETER ERROR

xxxxxxxxx

↑

(where the arrow indicates the offending parameter).

File organization software

TITLE

File organization software

DESCRIPTION

General

The file organization software comprises two parts:

- 1 A generator, #XPJZ, to generate a skeleton PLAN source segment of a specified file organization or file copy utility program. The source may be generated to cards, paper tape, disc, magnetic tape or the line printer
- 2 The file organization or file copy utility program as generated by #XPJZ

(The specifications of these commence on pages 125 and 139 respectively).

#XPJZ will generate file organization programs to the user's specifications. The programs can have input from one or two files with output to a single file. Any of these files may be held on paper tape, punched cards, magnetic tape, or any unified direct access standards device. User-coded segments can be added to the generated program to edit any of these files; however, the files must be in the format required by the file organization program (see page 141).

The user can generate a file organization program for a specific purpose (transfer data from cards to direct access, say) or he can generate a multi-purpose program for which the number of input files and the medium of the files can be determined by variable run-time parameters.

#XPJZ will generate file copy utility programs to the user's specifications. The programs can have input from several direct access files, with output to a composite magnetic tape file or input from one composite magnetic tape file with output to several direct access files. User-coded segments can be added to the generated program to edit any of these files; however the files must be in the format required by the file organization program (see page 143).

The user can generate a file copy utility program for a specific purpose (for example: to transfer direct access files to tape) or he can generate a multi-purpose program for which the medium of the input and output files can be specified by run time parameters.

The skeleton PLAN source segments generated by #XPJZ together with user-coded segments (if any) is subsequently compiled by a PLAN compiler. The file organization subroutines must be present in a subroutine group/file when consolidating the program.

USES OF THE FILE ORGANIZATION PROGRAM

- 1 To load or reorganize direct access serial, sequential or random files. Sequential files will be set up to a specified bucket packing density, and with indexes as required. Overflow buckets will not be used.
A user-coded randomizing routine will be required in the generated program for the loading or reorganizing of a random file
- 2 To transfer data from one file to another, with the option of inserting or appending data from a second file. The files can be on different media
- 3 To analyze and update files. A file organization program containing suitable user-coded segments can be used to analyze and update files on any medium

USES OF THE FILE COPY UTILITY PROGRAM

- 1 To batch copy selected direct access files to composite magnetic tape subfile
- 2 To batch load/re-organize direct access files from selected composite magnetic tape subfiles. The facilities available are as for item 1 in *Uses of the file organization program*

USING THE FILE ORGANIZATION SOFTWARE

The user should read the complete specification of the file organization software and then:

File organization software

- 1 Prepare for #XPJZ the parameters which will give the required file organization or file copy utility program
- 2 Run #XPJZ to generate onto cards, paper tape, magnetic tape, direct access or the line printer the skeleton source segment of the desired program
- 3 Compile and consolidate this source segment together with any user-coded segments.

The following file organization subroutines must be present in a subroutine group/file when the program is consolidated. The order of the subroutines is important and must be in the sequence given below (reading down the columns):

HXPJZCNTRL2	HRAIPT	HXPJZWRBKT
HXPJZCNTRL3	HRAIMT	HXPJZDARTNS
HRACMO	HRACDO	HXPJZOPF
HRAICM	HRAPTO	HRAOPF
HXPJZMTO	HRAMTO	HRACN50
HXPJZIMT	HRAIDA	HXPJZCLOF
HXPJZPRINT	HXPJZPIRDAO	HRACLOF
HXPJZPIRIDA	HXPJZDAO	HXPJZMOVE
HXPJZRDBKT	HXPJZPIRDAS	HRAMOVE
HXPJZPIRMTO	HXPJZDAS	HRAERR1
HXPJZPIRIMT	HXPJZPIRDAQ	HXPJZCN40A
HRAPIRCMO	HXPJZDAQ	HRACLEAR
HRAPIRICM	HXPJZPIRDAR	HRACN40
HRAEDITIN1	HRADAR	HRAMESS
HRAEDITIN2	HXPJZNMPAR	HJZESAWR
HRAEDITOUT	HXPJZTAGREC	HRARND
HRAICD	HRATAGREC	

- 4 Prepare the necessary parameters and files for the file organization or file copy utility program
- 5 The file organization or file copy utility program can now be run

NAME

#XPJZ

VERSION

Mark 3A

TITLE

Generate a segment for a file organization or file copy utility program

HARDWARE REQUIREMENT

3648 words of store

1 card reader or 1 paper tape reader

1 line printer

1 card punch or 1 paper tape punch or 1 magnetic tape deck or a direct access storage unit

DESCRIPTION

General

#XPJZ is used to generate a skeleton PLAN source segment of a required file organization or file copy utility program.

#XPJZ will output a source segment on cards, paper tape, magnetic tape, direct access, or it may output on the line printer. This segment together with any user-coded segments (see below) should then be compiled and consolidated. The file organization subroutines must be present at consolidation (see page 124).

FILE ORGANIZATION PROGRAMS

File organization programs can be generated which can accept input from one or two files and output to a single file. Each file can be held on paper tape, cards, magnetic tape or direct access. Multi-purpose programs can be generated by specifying more than one media for each input and output file. The actual number of input files and the media containing each file can be specified in run-time parameters to the generated program.

Optionally, programs can be generated with the facility to double buffer magnetic tape and direct access files and multibucket direct access files as follows:

- 1 Double buffering for any of the following:
 - (a) Input from direct access files (see Note)
 - (b) Input from magnetic-tape files
 - (c) Output to non-random direct access files (see Note)
 - (d) Output to magnetic tape files
- 2 Multibucketing for any of the following:
 - (a) Input from direct access files (see Note)
 - (b) Output to non-random direct access files (see Note)

- 3 Multibucketing in conjunction with double buffering for the files given in 2 above (see Note)

Note: This does not include transfers of overflow buckets from input files, and index buckets to sequential output files which are always performed by single bucket transfers.

The double buffer and multibucket facility is incorporated in a program by specifying a PIR field in the #XPJZ parameter for the file (see section *control*, page 129). The buffer size specified to #XPJZ for a PIR file should be the maximum required to incorporate the desired double buffering/multibucketing at run time for the appropriate file. The actual combination of double buffering/multibucketing can be specified by run-time parameters to the generated program. However the combination achieved for a given run is determined within the constraints of the maximum buffer size specified at generate time and the bucket/block size of the files to be processed.

For a description of double buffering and multibucketing see *Direct Access*, Chapter 18.

File organization programs without PIR files can be run in compact mode (15AM and DBM) only.

File organization programs with PIR files can be run in any valid combination of compact mode (15AM), extended data mode (22AM), direct branch mode (DBM) and extended branch mode (EBM).

FILE COPY UTILITY PROGRAMS

File copy utility programs can be generated which can accept input from several direct access files with output to one composite magnetic tape file and/or accept input from one composite magnetic tape file with output to several direct access files. The actual direct access files and the input and output media are specified in run time parameters to the generated program.

File copy utility programs can only be run in compact mode (15AM, DBM).

USER-CODED SEGMENTS

User-coded segments can be incorporated into the generated file organization or file copy utility program for the editing of both input and output files (HRAEDITIN1, HRAEDITIN2, HRAEDITOUT), for the randomizer when outputting to direct access random files (HRAEND), and for dealing with overflow, in place of the Housekeeping System, on random direct access output files (HRAENDREC). The absence of any of the user-coded segments will result in dummy file organization subroutines of the same names being incorporated into the generated program during consolidation. For this reason the user should ensure that any user-coded segments are presented at consolidation time before the file organization subroutines.

Within the user-coded segments the user can process other files, either input or output, if he so desires. User-coded segments may contain calls to other segments as desired. Records cannot be generated within a user-coded segment for output by the generated program; input records can however, be amended or rejected.

Editing routines

A generated file organization program can read in records from one or two input files, either concurrently or serially, and output these to another file.

A generated file copy utility program can read in records from an input file serially and output these to another file. It will pass to the user-coded segments details of the direct access file currently being processed.

The following user-coded segments can be incorporated into the generated program while it is being assembled:

HRAEDITIN1	to process input file 1
HRAEDITIN2	to process input file 2 (file organization program only)
HRAEDITOUT	to process the output file

On entry to a user-coded HRAEDITIN1, HRAEDITIN2, or HRAEDITOUT, accumulators 1, 3 and 7 are set as follows:

Accumulator 1	contains the address of the record to be edited. If it has the value zero, there are no further records to be presented, and the file has been closed
Accumulator 3	(file copy utility programs only) contains the address of word 0 of an area which contains the filename in words 8, 9 and 10 and the generation number in word 12, of the direct access file being processed
Accumulator 7	contains the link to which the editing routine must exit on completion

Before exiting, the editing routine must give the following information in accumulator 1. The state of the other accumulators is irrelevant and if accumulator 1 was zero on entry, its state on exit is irrelevant.

- 1 If the record is not to be output, accumulator 1 must be set negative
- 2 If the record is to be output and has not been moved by the editing routine, then accumulator 1 may be either left in its original state or set to zero
- 3 If the record is to be output and has been moved by the editing routine to another area, then accumulator 1 must contain the new address of the record

A lower common area HRACOMINF contains the following information, which may be useful. It must not be altered by the editing routines.

Word 0 = 0 if there is no user information.

= 1 if user information is being taken from input file 1.

= 2 if user information is being taken from input file 2.

Word 1 = the address of the first word of the 16-word buffer for holding user information.

Word 2 = Bits 0 to 8 = the key length.

Bits 9 to 23 = the number of characters before the key in the record.

Restrictions

- 1 Records input either from magnetic tape or direct access files must not have their record lengths altered in the main program buffers. Such records may be moved to a buffer provided by the user and have their lengths altered there. If this is done, the new address of the record must be given in accumulator 1 on exiting from the user-coded segment
- 2 Following the processing of a record by a user-coded segment the main program assumes that the first word of the record contains the record length, held as a binary number, and that the characters in the position initially occupied by the record key are in fact the record key
- 3 If the user wishes to process magnetic or direct access files by a user-coded segment, units 4 to 15 should be used as the file unit numbers
- 4 If two input files are presented for sequential processing HRAEDITIN1 is the initial user-coded segment to be entered by the generated program that is the first input record entry is for the first record on input file 1

Randomizing routine

If records are being output to a direct access random file, a user-coded segment called HRARND must be supplied for address generation.

On entry to the segment, accumulators 1, 3 and 7 are set as follows:

Accumulator 1 contains the address of the record key

Accumulator 3 (file copy utility programs only) contains the address of word 0 of an area which contains the filename in words 8 to 10 and the generation number in word 12, of the direct access file being processed

Accumulator 7 contains the link to which the randomizing routine must exit on completion

HRARND must calculate the logical number of the bucket to which the record is to be written, and place this in accumulator 1 before exiting through the link in accumulator 7.

Overflow routine

When records are output to a direct access random file, bucket overflow may occur. Normally the generated program will follow the Direct Access Housekeeping system of putting a tag in the home bucket for an overflow record and storing the record in another bucket in the same seek area. If a different overflow system is to be used, a user-coded segment named HRATAGREC must be provided.

On entry to this segment, accumulators 1, 2, 3, 4 and 7 are set as follows:

Accumulator 1	contains the number of the bucket into which the tag will be inserted by the main program. When accumulator 2 is zero, the contents of accumulator 1 are irrelevant
Accumulator 2	either (a) contains the address of the record to be written away as an overflow record or (b) is zero, in which case there are no further records to be output; all buckets in store must be written away
Accumulator 3	contains the address of a buffer which may be used for accessing a bucket. This is the same on each entry to the segment
Accumulator 4	(file copy utility programs only) contains the address of word 0 of an area which contains the filename in words 8 to 10 and the generation number in word 12, of the direct access file being processed
Accumulator 7	contains the link to which the overflow routine must exit on completion

Before exiting, HRATAGREC must give the following information in accumulator 2. The state of the other accumulators is irrelevant.

- 1 Unless the files are to be closed immediately, accumulator 2 must contain the address of a tag to be put into the home bucket
- 2 If the files are to be closed immediately and the run is to be terminated, accumulator 2 must be set negative

HRATAGREC must write away, as an overflow record, the record whose address it has been given. Also, HRATAGREC must create a tag, which the main program will put into the record's home bucket. The record may already be an overflow record which is being moved from its current bucket to make room for a record which belongs in this bucket. In such a case the main program removes the old tag in the home bucket and inserts the new tag.

Notes:

- 1 The output file is open whenever HRATAGREC is called, and has a file reference number of 3
- 2 HRATAGREC must update the current overflow pointer in the first bucket of the appropriate seek area when this updating is required
- 3 If this is subsequently to be used as an input file for a file organization program, or for other software using Direct Access Housekeeping, tags and overflow records must be in Housekeeping format

Input

Parameters to the program can be input from cards or paper tape. The parameters are described in the section *Control*, page 129.

Output

The skeleton source segment output from the program may be on cards, paper tape, magnetic tape, direct access, or on the line printer. The segment is named SKELETONSEG.

The output on cards consists of the segment or segments required in PLAN source program format, sequence numbers for each segment beginning at 01 are in columns 79 and 80. There is the option of following each segment with a #SWITCH or #FINISH directive.

The output on paper tape consists of the segment or segments required in PLAN source program format. There is the option of following each segment with a #SWITCH or #FINISH directive.

The magnetic tape output consists of a file in a format acceptable to the PLAN compiler #XPLT. The output file may be renamed except when appending further subfiles to it. Each segment output to the tape is contained in a subfile named xxxxSKELETON where xxxx is the name of the program for which the segment is required. Each segment is named SKELETONSEG and is provided with a steering line 'LIST, OBJECT'.

The output to direct access will be in COSY format acceptable to the PLAN compilers #XPLZ and #XPLT; consolidation is required using #XPCK. The output file must have previously been allocated with one block buckets by #XPJC. The source segment which is output may be appended to a file which already contains source programs in COSY format. Each segment is named SKELETONSEG and is held in a subfile named xxxxVVVVVVVV where xxxx is the name of the program being generated.

Mode compatible file organization programs

It is possible that with the allocation of very large PIR buffers, a file organization program requires more than 32K words of store. If this limit is exceeded then the PLAN 4 compiler #XPLT must be used, and a steering segment with #PMODE directive presented to #XPLT prior to the #XPJZ generated source and user-coded segments (if any). In addition the title line of SKELETONSEG should be altered to:

```
#PROGRAM xxxxnn/SKELETONSEG(22AM,15AM,DBM,EBM)
```

where xxxx and nn are the program name and priority respectively.

Line printer output

Error messages are output to the line printer (see section *Line printer messages*, page 137) preceded by the following heading line:

```
#XPJZ MKn dd/mm/yy hh/mm/ss SEGMENT GENERATOR PAGEy
```

Extensions of a direct access output file are reported by the line:

```
PROGRAM xxxx fn fgn EXTENDED BY nnnnnn BUCKETS
```

CONTROL

This section describes the parameters that may be input to the program. Further details of parameter formats are given in the section *Parameter formats*, page 20.

There are four types of parameters that may be input to the program. Types 1 and 2 define the requirements for the generated program. Type 3 defines the output medium for the PLAN source segment. Type 4 is an optional parameter type and may be input to rename the output file.

Parameter type 1

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJZ) and parameter type	XPJZ1	5 (fixed)
2	Name and priority to be given to the generated program	4 alphanumeric characters followed by 2 digits	6 (fixed)
3	Maximum number of characters in the largest record key to be handled by the generated program	0 to 64	2
4	Multibucket and/or double buffer facilities required	PIR (must be omitted if file copy utility program, otherwise may be omitted if not required)	3 (fixed)
*5	Buffer size for input file 1 (see Note 1)	Either 20 to 9999 (in words) if field 4 omitted or	6

File organization software
#XPJZ

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
	20 to 120852 (in words) if field 4 specified	
*6 Maximum block/bucket size for input file 1	20 to 9999 (in words) (must be omitted if field 4 is omitted)	4
*7 Device type for input file 1	PT, CD, MT, DA or CM where PT = paper tape CD = cards MT = magnetic tape DA = direct access CM = composite magnetic tape (see Note 4) (DA and/or MT must be specified if field 4 is specified. If CM is specified a file copy utility program is generated and DA must be specified in field 6 of parameter type 2)	2 (fixed) per type
*8 Buffer size for input file 2 (see Note 1) (file organization program only otherwise omitted)	Either 20 to 9999 (in words) if field 4 omitted or 20 to 120852 (in words) if field 4 specified	6
*9 Maximum block/bucket size for input file 2 (file organization program only otherwise omitted)	20 to 9999 (in words) (must be omitted if field 4 omitted)	4
*10 Device type for input file 2 (file organization program only otherwise omitted)	PT, CD, MT or DA where PT = paper tape CD = cards MT = magnetic tape DA = direct access (DA and/or MT must be specified if field 4 is specified)	2 (fixed) per type

*The group of parameter fields 5, 6 and 7 must be enclosed in parentheses. Parameter fields 8, 9 and 10 are optional and if specified must also be enclosed in parentheses.

Parameter type 2

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJZ) and parameter type	XPJZ2	5 (fixed)

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
2 Size of buffer in the generated program for holding indexes (see Note 2)	0 to 2048 (in words) Note: This may be omitted	4
3 Multibucket and/or double buffer facilities required	PIR (This must be present if field 4 present in parameter type 1, and must be omitted if field 4 was not present in parameter type 1)	3 (fixed)
*4 Buffer size for output file (see Note 1)	Either 20 to 9999 (in words) if field 3 omitted or 20 to 120582 (in words) if field 3 specified	6
*5 Maximum block/bucket size for output file	20 to 9999 (in words) (must be omitted if field 3 is omitted)	4
*6 Device type for output file	PT, CD, MT, DA or CM where PT = paper tape CD = cards MT = magnetic tape DA = direct access CM = composite magnetic tape (see Note 4) (DA and/or MT must be specified if field 3 is specified. If CM is specified a file copy utility program is generated and DA must be specified in field 7 of parameter type 1)	2 (fixed) per type

*The group of parameter fields 4, 5 and 6 must be enclosed in parentheses.

Parameter type 3

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJZ) and parameter type	XPJZ3	5 (fixed)
2 Device type for PLAN source output	PT, CD, MT, DA or LP where PT = paper tape CD = cards MT = magnetic tape DA = direct access LP = line printer	2 (fixed)

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
	Note: If LP is specified parameters 3 to 7 should be omitted	
3 For cards or paper tape source output: segment terminator to be output	S or F where S = #SWITCH F = #FINISH	1 (fixed)
For magnetic tape or direct access source output: append segment	A Note: This may be omitted, but if specified the output source segment subfile will be added to the existing file	1 (fixed)
*4 Name of the source output file	Up to 12 alphanumeric space (V) or minus (-) characters with the first character alphabetic	12
*5 Generation number of the source output file: for magnetic tape file for direct access file	0 to 8388607 -1, 0 to 4095 Note: If this is not specified it is taken as 0	7 4
*6 Serial number: for magnetic tape for direct access	0 to 37777777 (an octal number) 000001 to 777777 (an octal number) Note: This need not be specified	8 6 (fixed)
*7 Serial number of direct access storage unit for extensions	000001 to 777777 (an octal number) Note: This is optional and may be omitted. If it is specified the program will extend the PLAN source program on to this unit if there is insufficient space available on the storage unit containing the current end of file bucket	6 (fixed)

*The group of parameter fields 4 to 7 must be enclosed in parentheses.

Parameter type 4 (optional)

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJZ) and parameter type	XPJZ4	5 (fixed)
*2 New file name	Up to 12 alphanumeric space (V) or minus (-) characters, with the first character alphabetic	12
*3 New generation number: for magnetic tape	0 to 8388607	7

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
for direct access	0 to 4095	4
*4 Retention period for magnetic tape	1 to 4095 (in days)	4
	Note: If this is omitted, it will be taken as 0	

*The group of parameter fields 2 to 4 must be enclosed in parentheses.

Each parameter field must be terminated by a comma except when followed by a parenthesis.

End of parameters

A card or line containing four asterisks (****) in columns 1 to 4 must follow a batch of parameters.

Notes:

- 1 Buffers must be large enough to hold the largest card or paper tape record, magnetic tape block or direct access bucket, and must be at least 20 words long. If PIR has been specified the buffer sizes should be calculated as the maximum buffer sizes to be used by the generated program for the appropriate file. The minimum size should be the size in words of the largest single transfer as a multiple of the direct access bucket or magnetic tape block size in words; plus 10 words for control information. This minimum size should be doubled if the generated program files will be double-buffered
- 2 When the output file of the generated program is on a direct access storage unit an area of main store may be reserved for holding indexes. The size of the index buffer should be equal to $(n-1)$ times (bucket size of the output file), where n is the number of levels of index, one index buffer being automatically provided for a sequential direct access output file. If the file is held on more than one unit and only one transport is available it is advisable to request the buffer area for the indexes to avoid frequent unit changes and improve the efficiency of the program
- 3 Each segment output to paper tape will be followed by a length of runout; each segment output to cards will be followed by one blank card
- 4 The presence of the device type CM results in a file copy utility program being generated, in which case the media specifiable by run time parameters are CM and/or DA alone

Examples

- 1 #XPJZ is required to generate the source of a file organization program #AAAA with a priority of 70. This program should be capable of inputting from a single file which can be direct access, magnetic tape or cards and outputting to a direct access file, magnetic tape or cards. The medium of the input and output files is to be specified at run time. The program should be capable of double-buffering to and from magnetic tape and direct access and multibucketing to and from direct access. The maximum key size that can be handled by the program is 39 characters; the maximum block/bucket size that can be handled by the program for both input and output is 512 words. A buffer is to be reserved for assembling two levels of index tables for the output file. The buffer area to be used for data transfers is to be reserved long enough to double-buffer with a multibucketing factor of four buckets. The source segment will be output on cards and terminated with a #FINISH directive.

The parameters required to achieve this are:
XPJZ1,AAAA70,39,PIR(4116,512,DA,MT,CD)
XPJZ2,1024,PIR(4116,512,DA,MT,CD)
XPJZ3, CD,F

- 2 #XPJZ is required to generate the source of a file copy utility program #AAAA, with a priority of 70. The program, #AAAA, should be capable of both reading and writing a composite magnetic tape file and therefore

File organization software
#XPJZ

reading and writing a direct access file. The medium of the input and output files is to be specified at run time. A buffer is to be reserved for assembling index tables for the output file. The maximum key size of any file to be handled is 39 characters. The source segment is to be appended to the direct access source file COSYSOURCE17, file generation number 0 which already contains other source files.

The parameters required to achieve this are:

XPJZ1,AAAA70,39(512,DA,CM)

XPJZ2,512(512,DA,CM)

XPJZ3,DA,A(COSYSOURCE1)

- 3 #XPJZ is required to generate the source of a file organization program #BBBB, with a priority of 67. This program should be able to input from one or two files. The first file can be on direct access, magnetic tape or punched cards, and the second, if used, on punched cards, the number of input files and the media for the files being specified at run time.

No buffer has been reserved for index tables.

The source segment will be output to magnetic tape, serial number 10234, which will be opened as a tape with the name SCRATCH7TAPE, renamed FOPLANSOURCE, file generation number 0, and given a retention period of 7 days.

The parameters required to achieve this will be:

XPJZ1, BBBB67, 10 (1024, DA, MT, CD)(100, CD)

XPJZ2 (1024, DA, MT)

XPJZ3, MT (SCRATCH7TAPE, 0, 10234)

XPJZ4 (FOPLANSOURCE, 0, 7)

OPERATING

Executive priority

The program as supplied has a priority of 80.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0	Input of parameters	The reader is assigned at the start of the run and released when the end of batch parameter has been read
LPO	Error messages, extension messages or output of the source segment for the program	The line printer is assigned when required
TPO or CPO	Output of the source segment for the program	The device is assigned when the program is ready to output a source segment, and is released when the segment has been output
MT0		The magnetic tape file is opened when first required and unloaded at the end of the run or when another magnetic tape file is required

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
DA0		The file is opened when the program is ready to output a source segment, and closed when the segment has been output

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJZ (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 If output is required on a magnetic tape or direct access, load the appropriate tape or storage unit	
3 Load the parameters in the appropriate reader	
4 To activate the program:	
(a) If parameters are on paper tape, input:	GO #XPJZ 20
(b) If parameters are on cards, input:	GO #XPJZ 21
5 At the end of the run the program will output one of the following:	
(a) In an error free run	HALTED:- HH
(b) If any error has occurred during the run (see section <i>Line printer messages</i> , page 137)	HALTED:- ZZ
6 For a further run, return to operating instruction 2	

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJZ during execution and loading. Prior to entry into #XPJZ it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

<i>Message</i>	<i>Reason</i>	<i>Action</i>
1 HALTED:- CP	There is no card punch available	Make a card punch available and input: GO #XPJZ
2 HALTED:- CR	There is no card reader available	Make a card reader available and input: GO #XPJZ
3 HALTED:- LP	There is no line printer available	Make a line printer available and input: GO #XPJZ

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
4	HALTED:- NEEDS TAPE *nnnnnnnn	A check on the magnetic tape serial number <i>nnnnnnnn</i> has failed, where <i>nnnnnnnn</i> is the serial number specified in the parameters	Either (a) If the correct tape is available load correct tape and input: GO #XPJZ or (b) If the correct tape is not available continue with the next set of parameters by inputting: GO #XPJZ 27
5	DISPLAY:- PE	Transfer failure on the line printer	No action. The run continues
6	HALTED:- PF	Incorrect punching of a card or paper tape block	Either (a) Remove the part of the segment output and to repeat the segment input: GO #XPJZ or (b) To continue with the next set of parameters input: GO #XPJZ 27
7	DISPLAY:- PL	A paper low condition has been detected	Load more paper and re-engage the printer
8	HALTED:- RF	A card or paper tape block has been misread	To re-read the parameter line, reload the offending card or paper tape block and input: GO #XPJZ
9	HALTED:- TP	There is no paper tape punch available	Make a paper tape punch available and input: GO #XPJZ
10	HALTED:- TR	There is no paper tape reader available	Make a paper tape reader available and input: GO #XPJZ
11	HALTED:- xx	An error has been detected when loading #XPJZ by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when executive attempts to open the storage unit it will output a storage unit not on-line message, for example:

#XPJZ; LDC *nnnnnn (the actual format of the message varies according to which Executive is being used)

The user then has two options to take:

- 1 Put the requested storage unit on-line and the program will continue

- 2 If the requested storage unit is not available then continue with the next set of parameters by typing
GO #XPJZ 27

LINE PRINTER MESSAGES

If any of the following messages are output on the line printer, the message HALTED:- ZZ is output at the end of the run.

Each of the following messages has the format:

PROGRAM *xxxx fn fgn message*

where

xxxx is the program name.

fn is the file name.

fgn is the file generation number.

(*fn fgn* are omitted from the line printer messages marked *.)

message is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
1 ABANDON ENTRY POINT USED	The console message GO #XPJZ 27 has been input
2 DIRECTORY SUBFILE FULL	There is no room for the new program <i>xxxx</i> subfile in the direct access output file
3 DUPLICATE PROGRAM NAME	A program with the name <i>xxxx</i> already exists. When appending to magnetic tape only the first program to be output to the tape in this run will be validated for this condition. When appending to a direct access file all program names will be checked
4 FLAW AREA FULL	An unsuccessful attempt has been made to write a bucket to the direct access output file
5 INCORRECT FILE FORMAT	The direct access output file either (a) Is not a one block/bucket file or (b) Has a corrupt bucket header and new subfiles cannot be added to the existing file
6 INCORRECT TAPE FORMAT	The magnetic tape output file is not in the correct subfile format
7 INSUFFICIENT EXECUTIVE CORE	The parameters for this file are ignored. The run continues with the next set of parameters
8 INSUFFICIENT SPACE	If output is to magnetic tape, the end of the file marker has been encountered. If this message is output when appending to magnetic tape, the trailer label will be repositioned to follow the last complete subfile. If output is to direct access, an unsuccessful attempt has been made to extend on to the current end of file storage unit
9 INSUFFICIENT SPACE ON * <i>csn</i>	If output is to direct access, and an extension storage unit was specified, an unsuccessful attempt has been made to extend on this storage unit with serial number * <i>csn</i>

<i>Message</i>	<i>Reason and/or action</i>
10 INTEGRITY CODE FAILURE	The parameters for this file are ignored. The run continues with the next set of parameters
11 NO SPACE SFD ON APPEND FILE	There is no space subfile directory on the direct access output file
12 NOT IN SYSTEM	The file <i>fn</i> with generation number <i>fgn</i> is not in the system
13 PARITY ERROR IN BKT <i>xxxxxxx</i>	A parity error has occurred when reading from the direct access output file
14 PARITY ERROR TSN <i>*tsn</i>	A parity error has occurred when reading from or writing to a magnetic tape with serial number <i>*tsn</i> . The message will indicate whether the error occurred during reading or writing. If this message is output when appending to magnetic tape, the trailer label will be repositioned to follow the last complete subfile
15 PREVIOUS TAPE ERROR	A previous error occurred when attempting to output another program to this tape during the run
*16 PARAMETER ERROR <i>xxxxxxxxxxxxx</i> ↑	where ↑ indicates the offending parameter. (This message is preceded by four asterisks (****) only)
17 PURGE DATE UNEXPIRED	The parameters for this file are ignored. The run continues with the next set of parameters
*18 SEQUENCE ERROR, PARAMETER XPJZ <i>t</i> SEQ <i>xxxx</i>	where <i>t</i> is the parameter type <i>xxxx</i> is the contents of columns 77 to 80 (This message is preceded by two asterisks (**) only)
19 SYSTEM CONTROL AREA FULL	An unsuccessful attempt has been made to extend the direct access output file
20 UNRECOGNISABLE REPLY	An unexpected reply has been given by Executive

File organization/file copy utility program (#XPJZ generated)

TITLE

File organization or file copy utility program (generated by #XPJZ)

HARDWARE REQUIREMENT

- 1 or 2 paper tape readers, card readers, magnetic tape decks or direct access files for input
- 1 card punch, paper tape punch, magnetic tape deck or direct access file for output
- 1 line printer (file copy utility program with output to composite magnetic tape only)

OPERATING MODE

File organization programs with PIR files may be run in any valid mode combination.

File copy utility programs or file organization programs without PIR files may be run in compact mode (15AM) and direct branch mode (DBM) only.

STORE SIZES

The sizes of completed file organization programs are variable and may be determined after compilation and consolidation of the source. The following formula gives an estimate of the number of words of store required:

- 1 For the required input and output facilities add the sizes given below. A size should not be doubled if the program is to handle two input files from the same medium.

<i>Medium</i>	<i>Words of store</i>
Paper tape input	140
Card input	280
Magnetic tape input	240 (150 if PIR files absent)
Direct access input	350 (200 if PIR files absent)
Paper tape output	120
Card output	160
Magnetic tape output	220 (131 if PIR files absent)
Direct access output	1276 (1400 if PIR files absent)
- 2 Add 1084
- 3 Either
 - (a) If the records can be input from cards, add four times the number of words required to hold the longest possible record key
 - or
 - (b) Otherwise add twice the number of words required to hold the longest possible record key
- 4 For the three files:
 - (a) For each PIR file which may be held on magnetic tape or direct access, add 24 plus the required buffer size plus the required maximum block/bucket size. For each non PIR file that may be held on direct access add 24 plus twice the required buffer size
 - (b) For each other file, add 24 plus the required buffer size
- 5 If direct access devices can be used for input or output or both, add 460 (or 160 if PIR files absent)

File organization/file copy utility program
(#XPJZ generated)

- 6 If the output file can be on magnetic tape, or on direct access, add 35
- 7 Add the required index area size and store sizes of user-coded segments
- 8 If output to direct access is possible and no user-coded overflow segment has been provided, add 158 (or 190 if PIR files absent)

DESCRIPTION

General

The PLAN source for the file organization or file copy utility program is generated by #XPJZ and then compiled and consolidated together with any user-coded segments. and the file organization subroutines (see *Using the file organization software*, page 123).

File organization program

A generated file organization program can be used to:

- 1 Transfer a file from one medium to another
- 2 Load a direct access serial sequential-indexed or random file
- 3 Reorganize a direct access sequential-indexed or random file

A generated file organization program basically will read in records from one or two files and, after optional editing and sequence checking, output to another file. The number of input files and the media holding the files involved is specified by parameter. Also specified by parameter is the size and position of the key within the records of the files involved. For direct access sequential output files buckets can be loaded to a specified packing density, and specified levels of indexes set up.

Double buffering and/or multibucketing of magnetic media files can be specified provided this facility was requested when the program was generated by #XPJZ.

User information (16 words) from a header or trailer card or paper tape block, a start of data sentinel or end of file trailer label on magnetic tape, or from the end of file bucket on direct access can be output to the output file. User information at the start of the output file will come from the start of an input file, and at the end of the output file will come from the end of an input file.

MODE OF PROCESSING

The mode of processing determines

- 1 When there are two input files, the order in which they are processed, and
- 2 The organization of a direct access output file

There are four modes of processing. These are serial, sequential, random and end-of-file serial.

Serial processing

- 1 The first input file is processed completely before any second input file is opened and processed
- 2 A serial file will be output to direct access. The direct access file need not have bucket headers set up on allocation. The generated program will write its own bucket headers. If necessary, the program will extend the file on to the current end-of-file storage unit

Sequential processing

- 1 If there are two input files they are both opened at the same time and their records merged to form a single output file. If records of the same key are input from the two files, then the record from the second input file is assumed to be a replacement for that on the first input file
- 2 A sequential file, with indexes if required, will be output to direct access. The direct access file need not have been allocated with bucket headers (see page 148). If a non PIR direct access file already contains data then

the records being input will be added beyond the current data in that file; if the output file has indexes and sequence checking has been specified then the generated program will check that the key of the first record input is higher than the current highest key recorded in the indexes for the file before outputting and amending the indexes; if the file does not have indexes and sequence checking is required the user must make sure that the key of the first record input is greater than the last on the file. If a PIR direct access file already contains data then the run is abandoned.

Optionally a dummy highest key may be specified which will be inserted as the last entry in each level of index

Random processing

- 1 Input files are opened one at a time as for serial processing
- 2 This mode is used when outputting to direct access random files. The file need not have been allocated with bucket headers (see page 147). If the file is not to be cleared any records in the file at the start of the run will not be overwritten. Keys or records input are presented to a user-coded randomizing routine for address generation. Tags are used when buckets become full

End-of-file serial processing

- 1 Input files are opened one at a time as for serial processing
- 2 This mode is used only when outputting to direct access serial files. The file need not have been allocated with bucket headers. Records are stored in the output file starting at the end-of-file bucket

File copy utility program

A generated file copy utility program can be used to:

- 1 Batch copy selected direct access files to successive subfiles of a composite magnetic tape output file
- 2 Batch copy back to direct access files, from selected subfiles of a composite magnetic tape input file

The files to be copied and the input/output media are specified by parameter. Also specified by parameter is the size and position of the key within the records of the files involved.

When the output media is specified as composite magnetic tape (CM) the generated file copy utility program will read in records from each input direct access file and, after optional editing and sequence checking, outputs to a simple subfile. The subfile is given the name and generation number of the direct access file being copied. The tape is written to 1900 magnetic tape housekeeping composite file formats, and if end-of-tape is reached a scratch tape is opened and used as a continuation reel. A list of the files copied together with a count of records read from each file and written to each subfile is output to the line printer.

When the input media is specified as composite magnetic tape (CM) the generated file copy utility program will, acting on each direct access file definition in turn, search the input tape for a subfile of the specified name. If the generation number is not that defined, the option is given of continuing, or processing the next output file definition (if any) or abandoning the run. The direct access file definitions must be presented in the order in which the subfiles are held on tape. If no subfile of the specified name can be found the run will be abandoned.

Each of the files specified in the definitions must have been previously allocated using #XPJC. The program will read in records from each input subfile and, after optional editing and sequence checking, output to a direct access file to the organization indicated by the mode of processing. The bucket packing, index setting-up and mode of processing facilities available are the same as for direct access files output by file organization programs.

Input

Parameters to the program may be input on cards or paper tape. The parameters are described in the section *Control*, page 144.

FILE ORGANIZATION PROGRAM

Input of data to a file organization program can be from one or two files, each of which can be held on cards, paper tape, magnetic tape or direct access.

File organization/file copy utility program
(#XPJZ generated)

Cards

Card files must be punched on 80-column cards, and the 64-character 1900 Series code must be used. Each card containing record information must bear the following sequence information:

Column 1 to n	A key for the record that is held or partly held on this card. This key is used to check the sequence of the cards. It must be the same number of characters as the key of the record, though it need not be the key of the record itself
Columns $n+1$ to $n+2$	The sequence number of the card within the cards of the same key containing a record. The sequence number must be in the range 01 to 99

The record must be punched in the remaining columns of the card. The first word of each record must be the record length punched as four decimal characters. Each record must start on a new card and when column 80 is reached on one card the contents of the record must be continued from column $n+3$ on the next card. Any space at the end of the last card of a record must be left blank.

The first record must be preceded by a header card and the last record followed by a trailer card in the following format:

Columns 1 to 4	HDDR for the header card
	EOFV for the trailer card
Columns 5 to 16	Name of card file
Columns 17 to 80	These may contain user information or may be omitted

Paper tape

Paper tape must be prepared using the I.S.O. standards for 8-track tape. Each record must be punched as a single block and terminated by a newline character. The first word of each record must give the length of the record in words, and must be punched as four decimal characters.

The first record must be preceded by a header block and the last record followed by a trailer block in the following format:

Word 0	HDDR for the header block
	EOFV for the trailer block
Words 1 to 3	Name of paper tape file
Words 4 to 19	These 16 words may contain user information or may be omitted

Each of these blocks must be terminated by a newline character.

If a file is punched on more than one reel of tape, each reel must be terminated by a standard end of tape block, which has the following format:

newline 5 deletes TC4 newline

Magnetic tape

Magnetic tape files must be in standard ICL 1900 Series magnetic tape housekeeping simple file format but only the following types of block will be processed:

Header labels

Start-of-data sentinels

Data blocks

Trailer labels

All other types of block will be ignored.

.

Direct access

Direct access files must be in standard 1900 Series direct access housekeeping formats. Files of any organization and with overflow can be input.

FILE COPY UTILITY PROGRAM

Input of data to a file copy utility program can be from a single composite magnetic tape file or from several direct access files.

The composite magnetic tape file must be in standard 1900 Series magnetic tape housekeeping format.

The direct access files must be in standard 1900 Series direct access housekeeping formats. Files of any organization and with overflow can be input.

Output

Output from a file organization program will be to a single file on paper tape, cards, magnetic tape or direct access. Direct access or magnetic tape files can be renamed. Either a named magnetic tape or a scratch tape can be specified for output, and scratch tape will be used for any continuation reels. Output from a file copy utility program will be to a single composite magnetic tape file or to one or more direct access files. Magnetic tape files can be renamed, and either a named tape or scratch tape can be specified for output. Scratch tapes will be used for any continuation reels. Output will be in the formats required for the input files.

FILE COPY UTILITY #AAAA GENERATED BY #XPJZ MK3A				
ANALYSIS OF THE SUBFILE WRITTEN ON 10/12/78 AT 09/12/14 TO THE MAGNETIC TAPES:=				PAGE 0001
TSN = 200007	FILENAME = VANSTAPE	FGN = 0000	RSN = 0000	RET = 0000
SUBFILE NAME	FGN	COUNTS		
WWDA=SERIAL	0000	00002500	00002500	
WWDA=SERIALB	0000	00002500	00002500	
WWDA=SERIALC	0000	00002500	00002500	
WWDB=RANDOM	0001	00000500	00000500	
WWDB=RANDOM	0003	00001000	00001000	
DCUPDATEFILE	0000	00000726	00000726	
DCMAST=FILEA	0000	00001652	00001652	
DCMAST=FILEB	0000	00001652	00001652	
DCMAST=FILEC	0000	00001089	00001089	
DCMAST=FILEC	0000	00001089	00001089	
WWDB=RANDOM	0006	00000500	00000500	
END OF ANALYSIS				

Figure 14 An example of an output report

Note: If an exception condition is detected by a file copy utility program during the processing of an input direct access file, an indication is given on the line printer by outputting four asterisks (****) in the first four print positions of the line containing the counts for the relevant file.

When output is to direct access, any second level overflow area will not be used. For sequential files any first level overflow areas will not be used and buckets can be packed to a specified percentage. Records will not be rejected if they exceed the maximum record size as specified on allocation; they will be output provided they can be accommodated within the usable space in a bucket after allowing for bucket packing density where specified. An exception is in the case of a PIR sequential file, when a record will be output provided it can be accommodated

within the usable space in a bucket even though it exceeds the bucket packing density (unless this is 100%). A count of such records will be displayed at the end of the run.

Note: George 3 and 4 users only. When direct access output is to a George 3 or 4 filestore file, the file must have been created with the **KEYWORDS** parameter to the **CREATE** command equal to a multiple of ten.

For file copy utility programs batch copying direct access files to tape, a line printer report of the files copied to the tape and the number of records input from each file before user editing and written to each subfile after user editing is output. An example of an output report is shown in Figure 14 (page 143).

CONTROL

This section describes firstly the sets of parameters to be input to a file organization program and secondly the sets of parameters to be input to a file copy utility program. Further details of punching formats of parameters are given in the section *Parameter formats*, page 2.

Parameters for a file organization program

The full set of parameters is input only if there are two input files. If there is only one input file, then the third card or paper tape block must be omitted.

FIRST PAPER TAPE BLOCK OR CARD: GENERAL INFORMATION

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name and sequence number	xxxx1 where xxxx has been chosen by the user	5 (fixed)
2 The number of input files for this run	1 or 2	1 (fixed)
3 Sequencing indicator	Y or N where Y = sequence of records input required N = no sequencing required	1 (fixed)
4 The number of the input file to supply user information from header cards/blocks, sentinels on tape, etc. (see <i>Input</i> , page 141)	0, 1 or 2 where 0 = no user information 1 = user information from input file 1 2 = user information from input file 2	1 (fixed)
5 Length of the key, in characters, for this run	0 to 99 (see Note 1)	2
6 Number of characters in record (including word 0 of record) which precede the key in the record	0 to 9999 (see Note 1)	4

SECOND PAPER TAPE BLOCK OR CARD: INPUT FILE 1

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name and sequence number	xxxx2 where xxxx has been chosen by the user	5 (fixed)
2 Type of device for input file 1 for this run	CD, PT, MT or DA where CD = card reader PT = paper tape reader MT = magnetic tape DA = direct access (DA or MT must be specified if field 3 is present)	2 (fixed)
3 Double buffering or multibucket facility	(DB) (DB, MBTnnn) (MBT) This field is optional and may be omitted. (See Note 7)	11 including parentheses
4 Name of the input file	Up to 12 alphanumeric, space (V) or minus (-) characters, with the first character alphabetic	12
5 The generation number of the input file	0 to 4095 (omitted for cards or paper tape)	4

THIRD PAPER TAPE BLOCK OR CARD: INPUT FILE 2

This has a sequence number of 3 and will be identical to the second paper tape block or card. It must be omitted if there is no second input file.

FOURTH PAPER TAPE BLOCK OR CARD: OUTPUT FILE

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name and sequence number	xxxx4 where xxxx has been chosen by the user	5 (fixed)
2 Type of output device for this run	CD, PT, MT or DA where CD = card punch PT = paper tape punch MT = magnetic tape DA = direct access (DA or MT must be specified if field 3 is present)	2 (fixed)

File organization/file copy utility program
(#XPJZ generated)

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
3	Double buffering or multibucket facility	(DB) (DB, MBTnnn) (MBT) This field is optional and may be omitted. See Note 7	11 including parentheses
4	Name of the output file (scratch tapes will be used for any continuation magnetic tape reels)	Up to 12 alphanumeric, space (V) or minus (-) characters with the first character alphabetic; or the four characters **** if a scratch tape is to be used	12
5	The generation number of the output file	0 to 4095 (omitted for cards or paper tape; need not be specified if field 4 is ****)	4
6	The retention period to be given to the output file	0 to 4095 (omitted for cards or paper tape)	4
7	Either If the direct access or magnetic tape file is to be renamed or to have its generation number changed the format of this field (which must be in parentheses) is: (New filename, new generation number) or To change the generation number of a direct access file only, the format of this field is: New generation number of direct access output file or Otherwise for cards, paper tape and a magnetic tape which is not to be renamed, the parameter must be omitted	Up to 12 alphanumeric, space (V) or minus (-) characters with the first character alphabetic 0 to 4095 0 to 4095	14 5 4
8	Mode of processing	E, Q, R or S where E = end-of-file serial Q = sequential R = random	1 (fixed)

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
	S = serial (See Note 2)	
9 File clear required (direct access only, otherwise omitted)	C or A where C = clear the file by writing bucket headers A = append to a non PIR sequential file and create bucket headers Note: This need not be specified (see Note 5)	1 (fixed)
10 Seek area packing density (direct access only, otherwise omitted)	0 to 99 Note: 0 = 100% (see Notes 3 and 4)	2
11 Bucket packing density (direct access only, otherwise omitted)	0 to 99 Note: 0 = 100% (see Notes 3 and 4)	2
12 Auxiliary header size (direct access only, otherwise omitted)	0 to 1022 (see Note 4)	4
13 Number of buckets for the level 1 index (direct access only, otherwise omitted)	0 to 99 (see Notes 3 and 4)	2
14 Number of buckets for each level 2 index (direct access only, otherwise omitted)	0 to 99 (see Notes 3 and 4)	2
15 Number of buckets for each level 3 index (direct access only, otherwise omitted)	0 to 99 (see Notes 3 and 4)	2

If a fifth paper tape block or card is to follow, field 15 must be terminated by a comma and a space.

FIFTH PAPER TAPE BLOCK OR CARD: DUMMY KEY FOR SEQUENTIAL DIRECT ACCESS OUTPUT FILE (SEE NOTE 6) (OPTIONAL)

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name and sequence number	xxxx5 where xxxx has been chosen by the user	5 (fixed)
2 Dummy key	Key of up to 64 alphanumeric characters	64

Each parameter field must be terminated by a comma; the complete set of parameters must be terminated by an asterisk (*).

Notes:

- The key specified here will be used to determine the size and position of the record key in all files to be processed during this run. If this differs from the key description recorded in the system control area for a particular direct access file, the generated program will report this by the message:

DISPLAY:- KEY ERROR ON FILE DAn

(see *Exception conditions*, page 155)

If, after performing this check on all the direct access files to be opened, one or more displays of the above message have been given for non-serial direct access files, then the generated program will halt with the message:

HALTED:- KX

and the run should be abandoned.

If the key information does not match for a serial file an indication will have been given, but the program will continue (assuming the key as specified by the parameters), as this is not likely to be a problem. For a non-serial file, the program cannot be continued

- 2 If the *mode of processing* field specified here does not agree with the file organization code recorded in the system control area for a direct access output file, then the generated program will halt with the message:

HALTED:- FT

and the run should be abandoned (see *Exception conditions*, page 155)

- 3 For direct access output files only. For non-sequential files, fields 10 to 11 and 13 to 15 should not be specified or specified as zero
- 4 Parameter fields 13 to 15 must be identical with the corresponding parameters to the file allocator if parameter 9 is not specified, and parameter fields 10 and 12 should not be specified or specified as zero since in this case these parameters are not relevant as the program assumes that bucket headers are already set up
- 5 If parameter field 9 is specified as C the direct access file is cleared before data is output to it. After the file is cleared it will be set up as described by parameter fields 10 to 15. Random and PIR sequential files are cleared entirely and serial files up to the last bucket used. Non PIR sequential files are cleared up to the last seek area used. Field 9 may be specified as A (but only for a non PIR sequential file), in which case data will be appended to information already contained in the direct access file, but the program will create its own bucket headers up to the last seek area used

If C or A is not specified, the program assumes that bucket headers are already set up, and if there is information in a non PIR file the new data will be appended to it. If there is information in a PIR file an error message will be output and the run should be abandoned
- 6 The fifth card or paper tape line is used to insert a dummy highest key as the last entry in each level of index. This parameter is optional and if omitted the terminating asterisk follows the last field on the fourth card or paper tape block
- 7 The field must be in parentheses. It is optional and may be omitted. Either DB or MBT may be specified first. MBT may be followed by *nnn* where *nnn* is the number of buckets transferred in a single transfer. If *nnn* is not specified the program will transfer as many buckets as are allowed by the buffer size. If specified for a non PIR program the field will be ignored.

EXAMPLE

The file organization program generated in the first #XPJZ example (see page 133) is to be run. #AAAA is required to reorganize the data held in the direct access file DAINDEXSEQIN, *fgn* 1 into the direct access file DAINDEXSQOUT, *fgn* 2 which is to be renamed DAINDEXSEQFL, *fgn* 3. The program should implement double-buffering and multibucket transfers for both files.

The parameters to achieve this are:

AAAA1,1,Y,0,15,4,

AAAA2,DA,(DB,MBT),DAINDEXSEQIN,1,

AAAA4,DA,(DB,MBT),DAINDEXSEQOUT,2,0,(DAINDEXSEQFL,3),Q,C,80,70,0,1,1,1,*

Parameters for a file copy utility program

The parameters consist of a type 1 and type 2 parameter and an optional number of type 4 and type 5 parameters.

TYPE 1 PAPER TAPE BLOCK OR CARD: GENERAL INFORMATION

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name and sequence number	xxxx1 where xxxx has been chosen by the user	5 (fixed)
2 Magnetic tape access definition	either 0 if to be input file or 1 if to be output file (see Note 1)	1 (fixed)
3 Type of run	CM	2 (fixed)

TYPE 2 PAPER TAPE BLOCK OR CARD: MTFILE

The composite magnetic tape to be used as the input or output file, according to field 2 of parameter type 1.

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name and sequence number	xxxx2 where xxxx has been chosen by the user	5 (fixed)
2 Filename	Up to 12 alphanumeric space (V) or minus (-) characters with the first character alphabetic; or the 4 characters **** (see Note 2)	12
3 Generation number	0 to 4095	4
4 Retention period	0 to 4095 (omitted for input tape file)	4
5 New filename	Up to 12 alphanumeric, space (V) or minus (-) characters with the first character alphabetic (omitted for input tape file; need not be specified for output tape file)	12
6 New generation number	0 to 4095 (omitted for input tape file; need not be specified for output tape file)	4

TYPE 4 PAPER TAPE BLOCK OR CARD: DAFILE

A type 4 parameter is presented for each direct access file to be copied to the magnetic tape, or from a subfile on the tape. A type 5 parameter may optionally follow a type 4 when the output file is a direct access sequential file.

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name and sequence number	xxxx4 where xxxx has been chosen by the user	5 (fixed)

File organization/file copy utility program
(#XPJZ generated)

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
2	Sequencing indicator	Y or N where Y = sequence of records input required N = no sequencing required	1 (fixed)
3	User information	Y or N where Y = yes N = no	1 (fixed)
4	Key length in characters	0 to 99 (see Note 6)	2
5	Number of characters in record (including word 0 of the record) which precede the key in the record	0 to 9999 (see Note 6)	4
6	Filename	Up to 12 alphanumeric, space (V) or minus (-) characters, with the first character alphabetic	12
7	Generation number	0 to 4095	4
The following fields (8 to 15) are omitted when defining a direct access input file.			
8	Retention period	0 to 4095	4
9	New generation number	0 to 4095	4
10	Mode of processing	E, Q, R or S where E = end-of-file serial Q = sequential R = random S = serial (see Note 7)	1 (fixed)
11	File clear required	C or A where C = clear the file by writing bucket headers A = append to a sequential file and create bucket headers Note: This need not be specified (see Note 3)	1 (fixed)
12	Seek area packing density	0 to 99 Note: 0 = 100% (see Note 4)	2

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
13	Bucket packing density	0 to 99 Note: 0 = 100% (see Note 4)	2
14	Auxiliary header size	0 to 1022 (see Note 4)	4
15	Number of buckets for the level 1 index	0 to 99 (see Note 4)	2
16	Number of buckets for each level 2 index	0 to 99 (see Note 4)	2
17	Number of buckets for each level 3 index	0 to 99 (see Note 4)	2

TYPE 5 PAPER TAPE BLOCK OR CARD: DUMMY KEY FOR DA SEQUENTIAL OUTPUT (OPTIONAL)

Dummy key for sequential direct access output file (see Note 5).

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name and sequence number	xxxx5 where xxxx has been chosen by the user	5 (fixed)
2	Dummy key	Key of up to 64 alphanumeric characters	64

Each parameter field must be terminated by a comma; the complete set of parameters must be terminated by an asterisk (*).

Notes:

- 1 If specified as zero, the program will open the composite magnetic tape file, defined in parameter type 2, as the input file. If specified as 1, the program will open the composite magnetic tape file as the output file
- 2 When defining an input file, this field should be the existing filename on the tape required to be opened. When defining an output file, this field, if specified as an existing filename on a tape, will cause the named tape to be opened. If, however, it is intended to rely on Executive's search for a tape with an expired retention period to be opened as the output file, this field should be specified as ****. Field 3 need not be specified in this case
- 3 If parameter field 11 is specified as C the direct access file is cleared before data is output to it. After the file is cleared it will be set up as described by parameter fields 12 to 17. A random file is cleared entirely, a serial file up to the last bucket used and a sequential file up to the last seek area used. Field 11 may be specified as A (but only for a sequential file), in which case data will be appended to information already contained in the direct access file, but the program will create its own bucket headers up to the last seek area used.
If C or A is not specified, the program assumes that bucket headers are already set up, and if there is information in the file the new data will be appended to it
- 4 For non-sequential files, fields 12 to 13 and 15 to 17 should not be specified or specified as zero.
If field 11 is not specified, fields 15 to 17 must be identified to the corresponding parameters to the file allocator, and fields 12 and 14 should not be specified or specified as zero as the program assumes that bucket headers are already set up and the parameters are not relevant
- 5 Parameter type 5 is used to insert a dummy highest key as the last entry in each level of index. This parameter is optional

File organization/file copy utility program
(#XPJZ generated)

- 6 The key specified here will be used to determine the size and position of the record key when the file is processed during the run. If this differs from the key description recorded in the system control area for a direct access output file, the generated program will report this by the message:

DISPLAY:- KEY ERROR ON FILE DAn

(see *Exception conditions*, page 155)

If this display is given for a non-serial direct access file, the generated program will halt with the message:

HALTED:- KX

and the run should be abandoned.

If the key information does not match for a serial file an indication will have been given, but the program will continue (assuming the key as specified by the parameters), as this is not likely to be a problem. For a non-serial file the program cannot be continued

- 7 If the mode of processing field specified here does not agree with the file organization code recorded in the system control area for a direct access output file, then the generated program will halt with the message:

HALTED:- FT

and the run should be abandoned (see *Exception conditions*, page 155)

EXAMPLE

- 1 The program generated in the second XPJZ example (see page 133) is to be run. #AAAA is required to copy the data from the direct access files DASERIAL, *fgn* 1, DAINDEXED, *fgn* 3, and DARANDOM, *fgn* 2 to a composite magnetic tape file. A tape with an expired retention period is to be opened and renamed COMPOSITE, *fgn* 1 with a new retention period of 300 days. No sequencing is required and no user information is supplied for any of the direct access files. The key parameters differ for each file as shown in the parameters below.

The parameters required to achieve this are:

AAAA1,1,CM,

AAAA2,***,,300,COMPOSITE,1,

AAAA4,N,N,0,0,DASERIAL,1,

AAAA4,N,N,10,4,DAINDEXED,3,

AAAA4,N,N,7,5,DARANDOM,2,*

- 2 #AAAA is required to copy the data from the subfiles DASERIAL, *fgn* 1 and DAINDEXED, *fgn* 3, written by #AAAA in example 1, back to the direct access files DASERIAL, *fgn* 1 and DAINDEXED, *fgn* 3. Retention periods are not required for the direct access files and the generation numbers are not to be changed.

The parameters required to achieve this are:

AAAA1,0,CM,

AAAA2,COMPOSITE,1,

AAAA4,N,N,0,0,DASERIAL,1,0,1,S,

AAAA4,N,N,15,4,DAINDEXED,3,0,3,Q,C,80,80,0,1,0,1,*

OPERATING

Executive priority

The program has the priority as specified in the parameters supplied to #XPJZ.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CRO or TR0	Parameters	The reader is assigned at the start of the run and released as soon as the parameters have been read and checked
LPO (File copy utility program writing to tape)	Listing of files copied to tape	A line printer is assigned if the output file is specified as CM and used throughout the run

For file organization programs the first input file uses unit 1, the second input file is used as unit 2 and the output file is used as unit 3.

For file copy utility programs the input file (or files) use unit 1, the output file (or files) use unit 3.

Peripherals required for the input and output files will be used when needed and released when no longer required. In general these peripherals will be required throughout the run, except for a file copy utility program where direct access files are opened and closed as required.

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the generated program #xxxx (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>)	HALTED:- LD
The message HALTED:- LD will only be output if the loading operation was successful	
2 Load any required direct access units	
3 Load any magnetic tapes which are required. Output tapes must have write permit rings. Input tapes must not have write permit rings	
4 Load the parameters in the appropriate reader	
5 To activate the program:	
(a) For parameters on paper tape, input:	GO #xxxx 20
(b) For parameters on cards, input:	GO #xxxx 21

For file copy utility programs continue at step 13

Running file organization programs

<i>Narrative</i>	<i>Console message</i>
6 Load any paper tape or card input files in the readers allocated by the program	
7 When a direct access file has been closed the program outputs the message:	DISPLAY:- CLOSED FILE <i>n</i>
where	
<i>n</i> is either 1 or 2 for input files or 3 for an output file	
8 When an end of paper tape reel has been read the program outputs:	HALTED:- LOAD NEXT PAPER TAPE REEL

File organization/file copy utility program
(#XPJZ generated)

	<i>Narrative</i>	<i>Console message</i>
	Load the next reel of paper tape and input:	GO #xxxx
9	After all the files have been closed, the program outputs: where <i>x y z</i> each consists of 7 decimal digits, and <i>x</i> is the number of records successfully input from file 1 (prior to any user editing) <i>y</i> is the number of records successfully input from file 2 (prior to any user editing) <i>z</i> is the number of records successfully output (after any user editing) Note: For sequential processing of two input files, the output count will be less than the total of the input counts, by at least the number of replacement records assumed	DISPLAY:- COUNTS <i>x y z</i>
10	At the end of the run the program halts with either: (a) If the run has been successful: or (b) If an error condition has occurred during the run:	HALTED:- HH HALTED:- ZZ
11	If the program halts (see <i>Exception conditions</i> , page 155) before reaching the end of the run, the files may be closed and the program reset by inputting:	GO #xxxx 22
12	For a further run return to operating instruction 2	

Running the copy utility programs

	<i>Narrative</i>	<i>Console message</i>
13	On successful copying of a direct access file from tape the program outputs: where <i>x y z</i> each consists of 7 decimal digits, and <i>x</i> is the number of records successfully input from a subfile on tape (before any user editing) <i>y</i> is zero <i>z</i> is the number of records successfully output to the direct access file (after any user editing)	DISPLAY:- COUNTS <i>x y z</i>
14	When processing of a direct access file is complete the program outputs: either (a) If no error condition occurred:	DISPLAY:- SERVOK <i>fn fgn</i>
	or	

Narrative

Console message

(b) If an error condition was detected:

DISPLAY:- SERVVER ∇ *fn fgn*

where

fn is the 12 character name of the
direct access file

fgn is the 4 digit generation number
of the direct access file

15 At the end of the run the program
deletes with the message:

DELETED:- HH

Exception conditions

The following is a list of exception conditions reported by #xxxx during execution and loading. Prior to entry into #xxxx it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception conditions* in FIND A BINARY PROGRAM in *Compiling Systems*. The user's attention is also drawn to the note under Figure 14, page 143).

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	DISPLAY:- A1	A group of index buckets found in a direct access file is not compatible with the index bucket layout described in the parameters	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>
2	HALTED:- A1 (file copy utility program only)	Incorrect parameters or formats of parameters. (The message will be output if the program is asked to use a peripheral device that was not specified in the #XPJZ parameters)	Correct and return to operating instruction 2
3	HALTED:- AB	A random file is full apart from any second level overflow area	Abandon the run
4	HALTED:- APPEND FACILITY NOT AVAILABLE (file organization program only)	Parameter type 4, field 8 has been specified as A or not specified for a PIR file which contains information	Abandon the run
5	DISPLAY:- BUCKET IGNORED	A parity error has occurred on reading a bucket from a direct access input file	The program omits this bucket and continues
6	DISPLAY:- BUFFER TOO SHORT	The buffers are not long enough to hold buckets from a direct access file or blocks from a magnetic tape input file	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>
7	HALTED:- C2	There is insufficient space in core for further file descriptions	Either (a) Re-run later or

<i>Message</i>	<i>Reason</i>	<i>Action</i>
		(b) Wait until any other program(s) using the device closes its file(s) then input: GO #xxxx
8 DISPLAY:- C3	The flaw area is full	Either (a) If the direct access output file is serial or random, run continues, although if it is a random file a bucket will have been lost or (b) If output is to a sequential file, the program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>
9 DISPLAY:- CO	Cylinder overflow has occurred on outputting to a direct access file	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>
10 DISPLAY:- CONTROL RECORDS <i>x</i> (file organization programs only)	A number of records exceeding the bucket packing density have been output to a PIR file <i>x</i> = number of records	This message is output at end of run after the message: DISPLAY:- COUNTS <i>x y z</i>
11 DISPLAY:- CP ERROR	An unsuccessful card output transfer. The message is preceded by a halt on an Executive message	The program continues but the record will not be complete
12 HALTED:- CP	A card punch is required	Make a card punch available and input: GO #xxxx
13 DISPLAY:- CR ERROR	An unsuccessful card input transfer. The message is preceded by a halt on an Executive message	The program continues, but if the missing card is not replaced the record will be rejected as being incomplete
14 HALTED:- CR	A card reader is required	Make a card reader available and input: GO #xxxx

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
15	HALTED:- CR ERROR	A parameter line has been misread	Abandon the run, return to operating instruction 4
16	HALTED:- D1	Retention period not exceeded	Abandon the run
17	DISPLAY:- EL	The sequential direct access output file already contains records when writing bucket headers is not specified	The program will add the new records following existing data
18	HALTED:- ERROR CODE A ON PARAMETER TYPE <i>n</i> (file organization program only)	Incorrect program name specified	Abandon the run
19	HALTED:- ERROR CODE B ON PARAMETER TYPE <i>n</i> (file organization program only)	Specified sequence number out of order	Abandon the run
20	HALTED:- ERROR CODE C ON PARAMETER TYPE <i>n</i> (file organization program only)	Field 1 is not terminated by a comma	Abandon the run
21	HALTED:- ERROR CODE D ON PARAMETER TYPE <i>n</i> (file organization program only)	An alphabetic field is too long	Abandon the run
22	HALTED:- ERROR CODE E ON PARAMETER TYPE <i>n</i> (file organization program only)	A numeric field is too long	Abandon the run
23	HALTED:- ERROR CODE F ON PARAMETER TYPE <i>n</i> (file organization program only)	A field is terminated incorrectly	Abandon the run
24	HALTED:- ERROR CODE G ON PARAMETER TYPE <i>n</i> (file organization program only)	An unknown device type has been specified	Abandon the run
25	HALTED:- ERROR CODE H ON PARAMETER TYPE <i>n</i> (file organization program only)	The program was not generated to process the specified device type	Abandon the run
26	HALTED:- ERROR CODE J ON PARAMETER TYPE <i>n</i> (file organization program only)	PIR cannot be specified for device types CD or PT	Abandon the run
27	HALTED:- ERROR CODE K ON PARAMETER TYPE <i>n</i> (file organization program only)	PIR field is specified incorrectly	Abandon the run

File organization/file copy utility program
(#XPJZ generated)

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
28	HALTED:- ERROR CODE L ON PARAMETER TYPE <i>n</i> (file organization program only)	Multibucket transfers have been specified for a magnetic tape file	Abandon the run
29	HALTED:- ERROR CODE M ON PARAMETER TYPE <i>n</i> (file organization program only)	PIR field is terminated incorrectly	Abandon the run
30	HALTED:- ERROR CODE N ON PARAMETER TYPE <i>n</i> (file organization program only)	Filename specified too long	Abandon the run
31	HALTED:- ERROR CODE P ON PARAMETER TYPE <i>n</i> (file organization program only)	A negative generation number has been specified for a magnetic tape file	Abandon the run
32	HALTED:- ERROR CODE Q ON PARAMETER TYPE <i>n</i> (file organization program only)	A negative generation number has been specified for an output file	Abandon the run
33	DISPLAY:- FE	A serial direct access output file has been extended	The program continues
34	HALTED:- FGN DISCREPANCY: <i>filename</i> (file copy utility program only)	A tape subfile with the specified name has been found but the generation number does not agree with the one specified	Either (a) To ignore the discrepancy input: GO #xxxx or (b) To skip to the next output file definition input: GO #xxxx 24 or (c) Abandon the run
35	HALTED:- FILE <i>nn</i> BUFFER TOO SHORT (file organization programs only)	Double buffering requested on a magnetic tape file, but the buffer is not long enough	Either (a) To continue the run with single buffering input: GO #xxxx or (b) Abandon the run
36	HALTED:- FILE <i>nn</i> DB NOT POSS (file organization programs only)	Double buffering requested on a direct access file, but the buffer is not long enough	Either (a) To continue the run with single buffering input: GO #xxxx or (b) Abandon the run

<i>Message</i>	<i>Reason</i>	<i>Action</i>
37 DISPLAY:- FILE <i>nn</i> SINGLE BUFFER (file organization programs only)	The program has been resumed after the error messages 35 or 36	The program continues
38 HALTED:- FILE <i>nn</i> MBT TOO LONG (file organization programs only)	Multibucket transfers requested on a direct access file but the buffer is not long enough	Either (a) To continue the run with less buckets transferred input: GO #xxxx or (b) Abandon the run
39 DISPLAY:- FILE <i>nn</i> MBT <i>nnn</i> (file organization programs only)	Either (a) The program has been resumed after the error message 38 or (b) The program was requested to transfer as many buckets as possible	The program continues
40 HALTED:- FT	The processing mode specified in the parameters does not agree with the file organization code held in the SCA	Abandon the run
41 HALTED:- G1	A direct access file is not in the system	Either (a) Load the storage unit holding the required file and input: GO #xxxx or (b) Abandon the run
42 HALTED:- G2	An integrity code check has failed	Either (a) Re-run later or (b) Wait until any other program(s) which is also using the file closes the file, then input: GO #xxxx
43 DISPLAY:- H1	Either (a) A direct access serial file is full and no more room can be found to extend the file on the device containing the control information for the file	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>

<i>Message</i>	<i>Reason</i>	<i>Action</i>
	or	
	(b) A sequential file is full apart from any first and second level overflow area	
44 HALTED:- HEADER ERROR	A card or paper tape file either lacks a header block or has a header block with an incorrect name	Either (a) If a correct file is available, load it and input: GO #xxxx or (b) Abandon the run
45 DISPLAY:- I1	A bucket number is out of range	Either (a) For output to a random file the run continues with records for the buckets out of range being omitted or (b) For output to a serial or sequential file, the program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY SERV ER <i>fn fgn</i>
46 DISPLAY:- IB	Insufficient buckets have been allocated to at least one of the levels of index of a direct access file	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i> The user should recalculate his index bucket requirements and re-allocate the file, and check that the correct levels have been specified in the parameters
47 DISPLAY:- IC	The number of blocks successfully read from a magnetic tape is not the same as the block count held in the trailer label	The program continues

<i>Message</i>	<i>Reason</i>	<i>Action</i>
48 DISPLAY:- K1	An unclearable parity error has been detected	Either (a) For output to a random file the run continues, with records to be loaded into unreadable buckets being omitted or (b) For output to a sequential file, the program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>
49 DISPLAY:- <i>key</i>	The record cannot be input or output because it is too long for the program buffers	The program continues but does not output the offending record
50 DISPLAY:- KEY ERROR ON DA <i>n</i>	The key values specified in the parameters do not match those held in the SCA for the direct access file <i>n</i>	The program continues but may be followed by the message: HALTED:- KX
51 HALTED:- KX	The key parameters specified in the parameters do not agree with the parameters held in SCA for a non-serial file. This is preceded by the console display KEY ERROR . . .	Abandon the run
52 DISPLAY:- LB	Long block	The run continues using the data already read
53 DISPLAY:- MT ERROR	Either (a) A non-standard sentinel has been found on an input tape or (b) There has been a failure to read a magnetic tape block	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>
54 HALTED:- NO RANDOMIZER ROUTINE	A user-coded randomizer has not been included in the program	Abandon the run

File organization/file copy utility program
(#XPJZ generated)

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
55	DISPLAY:- P0	File is zero length or bucket out of range	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>
56	DISPLAY:- P1	F.I.P. entry cannot be read	
57	DISPLAY: P2	Invalid indexes specified	
58	DISPLAY:- P3	Overflow specified on serial or random files	
59	DISPLAY:- P4	Indexes specified on serial file	
60	DISPLAY:- P5	Usable buffer space less than minimum record length	
61	DISPLAY:- P6	Seek area packing density greater than 100%	
62	DISPLAY:- P7	Parity error	
63	DISPLAY:- P8	Flaw area full	
64	DISPLAY:- P9	Insufficient space for auxiliary header	Abandon the run Abandon the run Abandon the run Abandon the run Abandon the run Abandon the run Abandon the run Abandon the run
65	DISPLAY:- P:	Seek area >256 buckets and only 2 word headers specified	
66	HALTED:- PA (file organization programs only)	Incorrect number of input files specified in parameters	
67	HALTED:- PB (file organization programs only)	Sequence checking field specified incorrectly in parameters	
68	HALTED:- PC (file organization programs only)	The input file to supply user information was not specified in parameter type 1, field 1	
69	HALTED:- PD (file organization programs only)	Sequence checking requested, but key length specified as zero, in parameter type 1	
70	HALTED:- PE (file organization programs only)	Specified key length in parameter type 1 too long for the key buffer reserved at generation time	
71	HALTED:- PF (file organization programs only)	The rename field in parameter type 4 is not terminated by a bracket	
72	HALTED:- PG (file organization programs only)	The rename field in parameter type 4 is not followed by a comma	

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
73	HALTED:- PH (file organization programs only)	An incorrect processing mode has been specified in parameter type 4	Abandon the run
74	HALTED:- PJ (file organization programs only)	Parameter type 4, field 8 has been specified incorrectly	Abandon the run
75	HALTED:- PK (file organization programs only)	Parameter type 4, field 8 has been specified as A for a non-sequential file	Abandon the run
76	HALTED:- PM (file organization programs only)	Parameter type 4, field 8 has been specified as C or A for use with end of file serial processing	Abandon the run
77	HALTED:- PN (file organization programs only)	Parameter type 4, field 8 is not terminated by a comma	Abandon the run
78	HALTED:- PP (file organization programs only)	A seek area packing density has been specified in parameter type 4 for a non-sequential file	Abandon the run
79	HALTED:- PQ (file organization programs only)	Specified auxiliary header size in parameter type 4 is not in the range 0 to 1022	Abandon the run
80	HALTED:- PR (file organization programs only)	Index levels have been specified in parameter type 4 for a non-sequential file	Abandon the run
81	HALTED:- PS (file organization programs only)	Invalid combination of index levels specified in parameter type 4	Abandon the run
82	HALTED:- PV (file organization programs only)	Indexes specified in parameter type 4 but key length was specified as zero	Abandon the run
83	HALTED:- PW (file organization programs only)	Parameter type 5, field 2 is not terminated by a comma	Abandon the run
84	HALTED:- PX (file organization programs only)	Parameter set is not terminated by an asterisk	Abandon the run
85	DISPLAY:- R1	GEORGE 3/4 filestore file size is not a multiple of 10K words	The program closes all files and halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>

File organization/file copy utility program
(#XPJZ generated)

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
86	DISPLAY:- RECORD INCOMPLETE DISPLAY:- <i>key</i>	The record has a card missing or out of sequence	The program continues but does not output the offending record
87	DISPLAY:- RECORD LENGTH ERROR DISPLAY:- <i>key</i>	The length of the record held on cards or paper tape is not that given in the first word of the record	The program continues but does not output the offending record
88	DISPLAY:- RECORD TOO LONG DISPLAY:- <i>key</i>	The record cannot be input or output because it is too long for the program buffers, or (for a file copy utility program or file organization program non PIR file) output because it exceeds the usable bucket space after allowing for bucket packing density if specified	The program continues but does not output the offending record
89	DISPLAY:- SEQUENCE ERROR DISPLAY:- <i>key</i>	The record is out of sequence	The message is followed by another: HALTED:- SQ See message 91 for action to be taken
90	DISPLAY:- SQ	When adding records to the end of a sequential direct access file, the key of the first new record is not greater than then highest key recorded in the indexes	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY:- SERV ER <i>fn fgn</i>
91	HALTED:- SQ	This is preceded by the console messages: DISPLAY:- SEQUENCE ERROR DISPLAY:- <i>Key</i>	Either (a) To output the offending record and to read the rest of the file without sequence checking input: GO #xxxx 23 or (b) To omit the offending record and continue sequence checking, input: GO #xxxx
92	HALTED:- SUBFILE <i>subfilename</i> NOT FOUND (file copy utility programs only)	The program has searched the tape file to the end of composite file trailer label and not found the specified subfile	Abandon the run
93	DISPLAY:- TAGGED REC MISSING DISPLAY:- <i>key</i>	The overflow bucket indicated by a tag does not contain the required record	The program continues, but does not output the record required

<i>Message</i>	<i>Reason</i>	<i>Action</i>
94 DISPLAY:- TE	An end of file card or paper tape block has been read but it does not have the correct file name and has been ignored	The program continues
95 DISPLAY:- TO	Tag overflow has occurred. This can only happen when outputting to a direct access random file	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY SERV ER <i>fn fgn</i>
96 DISPLAY:- TP ERROR	An unsuccessful paper tape output transfer. The message is preceded by a halt on an Executive message	The program continues but the record will not be complete
97 HALTED:- TP	A paper tape punch is required	Make a punch available and input: GO #xxxx
98 DISPLAY:- TR ERROR	An unsuccessful paper tape input transfer. The message is preceded by a halt on an Executive message	The run continues and presents the next record. Another attempt may be made to read this record by resetting the tape
99 HALTED:- TR	A paper tape reader is required	Make a tape reader available and input: GO #xxxx
100 HALTED:- TR ERROR	A parameter line has been misread	Abandon the run or return to operating instruction 4
101 HALTED:- UE	An unexpected error has occurred	Abandon the run. #xxxx should be reloaded for any further run
102 HALTED:- UR	An unexpected reply has been given by Executive	Abandon the run
103 DISPLAY:- ZK	Index tables are to be set up for a sequential file, or tags are needed for a random output file, but the key length is zero	The program closes all files and either halts with the message: HALTED:- ZZ or, if a file copy utility program, displays: DISPLAY SERV ER <i>fn fgn</i>
104 HALTED:- xx	An error has been detected when loading #xxxx by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

NAME

#XRAA

VERSION

Mark 3

TITLE

Direct access composite file structure report

HARDWARE REQUIREMENT

2752 words of store (minimum)

1 card reader or 1 paper tape reader

1 line printer

1 direct access unit

DESCRIPTION**General**

The program outputs to a line printer details of the organization of a composite file, which must be set up in accordance with ICL 1900 Series direct access composite file standards. The program classifies subfiles as being of three kinds:

- 1 Directory subfiles. These contain subfile descriptions which index subfiles at the next level. Each successive fragment of a directory subfile is headed by a directory description which defines the fragment itself.
- 2 Data subfiles. These contain user data only. Each fragment is indexed by a subfile description contained in a directory subfile.
- 3 Free space subfiles. These are identical in format to data subfiles, but contain only unused space, and have neither name nor generation number.

The level of any subfile is defined as being one lower (that is, numerically higher) than the directory subfile which contains the subfile description for its first fragment.

Thus the top level directory subfile, whose first fragment starts at the physical beginning of the file, is at level 0. All subfiles having their descriptions for the first fragments in this directory are therefore at level 1, and so on. The program requires that the directory subfile at level 0 should have the standard name **COMPOFILE** in the directory description of its first fragment. Only in this way can the program identify the file as being composite.

Subfiles are reported in the order in which the subfile descriptions for their first fragments are encountered. Whenever the program has reported a directory subfile, it will then report all subfiles which have the subfile descriptions for the first fragment in the directory *before* reporting the next subfile at the same level as the directory subfile.

Input

Input to the program consists of a direct access composite file on which a report is to be made; parameters to the program may be input on cards or paper tape. The file must conform to the ICL 1900 Series direct access composite file format. The parameters are described in the section *Control*, page 171.

#XRAA

Output

The report on the composite file is output on the line printer. The layout of the possible report formats is given below.

Three possible report formats are available to the user: CHART, SHORTLIST and LIST. Each report is given a two line heading as follows:

Line 1

DATE: *dd/mm/yy* *format* TIME: *hh/mm/ss*

where

dd/mm/yy is the date, month and year on which the report was given.

format may be either CHART, SHORTLIST or LIST, and describes the format of the report that follows.

hh/mm/ss is the time in hours, minutes and seconds at which the report was started. The time is given assuming a 24-hour clock, where 00/00/00 is midnight. The time is completely omitted if the program is run on a configuration that does not have equipment suitable for determining real time.

Line 2

SUBFILE STRUCTURE REPORT OF *filename* FGN:*nnnn* VERSION NO.: *mmmm*

where

filename is the 12-character name of the file.

nnnn is the file generation number.

mmmm is the version number of the file.

The final two lines of each report are:

TOTAL SPACE USED: *nnnnnnnn* BUCKETS

TOTAL UNUSED SPACE: *fffffff* BUCKETS

where

nnnnnnnn is a decimal integer of up to 8 digits giving the total number of buckets used for directory or data subfiles.

fffffff is a decimal integer giving the total number of buckets contained in free space subfiles.

The formats of the reports themselves are described in detail below.

CHART

The CHART form of report gives a diagrammatic representation of the structure of the composite file, and also basic information about each subfile.

The report consists of a set of rectangular boxes spread across and down the page. Each box represents all the fragments of a particular subfile, and it contains information about that subfile as follows:

<i>subfilename</i>	
SFGN	<i>sfgn</i>
LEVEL	<i>l</i>
FRAGS	<i>gggg</i>
BKTS	<i>bbbbbb</i>

where

subfilename is the name of the subfile represented by the box. Subfiles containing unused space are given a subfilename of 12 zero characters.

sfgn is the generation number of the subfile, given as up to four decimal digits. This line is omitted for free space subfiles and for the level 0 directory COMPOFILE. For all other directories and for data subfiles, if no generation number is given in the file, the program assumes it is zero.

l is the level of the subfile, where 1 is the highest level for a data subfile.

gggg is the number of fragments in the subfile given as a decimal integer in the range 0 to 9999.

bbbbbb is the total number of buckets in the subfile given as a decimal integer in the range 0 to 999999.

The boxes representing the subfiles are given in a series of columns across the page. All the subfiles in the extreme left hand column are at level 1, subfiles in the next column are at level 2, and so on for each successive level across the page.

The boxes are connected to each other by vertical or horizontal lines which represent the structural relationship of the corresponding subfiles. If a column of subfile boxes is connected by a vertical line, this means that subfiles are all at the same level and indexed by the same directory subfile at the next higher level. If two subfile boxes are connected by a horizontal line, this means that the subfile on the left is a directory subfile, which indexes the subfile on the right. Thus every horizontal row is terminated with a data subfile (or free space subfile) at the right hand end of the row.

The fact that successive subfile levels are printed across the page, and not down it, means that the CHART report may be used for composite files that have any number of subfiles at each level. It implies a restriction, however, on the number of levels that the composite file may have, since this is dependent on the number of print positions of the line printer. In order to allow the maximum number of levels to be handled by the line printer, therefore, the directory subfile at level 0 is represented by a box on the centre of the page, *above* the remainder of the CHART, and not in the extreme left hand column. In this way, an extra level may be reported without sacrificing clarity.

Line printer restriction on CHART facility

In view of the fact that successive levels of subfiles are printed across the page, the number of levels a composite file may have, if the CHART format is to be used, is limited by the number of print positions on the available line printer.

<i>No. of print positions on line printer</i>	<i>Maximum depth for CHART</i>
96	level 4
120	level 5
160	level 7

If a printer with sufficient print positions is not available, there is an option at run time to change the request for CHART to a request for a SHORTLIST report.

SHORTLIST

The SHORTLIST report gives basic information about each subfile in a format that also allows the structure of the composite file to be easily followed. This form of report may be given for a composite file that contains any number of subfiles to a depth of up to 9999 levels.

Information about each subfile is printed on separate lines as follows:

LEVEL *llll* *subfilename* (*sfgn*) *ddf* FRAGS :*gggg* BUCKETS :*bbbbbb*

where

llll is the level of the subfile, given in up to 4 decimal digits, where level 1 is the highest level for a data subfile.

subfilename is the name of the subfile. Subfiles containing unused space are given a subfilename of 12 zero characters.

sfgn is the generation number of the subfile. This item is omitted for free space subfiles and for the level 0 directory COMPOFILE. For all other directories and for all data subfiles, if no generation number is given in the file, it is assumed to be zero.

#XRAA

ddf indicates the contents of the subfile, and may be either DATA, DIRECTORY or FREE SPACE.

gggg is the number of fragments that make up the subfile given as a decimal integer in the range 0 to 9999.

bbbbbb is the total number of buckets in the subfile, given as a decimal integer in the range 0 to 999999.

The report lines are printed with single spacing except when a subfile is reported at a different level to the previous one, in which case the change of level is indicated by an extra line of spacing.

LIST

The LIST report gives all the information that is available within the file about each individual subfile. This form of report may be given for a composite file of any number of subfiles to a depth of up to 9999 levels.

Information about each subfile is printed on a separate page, and consists of a number of lines as follows:

Line 1

subfilename SFGN: *sfgn*

where

subfilename is the 12-character name of the subfile being reported. Subfiles containing only unused space are given a subfilename of 12 zero characters.

sfgn is the generation number of the subfile. This item is omitted for free space subfiles and for the level 0 directory COMPOFILE. For all other directories, and for data subfiles, if no subfile generation number is given in the file, it is assumed to be zero.

Line 2

LEVEL: *llll ddf* SUBFILE TYPE: *xxxx*

where

llll is the level of the subfile, given in up to 4 decimal digits, where level 1 is the highest level for a data subfile.

ddf indicates the contents of the subfile and may be either DATA, DIRECTORY or FREE SPACE.

xxxx is the standard ICL expression of up to 4 characters that indicates the type of data contained in the subfile, if it is a data subfile.

Line 3

DATE WRITTEN: *dd/mm/yy* MAX. NO. USABLE WORDS PER BUCKET : *pppp*

where

dd/mm/yy is the date, month and year on which the subfile was written.

pppp is the maximum number of usable words in each bucket of the subfile, if it is a data subfile.

Line 4

RECORDS: *length rrrr* RECORDS PER BUCKET

where

length is either VARIABLE LENGTH for variable length records or *mmmm* WORDS LONG for fixed length records where *mmmm* is a decimal integer.

rrrr is a decimal integer indicating the number of records per bucket.

Line 5

SEQ. FIELD: *mmmm* CHARS LONG STARTING *cccc* CHARS FROM *be*

where

mmmm is the number of characters in the sequencing field of data records.

cccc is the number of characters from the beginning or the end of the data record at which the sequencing field begins.

be is either BEGINNING or END.

Line 6

RECORD TERMINATOR: *xx*

where

xx is either the one- or two-character expression which is used to indicate when the end of each data record has been reached, or is NONE if no such terminator is used.

Line 7 et seq.

FRAGMENT NO.	START BUCKET	END BUCKET	NO. BUCKETS
<i>nnnn</i>	<i>sssss</i>	<i>eeeeee</i>	<i>bbbbbb</i>
<i>nnnn</i>	<i>sssss</i>	<i>eeeeee</i>	<i>bbbbbb</i>

and so on.

where

nnnn is the number of the fragment, starting at 1.

sssss is the logical bucket number of the start of the particular fragment.

eeeeee is the logical bucket number of the last bucket of the particular fragment.

bbbbbb is the number of buckets in that particular fragment.

One line of information is output for each fragment in the subfile. Where no information is recorded in the file for any item about a particular subfile, all reference to that item is omitted.

CONTROL

The program requires two control statements for each file to be reported; the first of these defines the composite file itself, and the second defines the format in which the report is to be output. Each control statement may be input either on a card, or as one line of paper tape, terminated by a new-line character.

First control statement

The format of the file definition control statement is:

REPORT (*csn:filename(fgn)*)

where

csn (optional) is the cartridge serial number of the disc holding the composite file. If omitted, the colon should also be omitted. If present it must be an octal integer of up to six characters.

filename (mandatory) is the filename of the composite file. This name may be up to 12 characters, and if it has been given as less than 12 characters, the program will assume that the remaining characters are spaces.

fgn (optional) is the file generation number of the composite file. It may be a positive integer in the range 0 to 4095, or if it is set to -1, the program will open the file of that name with the highest generation number. If omitted, the enclosing internal parentheses should also be omitted, and the program will operate as if it were given as -1.

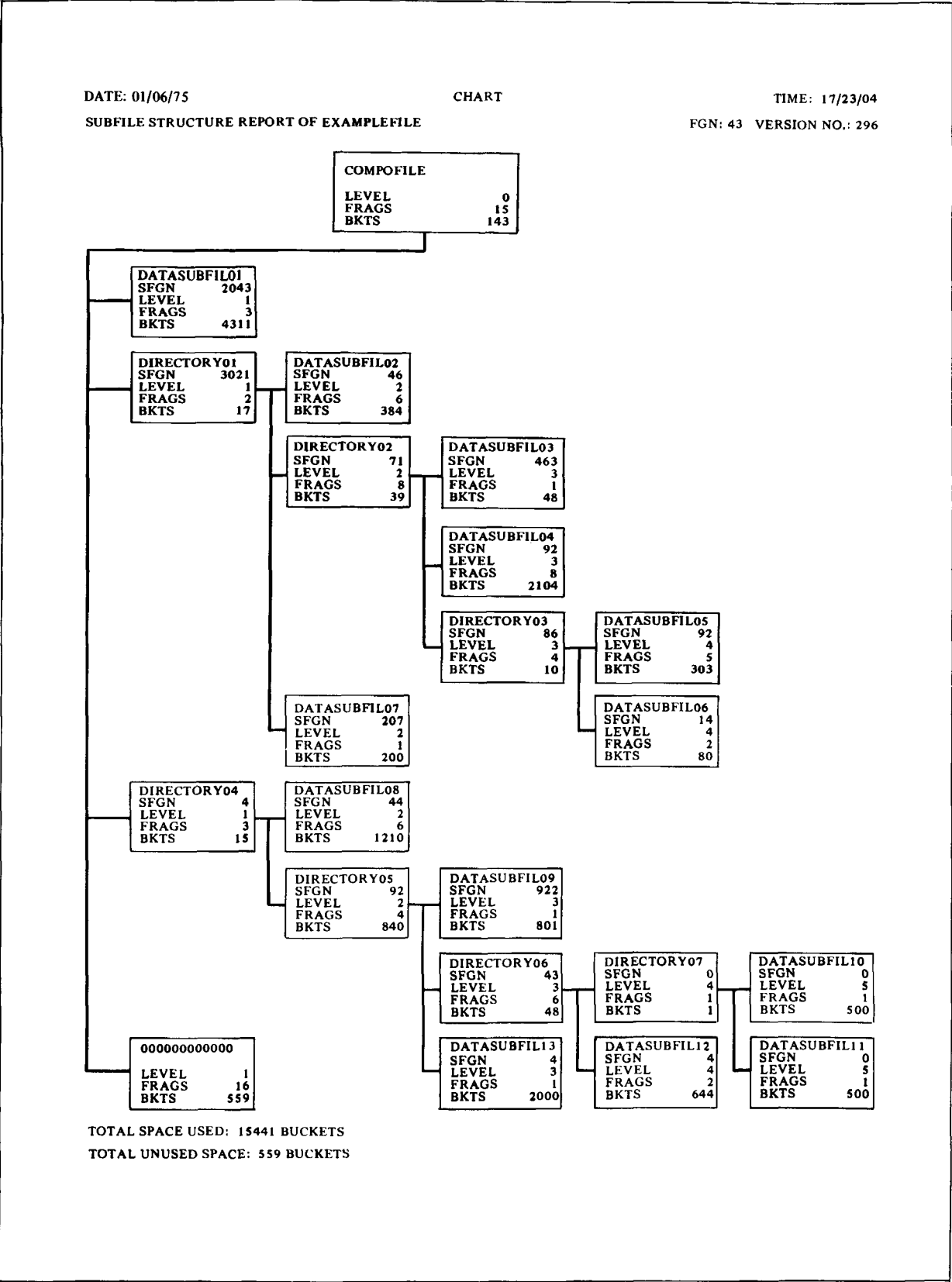


Figure 15 Composite file structure report

Second control statement

The format of the output definition control statement is:

FORMAT: SHORTLIST,LIST,CHART(*n*)

where

SHORTLIST,LIST,CHART describe the report facilities required. Any or all may be requested, and may be specified in any order. If more than one is requested, successive items must be separated by commas.

n (optional) indicates the number of print positions on the line printer available to be used for the CHART format, and must be either 96, 120 or 160. If present, it must be enclosed in brackets and immediately follow the CHART parameter. If CHART is omitted, *n* must also be omitted, and the program will use the first printer that is available. If CHART has been requested, the program ascertains from the file the minimum number of print positions necessary to output a CHART for that particular file.

Then:

- 1 If *n* has been specified and is too small, the program will halt with a request for a larger printer (see the section *Exception conditions*, page 174)
- 2 If *n* is large enough or has been omitted then the program attempts to allocate the required printer, using property codes

If a suitable printer is not available, then the user has the option at run time for changing the request for CHART to a request for SHORTLIST. (The program ensures that SHORTLIST is not performed twice.)

Example**A CHART REPORT**

The example Figure 15, page 172 is a CHART report of a composite file named EXAMPLEFILE with file generation number of 43 and version number 296. In all standard form composite files, the highest level subfile is the directory subfile at level 0, which has the standard subfilename of COMPOFILE. The box representing the subfile is printed at the centre of the head of the CHART. In this example, this subfile has 15 fragments which together take up 143 buckets.

The line from the bottom of COMPOFILE which links it to the subfiles DATASUBFILO1, DIRECTORY01, DIRECTORY04 and 000000000000 indicates that the subfiles of these names are directly indexed by COMPOFILE and are one level lower (that is, at level 1).

DATASUBFILO1 has no subfile boxes on its right, and this indicates that it is a data subfile. The subfile DIRECTORY01, however, does have a subfile on its right and is therefore a directory subfile. It is linked to three subfiles in the level 2 column: DATASUBFILO2, DIRECTORY02 and DATASUBFILO7 and this means that it indexes all three subfiles, and only those three subfiles. DATASUBFILO2 has no further subfiles on its right and is therefore data, while the subfile DIRECTORY02 is a directory subfile indexing DATASUBFILO3, DATASUBFILO4 and DIRECTORY03. And so on throughout the file.

The subfile at level 1 with the subfilename of 000000000000 is a free space subfile containing only empty space to be used if the composite file is expanded.

The final two lines are self explanatory.

OPERATING**Executive priority**

The program as supplied has a priority of 70.

#XRAA

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0	Control statements defining the composite file and the report required	The reader is assigned at the start of the run and released when the control statements have been read
LP0	Output structure report	The line printer is assigned when the file has been opened and released at the end of the run
DA0	Composite file on which the report is to be made	The disc unit is opened when the control statements have been read, and is closed at the end of the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XRAA (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Ensure that the composite file is on-line	
3 Load the control statements in the appropriate reader	
4 To activate the program:	
(a) If control statements are on paper tape, input:	GO #XRAA 20
(b) If control statements are on cards, input:	GO #XRAA 21
5 The program will print the structure of the file in each of the formats requested, release the peripherals and halt with the message:	HALTED:- HH
6 For a further run return to operating instruction 2	

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XRAA during execution and loading. Prior to entry into #XRAA it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

<i>Message</i>	<i>Reason</i>	<i>Action</i>
1 HALTED:- CR	A card reader is not available	Make a card reader available and input: GO #XRAA
2 HALTED:- EI	An integrity code check has failed	Either (a) Re-run later or (b) Wait until any other program(s) using the file closes the file, then input: GO #XRAA

<i>Message</i>	<i>Reason</i>	<i>Action</i>
3 HALTED:- E2	Executive cannot hold any further file descriptions	Either (a) Re-run later or (b) Wait until any other program(s) using the device closes its file(s) and then input: GO #XRAA
4 HALTED:- ED: <i>filename fgn csn</i> NOT IN SYSTEM	The file defined in the first control statement is not on-line	Ensure that the file is on-line and input: GO #XRAA
5 HALTED: F1	File is not a standard form composite file	Abandon the run
6 HALTED:- K1	An unclearable parity error has been detected	Abandon the run
7 HALTED:- LP	A line printer is not available	Make a line printer available and input: GO #XRAA
8 HALTED:- LP (<i>nnn</i>)	The program requires a printer with at least <i>nnn</i> print positions in order to output a CHART, where <i>nnn</i> is either 120 or 160	Either (a) If the specified printer is available, assign it to the program and input: GO #XRAA or (b) If the specified printer is not available, cause the program to omit the CHART by inputting: GO #XRAA or (c) If the specified printer is not available, cause the program to output a SHORTLIST instead of a CHART by inputting: GO #XRAA 25 The program ensures that SHORTLIST is not performed twice
9 HALTED:- NC	A CHART report has been requested in the control statements but the file is too deep	Either (a) To continue to next report, if any, input: GO #XRAA or (b) To replace CHART by a SHORTLIST input: GO #XRAA 25 The program ensures that SHORTLIST is not performed twice

#XRAA

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
10	HALTED:- P1	There is an error in the REPORT parameter in the first control statement	Correct and reload first control statement, and input: GO #XRAA
11	HALTED:- P2	There is an error in the file definition parameters in the first control statement	Correct and reload first control statement, and input: GO #XRAA
12	HALTED:- P3	The first control statement contains characters beyond the final parenthesis	Correct and reload first control statement, and input: GO #XRAA
13	HALTED:- P4	There is an error in the FORMAT parameter in second control statement	Correct and reload second control statement, and input: GO #XRAA
14	HALTED:- P5	There is an error in the mode parameters in the second control statement	Correct and reload second control statement, and input: GO #XRAA
15	HALTED:- P6	There is an error in the number of print positions specified in the second control statement	Correct and reload second control statement, and input: GO #XRAA
16	HALTED:- PL	Paper low condition has been detected	Load more paper and re-engage the printer
17	HALTED:- ST	Insufficient store available	Make more store available and input: GO #XRAA
18	HALTED:- TR	A tape reader is not available	Make a tape reader available and input: GO #XRAA
19	HALTED:- xx	An error has been detected when loading #XRAA by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

NAME

#XRAF

VERSION

Mark 6

TITLE

Off-line print from a direct access file

HARDWARE REQUIREMENT

2496 words of store (minimum)

1 card reader or 1 paper tape reader

1 line printer (optionally with Extended Repertoire Character Set)

1 direct access unit

DESCRIPTION**General**

The program prints information from a simple or composite file on direct access. If the file is composite, words 1 to 3 of the first record of the file must contain the name COMPOFILE and the file must conform to the ICL 1900 Series standard format for a composite file.

Optionally printing in Extended Repertoire Character Set (ERCS) may be requested.

Input

The input consists of a direct access print file produced by FORTRAN or Algol object programs. For a formatted data file produced by a FORTRAN program there is the option, by setting Bit 0 of Word 30, of taking the first data character of each record to be the FORTRAN paper feed control character which will be interpreted as follows:

<i>P.F.C. character</i>	<i>Space before printing</i>
space	One line
0	Two lines
1 to 7	Throw to Channel 1 to 7
+	No advance

Note: Any other character will cause an error.

If the size of a record exceeds the maximum length of print line specified or available, the excess characters will be printed on the following line. For ERCS printing, the excess characters will be printed in upper case until a further shift character is met. An error HALTED:- R2 is flagged the first time a record exceeds the print width and at the end of the run a count of the number of records in excess is displayed.

If printing in ERCS is required it should be noted that any shift characters will be included in the count of print positions. This count can take a value of up to 96, 120 or 160 characters depending on the line width specified (see *Operating Instructions*, page 180). Thus the presence of shift characters will result in a reduction of the maximum length of the line which can actually be printed.

Any other file having the ICL 1900 Series standard direct access file formats with data information starting at the first character position of the second word of the record may be printed by means of this utility. A bucket may

#XRAF

consist of more than one record.

Output

Output is to the line printer and consists of title lines and data lines. After printing the first title line, containing the file name, generation number and version number, the program will throw to the head of the next form.

In the case of simple files, the data will then be printed.

For a composite file, before a data subfile within the specified file (or the whole file if no subfile is specified) is printed, a heading line is printed for each subfile encountered before the first data subfile. Each heading contains a subfile name, the subfile generation number, if present, and the level of that subfile, level 1 being defined as the highest level at which a data subfile may occur.

Each subfile heading is printed only at the beginning of that subfile; therefore, where there are several subfiles within a particular subfile, the higher level subfile heading will be printed prior to printing only the first of the lower level subfiles. No indication of fragmentation is given. Subfiles are printed in the order in which the subfile descriptions for their first fragments are encountered. Subfiles whose subfile names begin with four zero characters are completely ignored.

The program will throw to head of form before printing the records contained in each data subfile.

If the use of FORTRAN paper feed control characters is specified, line spacing will be determined by these characters. However, if no control characters are specified, the records will be printed with single spacing.

ALIGNING PRINTER PAPER (OPTIONAL)

Before printing commences, a special printout is made to assist the operator in aligning the stationery. The printout consists of a throw to channel 1 followed by 7 lines of numeric data. Each print position in the line contains the tens digit of its position.

Print positions 1 to 9 contain 0

Print positions 10 to 19 contain 1

Prints positions 20 to 29 contain 2

etc.

Print positions 100 to 109 contain 0

Print positions 110 to 119 contain 1

Print positions 120 to 129 contain 2

etc.

The printout is preceded by the console message:

DISPLAY:- LINE UP PRINTER PAPER

and is immediately followed by the disengaging of the printer, which leads to the console message:

UNIT *n* FIX

If a further output of the line-up data is required the operator re-engages the printer. This procedure will not cause the DISPLAY event to re-occur and may be repeated as many times as required.

When the operator is satisfied with the alignment of the paper he will input:

GO #XRAF 24

This message will lead to a continuation of the main program. At the first printer order encountered, a further output of the console message:

UNIT *n* FIX

will occur. The operator will re-engage the printer and the program will continue without further interruption.

The paper alignment procedure will also occur after the paper low condition has been reported.

The paper line-up procedure may be omitted by the typing of the console message:

ON #XRAF 18

before initial entry to the program.

CONTROL

The program #XRAF requires control statements which may be punched on cards or paper tape. There are three statements, the first and third being mandatory and the second optional. Each statement should start on a new card or line and should be terminated by a space (for cards) or newline (for paper tape).

First control statement

This statement defines the file that contains the user's data. It has the following format:

INP(*csn:fn(fgn) . sfn(sfgn)*)

where

INP is the directive.

csn (optional) is the six-digit octal serial number of the disc containing the file, and the colon should only be present if the field is present.

fn is the name of the direct access file and consists of up to 12 alphanumeric, space (V) or minus (-) characters beginning with an alphabetic character. If less than 12 characters are specified, the remaining characters are assumed to be spaces.

fgn (optional) is the generation number of the file, which must be in the range 0 to 4095. If *fgn* is absent, a value of -1 will be assumed, causing the program to open the file on-line with the specified file name and the highest generation number. The parentheses enclosing this field should be omitted if the field is absent.

sfn (optional) is the name of the subfile to be printed if the file is a composite file. The subfile specified may be at any level, and may be a directory or data subfile. If this field is absent the preceding full stop should be omitted.

sfgn (optional) is the generation number of the subfile specified. If *sfgn* is absent, the program will not check for a subfile generation number but will take the first subfile it encounters with the correct subfile name. The parentheses enclosing this field should be omitted if the field is absent.

For a composite file, if a subfile name is not specified the contents of all the data subfiles will be printed.

Second control statement

The statement determines the positions in the file or subfile at which printing will commence. It may be used when switch 0 is on.

This statement is optional and, if it is omitted, printing will start at the beginning of the file or subfile. The format is as follows:

PRT*n*

where

PRT is the directive.

n is the decimal number in the range 0 to 999999, of the page throw at which printing is to start. Records will be read until the record containing the *n*th page throw is read; printing will commence with this record.

Third control statement

This statement signifies the end of the control statements. It is mandatory. The format is as follows:

OPERATING**Executive priority**

The program as supplied has a priority of 70.

#XRAF

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0	Control statements	The reader is assigned when the program is entered and released when the control statements have been read and checked
LPO	The listing of the user's file	The printer is used throughout the run
DA0	The user's file	The disc unit is used throughout the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XRAF (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required disc unit	
3 Load the parameters in the appropriate reader	
4 Ensure that the printer is set up with the stationery and format loop required	
5 If FORTRAN paper feed control characters are included in records, input:	ON #XRAF 0
6 Set a switch for the print line width required, if other than 120 character positions: (a) For 96 character positions, input: (b) For 160 character positions, input:	ON #XRAF 1 ON #XRAF 2
7 If it is required to by-pass the paper alignment procedure, input:	ON #XRAF 18
8 If printing in ERCS is required, input:	ON #XRAF 23
9 To activate the program: (a) For parameters on paper tape, input: (b) For parameters on cards, input:	GO #XRAF 20 GO #XRAF 21
10 If operating instruction 7 was carried out, this instruction and operating instruction 11 will be omitted. Otherwise the console message: will be followed by the output of alignment data to the line printer (see <i>Aligning printer paper</i> , page 178)	DISPLAY:- LINE UP PRINTER PAPER
11 When satisfied that the paper is lined up correctly to continue with the main program input:	GO #XRAF 24

<i>Narrative</i>	<i>Console message</i>
12 At the end of the run the program will halt with the message:	HALTED:- HH
13 For a further run return to operating instruction 2	

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XRAF during execution and loading. Prior to entry into #XRAF it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

<i>Message</i>	<i>Reason</i>	<i>Action</i>
1 HALTED:- C2	Executive cannot hold any further file descriptions	Either (a) Re-run later or (b) Wait until any other program(s) which is also using the device closes its file(s) then input: GO #XRAF
2 HALTED:- CR	There is no card reader available	Make a card reader available and input: GO #XRAF
3 HALTED:- ED <i>fn fgn csn</i> NOT IN SYSTEM	The file specified in the first control statement is not in the system. Note: <i>fgn</i> and <i>csn</i> will be output only if given in the control statement	Make the correct file available and input: GO #XRAF
4 HALTED:- EL	Switch 23 is set. There is no line printer with ERCS available	Make a line printer available and input: GO #XRAF
5 HALTED:- EX	An unidentified Executive reply has been received after an attempt to open a file	Abandon the run
6 HALTED:- F1	A subfile printout has been requested in the control statement, but the file is not a standard format composite file	Either (a) Abandon the run or (b) Print the whole of the simple file by inputting: GO #XRAF
7 HALTED:- G2	An integrity code check has failed	Either (a) Re-run later or (b) Wait until any other program(s) using the file closes it and then input: GO #XRAF

#XRAF

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
8	HALTED:- K2	An unclearable parity error has been detected	Abandon the run
9	HALTED:- LP	There is no line printer available	Make a line printer available and input: GO #XRAF
10	HALTED:- N2	Switch 23 is set. A printer with ERCS but not the specified print line width is assigned	Either (a) To accept the line width of the assigned printer as the maximum record length, if less than the requested width, input: GO #XRAF or (b) To assign the correct printer, TAKE the printer from #XRAF and GIVE #XRAF the required printer as unit 0, and input: GO #XRAF
11	HALTED:- P1	There is an error in the first control statement	Either (a) Abandon the run or (b) Correct the control statement and continue by inputting: GO #XRAF
12	HALTED:- P2	There is an error in the second or third control statement	Either (a) Abandon the run or (b) Correct the control statement and continue by inputting: GO #XRAF
13	HALTED:- PC	Property codes are implemented and a printer without the specified print line width is assigned	Either (a) To accept the line width of the assigned printer as the maximum record length, if less than the requested width, input: GO #XRAF or (b) To assign the correct printer, TAKE the printer from #XRAF and GIVE #XRAF the required printer as unit 0, and input: GO #XRAF

<i>Message</i>	<i>Reason</i>	<i>Action</i>
14 HALTED:- PL	Paper low condition has been detected on the line printer. The program disengages the printer	Load more paper and re-assign the printer, then input: GO #XRAF
15 HALTED: R1	An unacceptable FORTRAN paper feed control character has been detected	Either (a) To terminate the run input: GO #XRAF 29 or (b) To cause the FORTRAN character to be treated as a space paper feed control character, input: GO #XRAF
16 HALTED:- R2	Record length is greater than print width	To print excess characters on the following line(s) input: GO #XRAF Note: This message will be output once only during the run. If it occurs, the following message will be output at the end of the run: DISPLAY:- <i>n</i> where <i>n</i> is the number of records exceeding the print width.
17 HALTED:- ST	More store is required (up to a maximum of 896 words)	Either (a) Make more store available to the program and input: GO #XRAF or (b) Re-run later
18 HALTED:- SY	The message: GO #XRAF 24 has been input at an inappropriate time	Restart the run
19 HALTED:- TR	There is no paper tape reader available	Make a tape reader available and input: GO #XRAF
20 HALTED:- <i>sfn sfgn</i> NOT IN FILE	The specified subfile is not in the file	Abandon the run
21 HALTED:- <i>xx</i>	An error has been detected when loading #XRAF by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception conditions</i> in FIND A BINARY PROGRAM in Compiling Systems

NAME

#XREB

#XREJ

VERSION

Mark 3A (#XREB)

Mark 3B (#XREJ)

TITLE

Create direct access file from paper tape data (#XREB)

Create direct access file from punched card data (#XREJ)

HARDWARE REQUIREMENT

5568 words of store (#XREB)

5504 words of store (#XREJ)

1 paper tape reader or 1 card reader

1 paper tape punch or 1 card punch (optional)

1 line printer

1 or more direct access devices

DESCRIPTION**General**

The programs #XREB and #XREJ transcribe user's data held on eight-track paper tape or 80-column punch cards to a single fixed user-specified record format and output the resulting records to a previously allocated two-word bucket header direct access file. The programs are overlaid from direct access units.

Paper tape data must contain only the following characters:

alpha-shift characters

FE₂ (newline)FE₃ (line feed)FE₄ (form feed)FE₅ (carriage return)

§ (transcribed on output as #74)

] (transcribed on output as #75)

↑ (transcribed on output as #76)

← (transcribed on output as #77)

blank tape, which is ignored

erase, which is ignored

TC₄ (end of transmission), which should not be used except to mark the physical end to a reel of tape.

#XREB #XREJ

Card data must be punched in the ICL 1900 Series 64 character card code.

The programs will carry out conversion of the input data in the manner indicated by the parameters, producing a direct access file conforming to ICL Direct Access Housekeeping standards, and setting up index tables if required. In addition, the program may be used to edit the input data, compare fields one with another or with constants, sequence check, accumulate totals and print them according to specified criteria.

The facilities provided by the utilities are discussed under the following headings:

Direct access file

Input

Output

Control statements

Criteria checking

Error procedure

End of parameter input

DIRECT ACCESS FILE

The programs will load data to serial or sequential files. If a sequential file organization is indicated the user may or may not require index tables. Before the programs are used, the direct access file that is to contain the data must be allocated and bucket headers written using the file allocator program #XPJC. The file must be large enough to contain the data. The programs are only suitable for direct access files with two-word bucket header formats.

Allocating the file

When the file is allocated, the following items must be specified:

- 1 Two-word bucket headers to be written
- 2 Index buckets, if required. #XREB or #XREJ will set up standard index tables within the index buckets marked as such, but will not set aside buckets for this purpose
- 3 Overflow buckets, if required. Overflow buckets are set up by #XPJC on the basis of the seek area packing density required. #XREB or #XREJ will not write data to overflow buckets
- 4 Auxiliary header space, if required. #XREB and #XREJ will write only in the usable bucket space provided and will therefore leave auxiliary header space unchanged

Validating the file

A comprehensive check is performed to ensure that the requirements of the user in respect of #XREB and #XREJ are identical with the facilities of the pre-allocated file. Therefore, some parameters specified for #XPJC must be repeated for #XREB and #XREJ.

The total number of index buckets required by #XREB and #XREJ must be the same as the number found on reading the file. The user must specify the file organization for #XREB and #XREJ, either serial or sequential, which will be checked against the file organization code in the file index and parameters. Indexing may only be requested when a sequential file organization is required. If a serial file organization is required, the presence of index or overflow buckets is regarded as an error. If a sequential file organization without index tables is required, the presence of index buckets on the file is regarded as an error.

If a sequential file is required the user must specify to #XREB and #XREJ both the position relative to the start of the record, and the length of the key by which a record is identified. This information must agree with similar information recorded in the file index and parameters.

For any one run of #XREB and #XREJ output records are of fixed length and therefore the fixed length indicator in the file index and parameters must be set.

#XREB or #XREJ investigates the output record in relation to the direct access file, and checks that:

- 1 The length of the specified record can be accommodated within the usable space in the bucket
- 2 There is a field in the output record corresponding in position to the record key

The user may specify the following additional items at run-time.

Bucket packing density

This is expressed as a percentage of the usable space in the bucket. The bucket packing density is relevant only to a sequential file that is intended for use as an overlay file. By the use of this feature, space may be left in each data bucket to accommodate insertions, thus minimizing recourse to overflow techniques and extending the period during which a file may be used without reorganization.

Fill character

If a fill character is specified, any unused space at the right hand end of the bucket, excluding any auxiliary header space, is filled with this character. If a fill character is not specified, the original contents of the buckets remain unchanged.

Opening the file

The program searches for the file with the specified name and generation number. The file name and generation number are not changed, the specific retention period is inserted and the version number is set to zero. It is not possible to open the file with the highest generation number on-line by specifying - 1 as the file generation number.

Restart

If required, the input may be made in stages by re-opening at the end of the file. This opening mode is for use only as a restart to a previous run with #XREB or #XREJ and is not acceptable in other circumstances.

When this mode is used, writing of output records continues from the beginning of the previous end of file bucket. The preceding bucket may not, therefore, be filled to its required packing density.

A facility is provided to close a run early. If required, opening at the end of the file may be used to continue from the point of closure.

Input

The input data is read from paper tape by #XREB or punched cards by #XREJ and consists of user's data records which must be punched in ICL 1900 Series codes. A record may be complete within a card or line or may extend over any number of cards or lines subject only to the limit of size set by the INP control statement specifying the format of the input data. Each record consists of fields which may be fixed or variable in length but must always be in the same order in the input stream. All fields specified must be present and a record may not contain more than 128 fields.

If fields are variable in length, characters representing field and record terminators which must be distinguishable from the characters comprising the user's data must be specified. A variable length field that is non-existent or zero may be represented by the field terminator only. A maximum length variable field may or may not include the field terminator, at the user's option, provided the length is correctly specified by the INP directive. For example, if the maximum length of a field is known to be twelve characters, and the field terminator is to be always present, thirteen characters must be specified as the length of the field. Field and record terminators are not transcribed to the output record.

INPUT DATA ON PAPER TAPE (#XREB)

The input may be either in the form of a continuous stream of data or it may be divided into lines by the introduction, at any point in the stream, of newline or other stop characters. Recognizable stop characters are FE₂ (newline), FE₃ (line feed), FE₄ (form feed) and FE₅ (carriage return). TC₄ (end of transmission) should not be used except as the physical end of reel of paper tape. Input may extend over more than one reel of paper tape. Normally, #XREB reads the data in blocks of 80 characters and not to the stop character. In conjunction with data or validation errors, reading to the stop character may be requested by setting switch 5 on after loading the program (see *Error procedures*, Page 203).

#XREB #XREJ

User's data may contain alpha-shift characters only. The occurrence of delta-shift characters or beta-shift is reported as an error, except that:

- 1 The characters \$,], ↑ and ←, which enter store as delta-shift characters, will be transcribed for output as single characters with the values of #74, #75, #76 and #77 respectively. Although these characters appear in store as two characters, they need be counted only as one
- 2 Record and field terminators may be delta-shift characters

For paper tape input a record terminator is mandatory, whether the fields be fixed or variable in length. If required, a newline or other stop character may be specified as a terminator in which case this character must not be used to divide the input into lines. The use of a stop character as a field terminator is not recommended. Unless specified as a terminator, all stop characters are ignored by the program.

If required, a field terminator may be specified for fixed length fields, that is, as a field separator. The fields will effectively be treated as variable in length and care must be taken to ensure that the length of the field, as specified to the INP statement, includes the terminator.

If appropriate, that is, if the input is divided into lines, fields may extend over the end of a line. Records may commence anywhere in relation to the line.

The input must be preceded by at least twelve inches of run-out and must be terminated by four asterisks, which are punched immediately after the record terminator for the last record read and followed by newline, at least two TC₄ characters, six or more erase characters and at least twelve inches of run-out. If the input extends over more than one reel of paper tape, each reel must terminate with a TC₄ character following the record terminator for the last record read, six or more erase characters and at least twelve inches of run-out.

INPUT DATA ON PUNCHED CARDS (#XREJ)

User's data may contain any symbols or alphanumeric characters punched in ICL 1900 Series 64 character card code.

A record will normally begin at the first column of a card, in which case any unused portion of the previous card will be ignored. If specified, however, a record may begin at the card column immediately following the end of the previous record, in which case a record terminator is mandatory whether the fields be fixed or variable in length.

Fields may extend beyond the end of a card on to the next card. If areas of blanks occur in a record, for example at the right hand of a card to enable the next field to begin the next card, these areas must be described as fields. This does not apply to blanks separating the end of one record from the beginning of the next; these are ignored as stated above.

If required, a field terminator may be specified for fixed length fields, that is, as a field separator, in which case a record terminator is also mandatory. The fields will effectively be treated as variable in length and care must be taken to ensure that the length of the field, as specified by the INP statement, includes the terminator.

Blank cards, if any, occurring in the data stream are normally ignored. If the circumstances are such that a blank card can represent valid data, for example a fixed length empty character field, a facility is provided to prevent the program from ignoring such cards (see the section *Operating instructions*, page 205).

The input is preceded (unless it is valid data) by two or more blank cards and is terminated by an end card with asterisks punched in columns 1 to 4 followed by at least 2 blank cards. Where records do not begin on a new card, the four asterisks should be punched immediately following the record terminator for the last record read.

TYPES OF INPUT FIELD

There are three types of input field as defined by the INP statement.

- 1 Character fields
- 2 Decimal fields, either integral, mixed integral and fractional, or fractional
- 3 Octal fields

Decimal fields and octal fields are converted to binary before being distributed for output to the direct access file.

Character fields

A character field may not be more than 128 characters in length. Fields of greater length must be specified as two or more consecutive fields.

Fields may contain:

For #XREJ, any character of the 1900 Series 64 character card code.

For #XREB, alpha-shift characters and the delta-shift characters \$,], ↑ and ←.

Apart from the last four characters mentioned, character fields are transcribed without change.

Decimal fields

The value of the integral portion, if any, of a field must be within the range -2^{46} to $+2^{46} - 1$.

Leading spaces are treated as non-significant zeros.

A plus or minus sign is acceptable in any non-significant character position. Minus overpunched numerals, that is, quotes and J to R representing -0 to -9 respectively, are acceptable in the first significant character position. The fractional portion of a field, if any, must not exceed 6 decimal places in length.

It is not necessary to include a decimal point in a mixed integral and fractional field but if a decimal point is specified, its identity and position will be checked. Any non-numeric character (with #XREB this must be an alpha-shift character) is acceptable as a decimal point with the following exceptions:

Plus or minus sign

Minus overpunched numerals (quotes and J to R)

Comma

Characters used as a field or record terminator

Different characters representing the decimal point may be specified for different fields. The decimal point is not included in either integral or fractional counts.

Octal fields

An octal field may not exceed 8 digits in length; an octal digit may not be greater than 7.

Leading spaces are treated as non-significant #0 characters.

Output

The output data consists of records written in accordance with ICL Direct Access Housekeeping formats. The first word of the record is in the form of a standard record header containing in Bits 12 to 23 a binary count of the length of the record, including the first word, in words. This count is kept and inserted by the programs and should not be included in the input data.

The input fields, after any conversion (where applicable), are distributed for output in the formats and positions within the record specified by the parameters. The completed records are written to the direct access file, filling data buckets to the required packing density.

The output record for a particular run with #XREB or #XREJ are fixed in the length for the duration of the run, according to the output format specified by the parameters.

Output fields are described in order of appearance in the output record and are related to input fields by field number. In this way, input fields may be placed after conversion anywhere in the output record. If required, the same input field(s) may be placed in more than one position in the record.

Constants may also be inserted in the record.

TYPES OF OUTPUT FIELD

The four types of output field that can be defined are:

#XREB #XREJ

- 1 Character fields
- 2 Numeric fields which are either integral, mixed integral and fractional, or fractional fields. These are derived from decimal input fields
- 3 Binary fields derived from octal input fields
- 4 Constants for insertion in the output record; these may be characters numeric or binary

Character fields

Character fields are transcribed for output without change except that with #XREB the delta-shift characters \$,], ↑, and ← are converted to single characters with the values respectively of #74, #75, #76 and #77.

A character field may not be more than 128 characters in length.

It is not necessary for a character field to begin at an integral word boundary. If it succeeds another character field, it may begin at the character address immediately following the end of the previous field.

A variable length input character field, which is of less than the maximum length specified and therefore does not fill the output field area, will be left-justified on output and unoccupied space at the right-hand end of the field will be space-filled. Similarly, if such a variable length input field is non-existent, that is, represented on input by the field terminator only, the entire output field will be space-filled.

If the length specified for an output field is either greater or smaller than the length specified for the relevant input field, the field on output will be respectively space-filled or truncated at the right-hand end.

Numeric fields

Numeric fields are binary quantities derived from integral, mixed integral and fractional or fractional input fields.

One, two or three words may be specified for a numeric field..

The integral portion of a field must not exceed double-length capacity.

The fractional portion of a field must be contained within a single word.

A mixed integral and fractional quantity will normally occupy two or three words. The user may specify less than 23 bits for the fractional portion of a field. If this is required, the fraction will be incorporated in the less significant or only word of the integer, with an implicit binary point that number of places to the left of the least significant end of the word. The value of the integer will be scaled accordingly. The resultant quantity must not exceed double-length capacity.

In all cases, if more words, up to the maximum of three, are specified than are permitted for the particular quantity, most significant word(s) will merely propagate the sign.

If, during the conversion of input data, the number of words specified for the output field is found to be insufficient to contain the input quantity without loss of significance, an error condition arises.

Binary fields

Binary fields derive from octal input fields.

One or two words may be specified for a binary field. If two words are specified for octal fields, which may not be more than eight characters long on input, the more significant word is always zero.

If, during conversion of input data, the number of words specified for the output field is found to be insufficient to contain the input quantity without loss of significance, an error condition arises.

Constants for insertion

Constants for insertion in the output record may be characters, numeric or binary and are subject to the same definitions and restrictions except where indicated. Constants are punched in the statement defining the output record.

For character constants, the number of characters is limited by the requirement that a parameter statement must be complete within one line of paper tape (#XREB) or one card (#XREJ).

For numeric constants, a conventional decimal point (that is, a period) must be punched if a fraction is required. A purely fractional field may commence with a decimal point or with an integral zero.

LINE PRINTER OUTPUT

The control statements input are listed following the heading:

XREB	}	<i>dd/mm/yy</i>
or		
XREJ		

Error messages in respect of control statements, data errors and validity check failures are output to the line printer. Optionally accumulated totals and a list of ignored records are output to the line printer.

CONTROL

The operation of the programs is controlled by a set of *control statements*, which may be punched on paper tape (#XREB) or 80-column cards (#XREJ).

Each control statement consists of a three character *directive* which is followed by a string of *parameters*. Each parameter is separated from the next by a comma. The last parameter of a statement is not followed by a comma, but by a space (#XREJ) or newline (#XREB).

Where a parameter is omitted the associated comma must be punched, unless there are no parameters subsequent to the omitted parameter. In this case, the last parameter specified is regarded as the end of the control statement, and must be followed immediately by the appropriate space or newline. This means that punching ceases in column 79 for the last card of a given control statement.

A control statement may be continued on one or more cards, or one or more lines of paper tape. Each parameter must be entirely contained within one card or line.

Excepting control statement EOP (End of Parameters), control statements may be input in any order. No control statement may be input more than once.

Paper tape control statements (#XREB)

A line punched on paper tape must not exceed 128 internal characters, including the two internal characters for newline.

Where a control statement extends over more than one line, each line except the last is terminated by a comma followed by newline. Each line must start with the appropriate directive.

Punched card control statements (#XREJ)

Each card of a control statement must have the appropriate directive punched in the first three columns. The first parameter is the sequence number of the card within the statement, and must always be punched even if the statement is contained entirely on the one card. All cards for a control statement must be input together and in correct sequence.

Note: In the format specification of each control statement the sequence number is represented by *seq*. For #XREB *seq* and the comma should be ignored.

Directives

There are eleven directives, five of which are mandatory for all runs, one is mandatory for the circumstances to which it applies (including all runs with #XREB) and the remainder are optional.

A brief description of the directives and their status follows. Each directive is described in detail in the following pages.

In the format specification of each control statement, parameters specified by italic characters must be replaced by the appropriate characters when the statement is used. Where a parameter is specified by roman characters,

#XREB #XREJ

those characters must be punched.

<i>Directive</i>	<i>Purpose</i>	<i>Status</i>	<i>Description</i>
DFL	Defines the direct access FiLe to be opened for output	Mandatory	Below
AFL	Provides Additional FiLe information concerning the direct access file	Mandatory	<i>Page</i> 193
INP	Provides a picture image of the INPut data	Mandatory	<i>Page</i> 194
TER	Specifies field and record TERminate symbols	Mandatory for #XREB and for variable length input data on cards (#XREJ); otherwise optional	<i>Page</i> 195
OUT	Describes the format of the OUTput data	Mandatory	<i>Page</i> 196
VAL	Specifies VALidity checks to be performed	Optional	<i>Page</i> 200
ACC	Specifies the fields to be ACCumulated for output totals	Optional but both must be present or both absent	<i>Page</i> 200
PRI	Specifies the conditions for PRInting the totals accumulated in accordance with the ACC directive		<i>Page</i> 201
IGN	Specifies the conditions for IGNoring reocrds	Optional	<i>Page</i> 201
ERR	Specifies an additional ERRor procedure	Optional	<i>Page</i> 202
EOP	Announces the End Of the Parameter input	Mandatory	<i>Page</i> 203

A description of each of the control statements is given below:

Control statements required to define the direct access file

DFL

This statement describes the direct access file (two-word bucket header format) which is to receive the user's data. The statement is mandatory.

The format is as follows:

- 1 For a file that is being written to for the first time

DFL,*seq*, OUT,*fn*,*fgn*,*frp*

where

fn is the 12 character file name (12 characters must be present)

fgn is the file generation number in the range 0 to 4095

frp is the retention period to be given to the file, in the range 0 to 4095

- 2 For a file that is to be opened at end of file for a restart

DFL,*seq*, END,*fn*,*fgn*

AFL

This statement provides essential information about the file additional to the information required for opening the file.

The statement is mandatory.

The format is as follows:

AFL,*seq*,*a*,*b*,*c*,*d*,*e*,*f*,*g*

where

a is the character with which unused space in the bucket is to be filled. Any alpha-shift character (#XREB) or any character from the ICL 1900 Series 64 character card code (#XREJ) may be used, with the exception of comma. If this parameter is omitted and a comma only punched, the original contents of the unused space will be left untouched.

b is the file organization code where

S = serial

Q = sequential

This code must agree with the code set up at allocation.

If S is indicated, no further parameters are required in this statement.

If Q is indicated:

c is the number of buckets per file for the first level of index.

d is the number of buckets per file area required for the second level of index.

e is the number of buckets per seek area required for the third level of index.

If any one of the above three parameters is zero it may be omitted but a comma must be punched. The total $c + d + e$ must not exceed 79, and may be zero if index tables are not required. The total number of index buckets requested must agree with the number set aside by the file allocator program #XPJC.

f is the data bucket packing density percentage. 100 may be specified to signify 100% packing but, if a zero is punched or if the statement is omitted and a comma only punched, 100% packing will be assumed.

g is of the form x/y where *x* represents the number of characters constituting the record key (in the range 1 to 64), and *y* represents the number of characters from the left-hand end of the record preceding the first character of the record key. This is checked against information recorded in the File Index and Parameters.

Examples

- 1 Statements on cards (#XREJ)

DFL,1,OUT,WEEKVPAYFILE,1,4095

AFL,1,,Q,1,2,2,50,8/4

The cards are the first and only card of each statement and therefore in each case the sequence number is 1.

DFL: Open for output the file named WEEKVPAYFILE with a file generation number of 1 and give it a maximum retention period of 4095 days.

AFL: Following the sequence number, no fill character is specified and therefore unused space at the right hand end of the bucket is left unchanged.

Sequential file organization is indicated. Further parameters are therefore required. The number of buckets required for the first, second and third levels of index is 1, 2 and 2 respectively. The buckets are to be no

#XREB #XREJ

more than 50% filled with records. (In practice this file will probably be operated, after loading, as an overlay file with an expected high rate of insertion or expansion.)

The key by which records will be indexed is 8 characters long and commences immediately following the record header.

- 2 These two statements on paper tape (#XREB) would appear as follows:

DFL,OUT,WEEK7PAYFILE,1,4095↓AFL,,Q,1,2,2,50,8/4↓

Control statements required to define the input data

INP

This statement describes the format of the INPut data by field pictures. All fields must be described in order of appearance in the input stream, up to and including the last field of the user's data record. Each field is described by an individual parameter as being of one of the following types:

- 1 Character
- 2 Decimal (integral and/or fractional)
- 3 Octal

A maximum of 128 fields may be described in any record.

The statement is mandatory.

The format is as follows:

INP,*seq,fp,fp,.....,fp*

where

each *fp* is a field picture parameter and may take any of the following forms.

Character fields

The form of a parameter for a character field is

nH

where

n is the maximum or actual number of characters in the field, and must be in the range 1 to 128.

H is the standard PLAN symbol for character fields.

Decimal fields

The formats of parameters for decimal fields are:

nH

nNn₁F

n₁F

nNs₁F

where

N and F indicate respectively the integral and fractional components of the field.

n is the maximum or actual number of characters in the integral portion of the field.

n₁ is the number of decimal places in the fractional portion of the field and must be of fixed length in the range 1 to 6

s is a non-numeric character representing the decimal point.

If it is desired to represent a purely fractional field commencing with a decimal point symbol, the notation $ONsn_1 F$ must be used.

Octal fields

The form of a parameter for octal fields is:

$n\#$

where

n is the maximum or actual number of digits in the field and must be in the range 1 to 8.

$\#$ is the standard PLAN symbol signifying an octal field.

TER

This statement specifies the field and record TERminate symbols and, for #XREJ only, the action to be taken at the end of a record.

The statement is mandatory for #XREB and for variable length input fields.

The format is as follows:

TER,*seq*,F=*a*,R=*b*,C

where

a is the field terminate symbol.

b is the record terminate symbol.

C indicates, for #XREJ only, that, when the end of a record is encountered, the beginning of the next record will be found in the card column immediately following the terminating symbol of the previous record. If this parameter is omitted, it is assumed that the next record begins on the next card and the remainder of the current card is ignored.

If a field terminator is present, a record terminator also is mandatory. The field terminator must be a different symbol from the record terminator. If C is present, a record terminator is mandatory but a field terminator is optional and may be omitted, provided the comma is punched.

For #XREB a record terminator is mandatory, although a field terminator need only be specified for variable length fields.

Examples

1 Statements on cards (#XREJ)

INP,1,14H,4H,6N,3N4F,ON*4F,4#

The TER statement is not used and therefore the fields are of fixed length, all records begin at card column 1 and no record terminating symbol is in use.

The input stream will contain, in sequence:

- (a) Two consecutive character fields 14 and 4 characters long respectively
- (b) A 6 digit integer
- (c) A mixed field consisting of a 3 digit integer and a 4 digit decimal fraction. This field contains no decimal point
- (d) A 4 digit decimal fraction preceded by a decimal point represented by an asterisk
- (e) A field containing 4 octal digits

#XREB #XREJ

Such an input record might be as follows:

SMITHVJVDVVVVVZ11VVV4257VV10275*50007721VV.....

- 2 The same field descriptions on paper tape (#XREB) would appear as follows:

INP,14H,4H,6N,3N4F,0N*4F,4#↓TER,,R=↓↓

As the input data is on paper tape, a record terminator is mandatory and therefore the TER statement is used. The fields are fixed in length and since a field terminator is not required, this parameter is omitted and comma only punched. Newline is selected as the record terminator. The format of the input record might be as follows:

SMITHVJVDVVVVVZ11V0042570010275*50007721↓

- 3 The same field description on paper tape (#XREB) with variable length fields would appear as follows:

INP,14H,4H,6N,3N4F,0N*4F,4#↓TER,F=!,R=↓↓

An exclamation mark is selected as a field terminator.

The format of the input record would be as follows:

SMITHVJVD!Z11!4257!10275!*50007721↓

The fifth field (*5000) is of the maximum length specified and is therefore not followed by the field terminator. Similarly the last field is not followed by the field terminator but only by the record terminator.

Control statement required to define the output record

OUT

This statement describes the format of the record to be OUTput to the direct access file. For any one run with #ZREB or #XREJ, the fields, and hence the records, are of fixed length as specified, and all fields must be described in order of appearance in the output record. Each field is described by an individual parameter as being of the following types:

- 1 Character
- 2 Numeric
- 3 Binary
- 4 Constants for insertion

A maximum of 128 output fields relative to the input fields may be specified. Output fields are related to their appropriate input fields by input field number (where field 0 represents the first field in the input stream).

An input field may be referred to by more than one OUT parameter. A maximum of 32 additional output fields may be created in this way. All such additional fields must be given identical descriptions.

Additional output fields may also be created by the insertion of constants in the output record, in which case if

L is the total length, in words, of all the constants, and

N is the number of constants required, then

L+N must not exceed 127 words.

Fields specified by INP but omitted under OUT will be ignored. The statement is mandatory.

The format is as follows:

OUT,seq,od,od,od.....

where

each od is a parameter representing an output description, and may take any of the following forms.

Character fields

The format of a parameter for a character field is:

nH/ifn

where

n is the number of characters to be output, in the range 1 to 128.

ifn is the relevant input field number.

Numeric fields

The formats of parameters for numeric fields are:

nN/ifn

or

$nNn_1 F/ifn$

where

N and F indicate respectively the integral and fractional components of the field.

n is the number of output words required, which may be 1, 2 or 3.

n_1 is the number of bits within which a fraction is to be contained, in the range 1 to 23.

ifn is the input field number.

Binary fields

The format of a binary parameter is:

nB/ifn

where

n is the number of output words required which may be 1 or 2.

ifn is the input field number.

Binary fields derive from octal input fields.

Constants for insertion

A parameter describing a constant for insertion in the output record must be punched as a literal, that is, the parameter must be enclosed in apostrophes (internal character #27).

The formats of parameters describing constants for insertion are:

' nH/c '

' nN/c ' or ' $nNn_1 F/c$ '

' nB/c '

where

the definition up to and including the solidus is the same as that used to describe output fields relative to input fields, that is, for character, numeric and binary types.

c represents the constant to be inserted.

The following restrictions apply to the make up of the constants themselves.

- 1 CHARACTERS The number of characters is limited by the requirement that a parameter must be complete within one line of paper tape (#XREB) or one card (#XREJ)

#XREB #XREJ

- 2 **NUMERIC** A conventional decimal point must be punched if a fraction is required. A purely fractional constant may commence with a decimal point or with an integral zero. Other limitations are stated in *Numeric fields*
- 3 **BINARY** Octal constants must be punched in the following format:
#c
where
c consists of up to 8 octal digits.

Other limitations are given in *Types of output field*, page 189.

Examples

- 1 Statements on cards (#XREJ)

OUT,1'3H/778',4H/1,14H/0,2N23F/3,IN23F/4,IN/2,IB/5

Following the record count word the output record will contain, in sequence, the following information:

- (a) The constant 778 in character format. This will constitute the first field in every record
- (b) A 4 character field derived from input field 1. Field 1 is the second input field
- (c) A 14 character field relating to input field 0. The first three fields are character fields, and therefore each field begins at the character position immediately following the end of the previous field

The total length of these fields is 21 characters which does not fill a whole number of words. Since the next item is a numeric field which must begin on an integral word boundary, the last character field is filled with three spaces at the right-hand end
- (d) A double-word numeric field assembled as a mid-point number, relating to input field 3
- (e) A single word fraction relating to input field 4
- (f) A single word integral numeric field relating to input field 2
- (g) A single word binary quantity relating to input field 5

If this output description is related to the input example shown under the INP statement, it will be seen that output field (g) derives from octal input

- 2 The same field description on paper tape (#XREB) would appear as follows:

OUT,'3H/778',4H/1,14H/0,2N23F/3,1N23F/4,1N/2,1B/5↓

Criteria checking

Criteria checking is an optional means of controlling the action of a program whilst the program is running. Depending on whether or not certain conditions are satisfied the program will, or will not, undertake a specified course of action.

Provision is made for four types of condition:

- Type 1 A certain number of buckets have been written since the condition last arose.
- Type 2 A certain number of records have been written since the condition last arose.
- Type 3 The end of the input data has been reached.
- Type 4 The comparison of a field with another field or with a constant gives a certain result.

Types 1 and 2 may be specified in parameters with the format:

B = n,

R = n,

where B and R indicate respectively the number of buckets written and the number of records written.

n is the number of buckets or records written.

Type 3 is specified by the parameter:

EOR.

Type 4 has a greater range of possibilities. Comparisons may be:

- 1 Of fields within a record
- 2 Of the same field in consecutive records
- 3 Of a field with a constant which may be in any of the forms specified by the OUT directive

The conditions which may be examined are:

1	Equality, represented by the symbol	=
2	Inequality, represented by the symbol	;
3	Greater than, represented by the symbol	>
4	Less than, represented by the symbol	<

A comparison is specified by a parameter format:

$F_n s F_{n_1}$ or $F_n s 'x'$

where

n and n_1 are input field numbers.

s is the symbol representing the condition to be tested (= ; > or <)

' x ' represents a constant defined in exactly the same way as constants defined by the OUT directive.

If n and n_1 are the same number, comparison is required of the same field in consecutive records. This implies storage of this field from the previous record. There is no specified limit to the number of such fields that may be stored but their total length may not exceed 64 words. This type of comparison may conveniently be used to check key sequence of records.

If a record is rejected for any reason, no fields from it are stored and therefore comparison is always with the last valid record written to the direct access file.

If n and n_1 are not the same number, comparison is required of different fields within the current record.

A statement may contain more than one criterion parameters. Each Type 4 statement may be either:

- 1 A simple parameter consisting of one comparison only in which case the action to be taken by the program depends on the success or failure of that one comparison, or
- 2 A compound parameter consisting of up to three comparisons which may be interrelated to form a compound condition, in which case the action to be taken by the program may depend on different combinations of the success or failure of the individual comparisons

Two possible relationships between comparisons are provided:

Logical AND represented by the symbol +

Logical exclusive OR represented by the symbol /

This may be expressed as follows:

Assuming that symbols a , b , and c represent three possible comparisons within one parameter, the permitted interrelations of these are:

a	action taken depends on fulfilling one comparison only.
$a + b$	action taken depends on fulfilling two comparisons.
$a + b + c$	action taken depends on fulfilling all three comparisons.

#XREB #XREJ

a/b action taken depends on fulfilling either a or b but not both.
 $a/b/c$ action taken depends on fulfilling only one of a , b , or c .
 $a/b + c$ action taken depends on fulfilling only one of a and b , AND c .
 $a/(b+c)$ action taken depends on fulfilling either a OR $(b+c)$ but not both.

No other compound relations are permissible.

Although fields are specified by input field numbers, comparison is made of fields in the format in which they appear in the output record. Therefore comparison cannot be made of fields which do not appear in the output record, although specified by the INP statement.

Binary fields will be compared arithmetically, character fields logically. Length and type of fields to be compared must be identical and this fact is checked during the input of control statements. However, if comparison is with a constant it is the user's responsibility to ensure the constants are set up in a form relevant to the output field with which it is to be compared, and of equal size.

Sequence checking

Comparisons of the same field in consecutive records may conveniently be used to check the sequence of record keys within the file. No other sequence checking facility is provided and it is strongly recommended that this facility should be utilized if a sequential file is being loaded.

If a record is rejected for any reason, no fields from it are stored and therefore its key will not participate in this sequence checking. Comparison with a previous record is always with the last valid record written to the direct access file.

Control statements required to define both criteria checks and actions dependent on them

VAL

This statement specifies VALidity checks, defined in terms of criterion parameters to be performed on the user's data.

The format is as follows:

VAL,seq,R,c,c,c,.....

where

R in the first parameter position, that is, immediately following the card sequence number if required for #XREJ, indicates that a record which fails any of the subsequent validation checks will be rejected. If this parameter is omitted, records that fail will nonetheless be written to the direct access file.

c is a criterion parameter. Up to 10 criterion parameters of type 4 may be input.

For details or error indications see *Error procedure*, page 203.

ACC

This statement specifies, by input field number, fields to be ACCumulated, and the print heading required for each column of accumulations.

This statement is optional, but if present, the PRI statement must also be used.

The format is as follows: .

ACC,seq,Fn/n₁Hheading,.....

where

n is the number of the input field.

n₁ is the number of characters in the heading, which must be in the range 1 to 20.

H is the standard PLAN symbol for character fields.

heading is the heading of a column of totals. Up to 20 characters can be punched.

Up to four parameters of this type, specifying four fields, may be input. The accumulation of each field is contained in a double-length word. Each accumulation of a specified field is repeated five times so that a total may be output on the occurrence of any one of five conditions representing five *levels* of accumulation, described under PRI below.

PRI

This statement specifies the conditions, defined in terms of criterion parameters, which will cause the accumulated totals to be PRInted out. This statement is optional but must be present if ACC is used.

The format is as follows:

PRI,*seq,c,c*,...

where *c* is a criterion parameter. Up to five criterion parameters may be input, and each parameter refers to the respective level of totals. For #XREJ only, a criterion parameter type 3, that is, EOR, must be specified.

When a level of accumulation is printed, that level is also set for accumulation to start again.

The level being output is signified by a number of asterisks preceding the print line and equal to the level printed.

PRI is the only one of the criterion checking directives that operates on Types 1, 2 and 3 parameters.

If a record is rejected for any reason, and therefore not written to the direct access file, fields from it are not included in any accumulated totals for printing, the record does not initiate the printing of any level by updating bucket or record counts, and fields are not stored for comparison with the next record.

Items are printed out as follows:

DECIMAL	Up to 14 digits
MIXED DECIMAL	Up to 14 digits integer and 6 decimal places preceded by a decimal point. The item cannot exceed double word capacity and the maximum size of the integer is dependent on the number of bits occupied by the fraction
FRACTIONAL	6 decimal digits preceded by a decimal point
OCTAL	16 octal digits preceded by #

Alternative accumulation procedures

Accumulated totals specified by the ACC directive, the printing of which is initiated by criteria of types 1, 2 or 4, normally include fields from the record that satisfies the specified criteria. If it is required that fields from a current record should not be included in the printed totals (as when printing of batch totals is to be initiated by change of control field), SWITCH 4 should be set ON after loading the program (see *Operating instruction 3*, page 205).

IGN

This statement specifies the conditions, defined in terms of criterion parameters, on which records will be IGNored. The statement is not examined until after a record has been assembled. If the conditions are satisfied the record is not written to the direct access file, no conditions indicated by VAL or PRI are examined and therefore it cannot initiate printing of any accumulated totals, and no fields from it are included in these totals.

The statement is optional.

The format is as follows:

IGN,*seq,c,c*,...

where *c* is a criterion parameter. Up to five criterion parameters of type 4 may be specified.

Examples

Statements on cards.

VAL,1R,F0>F0,F2;'4H/ZZZV',F5<'1N/4000'+F5>'1N/900'

#XREB #XREJ

This statement would cause the following conditions to be checked:

- 1 Field 0 is always greater in the current record than in the preceding record. If field 0 represents a record key, this would check the sequence of records in the field
- 2 Field 2 is not equal to **ZZZV**
- 3 Field 5 is in the range 901 to 3999 inclusive

The first parameter immediately following the card sequence number is R, therefore all records on which these checks fail will not be written to the file.

ACC,1,F5/19HTOTALVANNUALVSALARY

PRI,1,F1;F1,R=200,EOR

These statements will cause:

- 1 Field 5 to be accumulated
- 2 The accumulated totals will be printed with a leading **TOTAL ANNUAL SALARY**
- 3 The totals will be printed and the level concerned reset to zero for further accumulations whenever field 1 does not equal the same field in the previous record, every time 200 records have been written to the direct access file and at the end of run

IGN,1,F2<'1N/0'

This statement would cause any record to be ignored in which field 2 was negative.

The same statements on paper tape (#XREB) would appear as follows:

VAL,R,F0>F0,F2,'4H/ZZZV',F5<'1N/4000'+F5>'1N/900'↓

ACC,F5/19HTOTALVANNUALVSALARY↓PRI,F1;F1,R=200,EOR↓

IGN,F2<'1N/0'↓

Control statement required to define additional error procedure

ERR

This statement defines the additional **ERRor** procedure required in respect of input data punching and format errors, and validity check failures.

The statement is optional.

The format is as follows:

ERR,seq,D,V,CP,LP (for #XREJ)

or

ERR,D,V,TP,LP (for #XREB)

where

D and V indicate that the additional procedure is to be applied to input data and validity errors respectively. Either or both may be specified.

CP/TP and LP signify respectively, either card punch (#XREJ) or tape punch (#XREB) and line printer, and represent the two possible channels of output for the copy record. Either or both may be specified.

Examples

- 1 Statements on cards

ERR,1,D,,CP

This statement indicates:

- (a) Additional procedure is required in respect of input data errors
 - (b) Note that a comma is punched for the omitted statement V
 - (c) Output is required to card punch only
- 2 The same statement on paper tape would appear as follows:

ERR,D,,TP↓

Note the substitution of TP for CP.

ERROR PROCEDURES

(For further details see also the section *Exception conditions*, page 205)

Errors on input of control statements

For all control statement errors the run is halted and an error message printed on the console typewriter. Directives are printed as they are received and a pointer, in the form of an upward pointing arrow in the subsequent line, indicates the position in a line at which an error is detected. In some cases it is not possible to output such a pointer or it is without significance with respect to the cause of the error. In these cases an additional explanatory message is output to the printer.

Errors on input of user's data

Error reports are to the line printer. The run is not normally interrupted on the occurrence of data errors. However, at the user's option, the program may be suspended in which case an additional message is output to console typewriter (see *Operating instructions*, page 204).

At the user's option, a copy of the card or line containing the error may be output to a tape punch for #XREB, or a card punch for #XREJ, line printer, or both a punch and a line printer. If this facility is required, the ERR statement must be used.

This additional procedure may be applied either to errors arising from punching and format errors in the user's input data discovered by the input program, to errors arising from validity check failures as defined by the VAL statement, or to both these conditions.

Note: If the additional error procedure is applied to errors arising from validity check failures as defined by the VAL control statement, any output produced on the punch or line printer will be a copy of the first or only card or line of the next input record, and not the current record.

It is intended that the identification of the invalid record should normally be given the message:

VALIDATION ERROR Vx KEY k

where *k* is the record key. Users are advised, wherever possible (even when loading serial files) to indicate a sequential organization to enable a meaningful key field to be specified in the AFL control statement. The additional procedure is provided only as a means of identification for exceptional cases when the key field is not significant or is non-existent.

Records in which validity checks fail will be written to the direct access file unless the first parameter of the VAL statement is R (see *Criteria checking*, page 198) in which case the record will be ignored.

Records in which punching or format errors are discovered by the input program will in all circumstances be rejected. When a record is rejected, no fields from it will be stored. When the subsequent record (that is, the next 80 characters, or the next line of switch 5 is set) is processed, any type 4 criterion parameter requiring comparison with a field in the preceding record will in fact refer to the last record written to the direct access file.

End of parameter input

EOP

The statement EOP (End of Parameter Input) is provided to indicate the end of control statements.

This statement is mandatory.

#XREB #XREJ

The format is as follows:

EOPV for #XREJ

EOP↓ for #XREB

When this statement is read the program checks that all the mandatory statements have been input. If this is the case, certain additional checks are imposed on the validity of the statements read and, if no error conditions are indicated, the program proceeds to the general data input operation. However, if additional statements are required, a message is output to this effect and the program is suspended. At this stage, additional control statements, terminated by another EOP statement, may be read by re-entering the program.

Unlike all the other control statements which may be input only once, EOP may be used as many times as required to suspend the input of control statements at any stage until such time as all mandatory control statements have been received.

OPERATING

Executive priority

The programs as supplied have a priority of 70.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
TR0 (#XREB) or CR0 (#XREJ)	Control statements and user's data	The reader is assigned at the beginning of the run and used throughout the run
TP0 (#XREB) or CP0 (#XREJ)	Copies of incorrect records	These are assigned, if specified by a control statement, when the first error occurs requiring its use
LP0	Listing of control statements, error reports, copy of records, accumulated totals and ignored records	The line printer is used throughout the run
DA0	Overlay file	The direct access file is opened when the program is entered and used throughout the run
DA1	Output file	The direct access file is opened when the input of control statements is successfully completed and closed at the end of the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Either	
(a) To transcribe data from paper tape, load the program #XREB	HALTED:- LD (The above message will only be output if the loading operation was successful)
or	
(b) To transcribe data from cards, load the program #XREJ	
(see FIND A BINARY PROGRAM in <i>Compiling Systems</i>)	

<i>Narrative</i>	<i>Console message</i>
2 Load the required disc cartridge	
3 If fields from the current record are not to be included in the accumulated totals (see <i>Alternative accumulation procedures</i> , page 201), set Switch 4 by inputting:	ON #XREx 4
4 For #XREB only, if data is to be read to 'stop' character, rather than in blocks of 80 characters, set Switch 5 by inputting:	ON #XREBS
5 Load the control statements and data on the appropriate peripheral. If it is not possible to load the data at this stage, the reader is disengaged for this purpose after reading the control statements	
6 If the program is required to halt on the occurrence of a punching or format error in the user's data set Switch 0 on:	ON #XREx 0
7 If the program is required to halt on the occurrence of a validity check failure as defined by the VAL statement, set Switch 1 on:	ON #XREx 1
8 For #XREJ only, if the data format is such that a blank card can constitute valid data and therefore must not be rejected, set Switch 2 on:	ON #XREJ 2
9 Input:	GO #XREx 20
10 Control statements are read and user's data is transcribed to the direct access file. When the end of data is encountered, check counts are output to the printer, the direct access file is closed, peripherals are released and the program halts with the message:	HALTED:- HH
11 To transcribe a new file of input data with a different set of control statements, return to operating instruction 2	
12 At any point during the input of user's data, the run may be terminated prematurely by setting Switch 3 on:	ON #XREx 3
This causes the program to complete the record currently being assembled. The run is then brought to a close as described in operating instruction 9, and the program halts with the message:	HALTED:- HH

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XREB or #XREJ during execution and loading. Prior to entry into #XREB or #XREJ it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

#XREB #XREJ

Conditions arising during input of parameters

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR (#XREJ only)	No card reader is available	Make a card reader available and input: GO #XREJ
2	HALTED:- TR (#XREB only)	No paper tape reader is available	Make a paper tape reader available and input: GO #XREB
3	HALTED:- LP	No line printer is available	Make a printer available and input: GO #XREx
4	HALTED:- OVRL ER n	An error in overlaying (n is in the range 1 to 7)	See <i>Running an overlaid program</i> in Chapter 3 of <i>Compiling Systems</i>

In the following conditions, additional information is output to the printer

5	HALTED:- P1	Punching or format error in control statement or violation of limitation	Either (a) Repunch, load correct control statements and return to operating instruction 3 or (b) Abandon the run

Printer output: a copy of the statement is output and beneath the last line printed an upward arrow (↑) points to the position at which the error was discovered.

6	HALTED:- P2	Discrepancy between the INP and OUT statements	Either (a) Redraft control statements, load correct control statements, and return to operating instruction 3 or (b) Abandon the run

Printer output:

(a) P2) FIELD a REFERENCED BY OUT DIRECTIVE NOT SPECIFIED BY INP DIRECTIVE

(b) P2) FIELD a DESCRIBED DIFFERENTLY BY INP AND OUT DIRECTIVES

where a is the number of the input field.

7	HALTED:- P3	Discrepancy in type 4 criterion parameter description	Either (a) Redraft control statements and return to operating instruction 3 or (b) Abandon the run

Printer output:

(a) P3) VAL DIRECTIVE CRITERION PAIR-- n --DIFFERENT SIZE FIELDS
PRI
IGN

(b) P3) VAL DIRECTIVE CRITERION PAIR-- n --DIFFERENT DATA TYPES

(c) P3) VAL DIRECTIVE CRITERION PAIR-- n --FIELD NOT OUTPUT

Fields for comparison must be of similar length and type, and except in the case of constants, must appear in the output record.

<i>Message</i>	<i>Reason</i>	<i>Action</i>
(d) P3) VAL DIRECTIVE CRITERION PAIR— <i>n</i> —STORAGE TABLE FULL PRI IGN		

This criterion parameter indicates comparison with a field stored from the previous record: there is not room in the table for this field.

(In these four types of printer output, *n* is the number of the criterion pair within the statement.)

8	HALTED:- P4	Error in specifying fields for accumulation	Either (a) Redraft control statements, load correct control statements and return to operating instruction 3 or (b) Abandon the run

Printer output:

(a) P4) ACC DIRECTIVE-FIELD *n* NOT SPECIFIED FOR OUTPUT

Only fields that occur in the output record may be accumulated.

(b) P4) ACC DIRECTIVE-FIELD *n* IS CHARACTER FIELD

It is not possible to accumulate fields specified as character fields.

(c) P4) ACC DIRECTIVE-FIELD *n* HAS 3 WORDS OUTPUT

Two words only are allowed for accumulated totals.

In these three types of printer output, *n* is the number of an input field required for accumulation.

9	HALTED:- P5	Further control statements are needed to complete a valid set	Either (a) Load missing control statements and input: GO #XREx or (b) Abandon the run

Printer output:

P5) PARAMETER SET INCOMPLETE

10	HALTED:- P6	The size of the output record, as specified by the OUT statement, is larger than 1021 words, that is, will not fit into a bucket of maximum size	Either (a) Redraft control statements, load correct control statement and return to operating instruction 3 or (b) Abandon the run

Printer output:

P6) OUTPUT RECORD TOO LARGE

11	HALTED:- P7	The record key description required by the AFL statement for sequential files is unsatisfactory	Either (a) Redraft control statements and return to operating instruction 3

#XREB #XREJ

<i>Message</i>	<i>Reason</i>	<i>Action</i>
		or
		(b) Abandon the run

Printer output:

(a) P7) INVALID KEY LENGTH

The length is not within the range 1 to 64 characters.

(b) P7) INVALID KEY POSITION

The position specified is outside the record defined by the OUT statement.

(c) P7) NO VALID KEY FIELD

There is no field as defined by the OUT statement, corresponding in length and position to this key:

Conditions arising concerning the specification of the direct access file

<i>Message</i>	<i>Reason</i>	<i>Action</i>
12 HALTED:- A2	An attempt to open at end of file has failed because: (a) The file was not originally created using #XREB/#XREJ (b) The file has been updated by another program since being closed by #XREB/#XREJ	Abandon the run. Re-run with correct file and/or correct control statements
13 HALTED:- A4	The structure of the bucket header or of the pre-allocated direct access file does not conform with the standards of Direct Access Housekeeping or with the file description specified by the AFL statement	Abandon the run. The user should check that bucket headers were written when the file was allocated. If not re-allocate the file using #XPJC and re-run
14 HALTED:- AL	Switches 22 and 23 set	To continue the run, input: GO #XREx
15 HALTED:- C2	Executive cannot hold any further file descriptions	Either (a) Re-run later or (b) Wait until any other program(s) using the device closes its file(s) and then input: GO #XREx
16 HALTED:- C3	The flaw area is full	Abandon the run
17 HALTED:- D1	An attempt has been made to open an output file, other than at the end of file, and the current date does not equal or exceed the date written plus the data retention period	Abandon the run

<i>Message</i>	<i>Reason</i>	<i>Action</i>
18 HALTED:- D3	The file to be used is corrupt or bucket headers were not written when the file was allocated	Abandon the run and re-allocate the file with #XPJC
19 HALTED:- G1	A specified file is not in the system	Either (a) Load the cartridge holding the required file and input: GO #XREx or (b) Abandon the run
20 HALTED:- G2	An integrity code check has failed	Either (a) Re-run later or (b) Wait until any other program using the file closes the file and then input: GO #XREx
21 HALTED:- G3	The file organization recorded in the File Index and Parameters differs from that specified in the AFL statement	Abandon the run. Re-run with corrected file and/or control statements. Ensure that a file with two-word bucket headers is in use
22 HALTED:- G4	The fixed length indicator in the File Index and Parameters is not set	Either (a) To ignore this failure input: GO #XREx or (b) Abandon the run. Re-run with corrected file
23 HALTED:- G5	A record is either (a) Larger than the usable bucket space or (b) Larger than the maximum record length recorded in the File Index and Parameters	Abandon the run
24 HALTED:- G6	Where a sequential file organization is indicated the position and/or length of the record key recorded in the File Index and Parameters does not agree with that specified by the AFL statement	Abandon the run. Re-run with correct file and/or control statements

#XREB #XREJ

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
25	HALTED:- H1	There is no space in the file to accommodate further data or to write the end of the file bucket	Abandon the run. Allocate a further area, using #XPJC and re-run
26	HALTED:- H2	Effective bucket size, calculated by applying bucket packing density to usable bucket space, is not sufficient to contain a record	Abandon the run. Re-run with correct control statements. Check that bucket headers were written when the file was allocated and if not re-allocate the file with #XPJC
27	HALTED:- H4	The index buckets for a level of index are full	Abandon the run. Re-run with correct file and/or control statements
28	HALTED:- K1	An unclearable parity error has been detected	Abandon the run
29	HALTED:- xx	An error has been detected when loading #XREB or #XREJ by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

Conditions arising from input of user's data

This section considers punching and format errors in the user's data records and violations of the limits set by the control statements and by the program. Normally error reports are made only on the printer, and since there is no output to the console typewriter, the run is not interrupted and no action is called for from the operator.

The format of the printer report is:

DATA ERROR *n* KEY *k*

where

n is a code representing in general terms the type of error.

k is the record key of the last record written to the direct access file.

The error codes represented by *n* are as follows:

<i>Code</i>	<i>Interpretation</i>
A	Non-permissible character encountered in a decimal or octal input field. For details of restrictions and limitations see the section <i>Input</i> , page 187
B	On conversion to binary, a decimal input field cannot be accommodated without loss of significance within the space indicated for the field
D	A fractional field or the fractional portion of a mixed field is not as specified. 1 If there is a symbol representing the decimal point, there are not the specified number of digits to the right of the point 2 If there is no symbol representing the decimal point, the number of digits in the entire field is less than that specified for the fraction
E	(#XREB only) A non-permissible delta-shift character has been encountered in the input stream. For details of limitations see the section <i>Input data on paper tape</i> , page 187
F	(#XREB only) A beta-shift has been encountered
G	Where a record terminator is specified by TER statement, this character is not found at the logical end of the input record, that is, immediately following the final field of a record

Code Interpretation

H Where a record terminator is specified by TER statement, this character is encountered before the logical end of the record

At the user's option, by means of an ERR statement, the error report above may be followed immediately by a copy of the card or line either to a line printer or to a card or paper tape punch or to both a line printer and a card or paper tape punch. For #XREB only, if Switch 5 is set the line, rather than the block of 80 characters, containing the error will be output on the line printer or paper tape punch.

If a printed copy is requested, an upward-pointing arrow is printed immediately below the line in which the error occurred, pointing to the character at which transcription ceased.

If a punched copy is required, the following message may be output to console typewriter:

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
30	HALTED:- CP (#XREJ only)	No card punch is available	} Make a punch available. Input: GO #XREx
31	HALTED:- TP (#XREB only)	No tape punch is available	

At the user's option by setting Switch 0 ON, the program may be required to halt on the occurrence of a data error (see operating instruction 6). In this case a message is output to the console typewriter:

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
32	HALTED:- DE	A data error on one of the types mentioned has occurred	Either (a) To cause the program to reject the failed record and continue transcription, input: GO #XREx or (b) Abandon the run

Other reports arising from input of user's data

This section considers four types of reports:

- 1 Validity check failure as defined by the VAL statement
- 2 Printing of totals as required by the ACC and PRI statements
- 3 Ignoring of records in accordance with the IGN statement
- 4 End of run reports

Normally, reports arising from these four circumstances are only made on the line printer. There is no output to the console typewriter, the run is not interrupted and no action is called for from the operator.

Validity check failure

The format of the printed report (which is output as many times as necessary per record) is

VALIDATION ERROR V *x* KEY *k*

where

x is a number (in the range 0 to 9) representing the number of the parameter within the VAL statement that has initiated printing.

k is the record key by which the record may be identified.

At the user's option, by means of an ERR statement, the report may be followed immediately by a copy of the first or only card or line of the next input record either to a line printer, or to a card or paper tape punch, or to both

#XREB #XREJ

a line printer and a card or paper tape punch. For #XREB only, if Switch 5 is set, the line, rather than the block of 80 characters, is output.

At the user's option, by setting Switch 1 ON, the program may be required to halt on the occurrence of a validity error (see operating instruction 7). In this case a message is output to the console typewriter.

<i>Message</i>	<i>Reason</i>	<i>Action</i>
33 HALTED:- VE	A validation check failure (as mentioned earlier) has occurred	Either (a) To continue transcription, rejecting or accepting the failed record as previously indicated by the VAL statement, input: GO #XREx or (b) Abandon the run

Printing of totals

The format of the printed output is as follows:

	HEADING 1	HEADING 2	HEADING 3	HEADING 4
*****	Total 1	Total 2	Total 3	Total 4

The first line contains as many headings (up to a maximum of 4) as are specified by the ACC statement.

The second line is preceded by 1 to 5 asterisks representing the level of totals being printed (see PRI statement). The format of the printing for different types of field is described in the section *Criteria checking*.

Should any total in process of accumulation exceed double-length capacity, that total is printed prematurely (that is prior to the addition causing it to overflow) and set to zero for accumulation to recommence. In this case the format of the printed output is as follows:

	HEADING 1	HEADING 2	HEADING 3	HEADING 4
*****			Total	

The headings are all printed. The asterisks, as before, indicate the level in which the overflow has occurred, but it would be noted that only this item of the level is printed.

Ignoring of records

The format of the printed output is as follows:

RECORD IGNORED KEY *k*

where *k* is the key by which the record may be identified.

End of run

When the end of data is detected, check counts are output on the line printer in the following format:

RI = *n*

RO = *n*

XD = *n*

XV = *n*

XI = *n*

CR = *n*

where

n is a number in the range 0 to 8388607.

RI is records input.

RO is records output to the direct access file.

XD is records rejected for data errors.

XV is records rejected for validity check failures.

XI is records ignored in accordance with the IGN statement.

CR is cards received, excluding cards ignored because blank (#XREJ only).

The run is then terminated with the console message referred to in operating instruction 10.

NAME

#XRPE

VERSION

Mark 4B

TITLE

Printer report of direct access records

HARDWARE REQUIREMENT

5056 words of core store (further store is requested according to the number of key parameters input)

1 direct access unit

1 card reader or 1 paper tape reader

1 line printer (optionally with Extended Repertoire Character Set)

DESCRIPTION**General**

This program produces edited printed output from a standard housekeeping format direct access file. Very powerful editing facilities are provided such that this program can in many cases do the work of a specially prepared program.

Data in a standard direct access file record can be treated as characters or as binary representation of decimal quantities. Fields within a record do not have to be output in the order that they appear on the file; the parameters allow a new order to be specified for the printout.

The user may specify by record number or by record key the point on the file at which printing is to commence, and the number of records to be printed. Alternatively, selective printing of consecutive records containing individually specified key values may be requested, or all records containing one specified key may be printed.

Facilities are provided to allow constant data, input as parameters, to be printed as page headings or to be included in the printing of each record (for example, as record headings).

Several options on the use of the page and the throw facilities of the printer are provided and page numbering can be specified. The program allows for line printer widths of up to 160 print positions.

Only one editing format can be specified, that is, all records printed will be edited in the same fashion.

Optionally, printing in Extended Repertoire Character Set (ERCS) may be requested.

Control statements specifying the records to be printed and the editing format are read in from cards or paper tape.

The facility to dump a version of the program with preset parameters is available (see *Operating*, page 222).

Input

The file from which the report is produced must be a simple file written in accordance with the ICL 1900 Series direct access housekeeping format. Bucket sizes may be 1, 2, 4 or 8 blocks.

All files whether serial, random or sequential will be accessed in the same way; records will be read in the order that they occur on the file but tags will be followed up. Thus serial and sequential files will be presented correctly but on a random file tagged records may be presented twice, once when the tag is found and followed up and once when the overflow record itself is read. Variable length records can be handled; if data accessed is outside the record limits, it will be replaced by spaces in the printout.

#XRPE

Output

Output is to the line printer and consists of selected items from a direct access file record. Specified headings may be output to each page and automatic page numbering starting from a given page number is available. The printed output for each record may occupy one or more lines of print. Items from the record may be output in any order, in any position on the print line, and spaces, constants and the current date obtained from Executive may be inserted as required. A line of print may occupy up to a maximum of 160 print positions.

On detection of the page overflow condition or on expiry of a given line count, whichever is required, the program will:

- 1 If headings are included in the report, throw to the appropriate channel or line space as indicated by the user, and print the headings

or

- 2 If headings are not included, throw to channel 1

Items from the record may be printed in character format or may be converted as follows:

- 1 Single or double-length binary integer to decimal integer
- 2 Single or double-length binary fraction to decimal fraction
- 3 Binary mixed integer with single or double-length integral and fractional component to mixed decimal and fractional number
- 4 Single or double-length binary integer to scaled mixed decimal or fractional number

ALIGNING PRINTER PAPER (OPTIONAL)

Before printing the selected data or after the "paper low" condition has been reported a special printout is made to assist the operator in aligning the stationery. The printout consists of a throw to channel 1 followed by 7 lines of numeric data. Each print position in the line contains the tens digits of its position:

Print positions 1 to 9 contain 0

Print positions 10 to 19 contain 1

Print positions 20 to 29 contain 2

etc.

Print positions 100 to 109 contain 0

Print positions 110 to 119 contain 1

Print positions 120 to 129 contain 2

etc.

The printout is preceded by the console message:

DISPLAY:- LINE UP PRINTER PAPER

and is immediately followed by disengaging the printer which leads to the console message:

UNIT *n* FIX

If a further output of the line-up data is required, the operator re-engages the printer. The line-up data is output again, the printer disengaged causing the console message:

UNIT *n* FIX

This procedure may be repeated as many times as required. When the operator is satisfied with the alignment of the paper he will input:

GO #XRPE 24

This will lead to a continuation of the main program. At the first printer order encountered a further output of the console message:

UNIT *n* FIX

will occur. The operator will re-engage the printer and the program will continue without further interruption.

The paper line-up procedure may be omitted by inputting the console message:

ON #XRPE 18

before initial entry to the program.

CONTROL

The operation of the program #XRPE is governed by control statements. Each control statement consists of an identifier (A, B or C) followed by a string of parameters, each of which is followed by a comma. These control statements must be input in the order A, B, C. They may be input on cards or on paper tape, and in the latter case the number of internal characters to a line must not exceed 80 excluding the newline terminator. Key parameters if specified must be input between control statements A and B. (See *Printing by record keys*, page 218.)

Control statement A defines the file and the records from which the report is to be made. It must be input on a single line or card.

Control statement B specifies the format of the print line(s) for each record.

Control statement C specifies the format of the print line(s) for page headings. Its presence is optional.

Control statements B and C may extend over several lines or cards and the identifier, which must be repeated in every line or card, must be immediately followed by a two-digit sequence number, starting at 01, (for example B01, B02, B03, etc.).

Control statement A

This control statement defines the file and records from which the report is to be made and the page end code. It takes one of two forms.

PRINTING BY RECORD COUNT

A, (*fn*(*fgn*))*a,b,c*,

where

A is the identifier, is obligatory, and must be followed by the comma.

(denotes the start of the file definition data and is obligatory.

fn is the obligatory filename and may consist of up to 12 characters, the first 9 of which must not contain a colon. If less than 12 characters are present the remainder will be space-filled.

(the second open bracket must be present only if *fgn* is specified.

fgn is the file generation number and is optional. If specified it must be a decimal integer in the range 0 to 4095. If no file generation number is specified, it will be set to zero.

) the first closed bracket must be present only if *fgn* was specified.

) the second closed bracket denotes end of file definition and is obligatory. It must be followed by a comma.

a is a decimal integer in the range 0 to 8388607 denoting the number of records to be ignored. If no records are to be ignored it must be specified as zero. It must be followed by a comma.

b is a decimal integer in the range 0 to 8388607 denoting the number of records to be printed. Its presence is obligatory. If its value is zero printing will continue to the end of the file or to a logical bucket number out of range on a random file. It must be followed by a comma.

c is the page code and may take one of the following four forms. Its presence is obligatory and it must be followed by a comma.

<i>Value of c</i>	<i>Effect</i>	
	<i>Headings specified (Control statement C present)</i>	<i>No headings specified (Control statement C absent)</i>
I(letter)	Headings will be printed on entry. No further headings will be printed and page overflow will be ignored	A throw to channel 1 will occur on entry. No further paper-throws will occur and page overflow will be ignored
O(letter)	Headings will be printed on entry, and thereafter each time page overflow is detected	A throw to channel 1 will occur on entry and each time page overflow is detected
R	As for O except that printing of the current record will be completed before throwing and printing headings	As for O except that printing of the current record will be completed before throwing to channel 1
Cnn	Headings will be printed on entry and thereafter on expiry of line count nn. Headings are not included in the line count	Throw to channel 1 on entry and thereafter on expiry of line count nn

PRINTING BY RECORD KEYS

- 1 To print a number of records starting at a record containing a specified key, K is input instead of the number of records to be omitted and control statement A is immediately followed by a card or line containing the key information. Control statement A will therefore take the form:

$A,(fn(fgn)),K,b,c,$

where

K indicates that a single key parameter card or line, specifying the starting key for printing, will follow control statement A. The remainder of control statement A parameters are as outlined on page 217.

The key parameter card or line will take the form:

$K/v/n,key,$

v is the number of characters preceding the key in the record.

n is the number of characters in the following key.

(n must be in the range 1 to 64, and ($n + v$) must be in the range 1 to 4088)

- 2 To print all consecutive records containing an individually specified key, K is input instead of the number of records to be omitted and is followed immediately by b , the number of keys to be specified without the separating comma. Control statement A will therefore take the form:

$A,(fn(fgn)),Kb,c,$

Kb indicates that b key parameter cards or lines specifying the keys of records to be printed will follow control statement A (b must be non-zero). The upper limits of b will be determined by the amount of store available at the time of inputting the key parameter information. The remainder of control statement A parameters are as outlined on page 217.

The key parameter cards or lines are of the same form as shown in 1 above. The keys must be input in the sequence in which they will be read from the file. However, they need not be of the same length or in the same position as the key specified for the file on allocation and held in the file index and parameters of the System Control Area

- 3 To print all records of one specified key, K is input instead of the number of records to be omitted, K is also input instead of the number of records to be printed. Control statement A is followed immediately by a single card or line containing the key information. Control statement A will therefore take the form:

$A,(fn(fgn)),K,K,c,$

The key parameter card or line is in the same form as shown in 1 on the previous page

Note: When the data is on paper tape the four delta-shift character \$] ↑ ← will be correctly converted to internal code; other delta shift characters will lead to the error halt K5. Each key parameter must be contained within one line and the limitation of 80 internal characters per line must be borne in mind.

End of file or logical bucket number out of range conditions

When the program is terminated because of these conditions, a display will be output to the console typewriter before reaching the normal halt. This should assist the user in determining whether or not all records specified have been printed.

Control statement B

This control statement specifies the format of the print line(s) relating print positions to fields in the record, and has the format:

Bss,d,d,d,.....d,d,

where

Bss, is the identifier B followed by a two-digit sequence number, starting at 01, denoting the number of card/lines within this control statement, (B01, B02, B03, etc.)

d,d,d,.... are directives describing the print line and may be any of the eleven types described below. A directive must not exceed 24 characters in length and must be wholly contained within one card/line.

Directives must be input in the sequence needed to describe every print position in every print line of the report from position 1 to the end of the data required in that line. A print line may not be defined as exceeding 160 characters in length; a warning DISPLAYCO message will be output if it is defined as exceeding 120 characters in length. If printing in ERCS is required it should be noted that any shift characters will be included in the count of print positions. This count can take a value of up to 160 characters. Thus the presence of shift characters will result in a reduction of the maximum length of the line which can actually be printed.

PRINTER SPACING

The first directive for a given print line must be one of the following:

T/n, Throw to channel *n* before printing; subsequent printing will commence in print position 1.

n must be in the range 1 to 7

L/n, Line space *n* lines before printing; subsequent printing will commence in print position 1.

n must be 1 or 2

INSERTION OF DATA EXTERNAL TO THE RECORD INTO THE PRINT LINE

In all cases the data is inserted commencing from the last print position filled, or, following directives L or T, from print position 1.

S/n, Space-fill the next *n* positions of the print line.

n must be in the range 1 to 120 (if it is required to space more than 120 positions two separate S/*n*, directives should be used).

H/n,x, Insert constant into the next *n* positions of the print line.

n is the number of characters in the constant. If printing in ERCS is required it should be noted that *n* includes any shift characters

x represents the constant which may contain any alphanumeric characters in the ICL 1900 Series 64-character code but must be wholly contained within one card/line. The constant must be terminated by a comma.

#XRPE

Example

H/8,EXT,INT.,

will insert the constant EXT,INT. into the next 8 positions of the print line.

Input on paper tape The four delta-shift characters S] ↑ ← may be included in the constant x and will be correctly translated on print out. The inclusion of such delta-shift characters as will cause the end of transfer of data will lead to a D2 error (number of characters H/ n not equal to n). Other delta-shift characters will be translated to spaces.

The limitation of 80 internal characters per paper tape line must be borne in mind

P/ n ,

Insert the current page number starting at n , into the next four positions of the print line. The page number will be incremented by 1 each time the headings are printed or, if no headings are specified, on throwing to channel 1 after detection of page overflow.

n must be in the range 1 to 9999.

Page numbers in excess of 9999 will be incorrectly printed

D,

Insert today's date into the next 8 positions of the print line.

The date is printed in the form $dd/mm/yy$ where dd is the day, mm is the month and yy is the year

INSERTION OF DATA EXTRACTED FROM THE RECORD DIRECTLY INTO THE PRINT LINE

As above, the data is inserted commencing from the last print position filled or, following directives L or T, from print position 1. If data accessed is not contained within the record spaces will be inserted in the printout.

A/ n_1/n_2 ,

Move to the next n_1 positions of the print line n_1 characters commencing at character n_2 of the record. If printing in ERCS is required it should be noted that n_1 includes any shift characters.

n_1 must be in the range 1 to 120 (if it is required to print a string of more than 120 alpha characters, two separate A/ n_1/n_2 , directives should be used).

n_2 must be in the range 1 to 4088.

(Character 0 word 0 of the record is specified by giving n_2 a value of 1.)

INSERTION OF DATA EXTRACTED FROM THE RECORD INTO THE PRINT LINE, AFTER CONVERSION FROM BINARY TO DECIMAL

If data accessed is not contained within the record spaces will be inserted in the printout. In these directives:

n and m represent the number of words within which a field is contained in the record, and must be 1 or 2.

n_2 and m_2 represent the character position within the record at which a field commences and must be in the range 1 to 4085 and must indicate an integral word boundary; that is $n_2 - 1$ or $m_2 - 1$ must be divisible by 4.

(Character 0, word 0 of the record is specified by giving n_2 or m_2 a value of 1.)

DECIMAL FIELDS

N/ $n/n_1/n_2$,

Convert the n words commencing at character n_2 of the record, from binary to decimal integer and insert into the next n_1 positions of the print line.

n_1 must be in the range 1 to 15.

If the binary integer is negative a minus sign will be floated before the decimal figure printed; allowance must be made for the sign when specifying n_1 . An error is reported if the result cannot be accommodated in the number of characters specified. Leading zeros are automatically converted to spaces but, if the binary integer is zero, a zero is printed in the least significant character position

SCALED DECIMAL FIELDS

This facility allows a quantity held in the record as a single or double-length binary integer to be multiplied by a factor of 10^{-x} and output as a mixed decimal or fractional number, that is, a decimal point is inserted to the left of the x th character position from the right-hand end of the converted decimal integer.

$Nx/n/n_1/n_2$, Convert the n words commencing at character n_2 of the record from binary to decimal, insert a decimal point to the left of the x th character position from the right-hand end of the converted decimal integer, and insert the mixed decimal or fractional number produced into the next n_1 positions of the print line (n_1 must be in the range 1 to 16, x must be in the range 1 to 14, and n_1 must be greater than x .)

The decimal point position must be included in the count n_1 . If the binary integer is negative a minus sign will be printed immediately to the left of the mixed number; allowance must be made for the sign when specifying n_1 .

An error is reported if the result cannot be accommodated in the number of characters specified. Leading zeros of the integral portion of the figure output are automatically converted to spaces

FRACTIONAL FIELDS

$F/n/n_1/n_2$, Convert the n words commencing at character n_2 of the record from binary to decimal fraction and insert the fraction, preceded by a space followed by a decimal point, in the next n_1 print positions. If the fraction is negative, the leading space will be converted to a minus sign. Allowance must always be made for the 2 fixed leading positions when specifying n_1 .

n_1 must be in the range 1 to 16

MIXED DECIMAL AND FRACTIONAL FIELDS

$N/n/n_1/n_2 F/m/m_1/m_2$, Convert a mixed binary number to decimal, N specifying the integral portion and F the fractional portion. If the mixed number is negative a minus sign will be floated before the integral position. The decimal point position must be included in the count m_1 . An error will be reported if the integral portion cannot be accommodated within the number of characters specified.

n_1 and m_1 must be in the range 1 to 15. It should be noted that for mixed fields there is no comma between the N portion of the directive and the F portion

Control statement C

This control statement is optional. It specifies the format of the page headings and has the format:

$Css,d,d,d,\dots,d$,

where

Css is the identifier C followed by a two digit sequence number, starting at 01, denoting the card/line within the control statement.

$d,d,d,\dots$ are directives describing the contents of headings.

All directives shown for control statement B are available for control statement C . Data extracted from the record to be included in page headings will be taken from the current record; that is, that record that is printed immediately following the page headings, whether the record is wholly or only partially contained on the new page.

Notes:

- 1 The total number of directives used within control statements B and C must not exceed 200. The mixed number directive $N/n/n_1/n_2 F/m/m_1/m_2$, and the scaled decimal directive $Nx/n/n_1/n_2$, count as two directives when calculating this total

#XRPE

- 2 The total number of characters inserted by control statements B and C using the H directive must not exceed 600
- 3 Because a very wide range of printer facilities may be specified, some illogical combinations of parameters may produce unexpected results. In particular, it should be noted that the design of the printer is such that only a paper throw will unset page overflow condition. If all three of the following conditions are met:
 - (a) Headings are specified
 - (b) Page end code O and R has been used to indicate that headings are to be printed each time page overflow is detected, and
 - (c) The headings in control statement C do not contain any paper throw directivesthe program will print the first page correctly, and will print the headings when the page overflow is detected. However, because the headings contain no paper throw directives, page overflow condition is not unset, and the program will, thereafter, print the headings after each subsequent line of the report

End of parameters

The end of parameters is indicated by a card or line containing **** in columns 1 to 4.

Example of a printer report with parameters read from cards

A, (XRPEFILE(4)), 0, 0, R,

This parameter opens a disc file XRPEFILE of generation number 4. All records on the file are to be printed. Page overflow will be checked only on completion of printing each record.

B01, L/2, H/5, NAME:, S/5, A/16/5, S/6, H/9, INITIALS, S/1, A/3/21, L/1, H/9, EXCHANGE:,

B02, S/1, A/12/25, S/10, H/7, NUMBER:, S/3, A/8/37,

This pair of parameters defines the print format of each record to be output. It consists of two consecutive lines of print, and the print for each record is separated by a single line. An example of a print for a record is:

Print positions:	1	11	33	43
	NAME:	SMITH	INITIALS:	AR
	EXCHANGE:	MAIDENHEAD	NUMBER:	62937

C01, T/1, S/1, H/4, DATE, S/1, D, S/4, H/23, *▽▽**▽▽*** ▽▽*****▽▽***** S/5, H/9, XRPE▽TEST,

C02, S/2, H/17, TELEPHONE▽NUMBERS, S/5, H/23, *****▽▽*****▽▽***▽▽**▽▽*, S/6,

C03, H/5, PAGE▽, P/1,

These three parameters define the heading to be output on the first line of each page as shown.

DATE dd/mm/yy *▽▽**▽▽***▽▽*****▽▽***** XRPE TEST TELEPHONE NUMBERS *****▽▽*****▽▽***▽▽**▽▽* PAGE*n*

OPERATING

Executive priority

The program as supplied has a priority of 50.

Use of peripherals

Device	Use	Assignment
CR0 or TR0	Parameters	The reader is assigned at the beginning of the run and released when the parameters have been read, analyzed and found correct

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
LPO	Edited report	The printer is assigned at the re-activate point, after the parameters have been read, analyzed and found correct
DAO	Data input	The file is opened at the re-activate point after the parameters have been read, analyzed and found correct

Running under Executive alone**OPERATING INSTRUCTIONS**

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XRPE (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required disc unit	
3 Load the parameters in the appropriate reader	
4 If required to by-pass the paper line-up procedure, input:	ON #XRPE 18
5 If printing in ERCS is required, input:	ON #XRPE 23
6 To activate the program (a) For parameters on paper tape, input: or (b) For parameters on cards, input:	GO #XRPE 20 GO #XRPE 21
7 The parameters are read and checked, the tape reader or card reader released, and the program halts with the message:	HALTED:- EP
8 To re-activate the program, input:	GO #XRPE
9 If operating instruction 4 was carried out, this instruction and operating instruction 10 will be omitted. Otherwise the console message: will be followed by the output of alignment data to the line printer (See <i>Aligning printer paper</i> , page 216)	DISPLAY:- LINE UP PRINTER PAPER
10 To continue with the program input:	GO #XRPE 24
11 The program commences to read the file and output the report. At the end of the report, the file is closed, the line printer is released and the program halts with the message:	HALTED:- HH
12 For a further run return to operating instruction 2	

#XRPE

Dump and restart procedure after parameter input

	<i>Narrative</i>	<i>Console message</i>
13	After operating instruction 7 has been completed (that is, the message HALTED:- EP has been output) it is possible to dump the program with all the parameter information recorded. The dumped program may then be reloaded later	
14	To activate the dumped program input:	GO #XRPE 22
15	The program then proceeds as from operating instruction 9	

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XRPE during execution and loading. Prior to entry into #XRPE it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- A0	First card/line is not control statement A	Either (a) Correct the control statements and return to operating instruction 3 or (b) Abandon the run by inputting: GO #XRPE 29
2	HALTED:- A1	There is a format error in the file definition (between brackets)	As for A0
3	HALTED:- A2	The file generation number is not in the range 0 to 4095	As for A0
4	HALTED:- A3	The end of page code has not been recognized	As for A0
5	HALTED:- A5	b is zero in control statement A when records are to be read by keys	As for A0
6	HALTED:- A6	First directive of parameter card/line B01 or C01 is not L or T	As for A0
7	HALTED:- AB	Run abandoned following GO #XRPE 29	If further runs required return to operating instruction 2
8	HALTED:- B0	First solidus (/) is missing from the directive	As for A0
9	HALTED:- B1	The comma is missing at the end of directive D	As for A0
10	HALTED:- B2	The second solidus (/) is missing from A, N, F or Nx directive	As for A0

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
11	HALTED:- B3	The comma is missing at the end of directive A	As for A0
12	HALTED:- B4	The comma is missing at the end of H, L, T, S or P directive or the middle of an H directive	As for A0
13	HALTED:- B5	The third solidus (/) is missing from N directive	As for A0
14	HALTED:- B6	The third solidus (/) is missing from F or Nx directive	As for A0
15	HALTED:- B7	The second solidus (/) is missing from F portion of N-F directive	As for A0
16	HALTED: B8	The comma is missing from the end of directive N, F, Nx or N-F	As for A0
17	HALTED:- B9	The first solidus (/) is missing from F portion of N-F directive	As for A0
18	DISPLAY:- C0	As defined by B or C control statements the number of print characters in line exceeds 120	If this was not expected the run should be abandoned at the HALTED EP event
19	HALTED:- C0	As defined by B or C control statements the number of print characters in a line is greater than 160	As for A0
20	HALTED:- C1	The number of print characters is less than 1 (A, N or F directive)	As for A0
21	HALTED:- C2	The starting page number is not in the range 1 to 9999 (P directive)	As for A0
22	HALTED:- C3	The printer spacing is not 1 or 2 (L directive)	As for A0
23	HALTED:- C4	The channel is not in the range 1 to 7 (T directive)	As for A0
24	HALTED:- C5	The number of print characters is greater than 15 (N directive)	As for A0
25	HALTED:- C8	The number of print characters is greater than 15 (F portion of N-F directive)	As for A0
26	HALTED:- C9	The number of print characters is less than 1 (H or S directive)	As for A0
27	HALTED:- CR	No card reader is available	Make the card reader available and input: GO #XRPE

#XRPE

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
28	HALTED:- D0	The length of the directive exceeds 24 characters	As for A0
29	HALTED:- D1	The record character address is not in the range 1 to 4088 (A directive)	As for A0
30	HALTED:- D2	The number of characters following H/n is not equal to <i>n</i> (or the comma is missing)	As for A0
31	HALTED:- D3	The length of the input area has been exceeded in specifying <i>n</i> in H/n directive	As for A0
32	HALTED:- D4	The number of binary words is not 1 or 2 (N, F or Nx directive)	As for A0
33	HALTED:- D5	The number of binary words is not 1 or 2 (F portion of N-F directive)	As for A0
34	HALTED:- D6	The record character address is not in the range 1 to 4085 (N, F, Nx or N-F directive)	As fo A0
35	HALTED:- D7	The record character address is not the beginning of a word (N, F, Nx or N-F directive)	As for A0
36	HALTED:- D8	There is an incomplete directive at the end of a card/line	As for A0
37	HALTED:- D9	Number of spaces or characters in S or A directive exceeds 120	As for A0
38	HALTED:- E0	There is a card sequence error within control statements B and C	As for A0
39	HALTED:- E1	The number of directives in control statements B and C is greater than 200	As for A0
40	HALTED:- E2	There is an unidentifiable directive (control statement B or C)	As for A0
41	HALTED:- E3	The total number of characters input by H directives is greater than 600	As for A0
42	HALTED:- ED	An E.D.S. file is not in the system	When the file has been put on-line input: GO #XRPE If the file is not available, abandon the run
43	HALTED:- F1	The number of print characters is insufficient to accommodate the field in converting N to decimal	Input: GO #XRPE 29 to abandon the run. Re-run with corrected control statements

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
44	HALTED:- G0	x in Nx directive not in range 1 to 14	As for A0
45	HALTED:- G1	The number of print characters in the Nx directive is greater than 16	As for A0
46	HALTED:- G2	n_1 is not greater than x in Nx directive	As for A0
47	DISPLAY:- H0	The program has been terminated due to EOF	None
48	DISPLAY:- H1	The program has been terminated because logical bucket number is out of range	None
49	HALTED:- IC	Either (a) An integrity code check has failed or (b) Executive cannot hold any further file descriptions	Abandon the run by inputting: GO #XRPE 29 Re-run later
50	HALTED:- K0	Incorrect number of key parameters, or B01 parameter card/line not in expected position	As for A0
51	HALTED:- K1	There is a format error in a key parameter	As for A0
52	HALTED:- K2	n in key parameter is not in range 1 to 64	As for A0
53	HALTED:- K3	$(n+v)$ in key parameter is not in the range 1 to 4088	As for A0
54	HALTED:- K4	In a key parameter a key is not terminated by a comma, or the number of characters in a key is not equal to n	As for A0
55	HALTED:- K5	The delta-shift character in a key is not convertible	As for A0
56	HALTED:- K6	A key parameter on paper tape exceeds one line	As for A0
57	HALTED:- NEEDS EL OF MINIMUM SIZE xxx	Switch 23 is set. No line printer with ERCS of this size is available. (xxx is 96, 120, 132 or 160)	Either (a) To assign a printer with ERCS, Give #XRPE the required printer as unit 0, and input: GO #XRPE #XRPE will not test that the printer Given is of the correct size. If the printer is too small, excess characters will be lost.

#XRPE

<i>Message</i>	<i>Reason</i>	<i>Action</i>
		or
		(b) Abandon the run by inputting: GO #XRPE 29
58 HALTED:- NEEDS LP OF MINIMUM SIZE <i>xxx</i>	Switch 23 is unset. There is no line printer of this size available. (<i>xxx</i> is 96, 120, 132 or 160)	Make a line printer available and input: GO #XRPE #XRPE will again test that a printer of the correct size is available. If not this message is repeated
59 DISPLAY:- PL	Paper low condition has been detected	Load more paper and re-engage the printer
60 HALTED:- SY	GO #XRPE 24 has been input in the wrong place	Abandon the run by inputting: GO #XRPE 29
61 HALTED:- TR	There is no tape reader available	Make the tape reader available and input: GO #XRPE
62 HALTED:- XX	DA exception condition that is not covered by the halts IC or ED	Abandon the run by inputting: GO #XRPE 29
The following three irrecoverable error conditions may be reported via the housekeeping package associated with the program.		
63 HALTED:- DA ERROR B ABANDON	A tag is missing, or there is no overflow record that corresponds to the tag	Abandon the run by inputting: GO #XRPE 29
64 HALTED:- DA ERROR F ABANDON	An unclearable parity error has been detected	Abandon the run by inputting: GO #XRPE 29
65 HALTED:- DA ERROR Z ABANDON	Either (a) There is an illegal record length (zero or larger than bucket size) or (b) There are too many levels of index or (c) There is an unidentifiable reply to a PERI instruction or (d) A second level overflow indicator has been set (in B9 of bucket header) but no second level overflow area is allocated on this file	Abandon the run by inputting: GO #XRPE

<i>Message</i>	<i>Reason</i>	<i>Action</i>
66 HALTED:- xx	An error has been detected when loading #XRPE by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

Chapter 3 GEORGE 3 system macro

TITLE

DAFAM

FUNCTION

To produce #XPJK file reports and/or cartridge reports.

RESTRICTIONS

The user must have the TRUSTED privilege.

There is a restriction of 15 on the number of reports which may be obtained in one run of the macro.

#XPJK reports require at least a 120-print-position line printer.

There are no safeguards in the macro to prevent the user allowing the #XPJK line printer output to be sent to a MOP console, although the size and format of the cartridge report is unsuitable for this form of output. A form of listing parameter which sends the line printer output to the monitoring file system is available and it is left to the user who wishes to use this facility in the MOP context to ensure, by means of the REPORT command, that object program output does not reach his console while the macro is running.

FORMAT

DAFAM *parameter list*

The parameter types in the parameter list are:

LISTING

ERROR

REPORT

They have no fixed position and if an optional parameter is omitted no null parameter is required in its place.

Listing parameter

The listing parameter consists of the characters *LP optionally followed by a filestore file description. The format is:

**LP file description*

or

**LP*

This parameter type is optional. If a file description is included, it specifies a file to which the program #XPJK is to write its line printer output. If no file description is given, it indicates that the line printer output is to be sent to the monitoring file system in the OBJECT category. In the MOP context this facility should be used with care (see *Restrictions*, above). If the listing parameter is omitted altogether the macro will create a workfile to receive the line printer output and, at the end of the run, LISTFILE it to a line printer and ERASE it.

Error parameter

The error parameter consists of the characters ER followed by a label. The format is:

ER label

This parameter type is optional. It specifies a point in the job description to which control should be passed if the macro detects an error in its parameters, if the run of the program fails to reach a satisfactory conclusion or, in the NOT MOP context, if a command error (such as might be caused by incorrect macro parameters) occurs in the macro.

DAFAM

Note: Like other system macros this macro uses labels beginning with 9 and, to avoid ambiguity, the error parameter should not specify a label beginning with that character.

Report parameters

Up to 15 report parameters may be specified. They may take any of the following forms, where *csn* is an octal number of between 1 and 6 digits. Non-significant zeros may be omitted.

Where

filename is the name of the file to be reported on,

gen.no is the generation number of that file:

- 1 **FILE(*filename*, *gen.no*, *csn*)**
for the file specified, the program prints the details of the file as held in the file index cell followed by a table of the file areas allocated to the file
- 2 **F*csn***
requests a table of all the files on the cartridge in alphabetical order. The table gives details of the files as held in the file index cells
- 3 **FA*csn***
as for report type F, plus, for each file, a list of its file areas in order of allocation
- 4 **M*csn***
as for report type F, plus a map of the cartridge
- 5 **MA*csn***
as for report type FA, plus a map of the cartridge

EXECUTION

The macro forms the report parameters specified into a format suitable for #XPJK and uses the program's entry point 2.

LIBRARY ENTRANTS ACCESSED

<i>Entrant</i>	<i>Mode of access</i>
:LIB.PROGRAM XPJK	EXECUTE

ERROR MESSAGES

The following error messages are output by the macro.

- 1 If there is a command error in the macro or if the user does not have the TRUSTED privilege the following message is displayed:
ERROR IN MACRO DAFAM: MACRO ABANDONED
- 2 When there are more than 15 report parameters specified the macro displays the message:
TOO MANY PARAMETERS SPECIFIED: FIRST *n* ACTIONED
and the first 15 report parameters are utilized
- 3 If no acceptable report parameters are presented to the macro the following message is displayed:
NO ACCEPTABLE REPORT PARAMS SPECIFIED

- 4 If the program counts 7000 cells in a chain in the System Control Area it will halt MX and the following message is displayed:
LOOP IN SCA?
- 5 If the program cannot obtain sufficient store to produce a map the program will halt ST and the macro will display the following message:
INSUFFICIENT STORE AVAILABLE FOR A MAP
- 6 If the program terminates in an unanticipated manner the following message is displayed:
XPJK HALTED UNEXPECTEDLY

TITLE

DAPRINT

FUNCTION

To print from direct access exofiles, filestore files, or storage units using #XPJL or #XPJQ.

RESTRICTIONS

The user must supply in a filestore file or the job source all the character peripheral input, that is the card or paper tape program parameters required by the program to be run, but the exofiles, filestore files and storage units to be printed are specified in the program parameters only, and no macro parameters are required to specify them. However this means that for filestore direct access files:

- 1 The names specified must be local names relative to the job's current directory
- 2 File generation numbers of 0 or - 1 will be interpreted as described below under *Execution*
- 3 There is no means of specifying a language code
- 4 Workfiles can not be specified

There is no restriction on the number of files printed in one run of the macro, but they must all be exofiles or all filestore files.

Storage unit serial numbers should not, of course, be included in the program parameters where these specify the names of filestore files. However on the printouts of such files a dummy storage unit number of 777777 will be inserted by the program.

The #XPJQ hardware block print facility is available only to users having the TRUSTED privilege.

The macro does not allow for the inclusion of a FIND or LOAD message in the terminating line of the program parameters. Such messages generated by DELETE instructions are not implemented by GEORGE 3 and the macro does not attempt to convert them to GEORGE commands.

FORMAT

DAPRINT *parameter list*

The parameter types in the parameter list are:

CHARACTER PERIPHERAL INPUT

ERROR

LISTING

PROGRAM NAME

FILESTORE FILES INDICATOR

They have no fixed position and if an optional parameter is omitted no null parameter is required in its place. Only one parameter of each type will be effective in any of the macros.

Character peripheral input parameter

The character peripheral input parameter consists of the characters *CR or *TR, optionally followed by a filestore file description. The format is:

**CR file description*

or

**TR file description*

DAPRINT

or

***CR**

or

***TR**

This parameter type is optional and indicates the peripheral type of the program parameters to be read by the program concerned. If a file description is included it indicates that the program is to read its parameters from the file specified. If no file description is given the program will read its parameters from the job source. If this parameter is omitted altogether, the program will read its parameters from the job source as card input.

Error parameter

The error parameter consists of the characters ER followed by a label. The format is:

ER *label*

This parameter type is optional. The parameter specifies a point in the job description to which control should be passed if the macro detects an error in the parameters, if the run of the program concerned fails to reach a satisfactory conclusion, or, in the NOT MOP context, if a command error (such as might be caused by incorrect macro parameters) occurs in the macro.

Note: Like other system macros, this macro uses labels beginning with 9, and to avoid ambiguity the error parameter should not have a label beginning with that character.

Listing parameter

The listing parameter in this macro consists of the characters *LP alone or followed by either a filestore file description or the four characters (PR) and a peripheral property name. The format is:

***LP**

or

LP *file description

or

LP(PR) *property name

This parameter is optional. If it is omitted the macro will CREATE a workfile to receive the line printer output, and, at the end of the run, LISTFILE it to any available line printer, regardless of the print width specified in the program parameters, and ERASE it.

LP *file description indicates that the line printer output is to be written to the filestore file specified. ***LP** alone indicates that the line printer output is to be sent to the monitoring file system.

LP(PR) *property name indicates that a workfile is to be CREATED to receive the line printer output, and, at the end of the run, LISTFILEd to a line printer having the specified property, and ERASEd. This facility is provided to allow the user to ensure that the output is listed to a printer suitable for the print width specified in the program parameters.

Note that only one listing parameter will be effective.

Program name parameter

The program name parameter consists of the four characters of the name of the program to be run. Thus the format is:

XPJL

or

XPJQ

This parameter is mandatory. It specifies which of the two printing programs is to be run by the macro.

Filestore files indicator parameter

The filestore files indicator parameter consists of the characters FILESTORE, of which only the first five are tested. Thus the format is:

FILESTORE

or

FILES

This parameter is optional. If present it indicates that all the files specified in the program parameters are filestore direct access files. If it is omitted all the files so specified will be assumed to be direct access exofiles. This parameter must of course be omitted if an #XPJQ hardware block print is required.

EXECUTION

The macro issues ASSIGN or ONLINE commands for the files to be printed using information derived from the program which is derived in turn from the user's program parameters. (Where those parameters specify an #XPJQ hardware block print for storage unit *nnnnnn* the file ONLINEd will be "SYFLE *nnnnnn*".)

Since for filestore files a file generation number of 0 is not allowed and a file generation number of -1 does *not* mean "the highest file generation number available", when a filestore file is to be printed, if the program parameters specify a file generation number of 0 (whether specifically or by omission), no file generation number will be given in the ASSIGN command issued by the macro, and if the program parameters specify a file generation number of -1, the file generation number given in the ASSIGN command will be -0.

When the macro is used in the NOT MOP context, in all cases where an ASSIGN or ONLINE command for a direct access file or exofile give rise to a command error, the macro uses the program's abandon entry point 27. Thus the line printer message ABANDON ENTRY POINT USED will occur not only when the storage unit specified for an exofile in the program parameters cannot be loaded (as it would in an Executive environment), but also when, for example, an exofile for which no storage unit serial number has been specified is not already on line, or a specified filestore file does not exist. In such cases the command error messages appearing in the job's monitoring file will indicate the cause of the abandonment.

In the MOP context if such an ASSIGN or ONLINE command is in error, a break-in will occur. At this point the user may issue a correct ONLINE or ASSIGN command, for instance if the file name was misspelt in the program parameters, and CONTINUE, or he may QUIT; but he should not attempt to use the abandon entry point.

LIBRARY ENTRANTS ACCESSED

<i>Entrant</i>	<i>Modes of access</i>
:LIB.PROGRAM XPJL	EXECUTE
:LIB.PROGRAM XPJQ	EXECUTE

ERROR MESSAGES

Error messages produced by the programs #XPJL and #XPJQ are described in the specifications of those programs in Chapter 2 of this manual.

If the program being run fails to halt HH or before halting HH produces the message DISPLAY DE, or in the NOT MOP context a command error (such as might be caused by incorrect macro parameters) occurs in the macro, then the macro will DISPLAY the general error message:

ERROR IN MACRO DAPRINT: MACRO ABANDONED

and GO TO the label specified in the Error parameter, if supplied. (Note that if the program deletes itself with a FIND or LOAD message, this is treated as an error situation. See *Restrictions*, page 235.)

DAPRINT

If the macro parameters do not include the program name parameter, the above general message will be preceded by the display:

MACRO PARAMETERS MUST SPECIFY XPJL OR -Q

TITLE

DAUTIL

FUNCTION

To run any one of the three UDAS utilities #XPJC, #XPJJ, #XPJX.

RESTRICTIONS

The user must supply in a filestore file or the job source all the character peripheral input, that is the card or paper tape program parameters required by the program to be run. The functions specified by these parameters may be batched in the usual way, but if a specified storage unit cannot be loaded, no attempt is made to use the abandon entry point 27 to go on to the next function.

The macro does not allow for the inclusion of a FIND or LOAD message in the terminating line of the program parameters. Such messages generated by DELETE instructions are not implemented by GEORGE 3 and the macro does not attempt to convert them to GEORGE commands.

To run #XPJX or to produce #XPJJ storage unit level reports the user must have the TRUSTED privilege.

FORMAT

DAUTIL *parameter list*

The parameter types in the parameter list are:

CHARACTER PERIPHERAL INPUT

LISTING

ERROR

PROGRAM NAME

They have no fixed position and if an optional parameter is omitted no null parameter is required in its place. Only one parameter of each type will be effective in the macro.

Character peripheral input parameter

The character peripheral input parameter consists of the characters *CR or *TR, optionally followed by a filestore file description. The format is:

**CR file description*

or

**TR file description*

or

*CR

or

*TR

This parameter type is optional and indicates the peripheral type of the program parameters to be read by the program concerned. If a file description is included it indicates that the program is to read its parameters from the file specified. If no file description is given the program will read its parameters from the job source. If this parameter is omitted altogether the program will read its parameters from the job source as card input.

DAUTIL

Listing parameter

The listing parameter consists of the characters ***LP** optionally followed by a filestore file description. The format is:

***LP** *file description*

or

***LP**

This parameter type is optional. If a file description is included, it specifies a file to which the program to be run is to write its line printer output. If no file description is given, it indicates that the line printer output is to be sent to the monitoring file system in the **OBJECT** category. In the **MOP** context, this facility should be used with care, since the format and, in most cases, the volume of the line printer output from the programs make it unsuitable for output on a console typewriter. If the listing parameter is omitted altogether the macro will **CREATE** a workfile to receive the line printer output and, at the end of the run, **LISTFILE** it to a line printer and **ERASE** it.

Error parameter

The error parameter consists of the characters **ER** followed by a label. The format is:

ER *label*

This parameter type is optional. The parameter specifies a point in the job description to which control should be passed if the macro detects an error in its parameters, if the run of the program concerned fails to reach a satisfactory conclusion, or, in the **NOT MOP** context, if a command error (such as might be caused by incorrect macro parameters) occurs in the macro.

Note: Like other system macros, this macro uses labels beginning with 9, and to avoid ambiguity the error parameter should not have a label beginning with that character.

Program name parameter

The program name parameter consists of the four characters of the name of the program to be run. The format is:

XPJC

or

XPJJ

or

XPJX

This parameter is mandatory. The parameter specifies which of the three programs is to be loaded and run by the macro.

EXECUTION

The required direct access exofiles are opened (or created) by unanticipated open-mode **PERI** instructions. Note that before creating a file **#XPJX** tries to open a file with the specified details to see if such a file already exists. A successful creation therefore will normally be preceded by the **GEORGE** monitoring file message **FILE (-----) NOT FOUND**.

In cases where the specification of a storage unit serial number in the program parameters is optional, if the required storage unit is not already on-line, the operator will be asked to load it only if its serial number is given in the program parameters, and in an Executive environment.

LIBRARY ENTRANTS ACCESSED

<i>Entrant</i>	<i>Modes of access</i>
:LIB.PROGRAM XPJC	EXECUTE
:LIB.PROGRAM XPJJ	EXECUTE
:LIB.PROGRAM XPJX	EXECUTE

ERROR MESSAGES

Error messages produced by the programs #XPJC, #XPJJ and #XPJX are described in the specification of those programs in Chapter 2 of this manual.

If the program being run fails to halt HH, or before halting produces the message DISPLAY DE, or in the NOT MOP context a command error (such as might be caused by incorrect macro parameters) occurs in the macro, then the macro will DISPLAY the general error message:

ERROR IN MACRO DAUTIL: MACRO ABANDONED

and GO TO the label specified in the error parameter (if supplied). (Note that if the program deletes itself with a FIND or LOAD message, this is treated as an error situation. See *Restrictions*, page 239.)

If the macro parameters do not include the program name parameter, the above general message will be preceded by the display:

MACRO PARAMETERS MUST SPECIFY XPJC/J/X

Note that if a storage unit specified in the program parameters cannot be loaded the program event message sent to the monitoring file storage by GEORGE will be:

ONLINE *DA0 FAIL

XREBJMAC1

TITLE

XREBJMAC1

FUNCTION

To write to a disc filestore file from data contained in one or more card or paper tape files using #XREB or #XREJ.

RESTRICTIONS

The user must supply in a filestore file or the job source the card or paper tape parameters required by the program to be run.

The macro does not allow for switch 3 to be used to terminate the run prematurely, for restrictions on the addition of data to the end of files see the END macro parameter, page 245.

The macro utilizes only off-line files.

FORMAT

XREBJMAC1 *parameter list*

The parameter types in the parameter list are:

PARAMETER INPUT

INPUT FILE

OUTPUT FILE

PRINT

PUNCH

SWITCH

END

SAVE

ERROR

With the exception of the parameter input parameter, the parameters have no fixed position.

Parameter input parameter

The parameter input parameter consists of the characters *CR or *TR optionally followed by a file description. These formats have the following functions:

*CR <i>file description</i>	Denotes that the parameters to #XREJ are being input from the serial filestore file described
*TR <i>file description</i>	Denotes that the parameters to #XREB are being input from the serial filestore file described
*CR	Denotes that the parameters to #XREJ are being input direct from the job source
*TR	Denotes that the parameters to #XREB are being input direct from the job source

The parameter input parameter is mandatory and must be the first entry in the parameter list, otherwise an error will occur. This parameter defines the file containing the parameters to #XREB/J and, if there are no input file parameters, the file containing the data for #XREB/J. In addition, it defines whether #XREB, for paper tape, or #XREJ, for cards, is to be used.

XREBJMAC1

Input file parameter

The input file parameter consists of the characters *TR or *CR, optionally followed by a file description. The formats are:

*TR <i>file description</i>	}	Denotes that data is to be read from the serial filestore described
or		
*CR <i>file description</i>	}	Denotes that the job source is to be used for data input to the appropriate program
*TR		
or		
*CR		

The input file parameters are optional and may be in any position within the parameter list, however, their order is significant. There may be any number of input file parameters, provided that the total number of parameters to the macro does not exceed 23. When input file parameters are present, the parameter input file is released after reading the parameter to #XREB/J, and the first input file is assigned. Subsequent input files are assigned in the order in which they appear in the parameter list until a data terminator is read.

If the input parameter is omitted, it is assumed that all the data required for #XREB/J is contained in the parameter input file. If some, but not all of the records are contained in the parameter input file, this file must also be defined as the list input file, otherwise the parameters to #XREB/J will be reread as data (see *Examples of parameters to XREBJMAC1*, page 246).

Output file parameter

The output file parameter has two formats:

*ED <i>file description</i>	The disc filestore file described is assumed to be in the correct format for #XREB/J to write to it, that is, bucket headers have been written and file index areas allocated, if required
or	
INIT (<i>file description</i> , KWORDS <i>m</i> , BUCK <i>n</i> , 1ST <i>x</i> , 2ND <i>y</i> , 3RD <i>z</i>)	The disc filestore file described in the parameter is used. The system macro INITIALIZE is called to write bucket headers and allocate file index areas, if required. The file description must be the first parameter within the parentheses. The parameters KWORDS <i>m</i> and BUCK <i>n</i> must be present and may occupy any position within the parentheses. They must be identical to those used in the CREATE command that set up this file. The other three parameters need be used only when an indexed sequential file is being written. They define the number of buckets to be used for each level of index

The output file parameter is compulsory but may occupy any position within the parameter list. For both forms of the parameter the disc file must previously have been created using a CREATE command (see Note 3, page 246).

Print parameter

The print parameter consists of the characters *LP optionally followed by a file description. The formats are:

*LP <i>file description</i>	The line printer output will be written to the filestore file described
*LP	The line printer output will be written to the job's monitoring file

The print parameter is optional but if present may occupy any position within the parameter list. If the print parameter is omitted, the line printer output will be written to a workfile which will be listed on a line printer.

Punch parameter

The punch parameter consists of the characters *TP or *CP, optionally followed by a file description. The formats are:

<i>*TP file description</i>	The paper tape output from #XREB will be written to the filestore file described
<i>*CP file description</i>	The card output from #XREJ will be written to the filestore file described
<i>*TP</i>	The paper tape output from #XREB will be sent to the job's monitoring file
<i>*CP</i>	The card output from #XREJ will be sent to the job's monitoring file

The punch parameter is optional but if present may occupy any position in the parameter list. If the punch parameter is omitted and paper tape or card punching is required by the program, then the output will be written to a workfile which will be listed on the appropriate punch.

Switch parameter

The switch parameter allows switches to be set at the start of the program. (For a description of the function of each switch, refer to the specification of #XREB/J in Chapter 2 of this manual). The switch parameter consists of the characters SW followed by a string of decimal numbers within parentheses. The numbers are separated by commas. The format is:

SW (*a, b, . . . , c*) This has the effect of setting on switches *a, b, . . . , c*

The switch parameter is optional but if present may occupy any position within the parameter list.

End parameter

The end parameter consists of the characters END. This parameter is only required when the parameters to #XREB/J indicate that the disc file is to be opened at the end of the file, in which case it is compulsory. Only serial disc filestore files may be opened at the end of the file.

The end parameter may be in any position in the parameter list.

Save parameter

The save parameter consists of the character SAVE followed by a file description. The format is:

SAVE file description

The save parameter is optional but is present may be in any position in the parameter list. This parameter describes the file to which a store dump will be written should an unexpected event occur during the running of #XREB/J. If this parameter is omitted, no store dump will be taken.

Error parameter

The error parameter consists of the character ER followed by a label. The format is:

ER label

The error parameter is optional but if present may be in any position in the parameter list. If present, this parameter defines a label to which control is to be passed in the event of a failure during the running of #XREB/J or an error in the macro. If this parameter is omitted, the next command after the macro will be obeyed if the run has failed or an error is detected in the macro, unless the user has specified command error action at a higher level.

EXECUTION

Before loading #XREB (for paper tape) or #XREJ (for cards) depending on the format of the first parameter to XREBJMAC1, the program #XK68 may be loaded to write bucket headers and file index areas to the disc filestore as determined by the output file parameter. The parameters to #XREB/J are read from the parameter input file. The disc filestore file is ASSIGNED to the program with the qualifier WRITE unless the end parameter is present in which case the qualifier OFFSET is used, so the file must have the traps required to allow the user access in the appropriate mode. The data is read from the input files in turn or, if there is no input file parameter, from the parameter input file, until a data terminator is found.

XREBJMAC1

When opening the disc file, the system macro XREBJMAC1 ignores the program halt, G4, which indicates that the fixed length record indicator is not set. All other program error halts cause the macro to be abandoned.

LIBRARY ENTRANTS USED

<i>Entrant</i>	<i>Mode of access</i>
:LIB.PROGRAM XK68	EXECUTE
:LIB.PROGRAM XREB	EXECUTE, READ
:LIB.PROGRAM XREJ	EXECUTE, READ

The last two entrants should be overlaid in a disc type filestore file.

ERROR MESSAGES

Details of most of the error messages may be found in the specification of #XREB/J in Chapter 2 of this manual. The error messages from this system macro INITIALIZE may be found in the ICL manual *Operating Systems GEORGE 3 and 4* (Edition 5, TP4345).

EXAMPLES OF PARAMETERS TO XREBJMAC1

- 1 XREBJMAC1 *CR, INIT(XREJOUT, KWOR10, BUCK1, 1ST1)

The job source is to be used for input of parameters and data to #XREJ. This disc filestore file XREJOUT is to have bucket headers written and first level file index area allocated

- 2 XREBJMAC1 *TR PARAM, *ED!, *TR DATA

The parameters to #XREB are to be read from the serial filestore file, PARAM, and the data from the serial filestore file, DATA. The disc filestore file already has bucket headers and file index areas

- 3 XREBJMAC1 *CR XREJIN, *CR XREJIN, *CR DATA2, *ED XREJOUT, END

The parameters to #XREJ and the first section of data are to be read from the serial filestore file XREJIN. The remainder of the data is to be read from the serial filestore file DATA2. The disc filestore file already has bucket headers and file index areas and is to be opened at the end of the file

NOTES ON THE XREBJMAC1 SYSTEM MACRO

- 1 Apart from the line printer output, only one medium may be used for basic peripheral input and output. For example, if a paper tape file is used in the parameter input parameter no card files may be specified
- 2 If it is required to read data from the job source, this data must be the last section of input or the only input file, and must contain an appropriate data terminator
- 3 Due to a discrepancy between the disc control area under GEORGE 3 and under Executive, if it is required to write to a serial disc file, the qualifier FIXED should not be used when the disc filestore file is CREATED

XRPMEMAC1

TITLE

XRPMEMAC1

FUNCTION

To print reports of disc or magnetic tape filestore files using #XRPE or #XRPM.

RESTRICTIONS

The user must supply in a filestore file or the job source all the character peripheral input, that is the card or paper tape program parameters required by the program to be run.

The macro utilizes only off-line files.

FORMAT

XRPMEMAC1 *parameter list*

The parameter types in the parameter list are:

PARAMETER INPUT

INPUT FILE

PRINT

PAPER LINE-UP

PROPERTY

SAVE

ERROR

They have no fixed position.

Parameter input parameter

The parameter input parameter consists of the characters *CR or *TR, optionally followed by a file description. The formats are:

*CR or *TR	Denotes that the parameters to #XRPE/M are to be input from the job source in the form of cards or paper tape respectively

*CR <i>file description</i>	Denotes that the parameters to #XRPE/M are to be input from the serial filestore described

or

*TR *file description*

The parameter input parameter is mandatory and maybe in any position within the parameter list. It describes the source of the parameters to #XRPE/M.

Input file parameter

The input file parameter consists of the character *ED or *MT followed by a file description. The formats are:

*ED <i>file description</i>	Denotes that the direct access type file described is to be printed using #XRPE

or

*MT <i>file description</i>	Denotes that the magnetic tape type file described is to be printed using #XRPM

XRPMEMAC1

The input file parameter is mandatory and may be in any position within the parameter list. It describes the disc or magnetic tape filestore file from which data is to be printed. This parameter also defines which program is to be used, that is #XRPE for disc type filestore files or #XRPM for magnetic tape type filestore files.

Print parameter

The print parameter consists of the characters *LP, optionally followed by a file description. The formats are:

*LP *file description* Denotes that the report will be sent to the filestore file described

or

*LP Denotes that the report will be sent to the job's monitoring file

The print parameter is optional and may be in any position within the parameter list. The parameter specifies whether the file report is to be sent to the filestore file described or the job's monitoring file. If the print parameter is omitted the report will be written to a workfile which will be listed on a line printer. In this case the paper line-up and/or property parameters may also be used.

Paper line-up parameter

The paper line-up parameter consists of the characters SETUP.

This parameter is optional and has no effect when the report output from #XPPE/M is written to a filestore file or to the job's monitoring file. If the parameter is present and the report is written to a workfile, before the workfile is listed, the line printer paper will be lined up using the SETUP procedure described in the ICL 1900 Series manual *Operating Systems GEORGE 3 and 4* (Edition 5, TP4345). For this purpose a second workfile will be created containing a test pattern of six lines of the form:

print positions	1	to	9	contain 0
print positions	10	to	19	contain 1
print positions	20	to	29	contain 2
.....				
print positions	100	to	109	contain 0
print positions	110	to	119	contain 1
print position	120	contains	2	

Note that, for this test pattern only, if a 160 print position line printer is used, the last 40 print positions will not be filled.

Property parameter

The format of the property parameter is identical to that used in the LISTFILE command described in the manual *Operating Systems GEORGE 3 and 4*.

The property parameter is optional and has no effect when the report output from #XRPE/M is written to a filestore or to the job's monitoring file. If the parameter is present and the report is written to a workfile when the workfile is LISTed, this parameter will be included in the command. Thus it is possible, for example, to ensure that a 160 print position line printer is available. The effect of the PROPERTY command is as described in the manual *Operating Systems GEORGE 3 and 4*.

Save parameter

The save parameter consists of the character SAVE followed by a file description. The format is:

SAVE *file description*

The save parameter is optional. The parameter describes the file to which a store dump will be written should an unexpected event occur during the running of #XRPE/M. If the parameter is omitted, no store dump will be taken.

Error parameter

The error parameter consists of the characters ER followed by a label. The format is:

ER *label*

The error parameter is optional. If present, the parameter defines a label to which control is to be passed in the event of either a failure during the running of #XRPE/M or an error in the macro. If the parameter is omitted, the next command after the macro will be obeyed, unless the user has specified command error action at a higher level.

EXECUTION

Depending upon the format of the INPUT FILE parameter, either #XRPE (for discs) or #XRPM (for magnetic tape) is loaded and entered. The report of the filestore file described in this parameter is output to the file described by the PRINT parameter.

LIBRARY ENTRANTS USED

<i>Entrant</i>	<i>Mode of access</i>
:LIB.PROGRAM XRPE	EXECUTE
:LIB.PROGRAM XRPM	EXECUTE

ERROR MESSAGES

Error messages produced by the program #XRPE are described in the specifications of that program in Chapter 2 of this manual.

EXAMPLES

1 XRPMEMAC1 *CR, *ED XRPEDS

The job source is to be used as a card parameter input to #XRPE. The data from the exchangeable disc filestore file, XRPEDS, will be output to a workfile and then listed on a line printer

2 XRPMEMAC1 *CR, *ED XRPEDS, SETUP, PROPERTY PRINT 160

This example is the same as example 1 except that before listing the workfile on a line printer with the property 'PRINT 160', the paper will be lined up

Notes:

- 1 There is no provision in XRPMEMAC1 to dump the program with all the parameter information recorded, or to request printing in Extended Repertoire Character Set
- 2 Since page overflow can never be set under the GEORGE 3 Operating System, there are only two criteria on which headings may be printed, on entry only, or on entry and thereafter on expiry of a line count (refer to the section on Control Statement A in the specifications of #XRPE/M)

Chapter 4 S-trusted and systems utility programs

NAME

#XPJ1

TITLE

Write direct access identifiers

HARDWARE REQUIREMENT

Up to 5440 words of store (6464 words if using EDS200)

1 paper tape reader or card reader or console typewriter

1 line printer

1 direct access storage unit (Twin EDS, EDS, DEDS, EDS 30, EDS60, or EDS200)

DESCRIPTION**General**

This program writes identifiers and test data on each track of a storage unit in succession. It condemns all blocks which have a parity error, and writes a skeleton system control area. This program will normally be used only on virgin storage units. In order to complete the setting up of the storage unit the reset system control area utility (#XPJ2) must subsequently be run. Where it is later necessary to reset the system control area only #XPJ2 need normally be run.

The system control area written by #XPJ1 is limited to block 1 and consists of the file index and parameter cells for the two system files (SYFLE and ICLSCAFLE) and their corresponding file area cells. The SCRATCH INHIBIT and EXCHANGEABLE bits will both be set.

If parameter field 5 (block address of flaw index) is specified, then the flaw index is written to this block. If the block address of the flaw index is not specified, the flaw index will be written to the following blocks:

Twin EDS	Block 8
EDS	Blocks 16
EDS 30	Block 64
DEDS	Block 16
EDS 60	Block 64
EDS200	Block 200

The standard number of flaw blocks assigned is as follows:

Twin EDS	8
EDS	24
EDS 30	96
DEDS	96
EDS 60	96
EDS 200	40

If the storage unit being initiated contains more flaws than specified above, as many additional flaw blocks as necessary will be assigned, up to a maximum of 120 (960 blocks plus 8 flaw indexes for EDS 200), but no free flaw areas will be assigned. In order to create free flaw areas in this case, function 2 of #XPJ2 (create additional system control area) should be used. The program will, in all cases, report on the number of free flaw areas.

#XPJ1

If **Executive** is using the storage unit for overlays then parameter field 5, if present, will be ignored and the creation of additional flaw blocks (if required) will be prevented from overwriting the overlay area; further, any flaws in the overlay area will be ignored.

It should be noted that no free file index and parameter cells, free file area cells or file area cells describing free areas are set up by this program.

Input

Input to the program consists of one direct access storage unit to which identifiers are to be written. The parameters to be input to the program are described in the section *Control*.

Output

Output from the program consists of one direct access storage unit with block identifiers and a skeleton system control area.

Line printer output shows the action taken on the parameters and, where appropriate, a list of flaws. The line printer messages relating to parameters are given in the section *Line printer messages*, page 255. The list of flaws is self-explanatory.

CONTROL

This section describes the one parameter type that may be input to #XPJ1. Further details of parameter formats are given in the section *Parameter formats* in Chapter 1, page 2.

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJ1) and parameter type	XPJ1	5 (fixed)
2	Storage unit type	TD, ED, DD, MD or HE where TD = Twin EDS ED = EDS DD = DEDS MD = EDS 30 or EDS 60 HE = EDS 200	2 (fixed)
3	Storage unit serial number	000001 to 777777 (in octal characters)	6 (fixed)
4	Number of cylinders	100, 200 or 203, 406, 160 or 822 Note: For Twin EDS and EDS the number of cylinders will either be 100, 200 or 203; for EDS 30 it will be 203; for EDS 60 it will be 406; for DEDS it will be 160; and for EDS 200 it will be 822	3 (fixed)
5	Block address of flaw index (a multiple of 8)	8 to 128 (8 to 512 for EDS 200) Note: If omitted the block address specified in the section <i>General</i> , page 251, will be taken	3

Each parameter field must be terminated by a comma. The parameter set must be followed by an asterisk (*).

A card or line containing four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a **Load** or **FInd** message. Further details are given in the section *Parameter formats* in Chapter 1.

Examples

Identifiers are to be written to EDS units 000157 and 763214, and to the EDS 30 unit 111111. The number of the EDS units are 100 and 200 respectively. The flaw index is to be written to block 72 of the EDS 30 unit and to the cylinders on block 16 of unit 763214.

The parameters to perform these functions would take the following form (if input on paper tape):

XPJ11,ED,000157,100*↓

XPJ11,MD,111111,203,72,*↓

XPJ11,ED,763214,200,16,*↓

where ↓ indicates a newline character.

OPERATING

Trusted status

The program has S-trusted status.

Executive priority

The program as supplied has a priority of 75.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
TR0 or CR0 or console typewriter	Input of parameters	The card or tape reader is assigned at the beginning of the run and released when the terminating parameter has been read
LP0	Output of reports and flaws	Used throughout run
DA0	Function unit	Used throughout the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJ1 (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required storage unit	
3 If the storage unit does not have a standard header, to set up the serial number, input: (see the section <i>The SD message</i> in Chapter 1, page 5)	SD w x y z
4 If the parameters are on paper tape or cards, load the parameters in the appropriate reader	
5 To activate the program:	
(a) For parameters on paper tape, input:	GO #XPJ1 20
(b) For parameters on cards, input:	GO #XPJ1 21

#XPJ1

	<i>Narrative</i>	<i>Console message</i>
	(c) For console typewriter parameters, input:	GO #XPJ1 22
6	If GO #XPJ1 22 was input, the program will output: Input: (see the section <i>The MS message</i> in Chapter 1, page 5)	DISPLAY:- MS? MS #XPJ1 = <i>parameter line</i>
7	When the run is complete the program will output one of the following:	
	(a) If no delete message was input in an error free run:	HALTED:- HH
	(b) If a delete message was input and a successful run has occurred:	DISPLAY:- HH DELETED:- <i>user message</i>
	(c) If there was an error in the input parameters:	HALTED:- ZZ

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJ1 during execution and loading. Prior to entry into #XPJ1 it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJ1 or (b) Abandon the run
2	HALTED:- EXEC NOT UDAS	Incorrect Executive in use	Abandon the run
3	HALTED:- FAILURE TO OPEN SYSTEM FILE	Another program has a file open on the unit	Either (a) Delete the other program or (b) Abandon the run
4	HALTED:- GO 20, 21 OR 22	GO 23-29 input on the console	To read the parameters, input: GO #XPJ1 <i>xx</i> where <i>xx</i> is either 20, 21 or 22
5	HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJ1 or (b) Abandon the run

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
6	DISPLAY:- PL	A paper low condition has been detected	Load more paper and re-allocate the printer
7	HALTED:- SL	There is insufficient store available	Either (a) Make more store available and input: GO #XPJ1 or (b) Abandon the run
8	HALTED:- TR	There is no tape reader available	Either (a) Make a tape reader available and input: GO #XPJ1 or (b) Abandon the run
9	HALTED:- UNIT NOW xxx CYLINDERS?	The unit has a valid control header specifying a different unit shape to those in the parameters. xxx is the number of cylinders specified in parameter field 4	Either (a) To implement the parameters input: GO #XPJ1 or (b) To read the next set of parameters input: GO #XPJ1 xx where xx is either 20, 21 or 22
10	HALTED:- xx	An error has been detected when loading #XPJ1 be means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in <i>FIND A BINARY PROGRAM in Compiling Systems</i>

LINE PRINTER MESSAGES

When #XPJ1 is first entered, the program identifier (name and mark number), the data and, if possible, the time, are output on the line printer. If any of the messages given below are output on the line printer, the message HALTED:- ZZ will be output on the console at the end of the run. With the first four messages given below, the parameter line will be printed with an upward pointing arrow beneath the parameter in error.

	<i>Message</i>	<i>Reason</i>
1	INVALID CHARACTER	The character is not of the expected type; for example, a letter instead of a number
2	PARAMETER TOO LONG	Either there are too many characters in the parameter or a comma has been omitted
3	PARAMETER TERMINATION ERROR	The message after the batch terminator (****) does not begin with the characters LO or FI
4	INVALID NO. OF CYLINDERS	The value of parameter field 4 is incompatible with parameter field 2

In addition to the messages given below relating to parameters, the following messages may be output.

	<i>Message</i>	<i>Reason</i>
5	NO IDENTIFIER ON TRACK n	Executive has failed to write identifiers on track n (EDS and Twin EDS only). The run will be abandoned on the storage unit when the flaw area has become full
6	FLAW AREA FULL	There are at least 121 flawed blocks on the disc (968 flawed blocks on the EDS 200) and so the disc has been abandoned with any identifiers written and possible zeros in some blocks
7	STORAGE UNIT IS NOT OF TYPE SPECIFIED	The unit has a valid control header but the number of blocks per track does not match those for the storage unit type specified in the parameters. The run will be abandoned for this storage unit
8	BLOCK 0 FLAW	A flaw has been detected in block 0. The unit is unusable
9	BLOCK 1 FLAW	A flaw has been detected in block 1. The unit is unusable
10	NO ROOM FOR FLAWS	Additional flaw free blocks cannot be obtained because Executive is using overlays from the storage unit
11	OVERLAYS ON UNIT: FIELD 5 NOT ALLOWED	If Executive is using overlays from the storage unit the flaw area must be in the area defined in section <i>General</i> , page 251
12	UNABLE TO ERASE BLOCK n	On EDS 30 or 60 block n is faulty but cannot be erased and entered in the flaw area. The reason why block n cannot be erased or entered in the flaw area is that block $n-1$ was also faulty. Block $n-1$ will have been erased and entered in the flaw area. In the event of a number of consecutive blocks being faulty only the first faulty block will be erased and entered in the flaw area and each of the remaining faulty blocks will cause an UNABLE TO ERASE BLOCK n message to be output. To overcome the problem of a program accessing the faulty but not erased block(s) a dummy file should be allocated over the offending area(s). If this is the only LP error message printed during the run (irrespective of how many time this same message is printed) then HALTED:- ZZ will not be output at the end of the run

NAME

#XPJ2

TITLE

Reset system control area

HARDWARE REQUIREMENT

Up to 7488 words of store

1 paper tape reader or card reader or console typewriter

1 line printer

1 direct access storage unit

DESCRIPTION**General**

This program can be used to perform three functions on a direct access storage unit:

RESET SYSTEM CONTROL AREA

This function is used to complete the initialization of a storage unit on which the write identifiers utility (#XPJ1) has been run, or to re-initialize a corrupt storage unit. The control header and flaw index must be valid; that is, the red tape record of these blocks must be correct. If they are not valid the write identifiers utility (XPJ1) must be used.

CREATE ADDITIONAL SYSTEM CONTROL AREA

This function creates an additional system control area according to the parameters input. All multi-cylinder system control areas created must be full width.

CHANGE STATE OF FLAGS

This function is used to set the EXCHANGEABLE and SCRATCH INHIBIT bits according to the parameters input.

Input

Input to the program consists of one direct access storage unit which is to have its system control area reset. The parameters to be input to the program are described in the section *Control*.

Output

Output from the program consists of one direct access storage unit with its system control area reset according to the parameters.

Line printer output is a report showing the action taken on the parameters. Details are given in the section *Line printer messages*.

CONTROL

This section describes the one parameter type that is to be input to #XPJ2 to perform the three functions described in the section *General*. Further details of parameter formats are given in the section *Parameter formats* in Chapter 1, page 2.

#XPJ2

Function 1: reset system control area

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJ2) and parameter type	XPJ21	5 (fixed)
2	Function identifier	1	1 (fixed)
3	Storage unit serial number	000001 to 777777 (in octal characters)	6 (fixed)
4	Number of overlay blocks (a multiple of 8)	0, S or 8 to 992 Note: 0 should be specified for units from which Executive is not required to overlay. With Twin EDS, 80 blocks should be specified if the overlays to be held are those of an Executive which does not include GEORGE 1S. S should normally be used in all other cases and will generate the following values: Twin EDS = 176 EDS = 280 Others = 256	4
*5	Number of blocks containing the file index and parameter cells	1 to 999	3
*6	Number of blocks containing file area cells	1 to 999	3

*If these fields are omitted the available area will be divided equally between file index and parameters and file area cells. If an old block remains, it will be set up as a file index and parameters block. If there are flaws in this area the program will omit the flawed blocks and the number of blocks written will be less than specified. If either of these fields is present, both must be specified and they must add up to the number of blocks available between block 0 and the flaw index.

Each parameter field must be terminated by a comma. If the optional field 5 and 6 are omitted, the succeeding comma(s) must also be omitted. The parameter set must be followed by an asterisk(*).

When run under Single-Programming Overlaid Executives, XPJ2 effectively removes the overlays from any storage unit processed by it, though in the case of the current overlay unit being reset, the overlays will remain available to the Executive on that unit (and can be copied to another unit) until the Executive is reloaded or the unit is taken off-line, that is, the loss will only become apparent when the Executive next searches for overlays on the reset unit.

When resetting the system control area, the existing flaw index and flaw area blocks are preserved; it is therefore essential to ensure that the SCA does not cover any part of an additional flaw area created by a previous run of #XPJ2. No check is made for this situation and doubly-defined areas could occur in ICLSCAFILE and on the unallocated area. If this situation does arise then #XPJ1 must be used first.

Function 2: create additional system control area

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJ2) and parameter type	XPJ21	5 (fixed)

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
2 Function identifier	2	1 (fixed)
3 Storage unit serial number	000001 to 777777	6 (fixed)
*4 Number of blocks containing file index and parameters cells	0 to 999 Note: The sum of this field and field 5 must be a multiple of 8	3
*5 Number of blocks containing file area cells	0 to 999 Note: The sum of this field and field 4 must be a multiple of 8	3
*6 Number of flaw blocks (a multiple of 8)	0 to 120	3
**7 Start cylinder number	0 to (number of cylinders on unit) - 1. See note <i>End block addresses</i> , below	4
**8 Start block number	0 to (number of blocks per cylinder on unit) - 1. See note <i>End block addresses</i> , below. Note: If the system control area to be created is less than one cylinder in length, this must be a multiple of 8. If it is to be more than one cylinder in length, this must be 0	4

*If the total number of system control area blocks requested is greater than one cylinder then it must be a multiple of a cylinder.

** If one of these fields is specified then both must be present. If they are omitted then the system control area is created in a suitable free area.

Each parameter field must be terminated by a comma. There parameter set must be followed by an asterisk (*). Each additional system control area must be defined by a full parameter set and must be followed by an asterisk.

To avoid double-defined areas in ICLSCAFILE when using Function 1, it is recommended that additional flaw area blocks be allocated only when the flaw index becomes full and then not in the low numbered cylinders.

Note: End block addresses. In the SCA file index, file area and flaw index cells use word addresses for chaining, the maximum address is therefore 8,388,607 which corresponds to a block address of 65535. For this reason, no additional SCA can be created with an end block address of 65535 or greater. (This only applies to EDS 60 and EDS 200.) Block 65535 corresponds to cylinder 221, block 119 on EDS 60, and to cylinder 138, block 399 on EDS 200.

Function 3: change state of flags

<i>Parameter fields</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJ2) and parameter name	XPJ21	5 (fixed)
2 Function identifier	3	1 (fixed)
3 Storage unit serial number	000001 to 777777 (in octal characters)	6 (fixed)

#XPJ2

	<i>Parameter fields</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
4	Exchangeable state	0 or 1 where 0 = non-exchangeable 1 = exchangeable	1 (fixed)
5	Scratch inhibit	0 or 1 where 0 = scratch permit 1 = scratch inhibit	1 (fixed)

Each parameter field must be terminated by a comma. The parameter set must be followed by an asterisk (*).

A card or line containing four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a LOad or FInd message. Further details are given in the section *Parameter formats* in Chapter 1.

Examples

1 FUNCTION 1

The system control areas of units 000162 and 000163 are to be reset. Unit 000162 is to be reset with standard parameters while unit 000163 is to be reset with no overlay area and 9 blocks of file index cells and 6 blocks of file area cells.

The parameters to perform these functions would take the following form (if input on paper tape):

XPJ21,1,000162,S,*↓

XPJ21,1,000163,0,9,6,*↓

****↓

where ↓ indicates a newline character

2 FUNCTION 2

An additional system control area is to be created on unit 000165. Eight blocks each of file index, file area and flaw area cells are required. The system control area is to be created on cylinder 10, starting at block 0.

The parameters to perform this function would take the following form (if input on cards):

card 1 XPJ21,2,000165,8,8,8,10,0,*

card 2 ****

3 FUNCTION 3

The flags of unit 000166 are to be set to exchangeable and scratch inhibit.

The parameters required to perform this function would take the following form (if input on paper tape):

XPJ21,3,000166,1,1,*↓

****↓

where ↓ indicates a newline character

OPERATING**Trusted status**

The program has S-trusted status.

Executive priority

The program as supplied has a priority of 75.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0 or console typewriter	Input of parameters	The card or tape reader is assigned at the beginning of the run and released when the terminating parameter has been read
LP0	Output of reports	Used throughout the run
DA0	Function unit	Used throughout the run

Running under Executive alone**OPERATING INSTRUCTIONS**

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJ2 (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required storage unit	
3 If the parameters are on paper tape or cards, load the parameters in the appropriate reader	
4 To activate the program:	
(a) For parameters on paper tape, input:	GO #XPJ2 20
(b) For parameters on cards, input:	GO #XPJ2 21
(c) For console typewriter parameters, input:	GO #XPJ2 22
5 If GO #XPJ2 22 was input, the program will output: Input: (See the section <i>The MS message</i> in Chapter 1)	DISPLAY:- MS? MS #XPJ2 = <i>parameter line</i>
6 When the run is complete the program will output one of the following:	
(a) If no delete message was input in an error free run:	HALTED:- HH
(b) If a delete message was input and a successful run has occurred:	DISPLAY:- HH DELETED:- <i>user message</i>
(c) If there was an error in the run:	HALTED:- ZZ

#XPJ2

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJ2 during execution and loading. Prior to entry into #XPJ2 it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJ2 or (b) Abandon the run
2	HALTED:- EXEC NOT UDAS	Executive currently in use is incompatible with Unified Direct Access Standards	Run under UDAS Executive
3	HALTED:- INCORRECT ENTRY	GO 23-29 input on the console	To read the parameters, input: GO #XPJ2xx where xx is either 20, 21 or 22
4	HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJ2 or (b) Abandon the run
5	DISPLAY:- PL	A paper low condition has been detected	Load more paper and re-allocate the printer
6	HALTED:- SL	There is insufficient store available	Either (a) Make more store available and input: GO #XPJ2 or (b) Abandon the run
7	HALTED:- TR	There is no tape reader available	Either (a) Make a tape reader available and input: GO #XPJ2 or (b) Abandon the run
8	HALTED:- xx	An error has been detected when loading #XPJ2 by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

LINE PRINTER MESSAGES

When #XPJ2 is first entered, the program identifier (name and mark number) the date and, if possible, the time, are output on the line printer.

If any of the line printer messages given below are output, then the message HALTED:- ZZ will be output on the console at the end of the run.

The line printer messages output fall into four groups:

Group 1

Every parameter line is printed on the line printer. If the parameter line is not actionable because of format errors, an upward pointing arrow will be printed beneath the parameter in error. One of the messages below will also be output to indicate what the actual error is. No action will be taken on this parameter line and the program will read in the next line of parameters.

<i>Message</i>	<i>Reason and/or action</i>
1 ERROR: ERROR IN FUNCTION TYPE	The function field does not contain 1, 2 or 3
2 ERROR: INVALID CHARACTER	The parameter line contains an unexpected character; for example, 9 in a serial number or a letter in the number of blocks
3 ERROR: INVALID PARAMETER TYPE	The fifth character in parameter field 1 is not 1
4 ERROR: INVALID TERMINATOR MESSAGE	The first two characters after the batch terminator (****) are neither LO or FI
5 ERROR: NAME INCONSISTENT WITH PROGRAM	The first four characters of parameter field 1 were not read as XPJ2
6 ERROR: NOT 6 DIGIT SERIAL NUMBER	The unit serial number does not consist of six digits
7 ERROR: FIELD TOO LONG	More characters than the permitted maximum for the field concerned have been read as a single parameter field. (This is often caused by an omitted comma)
8 ERROR: TERMINATOR EXPECTED	The comma terminating the last parameter field in a line has not been followed by an asterisk
9 ERROR: LINE INCOMPLETE	End of record has been unexpectedly found while scanning a parameter field
10 ERROR: VALUE OUT OF RANGE	The value of the field concerned is outside the range specified in the section <i>Control</i>
11 ERROR: FIELD MISSING	A field of zero length has been encountered
12 ERROR: ZERO LENGTH ADDITIONAL SCA	Applies to function2 only. Parameter fields 4, 5 and 6 are all zero

Group 2

Group 2 messages are those that are output because of certain invalid combinations of parameters due either to the values specified in the parameters or to the state of the unit.

<i>Message</i>	<i>Reason and/or action</i>
13 ERROR IN FIP/FA REQUEST. TOTAL = xxx	Applies to function 1 only. The number of file index and parameter blocks and file area blocks requested does not equal the number available between block 0 and the flaw index. xxx is the number of blocks available
14 ERROR IN OVERLAY REQUEST	Applies to function 1 only. The number of overlay blocks requested is not a multiple of 8

#XPJ2

<i>Message</i>	<i>Reason and/or action</i>
15 ERROR: NEED FA BLOCKS	Applies to function 1 only. Either because of flaws of for some other reason the program is unable to place any file area blocks. The request should be altered to give more blocks of file area cells
16 ERROR: NEED FIP/FA REQUEST	Applies to function 1 only. There is no flaw index to suggest the limit of the original system control area and it is necessary to specify the size
17 USED O. AREA LARGER THAN REQUESTED. USED AREA = xxx	Applies to function 1 only. The number of overlay blocks requested is not as large as the area Executive is currently using to overlay from. xxx is the size of the current Executive overlay area
18 BLOCK REQ. NOT MULT OF CYL. LENGTH	Applies to function 2 only. The total number of system control area blocks requested is greater than the number than can be held in one cylinder but does not constitute a multiple of the number of blocks that can be held in a cylinder
19 ERROR: FIP/FA REQ NOT MULT OF 8	Applies to function 2 only. The sum of the file index and parameters blocks and file area blocks requested is not a multiple of 8
20 ERROR: FLAW AREA REQ NOT MULT OF 8	Applies to function 2 only. The number of flaw area blocks requested is not a multiple of 8
21 AREA SPECIFIED NOT WITHIN BOUNDS	Applies to function 2 only. Area specified in parameters overflows boundaries of unit or is in a block inaccessible using the addressing conventions of UDAS. See note <i>End block addresses</i> , page 259
22 ERROR: START BLOCK NO.	Applies to function 2 only Either (a) The system control area requested is less than one cylinder in length, but the start block number is not a multiple of 8 or the area extends across a cylinder boundary or (b) A multi-cylinder system control area has been requested, but the start block number is not 0 or (c) The start block number is greater than or equal to the number of blocks per cylinder on the unit
23 ONLY FUNCTION 1 AFTER XPJ1 ALLOWED	Applies to functions 2 and 3 only. Indicates that an attempt has been made to change the state of flags or create additional SCA on a unit which has not been reset after running #XPJ1
24 AREA FOUND NOT WITHIN BOUNDS	Applies to function 2 only. The start cylinder and start block of the additional SCA were not specified. There is no free area large enough to accommodate the additional SCA in the part of the disc accessible using the addressing conventions of UDAS

Group 3

Group 3 messages indicate some kind of failure during the run other than those already specified.

<i>Message</i>	<i>Reason and/or action</i>
25 AREA SPEC. IS NOT FREE	Applies to function 2 only. A specified area has been requested but it is already wholly or partially allocated
26 NO FREE AREA	Applies to function 2 only. All space on the unit is already allocated
27 FREE AREA TOO SMALL	Applies to function 2 only. A specific area has not been specified and the total number of blocks requested is greater than the size of any free area available
28 NED FA BLOCKS	Applies to function 2 only. Insufficient room available to create new file area cells. More file area blocks must be specified
29 NO ROOM FOR NEW FI	Applies to function 2 only. No unflawed block available in new flaw area in which to write new flaw index. Set up the new flaw area elsewhere on the unit. Setting up of new file index and file area cells will be complete at this point
30 NOT ENOUGH EXEC CORE	Limit of files or file areas opened by Executive. Re-run job later
31 TOO MANY FLAW INDEXES	Applies to function 1 only. There are more than 8 flaw indexes on the unit. The program does not allow more than 8 flaw indexes
32 UNIT NOT AVAILABLE	The program is unable to open the system file on the specified unit because another program was using the unit at the time the attempt was made. Re-run job later
33 BLOCK 1 CORRUPT	The unit is corrupt and needs initialising by #XPJ1
34 FLAW INDEX CORRUPT	Applies to function 1 only. The unit is corrupt and needs initialising by #APJ1.
35 WRITE FAIL BLOCK 0	Applies to function 1 only. The unit is corrupt and needs initialising by #XPJ1

Group 4

There is a continuing report on all pairs of blocks that are not used in the areas where the system control area is created because either one is flawed.

<i>Message</i>	<i>Reason and/or action</i>
36 BLOCKS FLAWED: xxx AND NEXT	Action continues after this message unless any of the error conditions given above arises. This report does not cause the program to halt ZZ. xxx is the number of the first block of the pair

NAME

#XPJ7

VERSION

Mark 1

TITLE

Multi-storage unit file reset

HARDWARE REQUIREMENTS

Up to 5000 words of store

1 paper tape reader or card reader or console typewriter

1 line printer

2 or more direct access storage units

DESCRIPTION**General**

This program is used if for some reason one storage unit of a multi-storage unit file has had to have its system control area reset. The program brings the reset unit into agreement with related storage units by performing one of two major functions, depending on whether the reset system control areas contained the file index and parameters cell of the partially lost files:

- 1 The program deallocates those file areas of the non-reset units which belong to the files starting on the reset unit. This is done after first checking that there is no file on the reset unit using area on that cartridge
- 2 The program allocates file areas on the reset cartridge as indicated by the file descriptions in the non-reset units. This is done after checking that the areas are not already allocated

Notes:

- 1 This program is intended for use immediately after the storage unit has been reset by #XPJ2. In order to set up a reset storage unit which is associated with many other units, XPJ7 may be run several times in succession provided that no associated storage unit is acted upon twice. No files or file areas should be allocated on the reset unit until all #XPJ7 runs are complete. For example, there should be no uncontrolled extension from other cartridges

- 2 *#XPJ7 and Twin E.D.S.* Operating problems can arise if an attempt is made to reset a cartridge against more than one associated unit at an installation with only a single twin E.D.S. handler.

The system file of the reset cartridge is kept open to write throughout all the actions arising from one parameter line. Executive will not disconnect a disc whose system file is open to write. The operator is therefore faced with the problem of mounting more than two discs in turn on one pair of spindles, without ever taking anything off-line.

To avoid this problem, the run should be presented as the same number of separate runs of #XPJ7 as there are associated discs, and each run accomplishes just the resetting of the fresh cartridge against one of its associated discs. The two discs required for each run should be mounted before the run is started

Input

Input to the program consists of the reset storage unit and all associated direct access storage units. The parameters to be input to the program are defined in the section *Control*.

#XPJ7

Output

Output from the program consists of correctly cross-referenced storage units. Line printer output is a report showing the action taken on the parameters and the actions performed on disc. Details are given in the section *Line printer messages*.

CONTROL

This section describes the one parameter type that may be input to #XPJ7. Further details of parameter formats are given in the section *Parameter formats* in Chapter 1.

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJ7) and parameter type	XPJ71	5 (fixed)
2	Serial number of reset storage unit	000001 to 777777 (in octal characters)	6 (fixed)
3	Serial number of associated unit. One or more of these parameters are specified on each line. They refer to units associated with the reset unit that was specified as parameter 2 on that line	000001 to 777777 (in octal characters)	6 (fixed)

Each parameter field must be terminated by a comma, and each parameter set terminated by an asterisk (*).

A card or line containing four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a LOad or FInd message. Further details are given in the section *Parameter formats* in Chapter 1.

Example

The unit 701413 has been reset by #XPJ2 and it is required to allocate on it areas described as being on it by files starting on 701417, and to remove from 701417 areas that used to be allocated to files starting on 701413.

The parameters to perform this function would take the following form (if input on cards):

Card 1 XPJ71,701413,701417,*

Card 2 ****

OPERATING

Trusted status

The program has S-trusted status.

Executive priority

The program as supplied has a priority of 75.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
TR0 or CR0 or console typewriter	Input of parameters	Used throughout the run
LP0	Listing action on parameters	Used throughout the run

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
DA0	Specified reset storage unit	Assigned while processing the complete set of units
DA1	Used in turn for each associated storage unit	Assigned while processing each specified associated unit

Running under Executive alone**OPERATING INSTRUCTIONS**

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJ7 (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output of the loading operation was successful	HALTED:- LD
2 Load the required reset storage unit and as many associated storage units as the number of drives available permit	
3 If the parameters are on paper tape or cards, load them into the appropriate reader	
4 To activate the program:	
(a) For parameters on paper tape, input:	GO #XPJ7 20
(b) For parameters on cards, input:	GO #XPJ7 21
(c) For console typewriter parameters, input:	GO #XPJ7 22
5 If GO #XPJ7 22 was input, the program will output: Input: (see the section <i>The MS message</i> in Chapter 1)	DISPLAY:- MS? MS #XPJ7 = <i>parameter line</i>
6 Load any additional storage units as requested by Executive	
7 When the run is complete the program will output one of the following:	
(a) If no delete message was input in an error-free run:	HALTED:- HH
(b) If a delete message was input and a successful run has occurred:	DISPLAY:- HH DELETED:- <i>user message</i>
(c) If there was an error in the run:	HALTED:- ZZ

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJ7 during execution and loading. Prior to entry into #XPJ7 it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

#XPJ7

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJ7 or (b) Abandon the run
2	HALTED:- INCORRECT ENTRY	GO 23-29 input on the console	To read the parameters, input: GO #XPJ7 <i>xx</i> where <i>xx</i> is either 20, 21 or 22
3	HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJ7 or (b) Abandon the run
4	HALTED:- SL	There is insufficient store available	Either (a) Make more store available and input: GO #XPJ7 or (b) Abandon the run
5	HALTED:- TR	There is no tape reader available	Either (a) Make a tape reader available and input: GO #XPJ7 or (b) Abandon the run
6	HALTED:- <i>xx</i>	An error has been detected when loading #XPJ7 by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in <i>FIND A BINARY PROGRAM</i> in <i>Compiling Systems</i>

LINE PRINTER MESSAGES

Error reports

If any line printer error messages are output, the message HALTED:- ZZ will be output on the console at the end of the run. Any parameter errors detected are indicated by an upward pointing arrow beneath the incorrect parameter.

Each message (except messages 2, 5, 11 and 12) when printed is followed by

REST OF THIS SET IGNORED

The program goes on to process the next line of parameters. The above line is printed as there may in some cases be additional (correct) serial numbers beyond the point reached which the program is not examining.

It should be noted that when more than one associated unit is specified and an error is found in a later parameter, processing will have been carried out for any parameter whose terminating comma lies to the left of the indicator arrow.

<i>Message</i>	<i>Reason and/or action</i>
1 ERROR: COMMA EXPECTED	Either there are too many characters in the parameter, or a comma has been omitted
2 ERROR: INSUFFICIENT PARAMETERS	None or only one serial number has been given
3 ERROR: INVALID END OF LINE	Either the line has ended in mid parameter or with no terminating asterisk, or what appeared to be a terminating asterisk is followed by other non-space characters
4 ERROR: INVALID PARAMETER TYPE	The fifth character of any parameter line must be 1
5 ERROR: INVALID TERMINATOR MESSAGE	The message after a line starting with four asterisks must be either a LOAD or FIND message
6 ERROR: NAME INCONSISTENT WITH PROGRAM	The first four characters of any parameter line must be XPJ7
7 ERROR: NOT 6 DIGIT SERIAL NO.	The number of characters in this parameter is less than 6, or the parameter is all zero, which is invalid
8 ERROR: NOT OCTAL CHARACTER	A character in this parameter is not in the range 0 to 7
9 ERROR: SCA ON RESET UNIT FULL	The program has no space in which to manipulate files on the reset unit. Additional file area cells should be created using #XPJ2 and then #XPJ7 should be rerun
10 ERROR: TRANSFER FAILED	A read or write transfer has failed
11 <i>csn</i> NOT AVAILABLE	The disc specified is being used by another program. Consequently the program is unable to open the system file
12 <i>csn</i> NOT ENOUGH EXEC CORE	Limit of files or file areas that can be opened by Executive at this time reached. Rerun job later

Line printer report on actions performed on disc

Note: When the reports described below are output on the line printer, they are preceded by the following headings:

FILE INDEX DISC	AREA DISCS	ACTION BLOCKS
<i>Report</i>		
<i>csn 1</i>	<i>csn 2 start block end block</i>	DELETED

Meaning

An area on an associated unit (*csn 2*) that has previously belonged to a file on the reset unit (*csn 1*) has been made into a free area.

Report

<i>csn 1</i>	<i>csn 2 start block end block</i>	INSERTED

Meaning

An area that had been on the reset disc (*csn 2*) previously belonging to a file on one of the associated units (*csn 1*) has again been reallocated on the reset unit.

#XPJ7

Report

csn 1 *csn 2 start block end block* NOT ATTEMPTED

Meaning

No action has been taken on the area described. This only occurs if there is insufficient store available for Executive to extend its lists for the program. The reset unit must be reset again by #XPJ2.

Report

csn 1 *csn 2 start block end block* USED

Meaning

An area described is already wholly or partially allocated prior to action being attempted. This message can only occur if areas or files have been assigned on the reset unit since the #XPJ2 run, or if #XPJ7 has already been run and inserted areas for the associated cartridge. In either case the reset cartridge must be reset again with #XPJ2.

Appendix 1 Specifications of utilities #XPJD, #XPJH and #XPJJ

Specifications of GEORGE 3 macros DAMAP and DAREPORT

NAME

#XPJD

VERSION

Mark 2

TITLE

Map and summary file report for direct access storage units (see Preface)

HARDWARE REQUIREMENT

For Twin E.D.S 3264 words of store

For E.D.S. 8 3648 words of store

For E.D.S. 30 5056 words of store

For E.D.S. 60 6912 words of store

1 card reader or paper tape reader (optional)

1 line printer

1 or more direct access storage units

DESCRIPTION**General**

The program #XPJD references the system control area of storage unit and produces:

- 1 A map of the storage unit showing storage space allocated to files and storage space unallocated
- 2 A summary file report in the form of a table, in which the files, which are specified in file index cells in the storage unit's system control area, are listed against the symbols by which they are represented in the map, together with data from their file index cells and an indication of whether they have file areas allocated to them on other storage units
- 3 (If appropriate) a table giving details of file areas on the storage unit allocated to files specified in file index cells in the system control areas of other storage units
- 4 A line recording the status of the scratch inhibit bit

A maximum of 100 storage units can be processed in one run of the program. Any storage unit conforming to unified direct access standards can be processed, subject to the following limits:

Minimum number of blocks per cylinder	16
Maximum number of blocks per cylinder	464
Maximum number of cylinders	1000

Input

One or more storage units on which reports are to be made. Input parameters, as described in the section *Control*.

Output

A report printed on the line printer using print positions 1 to 120. Each page is headed by an identifying title giving the date and time at which the run began and, if appropriate, the unit serial number of the storage unit being reported on that page.

#XPJD

XPJD MK2 11/02/73 18/14/77 MAP AND SUMMARY FILE REPORT FOR STORAGE UNIT *15253										PAGE 1
CYL	BLOCK	CYL	BLOCK	CYL	BLOCK	CYL	BLOCK	CYL	BLOCK	
8 24 40 56 72	8 24 40 56 72	8 24 40 56 72	8 24 40 56 72	8 24 40 56 72	8 24 40 56 72	8 24 40 56 72	8 24 40 56 72	8 24 40 56 72	8 24 40 56 72	
U 16 32 48 64	U 16 32 48 64	U 16 32 48 64	U 16 32 48 64	U 16 32 48 64	U 16 32 48 64	U 16 32 48 64	U 16 32 48 64	U 16 32 48 64	U 16 32 48 64	
U00	S S S S S S S S S S	U01	U U U U U U U U U U	U02	H H H H H H H H H H	U03	M M M M M M M M M M	U04	K K K K K K K K K K	
U05	S S S S S S S S S S	U06	U U U U U U U U U U	U07	H H H H H H H H H H	U08	M M M M M M M M M M	U09	F F F F F F F F F F	
U10	S S S S S S S S S S	U11	U U U U U U U U U U	U12	H H H H H H H H H H	U13	M M M M M M M M M M	U14	U U U U U U U U U U	
U15	S S S S S S S S S S	U16	U U U U U U U U U U	U17	H H H H H H H H H H	U18	M M M M M M M M M M	U19	U U U U U U U U U U	
U20	S S S S S S S S S S	U21	U U U U U U U U U U	U22	H H H H H H H H H H	U23	M M M M M M M M M M	U24	U U U U U U U U U U	
U25	S S S S S S S S S S	U26	U U U U U U U U U U	U27	H H H H H H H H H H	U28	M M M M M M M M M M	U29	U U U U U U U U U U	
U30	S S S S S S S S S S	U31	U U U U U U U U U U	U32	H H H H H H H H H H	U33	M M M M M M M M M M	U34	U U U U U U U U U U	
U35	S S S S S S S S S S	U36	U U U U U U U U U U	U37	H H H H H H H H H H	U38	M M M M M M M M M M	U39	U U U U U U U U U U	
U40	S S S S S S S S S S	U41	U U U U U U U U U U	U42	H H H H H H H H H H	U43	M M M M M M M M M M	U44	U U U U U U U U U U	
U45	S S S S S S S S S S	U46	U U U U U U U U U U	U47	H H H H H H H H H H	U48	M M M M M M M M M M	U49	U U U U U U U U U U	
U50	S S S S S S S S S S	U51	U U U U U U U U U U	U52	H H H H H H H H H H	U53	M M M M M M M M M M	U54	U U U U U U U U U U	
U55	S S S S S S S S S S	U56	U U U U U U U U U U	U57	H H H H H H H H H H	U58	M M M M M M M M M M	U59	U U U U U U U U U U	
U60	S S S S S S S S S S	U61	U U U U U U U U U U	U62	H H H H H H H H H H	U63	M M M M M M M M M M	U64	U U U U U U U U U U	
U65	S S S S S S S S S S	U66	U U U U U U U U U U	U67	H H H H H H H H H H	U68	M M M M M M M M M M	U69	U U U U U U U U U U	
U70	S S S S S S S S S S	U71	U U U U U U U U U U	U72	H H H H H H H H H H	U73	M M M M M M M M M M	U74	U U U U U U U U U U	
U75	S S S S S S S S S S	U76	U U U U U U U U U U	U77	H H H H H H H H H H	U78	M M M M M M M M M M	U79	U U U U U U U U U U	
U80	S S S S S S S S S S	U81	U U U U U U U U U U	U82	H H H H H H H H H H	U83	M M M M M M M M M M	U84	U U U U U U U U U U	
U85	S S S S S S S S S S	U86	U U U U U U U U U U	U87	H H H H H H H H H H	U88	M M M M M M M M M M	U89	U U U U U U U U U U	
U90	S S S S S S S S S S	U91	U U U U U U U U U U	U92	H H H H H H H H H H	U93	M M M M M M M M M M	U94	U U U U U U U U U U	
U95	S S S S S S S S S S	U96	U U U U U U U U U U	U97	H H H H H H H H H H	U98	M M M M M M M M M M	U99	U U U U U U U U U U	

XPJD MK2 11/02/73 18/14/77 MAP AND SUMMARY FILE REPORT FOR STORAGE UNIT *15253										PAGE 3								
FILE CODE	FILE IDENTITY	GEN NO	VERS NO	INI CODE	DATE WRITTEN	DATA REIN	FILE REIN	FILE ORGN	VAR INH FL	WDS? /REC	BLKS /BKT	KEY LTH	STRT POSN	NO OF AREAS	TOTAL BKTS	E0F BKT	2ND LEV ST BKT	
S	ICLSCHFILE	0	0	3	24/08/72	U	U	SEQL	YES	YES	0	1	0	0	1	320	1	0
A	PROGRAM S050	3	1	3	07/11/74	U	U	SERL	NO	NO	0	1	0	0	1	720	1	0
B	PROGRAM JJJJ	0	1	3	14/08/74	U	U	SERL	NO	NO	0	1	0	0	1	160	5	0
C	XPB001STRAP	1	0	3	24/08/72	U	U	SERL	NO	NO	0	1	0	0	1	24	12	0
D	XPBUC/COPY	U	2	3	07/11/74	U	U	SEKL	NO	NO	0	1	0	0	6	480	1	0
E	SEMICOMPFILE	3	2	3	11/10/72	U	U	SERL	NO	NO	0	1	0	0	1	560	1	0
F	XPJCFILEAKE	U	1	3	04/10/73	U	U	SERL	YES	NO	126	1	0	0	1	80	1	0
G	XPES OUTPUT	U	0	3	20/10/72	U	U	SERL	YES	NO	126	1	0	0	1	80	6	0
H	AMENDXREH-EJ	7	1	1	30/11/73	U	U	SERL	NO	NO	0	1	0	0	1	880	1	0
I	ICLK6E06MCRU	U	0	3	24/08/72	U	U	SERL	YES	NO	126	1	0	0	1	80	1	0
J	XPES OUTPUT	U	0	3	15/11/74	U	U	SERL	YES	NO	126	1	0	0	1	80	6	0
K	OUTPUT-2	U	0	3	24/08/72	U	U	SERL	NO	NO	4	1	0	0	1	80	6	0
L	GLYNCUSYXMB	3	0	3	24/08/72	U	U	SERL	NO	NO	0	1	0	0	6	720	1	0
M	WKFILE-1	2	0	3	17/01/73	U	U	SERL	YES	NO	128	1	0	0	9	720	1	0
N	AKRATESFILE	U	0	3	30/07/74	U	U	SERL	NO	NO	120	1	0	0	1	160	1	0
O	ISOVSKE	U	0	3	07/09/72	U	U	SEQL	NO	NO	9	1	3	4	1	80	16	0
P	EKCOPYORANGE	U	0	3	07/08/73	U	U	SEKL	YES	NO	126	1	0	0	1	160	1	0
Q	XREJHUCKETST	U	0	3	30/03/73	U	U	SERL	NO	NO	23	4	0	0	1	20	2	0
R	XPMEHUCKT1	U	2	2	06/03/74	U	U	SERL	NO	NO	126	1	0	0	1	320	1	0
S	XMR OUTPUT	U	14	3	11/09/72	U	U	SERL	YES	NO	126	1	0	0	1	80	1	0
T	XREBXREJCOMB	U	19	3	15/11/74	U	U	SEKL	NO	NO	0	1	0	0	1	160	5	0
U	GLYNCUSYXMB	2	0	3	07/11/74	U	U	SEKL	NO	NO	0	1	0	0	1	1440	1	0

Figure 16 Map and storage file report for EDS 8

\XPJD MK2		11/02/75	16/23/45	MAP AND SUMMARY FILE REPORT FOR STORAGE UNIT #720032												PAGE 1		
CYL		BLOCK																
8 24 40 56 72 88 104 120 136 152 168 184 200 216 232 248 264 280																		
U 16 32 48 64 80 96 112 128 144 160 176 192 208 224 240 256 272 288																		
U00 S S S S S S S S S S S S S S S S S S S S S S S S S S S S S S																		
U01 S S S S S S S S S S S S S S S S S S S S S S S S S S S S S S																		
U02 S S S S S S S S S S S S S S S S S S S S S S S S S S S S S S																		
U03 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U04 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U05 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U06 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U07 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U08 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U09 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U10 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U11 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U12 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U13 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U14 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U15 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U16 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U17 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U18 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		
U19 A A A A A A A A A A A A A A A A A A A A A A A A A A A A A A																		

\XPJD MK2		11/02/75	16/23/45	MAP AND SUMMARY FILE REPORT FOR STORAGE UNIT #720032												PAGE 12		
FILE CODE	FILE IDENTITY	GEN NO	VERS NO	INI CODE	DATE WRITTEN	DATA RETN	FILE RETN	FILE ORGN	VAR REC?	INH ACI?	FL WDS	BLKS /REC	KEY LTH	STRT POSN	NO OF AREAS	TOTAL BKTS	EOF BKT	2ND ST BKT
S	ICLSCAFILE	U	0	3	20/03/74	U	0	SERL	YES	YES	U	1	0	U	2	416	1	U
A	WHAT A LOT	U	0	3	20/03/74	U	0	SERL	YES	NO	126	1	U	U	1	34040	2349	U
M	DCUSYFILE CR	59	0	2	20/03/74	U	0	SERL	YES	NO	125	1	U	U	1	3256	1	U
C	DCUSYFILE CM	56	U	2	20/03/74	U	U	SERL	YES	NO	125	1	U	U	1	4144	1	U
D	DCUSYFILE KJ	15	U	2	20/03/74	U	0	SERL	YES	NO	125	1	U	U	1	1776	1	U
E	DCUSYFILE KS	72	U	2	20/03/74	U	0	SERL	YES	NO	125	1	U	U	14	2928	1	U
F	SUBGROUPSMDH	U	0	3	20/03/74	4095	0	SERL	YES	NO	125	1	U	U	1	176	124	U
G	DCUSYINSTALU	11	U	2	20/03/74	U	0	SERL	YES	NO	125	1	U	U	1	1776	1	U
H	SUBGROUPS-RS	3	U	3	20/03/74	U	0	SERL	YES	NO	125	1	U	U	1	4736	1	U
I	TESTXPJWFILE	U	U	3	27/06/74	U	0	SERL	NO	NO	120	1	U	U	1	3080	1	U
J	RDACUBUL1	U	U	3	07/12/74	U	0	SERL	YES	NO	126	1	U	U	1	208	1	U
K	RDACUBUL3	U	U	3	07/12/74	U	0	SERL	YES	NO	126	1	U	U	1	296	1	U
L	RDACUBUL4	U	U	3	07/12/74	U	0	SERL	YES	NO	126	1	U	U	1	296	1	U
M	RDACUBUL5	U	U	3	07/12/74	U	0	SERL	YES	NO	126	1	U	U	1	1184	1	U
N	RDACUBUL2	U	U	3	08/12/74	U	0	SERL	YES	NO	126	1	U	U	1	296	1	U
SCRATCH INHIBIT CLEAR																		

Figure 17 Map and storage file report for EDS 30/60

An example of the layout of the map of an EDS 8 unit is shown on page 284 and an example of the layout of the map of an EDS 30/60 is shown on page 285.

Fifty-five symbols are available to represent storage space allocated to files specified in file index cells on the same storage unit. If more than 55 such files exist the message "MORE THAN 55 FILES ON STORAGE UNIT" is printed at the end of the report and the storage space used by the additional files is represented on the map by blanks. Storage space allocated to files which are specified in file index cells on other storage units is also represented on the map by blanks but data about such space is given, if appropriate, in a table headed "FILE AREAS ALLOCATED TO FILES SPECIFIED IN FILE INDEX CELLS ON OTHER STORAGE UNITS".

Storage space which forms part of an unallocated file area is represented by the symbols '<', '~', and '>'.

Storage space which is not referenced by the system control area is represented by zeros. If such symbols are present, the message "ZERO ON MAP = UNREFERENCED AREA" is printed at the beginning of the report.

Storage space which is referenced more than once by the system control area is represented by '/' symbols. If such symbols are present, the message "'/' ON MAP = DOUBLY DEFINED AREA" is printed at the beginning of the report.

The summary file report takes the form of a table with data under the following headings:

FILE CODE	The symbol by which the file is represented on the map
FILE IDENTITY	The name of the file
GEN NO	The generation number of the file
VERS NC	The version number of the file
INT CODE	Integrity code (1, 2 or 3)
DATE WRITTEN	The date on which the file was last written to
DATA RETN	The data retention period (retention period 1)
FILE RETN	The file retention period (retention period 2)
FILE ORGN	File organization (serial sequential or random)
VAR REC?	Variable or fixed length records? (yes or no)
INH FL ACT?	Inhibit Executive's flaw recovery action? (yes or no)
WDS/REC	The maximum number of words per record
BLKS/BKT	The number of blocks per bucket
KEY LTH	Key length in characters
STRT POSN	The number of characters preceding the key in the record
NO OF AREAS	The number of file areas in the file, including any which may be on other units
TOTAL BKTS	Size of the file in buckets
EOF BKT	The number recorded in the file index cell as the bucket number of the end of file bucket
2ND LEVEL ST BKT	The bucket number of the start bucket of the second level overflow area (zero if none)

If a file has file areas allocated to it on other storage units, an asterisk will be printed before the file's name.

Apart from the FILE CODE, the data given is that held in the file's file index cell.

If appropriate a second table is given under the heading:

FILE AREAS ALLOCATED TO FILES SPECIFIED IN FILE INDEX CELLS ON OTHER STORAGE UNITS.

Data from each such file area's file area cell is given under the following sub-headings:

OTHER UNIT	The serial number of the storage unit holding the file's file index cell
START CYL	The number of the first cylinder used by the file area
END CYL	The number of the last cylinder used by the file area
START BLOCK	The number of the first block used in each cylinder
END BLOCK	The number of the last block used in each cylinder
NO OF BLKS	The total number of blocks used by the file area
FULL SEEK AREA	Complete cylinders used ? (yes or no)

CONTROL

This section describes the one parameter type that may be input to #XPJD. Further details of parameter formats are given in the section *Parameter formats* in Chapter 1.

<i>Parameter</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name and parameter type	XPJD1	5 (fixed)
*2 The serial number of the storage unit to be analysed	1 to 777777	6

*This parameter field may be repeated if more than one storage unit is to be analysed. The field or fields must be enclosed in parentheses. Continuation cards or lines may be given if these parameter fields extend over more than one card or line. Parameter field 1 must be specified in continuation cards or lines.

Each parameter field must be terminated by a comma except when followed by a parenthesis.

A card or line containing four asterisks (****) in columns 1 to 4 must follow the last parameter line. This may be followed by a LOad or FInd message. Further details are given in the section *Parameter formats* in Chapter 1.

Examples

PARAMETER ON CARDS

Reports are to be made on the four storage units with serial numbers 724723, 721341, 001415 and 767431. The program is to delete itself with the message FI #XPJW 12000.

The parameters to perform this function can take the following form:

```
card 1    XPJD1(724723,721341,1415)
card 2    XPJD1(767431)
card 3    ****FI #XPJW 12000
```

PARAMETERS ON PAPER TAPE

Reports are to be made on the three storage units with serial numbers 000020, 000023 and 000024.

The parameters to perform this function can take the following form (if input on paper tape):

```
XPJD1(20)↓
XPJD1(23,24)↓
****↓
```

where ↓ indicates a newline character.

When #XPJD is being run under GEORGE 3 or GEORGE 4, parameters may be given with an ENTER 2 command for reading by the GEORGE Command Issuer. An example is given in the paragraph *Running under*

#XPJD

GEORGE 3 or GEORGE 4 in the OPERATING section. The maximum length of a parameter input in this way is 76 characters.

As an alternative to giving parameters in the above format, one unit serial number may be input to word 0 of the program by an Executive ALter command and the program activated by the command GO #XPJD 23 or ENTER 3. An example is given in the paragraph *Running under Executive alone* in the OPERATING section.

OPERATING

Trusted status

The program has Q-trusted status.

Executive priority

The program as supplied has a priority of 30.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
TR0 or CR0 (optional)	Input of parameters	The reader is assigned at the beginning of the run and released when all the parameters have been read and analyzed
LPO	Output of reports	The line printer is used throughout the run
DA0	Subject of the storage report	Storage units are used one at a time

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJD (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required storage units either	
3 Load the parameters in the appropriate reader	
4 To activate the program	
(a) If parameters are on paper tape, input:	GO #XPJD 20
(b) If parameters are on cards, input:	GO #XPJD 21
or	
3 Input the serial number xxxxxx of the storage unit to be reported on:	AL #XPJD 0 *xxxxxx
4 To activate the program, input:	GO #XPJD 23
5 If it is required to omit a requested storage unit because it is unavailable or unreadable, input:	GO #XPJD 27

<i>Narrative</i>	<i>Console message</i>
6 At the end of the run the program will output one of the following:	
(a) If a delete message was correctly input, and a successful run has occurred:	DELETED:- <i>user message</i>
(b) If no delete message was input in an error free run:	HALTED:- HH
(c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:	DISPLAY:- DE HALTED:- HH
(d) If any error occurred during the run (except the error described in (c) above), whether or not a delete message was input:	HALTED:- ZZ
7 For a further run, return to operating instruction 2	

Running under GEORGE 3 or GEORGE 4

GEORGE COMMANDS

(In these examples reports are to be made on storage units 711000 and 711001).

LOAD :LIB.PROGRAM XPJD

CREATE !

ASSIGN *LP0,!

LISTFILE !, *LP

ERASE !

ENTER 2, PARAM(XPJD1(711000,711001),****)

or

LOAD :LIB.PROGRAM XPJD

CREATE !

ASSIGN *LP0,!

LISTFILE !, *LP

ERASE !

ONLINE *CR0

((((

ENTER 1

XPJD1(71100,711001)

))))

Exception conditions

The following is a list of exception conditions reported by #XPJD during execution and loading. Prior to entry into #XPJD it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:-LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

#XPJD

	<i>Messages</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJD or (b) Abandon the run
2	HALTED:- NEEDS INPUT PARAMETERS	The first parameter card or line held four asterisks	Either (a) Load the correct parameters in the appropriate reader and input: GO #XPJD or (b) Abandon the run
3	HALTED:- NEEDS LP OF MINIMUM SIZE 120	The program requires a line printer with at least 120 print positions across the page	Either (a) Make a line printer available and input: GO #XPJD or (b) Abandon the run
4	DISPLAY:- PE	Transfer failure on the line printer	No action. The run continues
5	DISPLAY:- PL	Paper low condition has been detected	Load more paper and re-engage the printer
6	HALTED:- RF	A card or tape block has been misread	To reread the parameter line, reload the card block and input: (a) GO #XPJD or (b) Abandon the run
7	HALTED:- ST <i>nnn</i>	The program requires <i>nnn</i> more words of store	Make the extra store available and input: (a) GO #XPJD or (b) Abandon the run
8	HALTED:- <i>xx</i>	An error has been detected when loading #XPJD by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

Line printer messages

If any of the messages given below are output on the line printer, the message HALTED:- ZZ will be output on the console at the end of the run.

Unless otherwise stated, the program abandons the current unit and proceeds to deal with the next.

Line printer messages are of the general form:

*****message*

where *message* is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
1 ABANDON ENTRY POINT USED	This is output should the operator use entry point 27 to abandon the report on a storage unit
2 BUCKET NUMBER OUT OF RANGE	The program has tried to read a block outside the limits of the storage unit. This probably indicates a corrupted system control area
3 INSUFFICIENT EXECUTIVE CORE	Executive cannot handle the file at this time. The run is immediately terminated
4 INTEGRITY CODE FAILURE	Another program is accessing the system control area
5 MORE THAN MAXIMUM NUMBER OF UNIT SERIAL NUMBERS	The program can process a maximum of 100 storage units in one run. If more than 100 unit serial numbers are given, only the first 100 are processed
6 MORE THAN 55 FILES ON STORAGE UNITS	Only 55 unique symbols are used for file identification. If more than 55 are present, the areas the extra files occupy on the map are left blank
7 PARAMETER ERROR xxxxxxx ↑	There is an error in the parameters. This line is followed by the last card or paper tape block with an upward pointing arrow to indicate the offending character. Storage units whose numbers have already been accepted by the program will be analyzed
8 PARITY ERROR ON CYLxBLOCKy	A parity error has occurred on block y of cylinder x
9 S.C.A. CORRUPT	There is incorrect information in the system control area. At least one 8-block area is beyond the map range
10 UNACCEPTABLE STORAGE UNIT SIZE	The storage unit has fewer than 16 or more than 464 blocks per cylinder, or more than 1000 cylinders
11 ZERO ON MAP = UNREFERENCED AREA	An area has been 'lost'. There is no reference to it in the system control area
12 '/' ON MAP = DOUBLY DEFINED AREA	The system control area is corrupt. There is more than one reference to some areas

NAME

#XPJH

VERSION

Mark 1

TITLE

Direct access file analysis or attempt in situ file reorganization (see Preface)

HARDWARE REQUIREMENT

6272 words of store

1 card reader or paper tape reader

1 line printer

1 or more direct access storage units

DESCRIPTION**General**

This program can perform two functions on a direct access file conforming to the ICL 1900 Series Direct Access Housekeeping format. The program can be used either to attempt to reorganize the file in situ by returning records from first and second level overflow areas to their home buckets and optionally report on the reorganization it has carried out, or to give an analysis of the file in one of three forms specified by the user, either full analysis, short analysis or summaries only analysis.

The program will operate on a whole file or on part of a file which must be a multiple of contiguous file areas.

The transfer buffers in the program are initially set to 1024 words each. The size of these buffers may be increased by specifying the amount of core store that the program may extend into thereby increasing the efficiency of the program.

The facilities are available in the form of two functions which operate as described below.

FUNCTION 1: REORGANIZE PART OR COMPLETE FILE IN SITU

The program examines each bucket for overflow records, tags and an extension chain. Provided that there is room in the home bucket any overflow records belonging to it will be restored to their correct position, tags being deleted as necessary. Records in an extension chain are moved, in order, to the home bucket where possible.

The reorganization may be carried out to a specific packing density.

If an overflow record is found for which no tag exists the program will either move the record to its home bucket, create a tag in the home bucket or leave the record where it is. The action taken depends on the space available in the home bucket. A tag will be deleted if the corresponding record cannot be found. The action the program takes is reported on the line printer.

When only part of a file is being reorganized and a second level overflow area is present, but not within the area specified for reorganization, the program will attempt to retrieve records from the second level overflow area if a bucket with an extension chain is encountered in the specified part of the file.

The current overflow bucket pointer held in the first bucket of a seek area is updated to point to the highest numbered bucket in that seek area which can accept overflow records.

Optionally, a report showing the state of the areas before and after reorganization in a format similar to that given for function 2 (see below) may be output on the line printer.

#XPJH

FUNCTION 2: ANALYZE PART OR WHOLE FILE

The program analyzes a complete file or specified areas of a file, and provides a report of the contents of buckets in the specified areas. It is intended to act as a guide towards determining whether a file should be reorganized or not.

The report output on the line printer can take one of three forms, full analysis, short analysis or summaries only analysis. The format of the output is shown in the section *Output*.

Full analysis

In a full analysis the details and summaries of every bucket are printed.

Short analysis

In a short analysis the details and summaries of all buckets except index buckets are printed.

Summaries only analysis

In this option only the summaries are printed.

Input

Input to the program consists of direct access data files conforming to the ICL 1900 Series Housekeeping format held on storage units conforming to Unified Direct Access Standards. The parameters to be input to the program are described in the section *Control*.

Output

The output from function1 is the files or part files in their reorganized state plus a report on the line printer, if required, of the reorganizations that have been carried out. The format of the report, which is printed in print position 1 to 96, is shown in Figures 18 to 20 below.

The analysis of files when using function 2 are output on the line printer using print positions 1 to 96. The possible forms that the report can take are shown in Figures 21 to 23 on the following pages.

#XPJH	MKxx	dd/mm/yy	hh/mm/ss	FILE ANALYSIS/REORGANISATION				PAGEyyy
WHOLE FILE REORGANISATION OF FILE xxxxxxxxxxxx				FGN xxxx		ON UNIT *xxxxxx		
FILE AREA NUMBER:xxx				ON UNIT *xxxxxx				
BUCKET TYPE		HOME RECORDS		O'FLOW RECORDS		TAGS		WORDS AVAIL.
		BEFORE	AFTER	BEFORE	AFTER	BEFORE	AFTER	
xxxxxx	INDEX							xxxx
xxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	2ND LEV O'F	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	2ND LEV O'F	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	2ND LEV O'F	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	1ST LEV O'F			xxxx	xxxx			xxxx
xxxxxx	1ST LEV O'F			xxxx	xxxx			xxxx
FILE AREA xxxx SUMMARY AND FINAL SUMMARY								
BUCKET TYPE		NUMBER UNUSED		NUMBER USED		NUMBER WITH OVERFLOW		
						1ST LEVEL	2ND LEVEL	
INDEX		xxxxxxx		xxxxxxx				
HOME		xxxxxxx		xxxxxxx		xxxxxxx	xxxxxxx	
1ST LEV O'F		xxxxxxx		xxxxxxx				
TOTAL NUMBER OF RECORDS = xxxxxxxx								
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxxx								
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx								

Figure 18 Full report of reorganization

# XPH	MKxx	dd/mm/yy	hh/mm/ss	FILE ANALYSIS/REORGANISATION				PAGEyy
WHOLE FILE REORGANISATION OF FILE xxxxxxxxxxxx				FGNxxx	ON UNIT *xxxxxx			
FILE	AREA	NUMBERxxx	ON UNIT *xxxxxx					
BUCKET	TYPE		HOME RECORDS BEFORE	AFTER	O'FLOW RECORDS BEFORE	AFTER	TAGS BEFORE AFTER	WORDS AVAIL.
xxxxxx	HOME		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	2ND LEV O'F		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	2ND LEV O'F		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	2ND LEV O'F		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	HOME		xxxx	xxxx	xxxx	xxxx	xxxx	xxxx
xxxxxx	END OF FILE							xxxx
xxxxxx	1ST LEV O'F				xxxx	xxxx		xxxx
xxxxxx	1ST LEV O'F				xxxx	xxxx		xxxx
xxxxxx	1ST LEV O'F				xxxx	xxxx		xxxx
FILE	AREA	xxxx	SUMMARY AND FINAL SUMMARY					
	BUCKET	TYPE	NUMBER UNUSED		NUMBER USED	NUMBER WITH OVERFLOW 1ST LEVEL 2ND LEVEL		
	HOME		xxxxxx		xxxxxx	xxxxxx	xxxxxx	
	1ST LEV O'F		xxxxxx		xxxxxx			
	2ND LEV O'F		xxxxxx		xxxxxx			
TOTAL NUMBER OF RECORDS = xxxxxxx								
NUMBER OF HOME BUCKETS WITH LESS THAN xxxxx WORDS SPARE = xxxxxxx								
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx								

Figure 19 Short report of reorganization

# XPH	MKxx	dd/mm/yy	hh/mm/ss	FILE ANALYSIS/REORGANISATION				PAGEyy
WHOLE FILE REORGANISATION OF FILE xxxxxxxxxxxx				FGN xxxx	ON UNIT *xxxxx			
FILE	AREA NUMBER	xxxx	ON UNIT *xxxxx	NUMBER USED	NUMBER WITH OVERFLOW			
	BUCKET TYPE		NUMBER UNUSED		1ST LEVEL	2ND LEVEL		
	INDEX		xxxxxx	xxxxxx				
	HOME		xxxxxx	xxxxxx	xxxxxx	xxxxxx		
	1ST LEV O'F		xxxxxx	xxxxxx				
TOTAL NUMBER OF RECORDS = xxxxxxx								
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxx								
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx								
FILE	AREA NUMBER	xxxx	ON UNIT *xxxxx	NUMBER USED	NUMBER WITH OVERFLOW			
	BUCKET TYPE		NUMBER UNUSED		1ST LEVEL	2ND LEVEL		
	INDEX		xxxxxx	xxxxxx				
	HOME		xxxxxx	xxxxxx	xxxxxx	xxxxxx		
	1ST LEV O'F		xxxxxx	xxxxxx				
TOTAL NUMBER OF RECORDS = xxxxxxx								
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORD SPARE = xxxxxxx								
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx								
FINAL SUMMARY								
	BUCKET TYPE		NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW			
					1ST LEVEL	2ND LEVEL		
	INDEX		xxxxxx	xxxxxx				
	HOME		xxxxxx	xxxxxx	xxxxxx	xxxxxx		
	1ST LEV O'F		xxxxxx	xxxxxx				
TOTAL NUMBER OF RECORDS = xxxxxxx								
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxx								
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx								

Figure 20 Summary report of reorganization

#XPJH

#XPJH	MKxx	dd/mm/yy	hh/mm/ss	FILE ANALYSIS/REORGANISATION		PAGEyyy
WHOLE FILE FULL ANALYSIS OF FILE xxxxxxxxxxxx				FGN xxxx = ON UNIT *xxxxxx		
FILE AREA	NUMBERxxxx	ON UNIT *xxxxxx				
BUCKET	TYPE	HOME RECORDS	O'FLOW RECORDS	TAGS	WORDS AVAIL.	
xxxxxxx	INDEX				xxxx	
xxxxxxx	INDEX				xxxx	
xxxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	2ND LEV O'F	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	1ST LEV O'F	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	1ST LEV O'F		xxxx		xxxx	
FILE AREA xxxx SUMMARY						
	BUCKET TYPE	NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW		
				1ST LEVEL	2ND LEVEL	
	INDEX	xxxxxxx	xxxxxxx			
	HOME	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	
	1ST LEV O'F	xxxxxxx	xxxxxxx			
TOTAL NUMBER OF RECORDS = xxxxxxx						
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxx						
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx						

#XPJH	MKxx	dd/mm/yy	hh/mm/ss	FILE ANALYSIS/REORGANISATION		PAGEyyy
FILE AREA NUMBERxxxx ON UNIT *xxxxxx						
BUCKET	TYPE	HOME RECORDS	O'FLOW RECORDS	TAGS	WORDS AVAIL.	
xxxxxxx	INDEX				xxxx	
xxxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	HOME	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	END OF FILE	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	1ST LEV O'F	xxxx	xxxx	xxxx	xxxx	
xxxxxxx	1ST LEV O'F		xxxx		xxxx	
xxxxxxx	1ST LEV O'F		xxxx		xxxx	
FILE AREA xxxx SUMMARY						
	BUCKET TYPE	NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW		
				1ST LEVEL	2ND LEVEL	
	INDEX	xxxxxxx	xxxxxxx			
	HOME	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	
	1ST LEV O'F	xxxxxxx	xxxxxxx			
TOTAL NUMBER OF RECORDS = xxxxxxx						
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxx						
FINAL SUMMARY						
	BUCKET TYPE	NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW		
				1ST LEVEL	2ND LEVEL	
	INDEX	xxxxxxx	xxxxxxx			
	HOME	xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx	
	1ST LEV O'F	xxxxxxx	xxxxxxx			
TOTAL NUMBER OF RECORDS = xxxxxxx						
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxx						
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx						

Figure 21 Full report of analysis

#XPJH	MKxx	dd/mm/yy	hh/mm/ss	FILE ANALYSIS/REORGANISATION		PAGEyyy
WHOLE FILE SHORT ANALYSIS OF FILE xxxxxxxxxxxx				FGN xxxx ON UNIT *xxxxxx		
FILE	AREA	NUMBERxxxx	ON UNIT *xxxxxx			
BUCKET	TYPE	HOME RECORDS		O'FLOW RECORDS	TAGS	WORDS AVAIL.
xxxxxxx	HOME	xxxx		xxxx	xxxx	xxxx
xxxxxxx	HOME	xxxx		xxxx	xxxx	xxxx
xxxxxxx	2ND LEV O'F	xxxx		xxxx	xxxx	xxxx
xxxxxxx	HOME	xxxx		xxxx	xxxx	xxxx
xxxxxxx	2ND LEV O'F	xxxx		xxxx	xxxx	xxxx
xxxxxxx	HOME	xxxx		xxxx	xxxx	xxxx
xxxxxxx	HOME	xxxx		xxxx	xxxx	xxxx
xxxxxxx	1ST LEV O'F	xxxx		xxxx	xxxx	xxxx
xxxxxxx	1ST LEV O'F	xxxx		xxxx	xxxx	xxxx
xxxxxxx	1ST LEV O'F	xxxx		xxxx	xxxx	xxxx
xxxxxxx	1ST LEV O'F	xxxx		xxxx	xxxx	xxxx
FILE	AREA	xxxx	SUMMARY			
	BUCKET	TYPE	NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW	
	HOME		xxxxxxx	xxxxxxx	xxxxxxx	
	1ST LEV O'F		xxxxxxx	xxxxxxx	xxxxxxx	
	2ND LEV O'F		xxxxxxx	xxxxxxx	xxxxxxx	
TOTAL NUMBER OF RECORDS = xxxxxxxx						
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxxx						
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx						

#XPJH	MKxx	dd/mm/yy	hh/mm/ss	FILE ANALYSIS/REORGANISATION		PAGE yyy
FILE	AREA	NUMBERxxxx	ON UNIT *xxxxxx			
BUCKET	TYPE	HOME RECORDS		O'FLOW RECORDS	TAGS	WORDS AVAIL.
xxxxxxx	HOME	xxxx		xxxx	xxxx	xxxx
xxxxxxx	END OF FILE					xxxx
xxxxxxx	1ST LEV O'F			xxxx		xxxx
xxxxxxx	1ST LEV O'F			xxxx		xxxx
xxxxxxx	1ST LEV O'F			xxxx		xxxx
FILE	AREA	xxxx	SUMMARY			
	BUCKET	TYPE	NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW	
	HOME		xxxxxxx	xxxxxxx	xxxxxxx	
	1ST LEV O'F		xxxxxxx	xxxxxxx	xxxxxxx	
	2ND LEV O'F		xxxxxxx	xxxxxxx	xxxxxxx	
TOTAL NUMBER OF RECORDS = xxxxxxxx						
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxxx						
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx						
FINAL SUMMARY						
	BUCKET	TYPE	NUMBER USED	NUMBER UNUSED	NUMBER WITH OVERFLOW	
	HOME		xxxxxxx	xxxxxxx	xxxxxxx	
	1ST LEV O'F		xxxxxxx	xxxxxxx	xxxxxxx	
	2ND LEV O'F		xxxxxxx	xxxxxxx	xxxxxxx	

Figure 22 Short report of analysis

#XPJH

#XPJH	MKxx	dd/mm/yy	hh/mm/ss	FILE ANALYSIS/REORGANISATION		PAGE.yyy
WHOLE FILE SUMMARY ANALYSIS OF FILE xxxxxxxxxxxx				FGN.xxxx	ON UNIT *xxxxxx	
FILE AREA	xxxx	SUMMARY				
	BUCKET	TYPE	NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW	
					1ST LEVEL	2ND LEVEL
	INDEX		xxxxxxx	xxxxxxx		
	HOME		xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx
	1ST LEV O'F		xxxxxxx	xxxxxxx		
TOTAL NUMBER OF RECORDS = xxxxxxxx						
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxxx						
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx						
FILE AREA	xxxx	SUMMARY				
	BUCKET	TYPE	NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW	
					1ST LEVEL	2ND LEVEL
	INDEX		xxxxxxx	xxxxxxx		
	HOME		xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx
	1ST LEV O'F		xxxxxxx	xxxxxxx		
TOTAL NUMBER OF RECORDS = xxxxxxxx						
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxxx						
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx						
FINAL	SUMMARY					
	BUCKET	TYPE	NUMBER UNUSED	NUMBER USED	NUMBER WITH OVERFLOW	
					1ST LEVEL	2ND LEVEL
	INDEX		xxxxxxx	xxxxxxx		
	HOME		xxxxxxx	xxxxxxx	xxxxxxx	xxxxxxx
	1ST LEV O'F		xxxxxxx	xxxxxxx		
TOTAL NUMBER OF RECORDS = xxxxxxxx						
NUMBER OF HOME BUCKETS WITH LESS THAN xxxx WORDS SPARE = xxxxxxxx						
NUMBER OF SECOND LEVEL OVERFLOW BUCKETS USED = xxx						

Figure 23 Summary report of analysis

CONTROL

This section describes the types of parameters associated with the two functions described in the section *General*. Further details of the parameter formats are given in the section *Parameter formats* in Chapter 1.

Function 1: reorganize part or whole file in situ

Parameter field	Range of values	Maximum number of characters
1 Program name (XPJH) and parameter type	XPJH1	5 (fixed)
2 Function code	1	1 (fixed)
3 Form of report required	F, A or S	1 (fixed)
	where	
	F = full	
	A = short	
	S = summaries only	
	Note: If omitted, only error messages will be output on the line printer	
4 Information required if any home buckets have less than the number of words specified in this parameter spare	1 to 1022	4
	Note: If this is not specified information regarding spare words in home buckets will not be printed in a short analysis or in any summaries	

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
5 Program store size	1 to 32 (in multiples of 1024 words) Note: This is optional and may be omitted	2
*6 Name of the file to be reorganized	Up to 12 alphanumeric, space (V) or minus (-) characters, with the first character alphabetic	12
*7 Generation number of the file	- 1, 0 to 4095 Note: If this is not specified, a generation number of zero is assumed	4
*8 Serial number of the unit holding the file	000001 to 777777 (in octal characters) Note: This parameter is optional and may be omitted	6 (fixed)
*9 Number of first file are to be reorganized	1 to 9999 Note: If this is not specified, the value 1 is assumed	4
*10 Number of file areas to be reorganized	1 to 9999 Note: If this is not specified, all file areas following the start file area will be reorganized	4
*11 Bucket packing density (as a percentage)	0 to 99 Note: If this is omitted, a packing density of 100% is assumed	2

*The group of parameter fields 6 to 11 must be enclosed in parentheses. Each parameter field must be terminated by a comma except when followed by a parenthesis.

Function 2: analysis part or whole file

<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1 Program name (XPJH) and parameter type	XPJH1	5 (fixed)
2 Function code	2	1 (fixed)
3 Form of analysis required	F, A or S where F = full A = short S = summaries	1 (fixed)
4 Summary information required if any home buckets have less than the number of words specified in this parameter spare	1 to 1022 Note: If this is not specified, information regarding spare words in home buckets will not be printed in any summaries	4

#XPJH

5	Program store size	1 to 32 (in multiples of 1024 words)	2
		Note: This is optional and may be omitted	
*6	Name of the file to be analysed	Up to 12 alphanumeric, space (V) or minus (-) characters with the first character alphabetic	12
*7	Generation number of the file	-1, 0 to 4095	4
		Note: If this is not specified, a generation number of zero is assumed	
*8	Serial number of the unit holding the file	000001 to 777777 (in octal characters)	6 (fixed)
		Note: This field is optional and need not be specified	
*9	Number of the first file area to be analysed	1 to 9999	4
		Note: If this is not specified, the value 1 is assumed	
*10	Number of file areas to be analysed	1 to 9999	4
		Note: If this is omitted, all file areas following the start file area will be analysed	

*The group of parameter fields 6 to 10 must be enclosed in parentheses. Each parameter field must be terminated by a comma except when followed by a parenthesis.

End of parameters

A card or line containing four asterisks (****) in columns 1 to 4 must follow a batch of parameters. This may be followed by a LOad or FInd message. Further details are given in the section *Parameter formats* in Chapter 1.

Examples

FUNCTION 1: REORGANIZE PART OR WHOLE FILE IN SITU

- 1 All file areas of the file MASTERFILEA, generation number 5, held on unit 000450 are to be reorganized. The core size of the program may be increased to 8K words. Buckets are, when appropriate, to be packed to 80% density. A full report of the reorganization is to be output on the line printer including information of the number of home buckets which have less than 60 words spare.

The parameters to perform this function would take the following form (if input on cards):

card 1 XPJH1,1,F,60,8(MASTERFILEA,5,000450,1,,80)

card 2 ****

- 2 The first two file areas of the file AMENDFILEB, generation number 0, is to be reorganized and a report containing summaries only is to be output. Bucket packing density is to be 100%.

The parameters to perform this function would take the following form (if input on paper tape):

XPJH1,1,S,(AMENDFILEB,0,,1,2)↓

****↓

where ↓ indicates a newline character

FUNCTION 2: ANALYZE PART OR WHOLE FILE

A short analysis including information on home buckets which have less than 35 words spare is to be given of file areas 2, 3 and 4 of file STOCKFILEC, generation number 20, held on storage unit 1000025.

The parameters to perform this function would take the following form (if input on cards):

card 1 XPJH1,2,A,35(STOCKFILEC,2,100025,2,3)

card 2 ****

OPERATING

Executive priority

The program as supplied has a priority of 60.

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
TR0 or CR0	Input of parameters	The reader is assigned at the beginning of the run and released when the terminator parameter has been read
LP0	Report and error messages	The printer is used throughout the run if a report is being output; otherwise assigned when an error occurs
DA0	Input medium	Used throughout the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJH (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required storage unit(s)	
3 Load the parameters in the appropriate reader	
4 To activate the program, either: (a) If parameters are on paper tape, input: or (b) If parameters are on cards, input:	GO #XPJH 20 GO #XPJH 21
5 When the run is complete, the program will output one of the following messages: (a) If a delete message was correctly input, and a successful run has occurred: (b) If the run was error free and no delete message was input:	DELETED:- user message HALTED:- HH

#XPJH

Narrative

- (c) If a delete message was input which did not start with the characters LO or FI, and the run was error free:
- (d) If any error had occurred during the run (except the one described in (c) above), whether or not a delete message was input:

Note: If this message is output at the end of the run, one or more error messages will have been output on the line printer (see the subsection *Line printer messages*, page 293)

Console message

DISPLAY:- DE
HALTED:- HH

HALTED:- ZZ

6 For a further run, return to Operating Instruction 2

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJH during execution and loading. Prior to entry into #XPJH it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJH or (b) Abandon the run
2	HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJH or (b) Abandon the run
3	DISPLAY:- PE	Transfer failure on the line printer	No action. The run continues
4	DISPLAY:- PL	Paper low condition detected on the line printer	Load more paper and re-engage the printer
5	HALTED:- RF	A read failure has occurred when reading parameters	Reset the last parameter read and input: GO #XPJH
6	HALTED:- TR	There is no tape reader available	Either (a) Make a tape reader available and input: GO #XPJH or (b) Abandon the run

<i>Message</i>	<i>Reason</i>	<i>Action</i>
7 HALTED:- xx	An error has been detected when loading #XPJH by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when Executive attempts to open the storage unit it will output a storage unit not on-line message, for example:

#XPJH; LDC * nnnnnn (the actual format of the message varies according to which Executive is being used)

The user then has two options to take:

- 1 Put the requested storage unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJH 27. The program abandons the current function

LINE PRINTER MESSAGES

If any of the error messages given below are output on the line printer, the message HALTED:- ZZ will be output on the console at the end of the run.

The formats of these messages fall into five categories:

Format 1

***fn fgn message

where

fn is the name of the file.

fgn is the file generation number.

message is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
1 FILE CONTRACTED BY ANOTHER PROGRAM	This message will occur if the file being analyzed/reorganized is contracted by another program during the run. The run will continue with the next function
2 FILE NOT IN SYSTEM	The run will continue with the next function
3 INSUFFICIENT EXECUTIVE CORE	The run will continue with the next function
4 INTEGRITY CODE FAILURE	The run will continue with the next function
5 PARAMETER ERROR xxxxxxxxxxxxxxxxxxxxx ↑	Where the arrow indicates the offending parameter. The run will continue with the next function
6 PURGE DATE NOT EXCEEDED	The run will continue with the next function
7 ABANDON ENTRY POINT USED	This message is output if entry point 7 is used during the running of the program

Format 2

lbn message key

where

lbn is the logical bucket number of the current bucket.

#XPJH

key is the key of the record in error.

message is one of the following:

Message

- 8 ****CREATED TAG**
- 9 ****DELETED TAG**

Reason and/or action

The message will only occur during a reorganization. The run continues by processing the next record

This message will only occur during a reorganization. The run continues by processing the next record

Format 3

lbn message bkt

KEY *key*

where

lbn is the logical bucket number of the current bucket.

bkt is the logical number pointed to by the erring record.

key is the key of the record in error.

message is one of the following:

Message

- 10 ****LOGICAL BUCKET NUMBER OUT OF RANGE**
- 11 ****DUPLICATE OVERFLOW RECORD FOUND IN BUCKET**
- 12 ****DUPLICATE TAG FOUND IN BUCKET**
- 13 ****OVERFLOW RECORD MISSING IN BUCKET**
- 14 ****OVERFLOW RECORD POINTS TO A HOME RECORD IN BUCKET**
- 15 ****TAG MISSING IN BUCKET**
- 16 ****TAG POINTS TO HOME RECORD IN BUCKET**
- 17 ****OVERFLOW REFERENCES INDEX BUCKET**

Reason and/or action

The run continues by processing the next record

This message will occur when an overflow record points to another overflow record. The run continues by processing the next record

This message will only be output when a tag points to another tag during an analysis. The run continues by processing the next record

The run continues by processing the next record

The run continues by processing the next record

This message will always occur in an analysis run if an overflow record is found for which there is no corresponding tag. During a reorganization run this message may be output if there is insufficient room for a tag to be created in the bucket referenced by the overflow record. The run continues by processing the next record

The run continues by processing the next record

The message will occur if a tag or a record in a first level overflow bucket points to an index bucket. The run continues by processing the next record

Format 4*lbn message*

where

lbn is the logical bucket number of the current bucket (that is, the one in error)*message* is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
18 **FLAW AREA FULL	The run continues by processing the next bucket
19 **INVALID BUCKET HEADER	This message refers to a bucket which is currently being analyzed/reorganized. The run continues by processing the next bucket
20 **PARITY ERROR	This message refers to a bucket which is currently being analyzed/reorganized. The run continues by processing the next bucket
21 **UNRECOGNISABLE REPLY	This message refers to a bucket which is currently being analyzed/reorganized. The run continues by processing the next bucket

Format 5*lbn message bkt*KEY *key*

where

lbn is the logical number of the current bucket.*bkt* is the bucket pointed to by the current record (that is, the one in error).*key* is the key of the current record.*message* is one of the following:

<i>Message</i>	<i>Reason and/or action</i>
22 **FLAW AREA FULL BUCKET	This message refers to a bucket which is pointed to be an overflow record or tag. The run continues by processing the next record
23 **INVALID BUCKET HEADER IN BUCKET	This message refers to a bucket which is pointed to by an overflow record or tag. The run continues by processing the next record
24 **PARITY ERROR IN BUCKET	This message refers to a bucket which is pointed to by an overflow record or tag. The run continues by processing the next record
25 **UNRECOGNISABLE REPLY IN BUCKET	This message refers to a bucket which is pointed to by an overflow record or tag. The run continues by processing the next record

NAME

#XPJJ

VERSION

Mark 1B

TITLE

Utilization and storage report (see Preface)

HARDWARE REQUIREMENT

3000 words of store

1 card reader or paper tape reader

1 line printer

1 or more direct access storage units

DESCRIPTION**General**

The program will print reports of the state of storage units at two separate levels, at **file level** and at **storage unit level**.

FILE LEVEL REPORT

At this level, the program will print a report of the storage used by the specified file, or files, by file areas giving details of the file as held in the file index and parameters cell.

Up to 40 files may be reported on.

STORAGE UNIT LEVEL REPORT

At this level five optional reports are available, as follows (the use of the letters in brackets is explained under *Control* on page 300):

- 1 Allocated areas. This is a report, in the same format as the file level report, of all files originating on the storage unit. This report also shows the setting of the scratch inhibit on the specified unit (A)
- 2 File areas allocated on the specified storage unit but originating from other storage units (E)
- 3 Unallocated areas on the storage unit (U)
- 4 Vacant file index and parameter cells and file area cells on the storage unit (V)
- 5 A summary of the files on the storage unit (S)

Up to 33 storage units may be specified.

Input

One or more storage units conforming to the 1900 Series Unified Direct Access Standards. The parameters that are input to the program are described in the section *Control*.

Output

The report on the storage unit is output on the line printer using print positions 1 to 96. Examples of reports are given in Figures 24 to 29 on the following pages.

#XPJJ	MK1	dd/mm/yy	hh/mm/ss	UTILISATION AND STORAGE REPORT SPECIFIED FILE REPORT												PAGE	yyy
		STORAGE UNIT SERIAL NO. xxxxxx															
FILE IDENTITY	GEN NO.	VERS NO.	INT CODE	DATE WRITTEN	DATA RETN	FILE RETN	FILE ORGN	OWN MON	F/ V	MAX REC	BLKS /BKT	KEY LTH	STRT POSN	EOF BKT	2ND LEV ST BKT	FILE AREAS	
xxxxxxxxxxxx	xxxx	xxxx	x	dd/mm/yy	xxxx	xxxx	x	xxx	x	xxxx	x	xx	xxxx	xxxxxxx	xxxxxxx	xxxx	
FILE AREA NO.	STORAGE UNIT	START CYL	END CYL	START BLOCK	END BLOCK	NO. OF BUCKETS	FULL SEEK AREA										
x	xxxxxx	xxxx	xxxx	xxxxxx	xxxxxx	xxxxxx	xxx										
x	xxxxxx	xxxx	xxxx	xxxxxx	xxxxxx	xxxxxx	xxx										
						TOTAL	xxxxxx										

Figure 24 File level report

#XPJJ	MK1	dd/mm/yy	hh/mm/ss	UTILISATION AND STORAGE REPORT SUMMARY FILE REPORT												PAGE	yyy
		STORAGE UNIT SERIAL NUMBER xxxxxx															
FILE IDENTITY	GEN NO.	VERS NO.	INT CODE	DATE WRITTEN	DATA RETN	FILE RETN	FILE ORGN	OWN MON	F/ V	MAX REC	BLKS /BKT	KEY LTH	STRT POSN	EOF BKT	2ND LEV ST BKT	FILE AREAS	
xxxxxxxxxxxx	xxxx	xxxx	x	dd/mm/yy	xxxx	xxxx	x	xxx	x	xxxx	x	xx	xxxx	xxxxxxx	xxxxxxx	xxxx	
xxxxxxxxxxxx	xxxx	xxxx	x	dd/mm/yy	xxxx	xxxx	x	xxx	x	xxxx	x	xx	xxxx	xxxxxxx	xxxxxxx	xxxx	
xxxxxxxxxxxx	xxxx	xxxx	x	dd/mm/yy	xxxx	xxxx	x	xxx	x	xxxx	x	xx	xxxx	xxxxxxx	xxxxxxx	xxxx	

Figure 25 Storage unit level report - summary

#XPJJ	MK1	dd/mm/yy	hh/mm/ss	UTILISATION AND STORAGE REPORT COMPLETE FILE REPORT												PAGE	yyy
		STORAGE UNIT SERIAL NO. xxxxxx		SCRATCH INHIBIT CLEAR													
FILE IDENTITY	GEN NO.	VERS NO.	INT CODE	DATE WRITTEN	DATA RETN	FILE RETN	FILE ORGN	OWN MON	F/ V	MAX REC	BLKS /BKT	KEY LTH	STRT POSN	EOF BKT	2ND LEV ST BKT	FILE AREAS	
xxxxxxxxxxxx	xxxx	xxxx	x	dd/mm/yy	xxxx	xxxx	x	xxx	x	xxxx	x	xx	xxxx	xxxxxxx	xxxxxxx	xxxx	
FILE AREA NO.	STORAGE UNIT	START CYL	END CYL	START BLOCK	END BLOCK	NO. OF BUCKETS	FULL SEEK AREA										
x	xxxxxx	xxxx	xxxx	xxxxxx	xxxxxx	xxxxxx	xxx										
x	xxxxxx	xxxx	xxxx	xxxxxx	xxxxxx	xxxxxx	xxx										
						TOTAL	xxxxxx										
FILE IDENTITY	GEN NO.	VERS NO.	INT CODE	DATE WRITTEN	DATA RETN	FILE RETN	FILE ORGN	OWN MON	F/ V	MAX REC	BLKS /BKT	KEY LTH	STRT POSN	EOF BKT	2ND LEV ST BKT	FILE AREAS	
xxxxxxxxxxxx	xxxx	xxxx	x	dd/mm/yy	xxxx	xxxx	x	xxx	x	xxxx	x	xx	xxxx	xxxxxxx	xxxxxxx	xxxx	
FILE AREA NO.	STORAGE UNIT	START CYL	END CYL	START BLOCK	END BLOCK	NO. OF BUCKETS	FULL SEEK AREA										
x	xxxxxx	xxxx	xxxx	xxxxxx	xxxxxx	xxxxxx	xxx										
x	xxxxxx	xxxx	xxxx	xxxxxx	xxxxxx	xxxxxx	xxx										
						TOTAL	xxxxxx										

Figure 26 Storage unit level report - allocated areas

#XPJJ MK1 dd/mm/yy hh/mm/ss UTILISATION AND STORAGE REPORT						PAGE yyy
FILE AREAS ASSIGNED TO FILES STARTING ON OTHER STORAGE UNITS						
STORAGE UNIT SERIAL NO. xxxxxx						
HOME STORAGE UNIT	START CYL	END CYL	START BLOCK	END BLOCK	FULL SEEK AREA	
xxxxxx	xxxx	xxxx	xxxxxxx	xxxxxxx	xxx	
xxxxxx	xxxx	xxxx	xxxxxxx	xxxxxxx	xxx	
xxxxxx	xxxx	xxxx	xxxxxxx	xxxxxxx	xxx	

Figure 27 Storage unit level report - file areas assigned to files starting on other storage units

#XPJJ MK1 d.t/mm/yy hh/mm/ss UTILISATION AND STORAGE REPORT						PAGE yyy
UNALLOCATED FILE AREAS						
STORAGE UNIT SERIAL NO. xxxxxx						
START CYLINDER	END CYLINDER	START BLOCK	END BLOCK	NO. OF BLOCKS DEFINED		
xxxx	xxxx	xxxxxxx	xxxxxxx	xxxxxxx		
xxxx	xxxx	xxxxxxx	xxxxxxx	xxxxxxx		

Figure 28 Storage unit level report - unallocated areas

#XPJJ MK1 dd/mm/yy hh/mm/ss UTILISATION AND STORAGE REPORT						PAGE yyy
VACANT CELLS IN THE SYSTEM CONTROL AREA						
STORAGE UNIT SERIAL NO. xxxxxx						
VACANT F.I.P. CELLS			VACANT FILE AREA CELLS			
xxxxxxx			xxxxxxx			

Figure 29 Storage unit level report - vacant cells

#XPJJ

Details of line printer messages are given in the subsection *Line printer messages* on page 304.

CONTROL

This section describes the parameters associated with the functions of #XPJJ described in the section *General*. Further details of the formats of parameters are described in the section *Parameter formats* in Chapter 1.

File level report parameters

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJJ) and parameter type	XPJJ1	5 (fixed)
*2	File name of the file to be reported on	Up to 12 alphanumeric, space (V) or minus (-) characters, with the first character alphabetic	12
*3	Generation number of the file to be reported on	- 1, 0 to 4095 Note: If this parameter is not specified the generation number will be assumed to be zero. If the generation number is specified as - 1 the file with the correct name and highest generation number will be reported on	4
*4	Serial number of the unit holding the file to be reported on	000001 to 777777 (an octal number) Note: A file may still be specified if this parameter is omitted	6 (fixed)

*The group of parameter fields 2 to 4 must be enclosed by parentheses and may be repeated. A group must be contained within one card or line, although several groups may appear on one card or line.

Each parameter field must be followed by a comma except when followed by a parenthesis.

Storage unit level report parameters

	<i>Parameter field</i>	<i>Range of values</i>	<i>Maximum number of characters</i>
1	Program name (XPJJ) and parameter type	XPJJ2	5 (fixed)
*2	Serial number of the storage unit to be reported on	000001 to 777777 (an octal number)	6 (fixed)
*3	Possible report types	A, E, U, V, S Note: These letters correspond to numbers 1 to 5 as described in the subsection <i>Storage unit level report</i> on page 297. Where more than one report is required the report types should be separated by commas	5

*The group of parameter fields 2 and 3 must be enclosed in parentheses and may be repeated. They must be contained on one card or line although several groups may appear on one card or line.

Each parameter field must be followed by a comma except when followed by a parenthesis.

End of parameters

A card or line with four asterisks (****) in columns 1 to 4 must follow a batch of parameters. These may be followed by a L_Oad or F_Ind message. Details are given in the section *Parameter formats* in Chapter 1.

Examples**1 FILE LEVEL REPORT**

Reports are required on files ANDYSFILE with generation number 0, JIMSFILE with generation number 1, CHRISFILE with generation number 123 and IANSFILE with a generation number assumed to be zero. JIMSFILE and IANSFILE are held on unit 717071 and CHRISFILE on unit 717007.

The parameters required to perform this function would take the following form (if input on cards):

card 1 XPJJ1(ANDYSFILE,0)(JIMSFILE,1,717071)

card 2 XPJJ1(CHRISFILE,123,717007)(IANSFILE,,717071)

card 3 ****

2 STORAGE UNIT LEVEL REPORT

A report of allocated areas is required on unit 717007, a report of allocated areas, unallocated areas and vacant file index and parameter cells and file area cells on unit 717071 and a summary of the files on unit 717047.

The parameters required to perform these functions would take the following form (if input on paper tape):

XPJJ2(717007,A)(717071,A,U,V)↓

XPJJ2(717047,S)↓

****↓

where ↓ indicates a newline character.

3 FINDING NEXT PROGRAM

If the parameters given in examples 1 and 2 are input as one batch, and it is required to load #XPJC after the termination of the program, the parameters would take the following form (if input on paper tape):

XPJJ1(ANDYSFILE,0)(JIMSFILE,1,717071)↓

XPJJ1(CHRISFILE,123,717007)(IANSFILE,,717071)↓

XPJJ2(717007,A)(717071,A,U,V)↓

XPJJ2(717047,S)↓

****▽▽FI#XPJC#DISC↓

where ↓ indicates a newline character

OPERATING**Trusted status**

The program has Q-trusted status.

Executive priority

The program as supplied has a priority of 50.

#XPJJ

Use of peripherals

<i>Device</i>	<i>Use</i>	<i>Assignment</i>
CR0 or TR0	Parameters	The reader is assigned at the beginning of the run and is released as soon as the terminator parameter has been read
LP0	Reports and error messages	The printer is assigned at the beginning of the run and is used throughout the run
DA0	Files to be reported on	Assigned at the beginning of the run and used throughout the run
DA2	Systems file	Used throughout the run

Running under Executive alone

OPERATING INSTRUCTIONS

<i>Narrative</i>	<i>Console message</i>
1 Load the program #XPJJ (see FIND A BINARY PROGRAM in <i>Compiling Systems</i>) The message HALTED:- LD will only be output if the loading operation was successful	HALTED:- LD
2 Load the required storage unit(s)	
3 Load the parameters in the appropriate reader	
4 To activate the program:	
(a) For parameters on paper tape, input:	GO #XPJJ 20
(b) For parameters on cards, input:	GO #XPJJ 21
5 When the run is complete, the program will output one of the following messages:	
(a) If a delete message was correctly input, and a successful run had occurred:	DELETED:- <i>user message</i>
(b) If the run was error free and no delete message was input:	HALTED:- HH
(c) If a delete message was input which did not start with the character LO or FI, and the run was error free:	DISPLAY:- DE HALTED:- HH
(d) If any error had occurred during the run (except the one described in (c) above) whether or not a delete message was input:	HALTED:- ZZ
Note: If this message is output at the end of the run, one or more error messages will have been output on the line printer (see below)	

EXCEPTION CONDITIONS

The following is a list of exception conditions reported by #XPJJ during execution and loading. Prior to entry into #XPJJ it must have been loaded correctly and this is signalled by HALTED:- LD. When a loading exception condition occurs before HALTED:- LD reference must be made to the section *Exception Conditions* in FIND A BINARY PROGRAM in *Compiling Systems*.

	<i>Message</i>	<i>Reason</i>	<i>Action</i>
1	HALTED:- CR	There is no card reader available	Either (a) Make a card reader available and input: GO #XPJJ or (b) Abandon the run
2	HALTED:- LP	There is no line printer available	Either (a) Make a line printer available and input: GO #XPJJ or (b) Abandon the run
3	DISPLAY:- PE	Transfer failure on the line printer	No action. The run continues
4	DISPLAY:- PL	Paper low condition has been detected	Load more paper and re-engage the printer
5	HALTED:- RF	A card or paper tape block has been misread	To reread the parameter line reload the card/block and input: GO #XPJJ
6	HALTED:- TR	There is no paper tape reader available	Either (a) Make a tape reader available and input: GO #XPJJ or (b) Abandon the run
7	HALTED:- xx	An error has been detected when loading #XPJJ by means of #XPEO, #XPMO, #XPXX or #TAPE	See section <i>Exception Conditions</i> in FIND A BINARY PROGRAM in <i>Compiling Systems</i>

The program uses a storage unit and this should be placed on-line as part of the operating instructions. In the event of the storage unit having not been placed on-line, when Executive attempts to open the storage unit it will output a storage unit not on-line message, for example:

#XPJJ; LDC * nnnnnn (the actual format of the message varies according to which Executive is being used)

The user then has two option to take:

- 1 Put the requested storage unit on-line and the program will continue
- 2 If the requested storage unit is not available then type GO #XPJJ 27. The program will abandon the present set of parameters and go to the next

#XPJJ

LINE PRINTER MESSAGES

If any of the messages given below are output on the line printer, the message HALTED:- ZZ will be output on the console at the end of the run.

- 1 If a card sequence error occurs, the following message is output:

** SEQUENCE ERROR,PARAMETER XPJJxSEQ xxxx

The card is processed normally

- 2 If an error occurs while reading the system control area of the unit being processed and reporting cannot continue, depending on the type of error, one of the following messages is output:

**** SCA CORRUPT ON *nnnnnn

**** SYFLEnnnnnn PARITY ERROR BUCKET xxxxxxx

**** SYFLEnnnnnn UNEXPECTED REPLY BUCKET xxxxxx

Only the storage unit specified in the message is ignored. The run continues normally

- 3 If a file report cannot continue one of the following messages is output:

xxxxxxxxxxxx mnnn	{	ABANDON ENTRY POINT USED
		NOT IN SYSTEM
		INTEGRITY CODE FAILURE
		INSUFFICIENT EXECUTIVE STORE

where

xxxxxxxxxxxx is the file name

nnnn is the generation number of the file

Only the file specified in the message is ignored and the run continues normally

- 4 If there has been a parameter error or there has been a request to report on more than 40 files or 33 storage units the following messages are output:

**** PARAMETER ERROR

**** TOO MANY FILES

**** TOO MANY UNITS

Each of these messages is followed by the image of the card or line with an upward pointing arrow beneath the incorrect field. Only the file or unit specified and the files or unit following on the same card or line are ignored. The run continues normally

TITLE

DAMAP

FUNCTION

To produce an #XPJD storage map of a specified disc storage unit.

RESTRICTIONS

The user must have the TRUSTED privilege.

Only one storage map on one storage unit may be produced in one run of the macro. However, the user is not required to supply any character peripheral input (program parameters) for #XPJD.

The #XPJD storage unit map will not fit on a 96-print-position line printer, and if the user's installation includes such a printer he should not leave it to the macro to LISTFILE the #XPJD output. If the user specifies that the line printer output is to be sent to his own file, he can then LISTFILE that file to a line printer having a suitable peripheral property.

There are no safeguards in the macro to prevent the user allowing the #XPJD line printer output to be sent to a MOP console, although the size and format of the storage unit map make it unsuitable for this form of output. A form of the listing parameter which sends the line printer output to the monitoring file system is available and it is left to the user who wishes to use this facility in the MOP context, to ensure (by means of the REPORT command) that object program output does not reach his console while the macro is running.

FORMAT

DAMAP *parameter list*

The parameter types in the parameter list are:

LISTING

ERROR

STORAGE UNIT

They have no fixed position and if an optional parameter is omitted no null parameter is required in its place. Only one parameter of each type will be effective in the macro.

Listing parameter

The listing parameter consists of the characters *LP optionally followed by a filestore file description. The format is:

**LP file description*

or

*LP

This parameter type is optional. If a file description is included, it specifies a file to which the program #XPJD is to write its line printer output. If no file description is given, it indicates that the line printer output is to be sent to the monitoring file system in the OBJECT category. In the MOP context this facility should be used with care (see *Restrictions* above). If the listing parameter is omitted altogether the macro will CREATE a workfile to receive the line printer output and, at the end of the run, LISTFILE it to a line printer and ERASE it.

Error parameter

The error parameter consists of the characters ER followed by a label. The format is:

ERlabel

DAMAP

This parameter type is optional. It specifies a point in the job description to which control should be passed if the macro detects an error in its parameters, if the run of the program fails to reach a satisfactory conclusion or, in the NOT MOP context, if a command error (such as might be caused by incorrect macro parameters) occurs in the macro.

Note: Like other system macros, this macro uses labels beginning with 9, and to avoid ambiguity the error parameters should not specify a label beginning with that character.

Storage unit parameter

The storage unit parameter consists of the characters SU followed by a DA storage unit serial number. The format is:

SU serial number

The serial number consists of six octal digits. Non-significant zeros must not be omitted. The number should not be enclosed in parentheses.

EXECUTION

The macro onlines to #XPJD the system file of the storage unit specified in the macro parameters and CREATES a workfile containing the necessary #XPJD parameter which it subsequently ERASEs. The contents of this workfile are designed to make #XPJD delete itself with the two-character message LO at the end of its run.

LIBRARY ENTRANTS ACCESSED

<i>Entrant</i>	<i>Mode of Access</i>
:LIB.PROGRAM XPJD	EXECUTE

ERROR MESSAGES

Error messages produced by program #XPJD are described in the specification of that program starting on page 273.

The only error message produced by the macro is the general message:

ERROR IN MACRO DAMAP: MACRO ABANDONED

after which the macro will GO TO the label specified by the error parameter, if supplied. This message will be produced if the #XPJD run fails to reach its expected termination, or, in the NOT MOP context, in the event of a command error in the macro arising from incorrect macro parameters or because the required storage unit cannot be loaded or because the user has not got the TRUSTED privilege.

DAREPORT

TITLE

DAREPORT

FUNCTION

To produce an #XPJJ file-level report on a specified DA exofile.

RESTRICTIONS

Only one report on one exofile may be produced by one run of the macro. However, the user is not required to supply any character peripheral input (program parameters) for #XPJJ.

FORMAT

DAREPORT *parameter list*

Parameter types in the parameter list are:

LISTING

ERROR

D.A. EXOFILE

They have no fixed position and if an optional parameter is omitted no null parameter is required in its place. Only one parameter of each type will be effective in the macro.

Listing parameter

The listing parameter consists of the characters *LP optionally followed by a filestore file description. The format is:

**LP file description*

or

**LP*

This parameter type is optional. If a file description is included, it specifies a file to which the program to be run is to write its line printer output. If no file description is given, it indicates that the line printer output is to be sent to the monitoring file system in the OBJECT category. In the MOP context this facility should be used with care, since the format and, in most cases, the volume of the line printer output make it unsuitable for output on a console typewriter. If the listing parameter is omitted altogether, the macro will CREATE a workfile to receive the line printer output and, at the end of the run, LISTFILE it to a line printer and ERASE it.

Error parameter

The error parameter consists of the characters ER followed by a label. The format is:

ER label

This parameter type is optional. It specifies a point in the job description to which control should be passed if the macro detects an error in its parameters, if the run of the program concerned fails to reach a satisfactory conclusion, or, in the NOT MOP context, if a command error (such as might be caused by incorrect macro parameters) occurs in the macro.

Note: Like other system macros, this macro uses labels beginning with 9, and to avoid ambiguity the error parameter should not have a label beginning with that character.

DA exofile parameter

The DA exofile parameter consists of the characters *DA followed by a DA exofile description. The format is:

DAREPORT

**DA exofile description*

The parameter is mandatory and specifies the DA exofile for which a file-level report is to be produced.

EXECUTION

The macro **ONLINEs** to **#XPJJ** the exofile specified in the macro parameters and **CREATEs** a workfile containing the necessary **#XPJJ** parameter, which it subsequently **ERASEs**. The contents of this workfile are designed to make **#XPJJ** delete itself with the two-character message **LO** at the end of its run.

LIBRARY ENTRANTS ACCESSED

<i>Entrant</i>	<i>Mode of access</i>
:LIB.PROGRAM XPJJ	EXECUTE

ERROR MESSAGES

Error messages produced by program **#XPJJ** are described in the specification of that program starting on page 297.

The only error message produced by the macro is the general message:

ERROR IN MACRO DAREPORT: MACRO ABANDONED

after which the macro will **GO TO** the label specified by the error parameter, if supplied. This message will be produced if the **#XPJJ** run fails to reach its expected termination, or, in the **NOT MOP** context, in the event of a command error in the macro arising from incorrect macro parameters, or because the specified **DA** exofile is not available. If the exofile description in the **DA** exofile macro parameter includes a storage unit serial number, a command error will occur if the required storage unit cannot be loaded. If no storage unit serial number is given, a command error will occur if the required exofile is not already on-line.

Index

Additional error procedure	202	GEORGE 3 macro for	235-238
Aligning printer paper		Data retention period, checking	117-121, 239-241
(off-line print)	178	DAUTIL	239-241
(record print)	216	Decimal fields	189, 194, 200
Allocator, of DA files	15-30, 186	Deleting	
GEORGE 3 macro for	239-241	file areas	17
Analysis of DA data files	69-80	files	15-30, 117-121, 239-241
Analysis of DA housekeeping format		fragmentation	7-13
files	283-295	Direct access	
Attempt in situ file reorganization	283-295	composite file structure report	167-176
		data file analysis	69-80
Batch copy of DA files	141	file allocator	15-30, 186, 239-241
Batch of parameters	3	file copy	7-13, 15-30, 139-165, 239-241
Binary fields	190, 197	file copy, dump and restore	81-116
Block address	6	file, creation and extension of in dumping	83
Block print	61-68	file creation from cards or tape	185-213
GEORGE 3 macro for	235-238	file retention	16, 117-121
Bucket packing density	187	file organization	139-165
Bucket print		file print (any data format)	61-68, 235-238
(data format)	61-68, 235-238	file print (housekeeping format)	47-59, 235-238
(housekeeping format)	47-59, 235-238	housekeeping file analysis	283-295
		identifiers	251-256
Cards, create DA file from	185-213	map and summary file report	273-281, 305-306
GEORGE 3 macro for	243-246	off-line print	178-183
Cartridge report	31-45	Dump and restore	81-116
GEORGE 3 macro for	231-233	Dump files	83
Change data retention period	117-121, 239-241		
Character fields	189, 190, 194, 197	Editing routines	126
CHART	168-169, 172, 173	ERCS	177-183
Checking		Error B,D,J,M or P	5.6
criteria	198	Error procedure	202, 203
sequence	200	Exception conditions	4
Composite file		Extended repertoire character set	177-183
copy	7-13	Extension of DA dump files	83
structure report	167-176		
Constants	190, 197	Fields	
Copy DA file	7-13, 15-30, 81-116, 139-165	input to #XREB/#XREJ	189
GEORGE 3 macro for	239-241	output by #XREB/#XREJ	190
Create		parameter	2
additional file area	16	File allocator	15-30
additional SCA	258	GEORGE 3 macro for	239-241
DA file from cards or paper tape	185-213, 243-246	File area	
new DA file	15	creating	16
Creation and extension of DA dump files	83	deleting	17
Criteria checking	198	File bucket or hardware block print	61-68
		GEORGE 3 macro for	235-238
DAFAM	231-233	File copy program	126, 139-165
DAMAP	305-306	introduction to	123-124
DAPPRINT	235-238	operating	154
DAREPORT	307-308	parameters for	149-152
Data file analysis	69-80	File creation from cards or paper tape	185-213
Data format print	61-68	GEORGE 3 macro for	243-246
		File level report	297

File or cartridge report	31-45	Organization, <i>see</i> File organization	
GEORGE 3 macro for	231-233	Overflow routine	127
File organization program	139-165		
generation of	125	Packing density	187
introduction to	123-124	Paper tape, create DA file from	185-213
operating	153	Parameter batch	3
parameters for	143-148	Parameter field	2
File organization software	123-165	Parameter formats	2
introduction to	123-124	Parameter set	3
File organization subroutines	124	Parameter type	3
File print		Parity errors	9
data format	61-68, 235-238	Print DA file	
housekeeping format	47-59, 235-238	data format	61-68, 235-238
off-line	177-183	housekeeping format	47-59, 235-238
Files, report of	273-281	off-line	177-183
Fill character	187	records	215-229, 247-249
Final summary	73	Printer paper, alignment of	178, 216
Format of parameters	2	Printer report, <i>see</i> Report	
Fractional fields	221	Printer spacing	219
Fragmentation, deletion of	7-13	Punching of parameters	3
Full report			
of analysis	286	Q-trusted utility programs	7-229
of contents of data file	69-80		
of reorganization	284	Random data file analysis	69-80
Generate a segment for file organization		totals	73
or file copy	125-138	Randomizing routine	127
GEORGE 3 system macros	4, 231-249	Record print (housekeeping format)	47-59
Group of parameters	3	Reorganize DA file	140, 283-295
		Report	
Hardware block print	61-68	file or cartridge	31-45, 231-233
GEORGE 3 macro for	235-238	map and summary file	273-281
Housekeeping format file print	47-59	of analysis	284
GEORGE 3 macro for	235-238	of composite file structure	167-176
HRACOMINF	127	of contents of data file	69-80
HRAEDITIN1	126, 127	of DA records	215-229, 247-249
HRAEDITIN2	126	of reorganization	284
HRAEDITOUT	126	of utilization and storage	297-304
HRARND	126, 127	Reset multi-storage unit file	267-272
HRATAGREC	126-128	Reset system control area	257-265
		Restore DA file	81-116
Identifiers, writing	251-256		
In situ file reorganization	283-295	S-trusted utilities	5, 251-272
		Scaled decimal fields	221
LIST	170	SD message	5
Load DA file	140	Sequence checking	200
		Sequential data file analysis	69-80
Macros	4, 231-249	totals	72-73
Magnetic tape		Serial data file analysis	69-80
block count checking	83	totals	72
dump file	82, 83	Set of parameters	3
Map and summary file report	273-281	Short report	
Mixed decimal and fractional fields	221	of analysis	287
Mode compatible file organization programs	129	of contents of data file	69-80
MS message	5	of reorganization	285
Multi-storage unit file reset	267-272	SHORTLIST	169
		Storage report	297-304
Non-trusted utility program	7-229	Storage unit level report	297
Numeric fields	190, 197	Structure report of composite file	167-176
		Subroutines, for file organization	124
Octal fields	189, 195	Summary file report	273-281
Off-line print	177-183	Summary report	
		of analysis	288

of contents of data file	69-80
of reorganization	285
System control area	
create	15, 285
multi-storage unit reset	267-272
reset	257-265
system macros	4, 231-249
Transfer DA file	140
Type of parameter	2
User-coded segments	126
Utilization and storage report	297-304
Validating file for #XREB/#XREJ	186
Validity check failures	202
Write DA identifiers	251-256
#XMFR	7-13
#XPJ1	251-256
#XPJ2	257-265
#XPJ7	267-272
#XPJC	15-30
GEORGE 3 macro for	239-241
#XPJD	273-281
GEORGE 3 macro for	305-306
#XPJH	283-295
#XPJJ	297-304
GEORGE 3 macro for	307-308
#XPJK	31-45
GEORGE 3 macro for	231-233
#XPJL	47-59
GEORGE 3 macro for	235-238
#XPJQ	61-68
GEORGE 3 macro for	235-238
#XPJR	69-80
#XPJW	81-116
#XPJX	117-121
GEORGE 3 macro for	239-241
#XPSZ	
generated program	139-165
introduction to	123-124
specification	125-138
#XRAA	167-176
#XRAF	177-183
#XREB	185-213
GEORGE 3 macro for	243-246
#XREJ	185-213
GEORGE 3 macro for	243-246
#XRPE	215-229
GEORGE 3 macro for	247-249

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.

[illegible]

